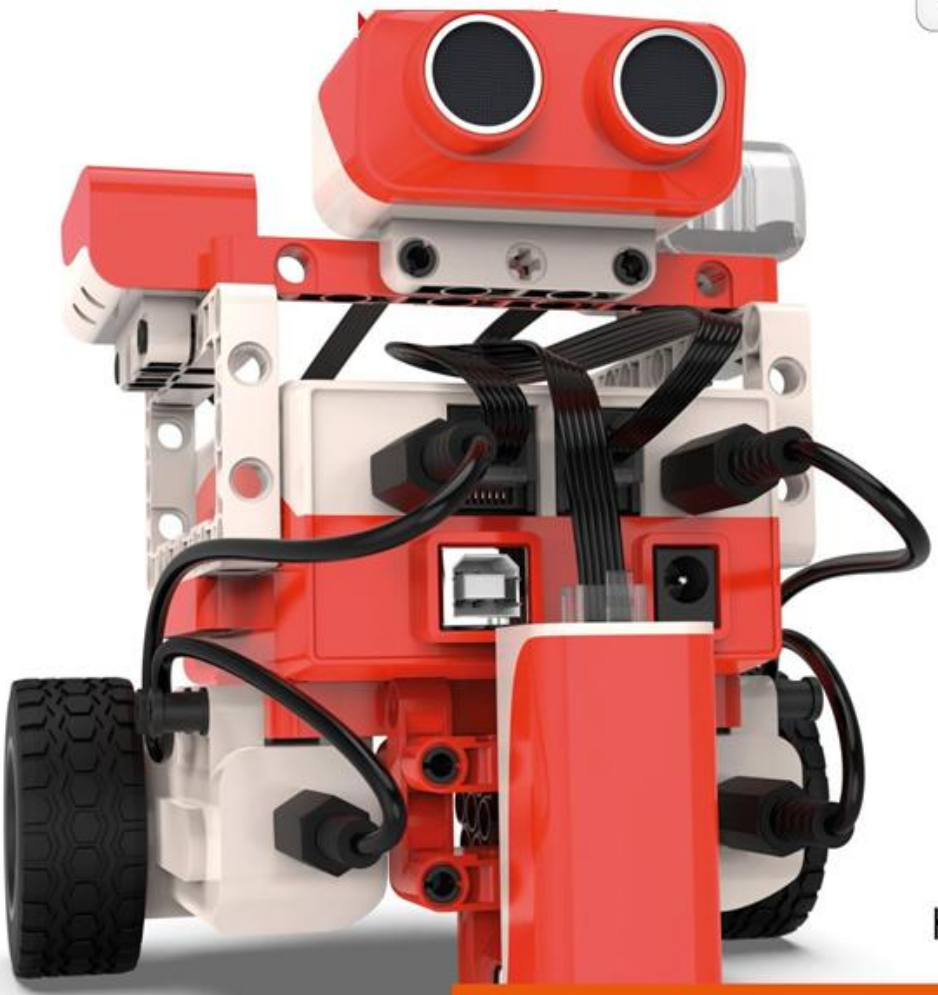
КИБЕРТЕХНИК

БСКомп

CYBER TOY

STEM



Корягин А.В.

Методические рекомендации

Методика построения
образовательного процесса
по направлению «Робототехника»
с использованием набора НикиРобот

МЕТОДИКА ПОСТРОЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА ПО НАПРАВЛЕНИЮ «РОБОТОТЕХНИКА» С ИСПОЛЬЗОВАНИЕМ НАБОРА КЛИК

Авторы: Корягин А. В.
Филимонов А.С

1. Введение

Дорогие читатели перед вами книга, в которой содержится методика работы с робототехническим конструктором нового поколения **КЛИК**.

Как известно, для реализации таких задач ФГОС, как интеллектуальное творческое развитие дошкольников и инженерно-технического творчества школьников рекомендовано использовать образовательные робототехнические конструкторы.

На образовательном рынке существуют множество образовательных конструкторов, которые в разной степени решают задачи в области обучения таким точным дисциплинам как: физика, математика химия, инженерия, программирование и т.д.

Как показала многолетняя практика преподавания робототехники, в наборах ценят две вещи:

- модульность и наличие разнообразия видов крепления (под силу ребёнку с 9 лет) с разнообразием деталей

- обширная функциональная возможность набора: разнообразие датчиков, количество актуаторов (моторов).

Лидирующую позицию по первой категории оценивания занимает наборы Lego, а по второй Arduino. И эволюционным звеном двух этих продуктов стал **КЛИК**.

КЛИК – представляет собой набор, состоящий из деталей, схожих по инженерному решению с деталями Lego technic, но имеющих ряд разнообразных преимуществ и электрокомпонентами, разработанными на базе плат Arduino и датчиков с модулями, совместимых с платами Arduino. Данное решение даёт ряд преимуществ:

- понижает возрастной порог обучения робототехнике;

- расширяет диапазон разработок роботов и роботизированных систем в научно-исследовательском, инженерно-техническом и спортивно-соревновательном ключе.

Первое преимущество вытекает из-за дизайна продукта и технических решений. Все электронные компоненты вложены в защитные пластиковые контейнеры. Данное решение защитит датчик или модуль от механических повреждений или случайном возникновении короткого замыкания. Очень хорошо развита система соединений деталей. Детали обладают от двух до трёх степеней свободы в области крепления и полностью совместимы с деталями Lego technic. Соединительные провода прочные и крепятся только в определённом положении. Данная технология позволяет снизить возрастной порог обучения робототехнике до 7 лет.

Второе преимущество связано с разнообразием аппаратной части Arduino систем. На сегодняшний день насчитываются более 90 датчиков и модулей, которые, непосредственно, разрабатывались под платы Arduino, не считая той электроники, которая

может быть совместима по техническим характеристикам. Набор содержит универсальный переходник для подключения любого датчика, совместимого с Arduino.

Программное обеспечение на данном моменте так разнообразно, что позволяет программировать устройства на Arduino с 7 лет как на графико-визуальном языке (разновидность Scratch), так и текстовом языке высокого уровня C++, Java и т.д.

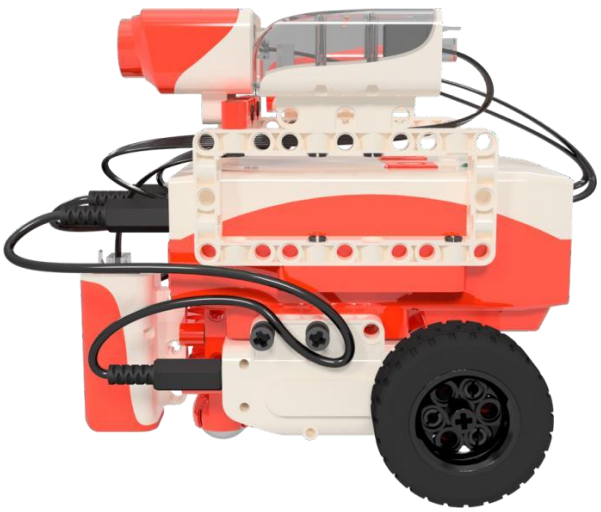
Робототехника развивается и расширяет горизонты познания. Будущее технического прогресса, как и науки – это комбинирование множества решений и направлений.

Данный методический сборник постарается помочь в изучении этого универсального робототехнического набора и реализовать его потенциал на учебный год.

Для кого эта книга

Данная книга – методическое пособие по работе с набором КЛИК и, в первую очередь, предназначена для преподавателей робототехники, информатики, физики и технологии, но и может использована в домашнем обучении родителями.

Сборник содержит методику обучения робототехнике детей, начиная с 7 лет и опирается на методику обучения применяемых в Lego и Arduino.



Содержание сборника

Книга содержит информацию разного уровня сложности, предназначенную для детей разной подготовленности.

Уровни сложности определяются в книге маркировками: A, B и C.

A – лёгкий уровень

B – средний уровень

C – сложный уровень

Сборник содержит:

- базовую информацию о комплекте набора **КЛИК** и технических характеристиках электронных компонентов и ПО.
- поурочные проекты в области робототехники.
- календарно-тематическое планирование на учебный год.
- ссылки на программы проекта.

Все коды из книги можно скачать по ссылке:
https://github.com/Antipat/CyberBot.git

Об авторе

Корягин Андрей – выпускник Пензенского государственного педагогического университета им. В.Г. Белинского физико-математического факультета.

Автор книги и рабочей тетради «Образовательная робототехника Lego WeDo», «Физические эксперименты с Lego EV3». Победитель конкурса «Молодой учитель 2011» Пензенской области в номинации информационные технологии. Ведущий специалист в области образовательной робототехники, обучения детей языкам программирования (Python, C++, C#) и 3D моделирования. Автор научных статей в области применения информационных технологий в образовании.

Оглавление

2. Содержание набора КЛИК

За основу управления электронной частью робота отвечает плата Arduino. Все электронные компоненты совместимы с данной платой. Весь набор выдержан в едином стиле. Элементам несущей конструкции и крепёжной системы служит пластик ABS, который устойчив к механическим, термическим и химическим воздействиям. Все корпуса, в которых заключена электроника съёмные.

1. Блок управления
2. Аккумулятор на 300 мАч, 7.2В
3. DC моторы
4. Сервопривод
5. Ультразвуковой датчик расстояния
6. Датчик линии спаренный
7. IR модуль
8. Датчик цвета
9. Bluetooth модуль
10. IR пульт
11. Соединительные провода
12. USB шнур
13. Детали для сборки робота
14. Крепёжные детали
15. Аккумулятор питания
16. Блок питания

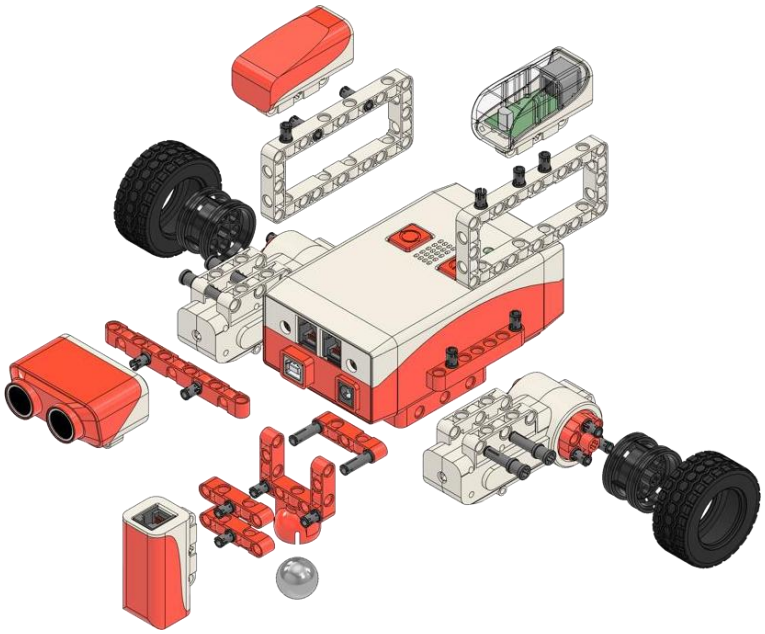


Рис.1 Содержание набора КЛИК

На Рис.1 представлен набор КЛИК, рассмотрим подробнее, что представляют его составные части.

2.1. Блок управления

Мозгом роботизированного устройства является блок управления, который имеет два вывода для подключения DC моторов и шесть выводов для подключение датчиков и модулей. Отдельно выведен USB порт для загрузки программы и порт питания для зарядки аккумулятора.

Блок оснащён пьезоэлементом для подачи звуковых сигналов и светодиодом для подачи световых сигналов. Также присутствуют кнопка включение/выключения и перезагрузки (сброс). Четвёртый порт необходим для подключения Bluetooth модуля.

Элементы блока управления:

- кнопка ВКЛ/ВЫКЛ
- кнопка Reset (перезагрузка)
- USB вход
- питание аккумулятора
- выводы для подключения DC моторов
- световой индикатор
- вывод звука

П1, П2, П3, П4, П5,П6 – порты для подключения датчиков и модулей.

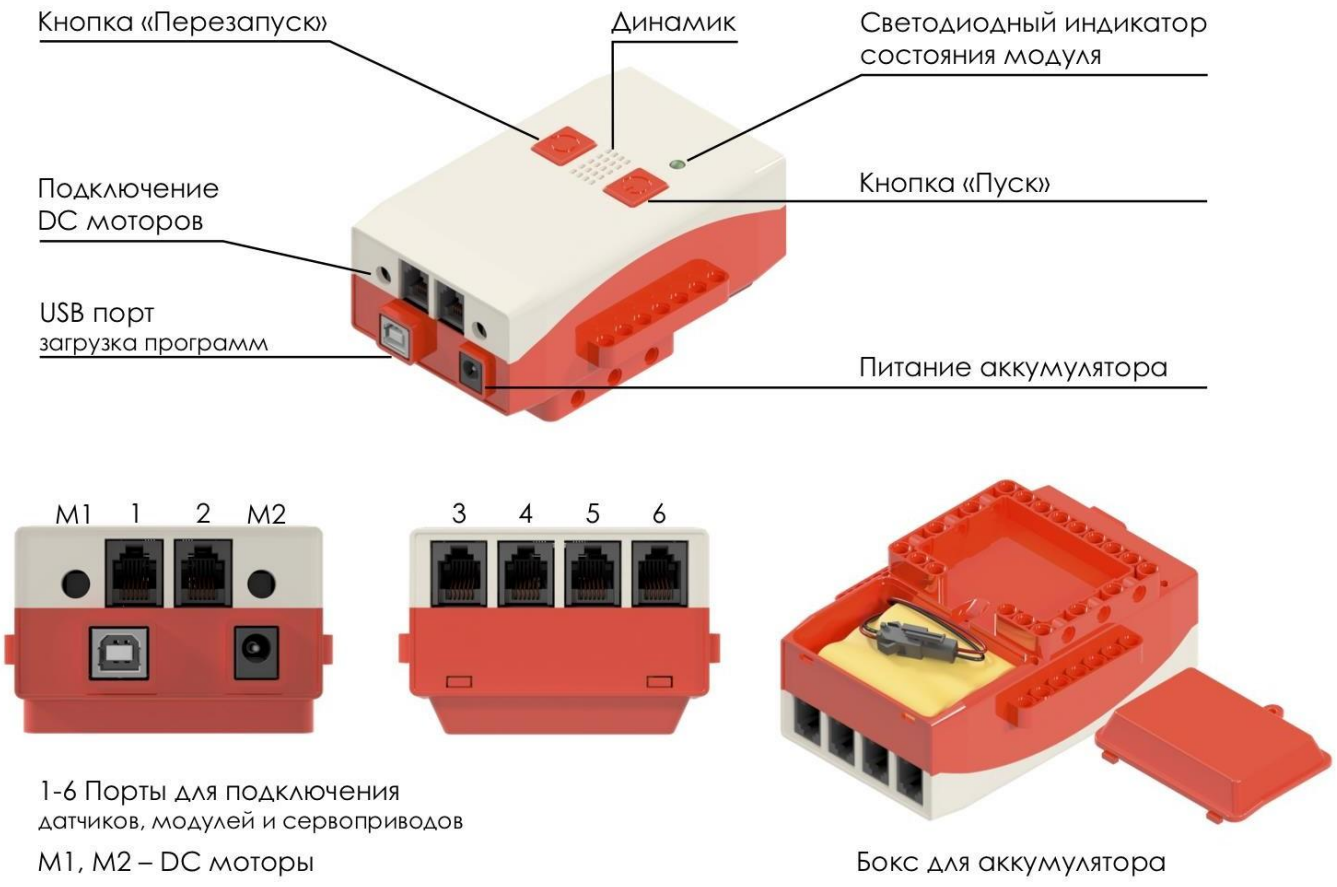


Рис.2-4 Структура блока управления КЛИК

2.2. Аккумулятор на 300 мАч, 7.2В

Мы рассмотрели внешнюю часть конструкции, теперь настало время заглянуть внутрь.

На рисунке 5 и 6 представлена внутренняя часть блока управления:

- **Arduino Uno**
- **расширение под плату Arduino Uno**

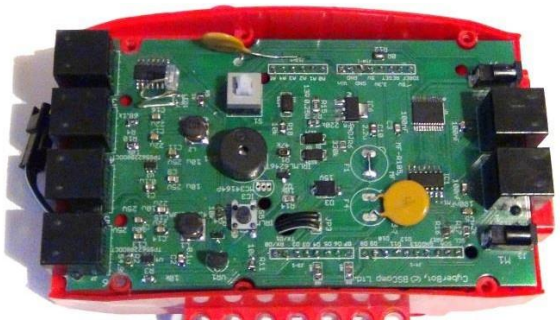


Рис.5 Внутренняя структура блока управления КЛИК

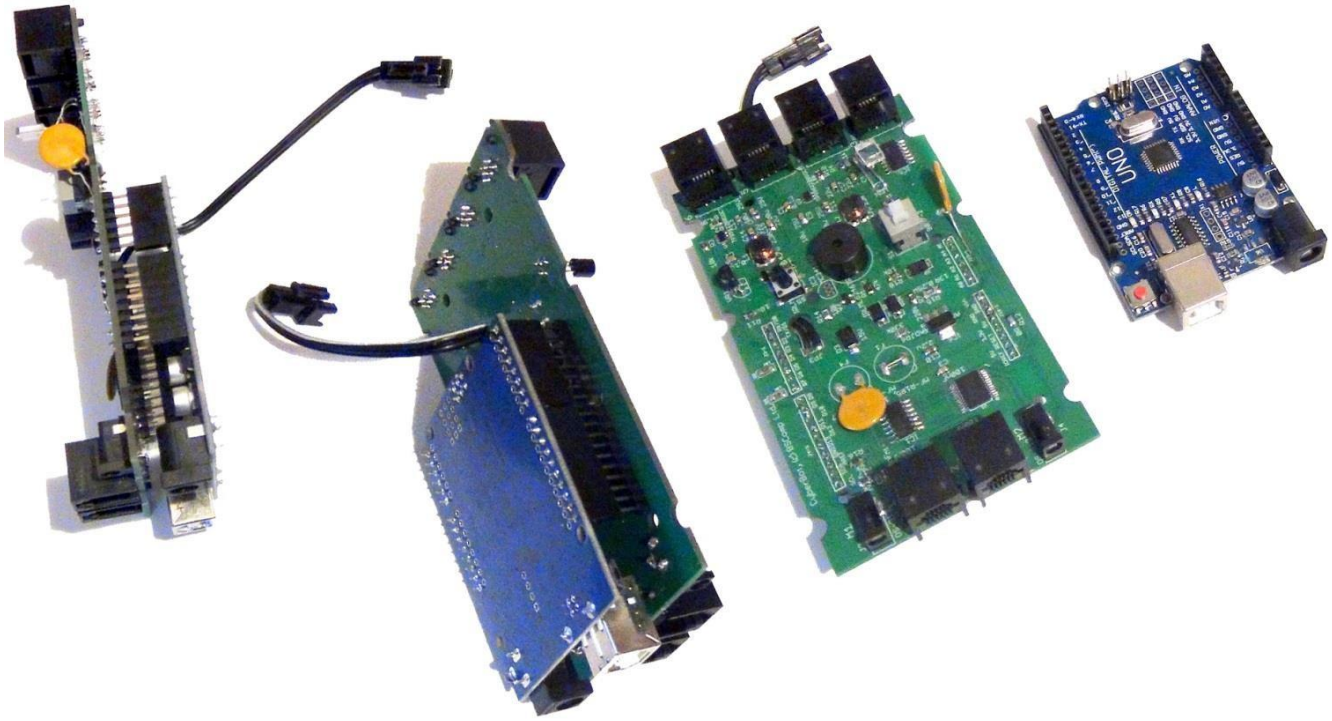
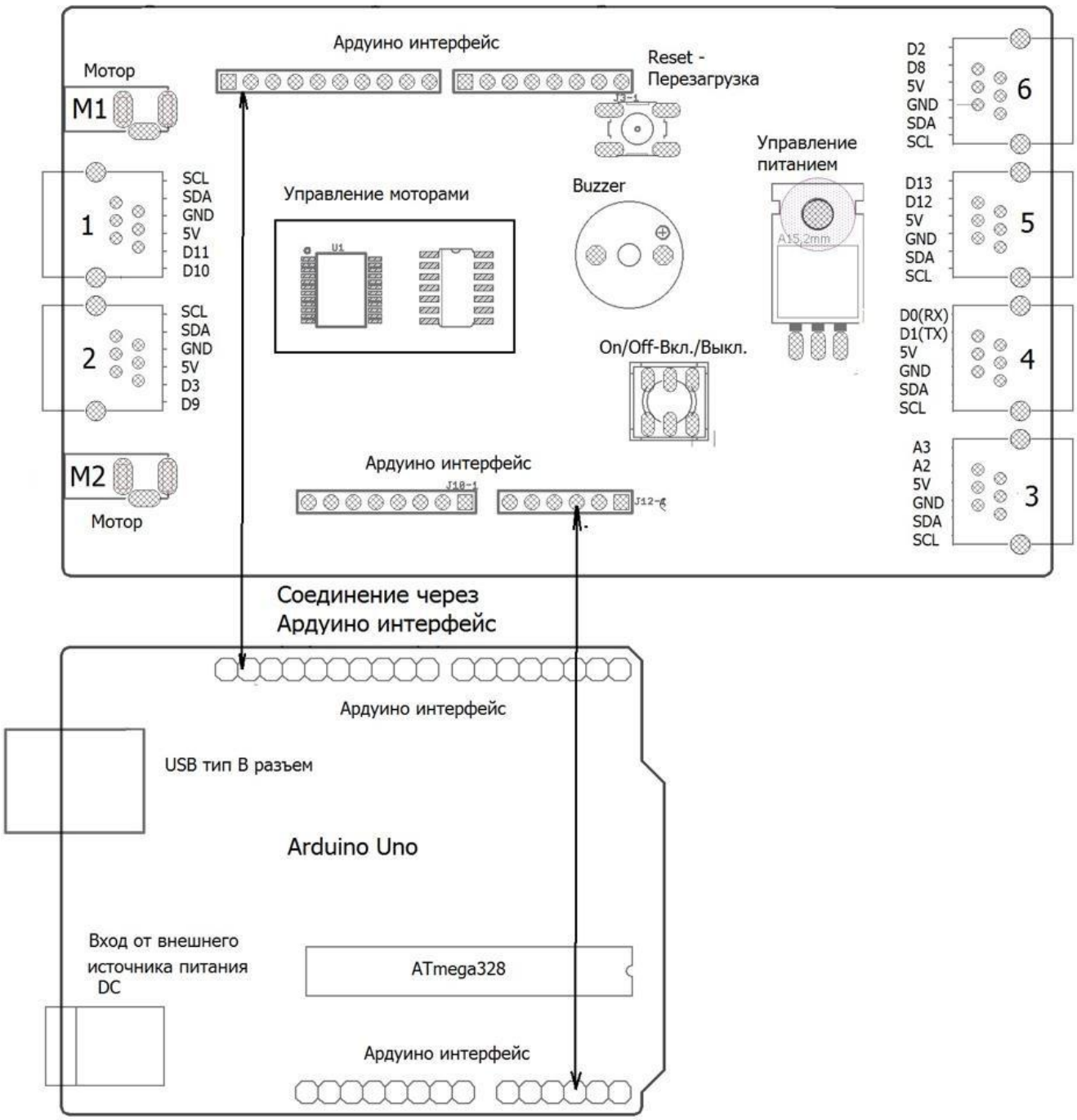


Рис.6 Внутренняя структура блока управления КЛИК

Соответствие портов КЛИКа и Arduino Uno

Порт КЛИК	Ардуино (Arduino Uno) (вход/выход)	Примечания. Рекомендуемые устройства и датчики для подключения.
1	D10, D11, SDA, SCL	Сервопривод, Ультразвуковой датчик расстояния, IR модуль, Датчик линии, Датчик цвета. Датчик касания, Цифровой вход/ выход Ардуино, I2C интерфейс.
2	D9, D3, SDA, SCL	Сервопривод, Ультразвуковой датчик расстояния, IR модуль, Датчик цвета. Датчик касания, Цифровой вход/ выход Ардуино, I2C интерфейс.
3	A3, A2, SDA, SCL	Датчик цвета, Аналоговый вход Ардуино, Дополнительные датчики Температуры, Влажности и.т.п. I2C интерфейс
4	D0(Rx), D1(Tx), SDA, SCL	Последовательный порт, Bluetooth модуль, Сервопривод, Ультразвуковой датчик расстояния, IR модуль, Датчик линии, Датчик цвета. Датчик касания, Цифровой вход/ выход Ардуино, I2C интерфейс.
5	D13, D12, SDA, SCL	Сервопривод, Ультразвуковой датчик расстояния, IR модуль, Датчик линии, Датчик цвета, Датчик касания, Цифровой вход/ выход Ардуино, I2C интерфейс.
6	D2, D8, SDA, SCL	Сервопривод, Ультразвуковой датчик расстояния, IR модуль, Датчик линии, Датчик цвета. Датчик касания, Цифровой вход/ выход Ардуино, I2C интерфейс.
M 1	D6, D7	DC мотор 1. Для управления мотора используется цифровой выход Ардуино. D6 - скорость, D7 - направление
M 2	D4, D5	DC мотор 2. Для управления мотора используется цифровой выход Ардуино. D5 - скорость, D4 - направление

Buzzer (Динамик) - выход A0
Включение последовательного порта - выход A4.


Ардуино интерфейс
Мотор
M1
Reset -
Перезагрузка
J3-1
Управление
питанием
D2
D8
5V
GND
SDA
SCL
6
SCL
SDA
GND
5V
D11
D10
1
Управление моторами
U1
Buzzer
A15.2mm
D13
D12
5V
GND
SDA
SCL
5
SCL
SDA
GND
5V
D3
D9
2
On/Off-Вкл./Выкл.
D0(RX)
D1(TX)
5V
GND
SDA
SCL
4
M2
Мотор
Ардуино интерфейс
J10-1
J12-6
A3
A2
5V
GND
SDA
SCL
3
Соединение через
Ардуино интерфейс
Ардуино интерфейс
USB тип B разъем
Arduino Uno
Вход от внешнего
источника питания
DC
ATmega328
Ардуино интерфейс

2.3. DC моторы

Данные моторы – это обычные электромоторы с редукторами на 3–5 Вольт. Оснащены дисковыми элементами с двух сторон для крепление колёс на шине или зубчатых колёс с использованием, как осей, так и штифтов. Моторы имеют легко отличимые провода со штекерами.

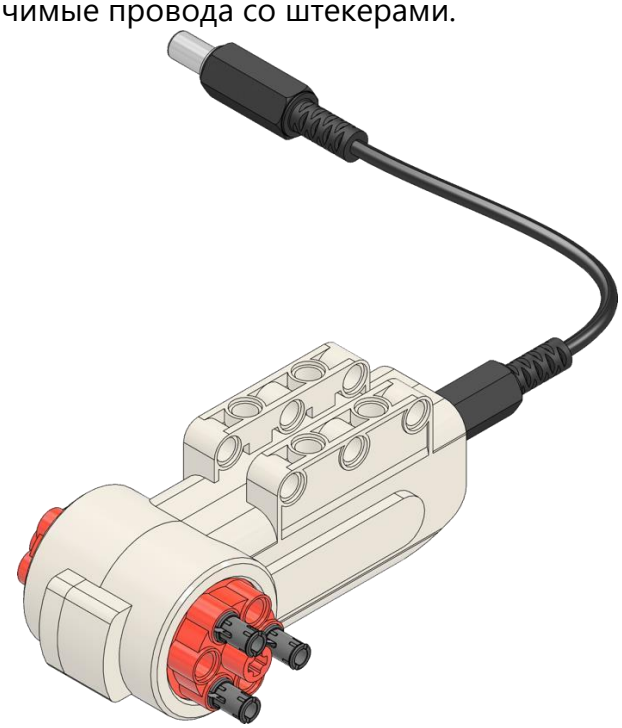


Рис.7 DC моторы КЛИК. Внешний вид

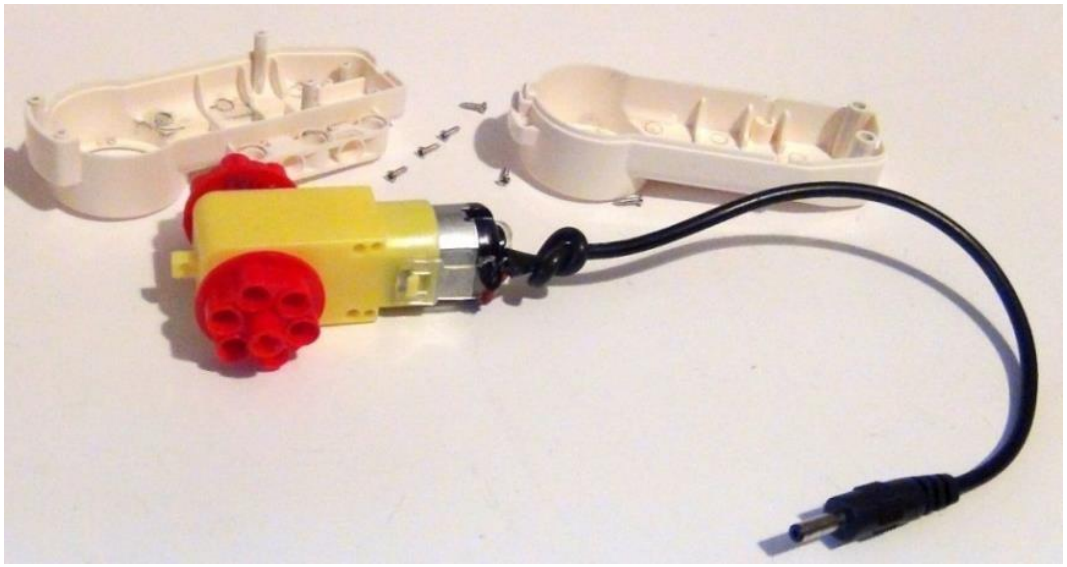


Рис.8 DC моторы КЛИК. Внутреннее строение

Для них зарезервированы пины на Arduino:
Левый мотор – 5 и 6 пин
Правый мотор – 4 и 5 пин

2.4. Сервопривод

Сервопривод предназначен для точного поворота. Точность поворота определяется градусной мерой. В наборе представлен сервопривод с градусом поворота $0^0 - 180^0$. Момент силы данного привода составляет 2 кг/см.

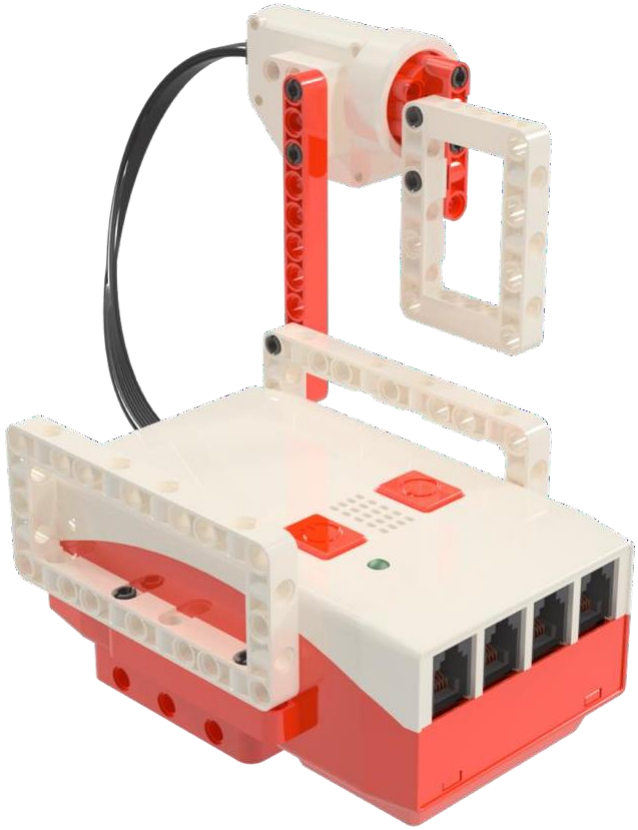


Рис.9 Сервопривод КЛИК. Внешний вид

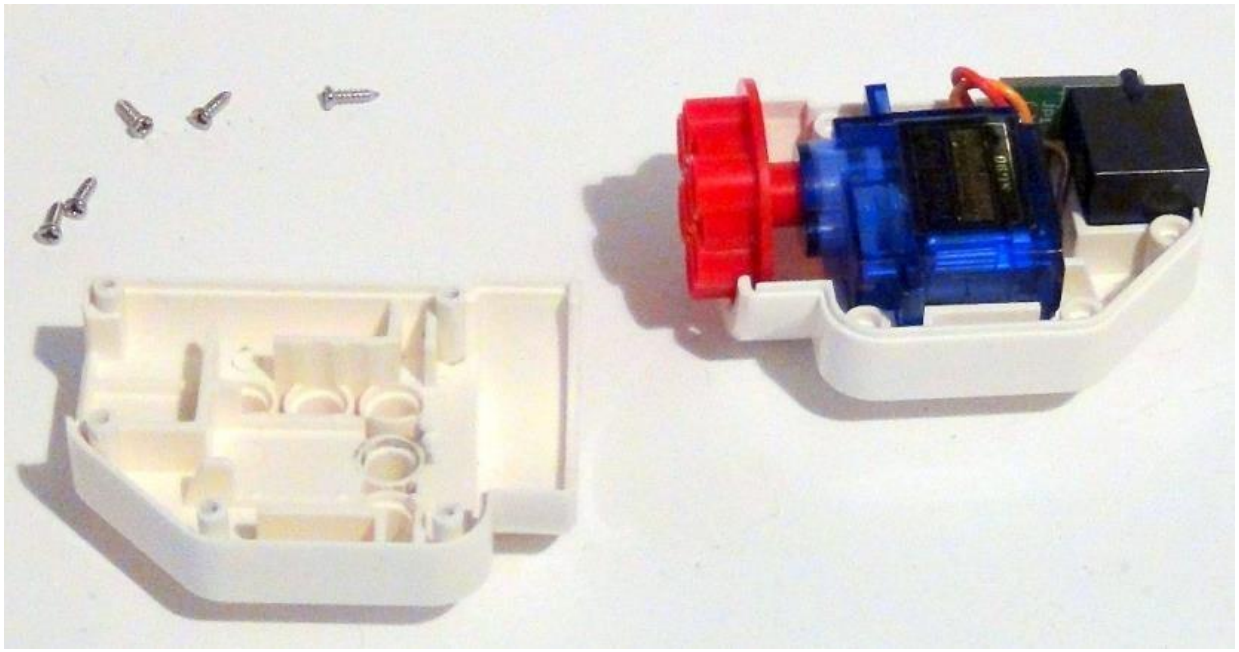


Рис.10 Сервопривод КЛИК. Внутреннее строение

2.5. Ультразвуковой датчик расстояния

Ультразвуковой датчик расстояния измеряет расстояния до объекта. Данный датчик часто применяется в робототехнике. В наборе идет датчик HC-SR04. Диапазон измерения до 4 метров.

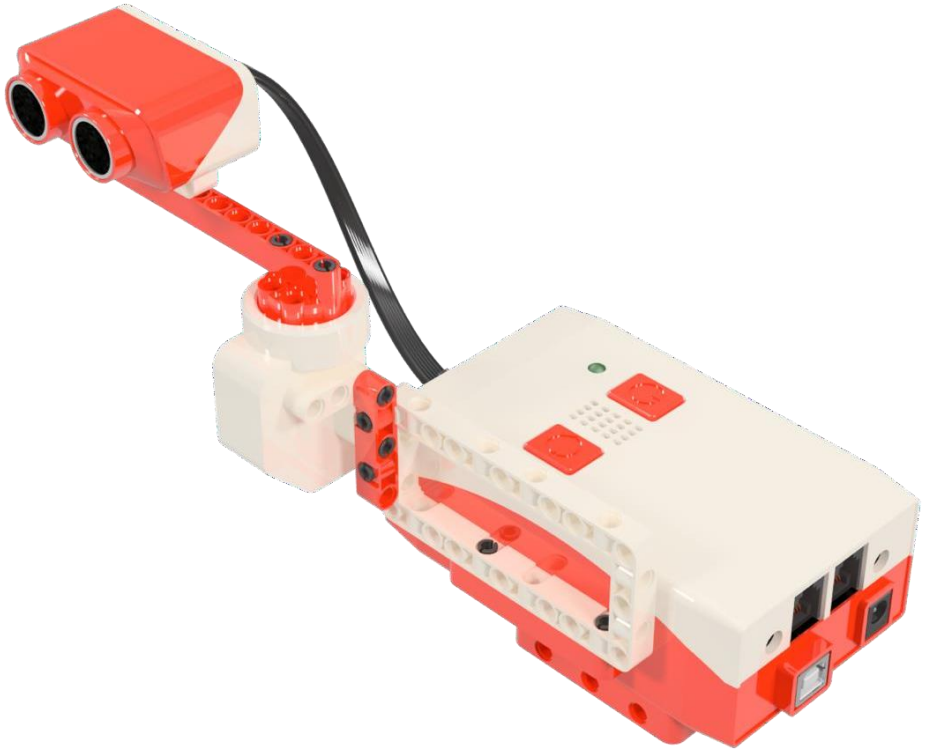


Рис.11 Ультразвуковой датчик расстояния КЛИК.

Рис.12 Ультразвуковой датчик расстояния КЛИК. Внутреннее строение

2.6. Датчик линии спаренный

Датчик линии в основном применяется для движение мобильного робота по чётко выделенной линии, либо белая, либо чёрная.

Рис.13 Датчик линии КЛИК

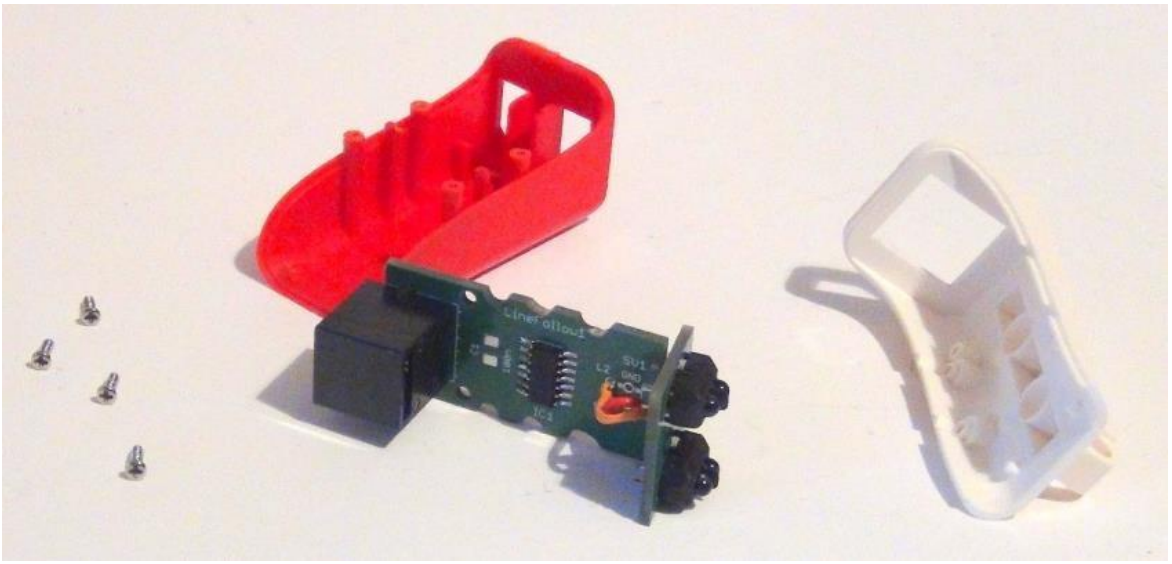


Рис.14 Датчик линии КЛИК. Внутреннее строение

2.7. IR модуль

ИК модуль (IR) предназначен для приёма сигналов в инфракрасном диапазоне. С помощью IR пульта можно управлять роботизированным устройством КЛИК.

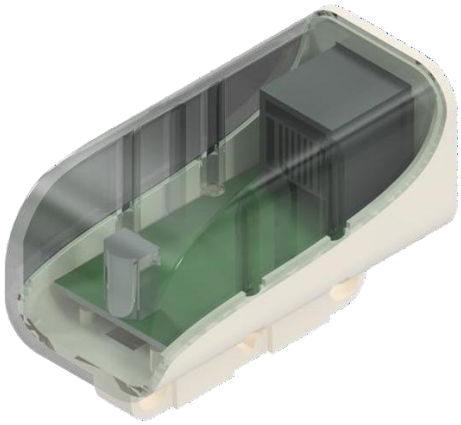


Рис.15 IR модуль КЛИК

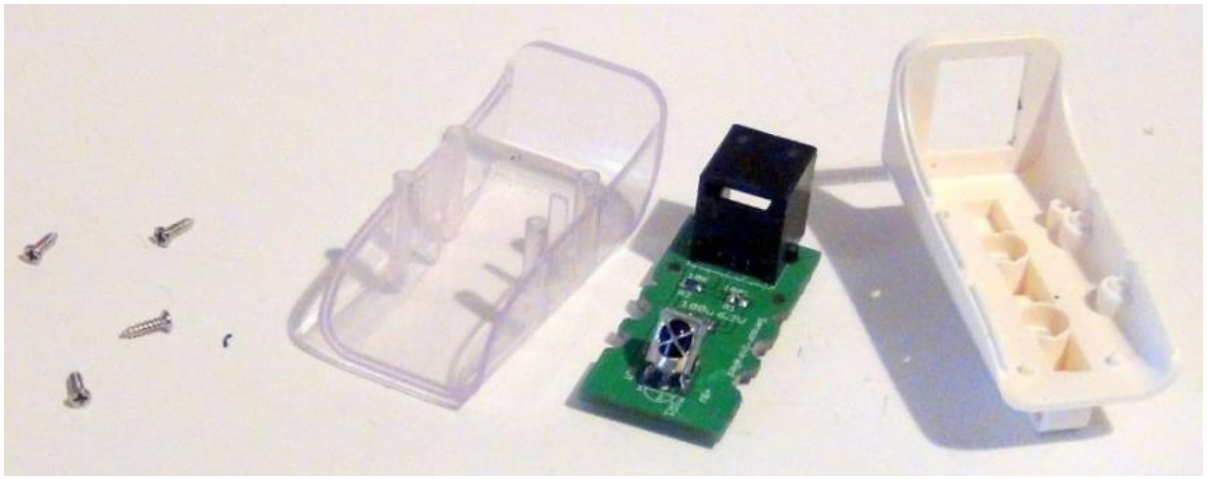


Рис.16 IR модуль КЛИК. Внутреннее строение

2.8. Датчик цвета

Датчик цвета применяется для определения цвета и освещённости. Применяется в устройствах, где необходимо распознать объекты разного цвета, либо провести замер освещённости.

Рис.17 Датчик цвета КЛИК

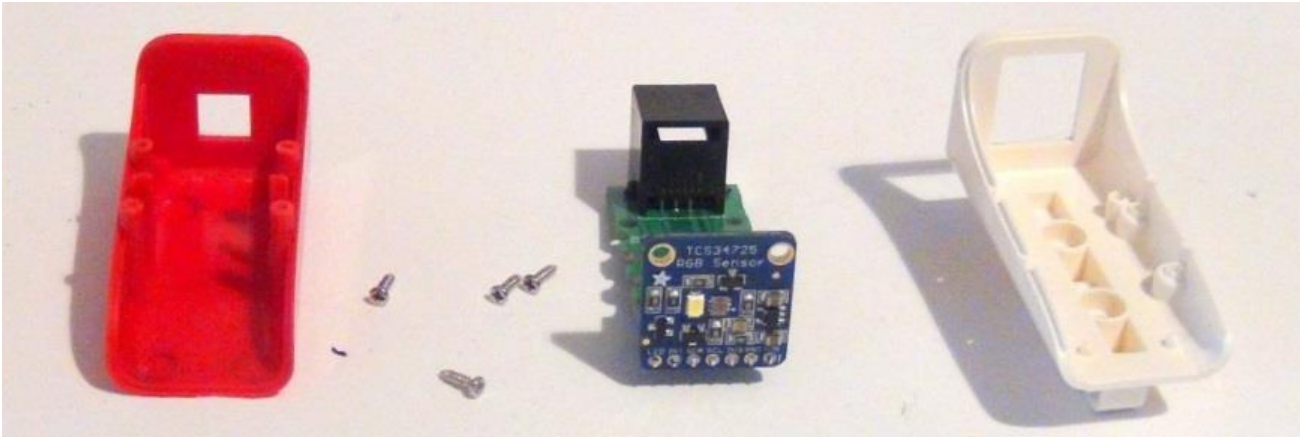


Рис.18 Датчик цвета КЛИК. Внутреннее строение

2.9. Bluetooth модуль

Данный модуль предназначен для дистанционного управления роботизированным устройством с помощью радиосигнала. Дальность управления на открытой местности порядка 10 -15 метров.

Рис.19 Bluetooth модуль КЛИК

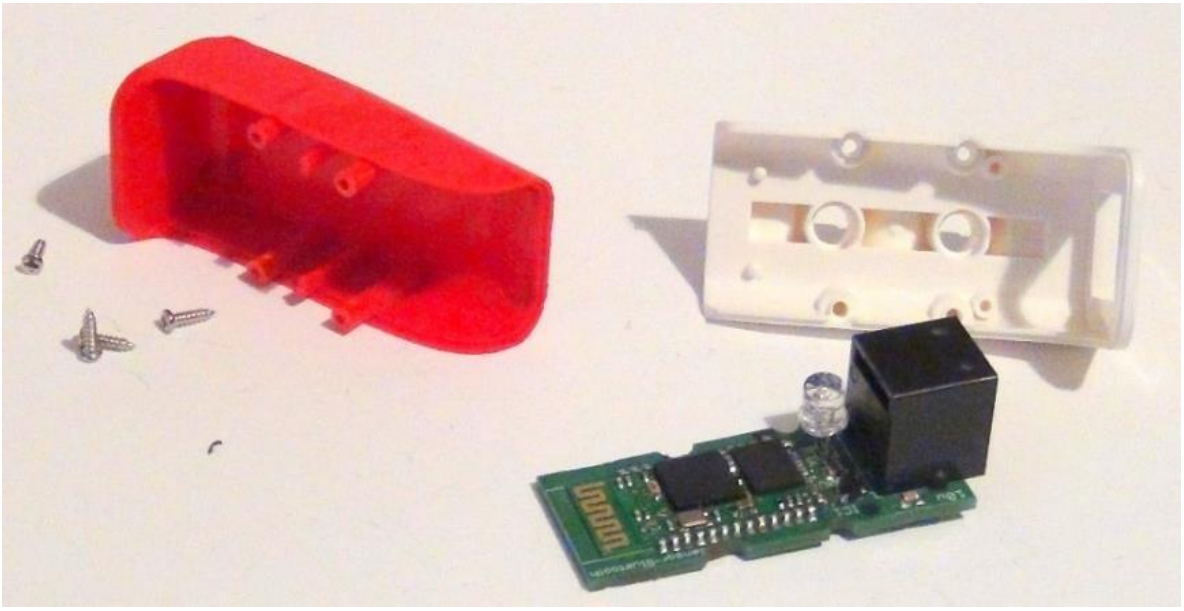


Рис.20 Bluetooth модуль КЛИК. Внутреннее строение

2.10. IR пульт

IR пульт применяется для управления роботом, совместно с IR модулем.

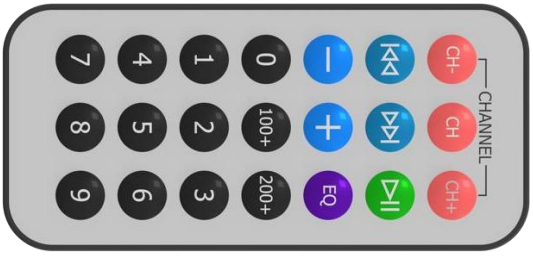


Рис.21 IR пульт КЛИК

2.11. Соединительные провода

Провода имеют штекеры модели 6P6, которые применяются для интернет подключения.

Рис.22 Соединительные провода КЛИК

2.12. USB шнур

Данный шнур предназначен для загрузки программ на Arduino и получения данных через последовательный порт.

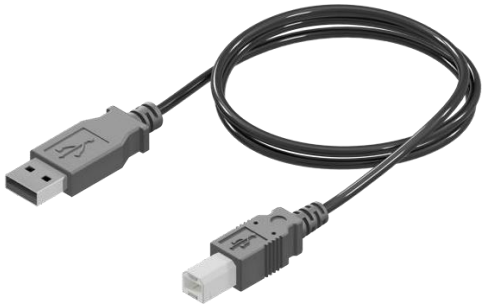


Рис.23 USB шнур КЛИК

2.13. Детали для сборки робота

Кроме электроники должна быть несущая конструкция, на которую и крепится вся электроника. Конструкторское решение обеспечивают детали: балки, рамки, зубчатые колёса, оси.

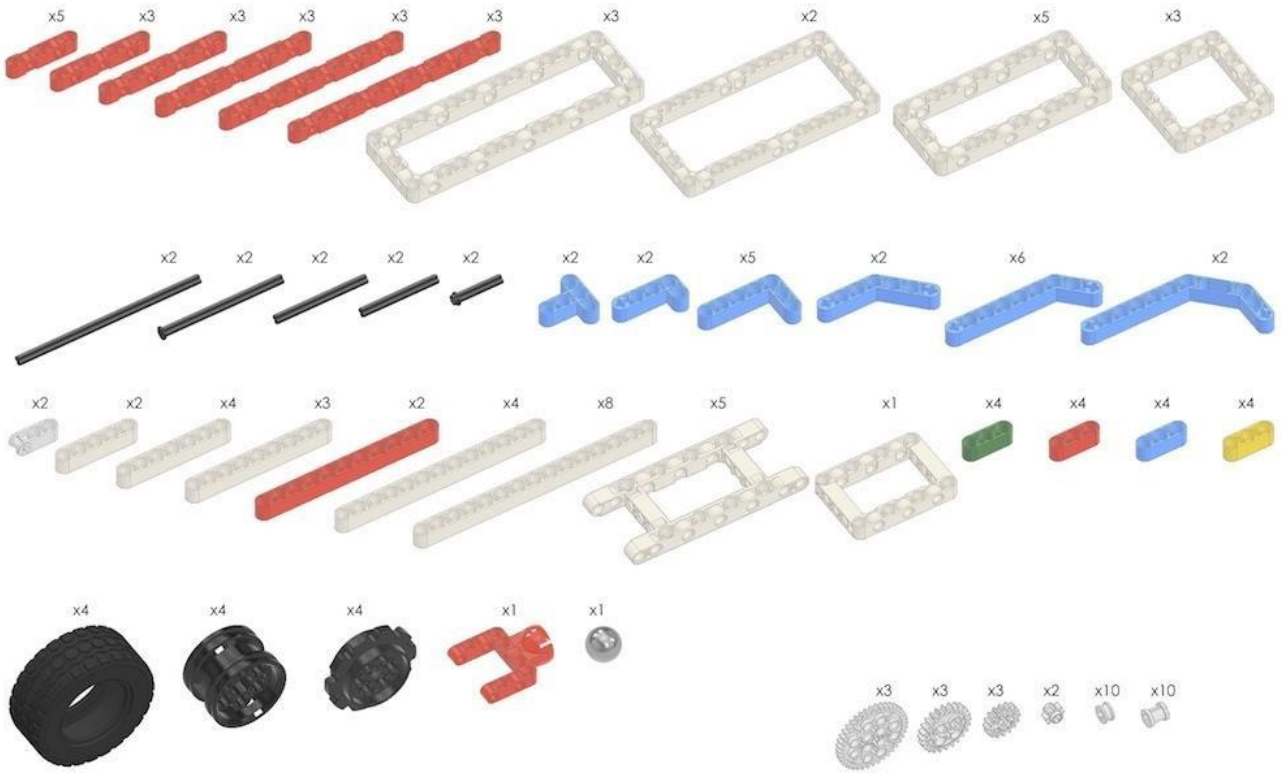


Рис.24 Детали КЛИК

2.14. Крепёжные детали

Для прочности конструкции робота применяют крепёжный материал – это штифты и втулки

Рис.25 Крепёж КЛИК

2.15. Аккумулятор питания

Многозарядный источник питания робота.

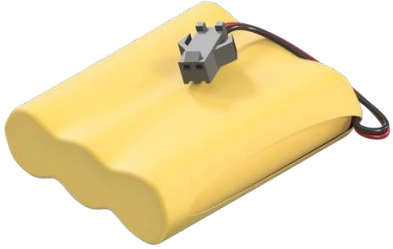


Рис.26 Аккумулятор КЛИК

2.16. Блок питания

Со временем аккумулятор разряжается и его необходимо, периодически, заряжать. Для этого есть блок питания.

Рис.27 Блок питания КЛИК

2.17. DC провод x2

Рис.28 DC провод x2

3. Программное обеспечение

Так как в основе «мозга» набора КЛИК лежат микроконтроллеры Arduino, то работа с ними возможна с помощью таких программных сред, как:

1. **Arduino ide**
2. **ArduBlock**
3. **MBlock3**
4. **MBlock5**

Возможно использовать и другое программное обеспечение, но перечисленные среды уже оптимизированы для данных плат и не требуют сложных настроек.

3.1. Программирование в среде Arduino ide

Данная среда была разработана производителями плат Arduino, поэтому в программировании и управлении плат не должно возникнуть проблем.

Официальная страница для скачивания программы:
https://www.Arduino.cc/en/Main/Software

Рис.29

Как видно, на странице присутствует множество вариантов для разных операционных систем. Рассмотрим процедуру установки Arduino ide на windows.

Для windows можно воспользоваться тремя способами установки:

- скачать установщик
- скачать архив с установщиком

- установить программу через магазин windows[1].

Самый простой способ – это первый. Нажимаем по надписи **«Windows Installer, for Windows XP and up»**.

После нажатия нас перебросит на страницу добровольной помощи разработчикам. Здесь вы можете оказать финансовую помощь, если захотите

Рис.30 Страница добровольной помощи разработчикам arduino ide

Можно не оказывать помощь и сразу перейти к скачиванию. Для этого вам нужно нажать на кнопку «**JUST DOWNLOAD**». Далее, начнётся процедура скачивания.

Запускаем скаченный файл и устанавливаем программу согласно всплывающим окнам. В течение установки будут появляться окна для разрешения установить файлы с определённой лицензией. Мы соглашаемся и продолжаем установку. В итоге на рабочем столе и в меню Пуск появится значок Arduino.

Запускаем программу

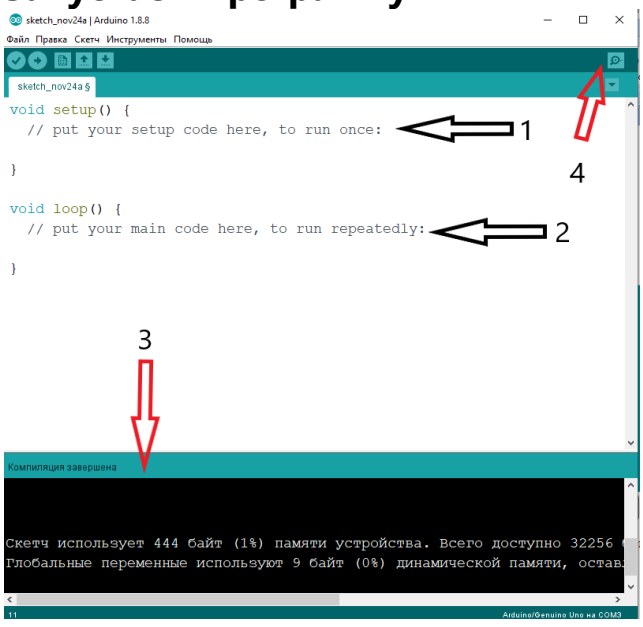


Рис.31 Программа arduino ide

Галочка вверху программы нужна для проверки вашего кода на наличие синтаксических ошибок. Код программ пишется на C++. Значок загружает ваш код на плату Arduino. По умолчанию структура кода выглядит так как на рис. 30 в коде

[1] Совместима с Windows 8.1 и Windows 10.

присутствуют две функции **void setup()** и **void loop ()** (стрелка 1 и 2) . Стрелка 3 указывает на область, где будет показан успех загрузки программы. Стрелка 4 указывает на значок «монитор порта» - вывод или ввод данных по COM порту. Во вкладке Файл - Примеры представлены примеры установленных библиотек

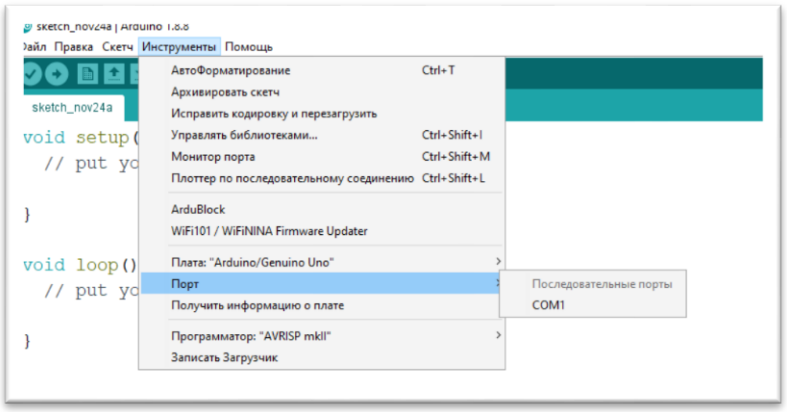


Рис.32 Выбор COM порта arduino

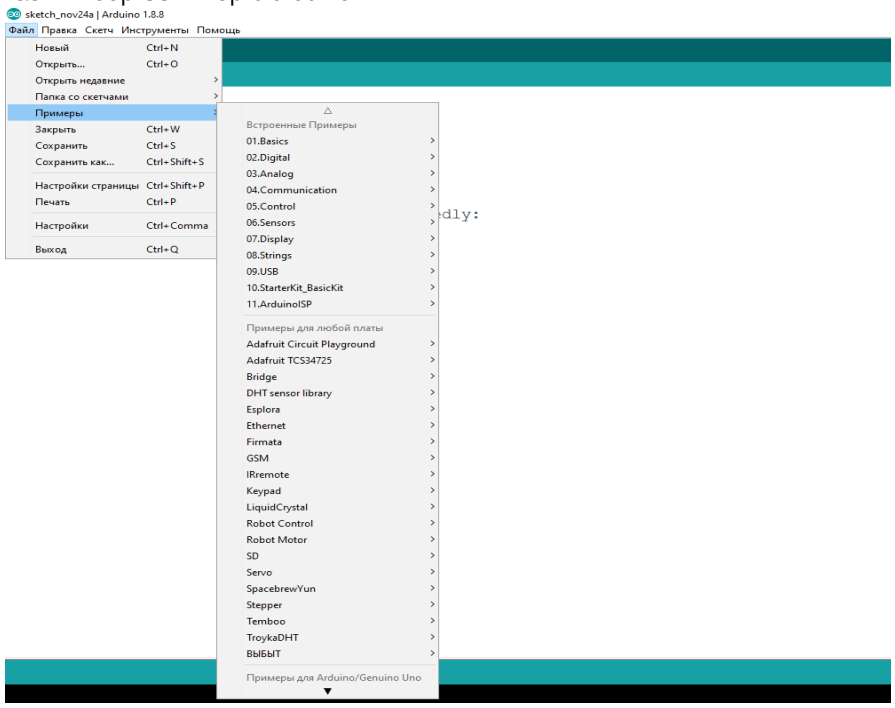


Рис. 33

Во вкладке инструменты, перед тем как загружать программу на плату, необходимо выбрать нужную нам платформу.

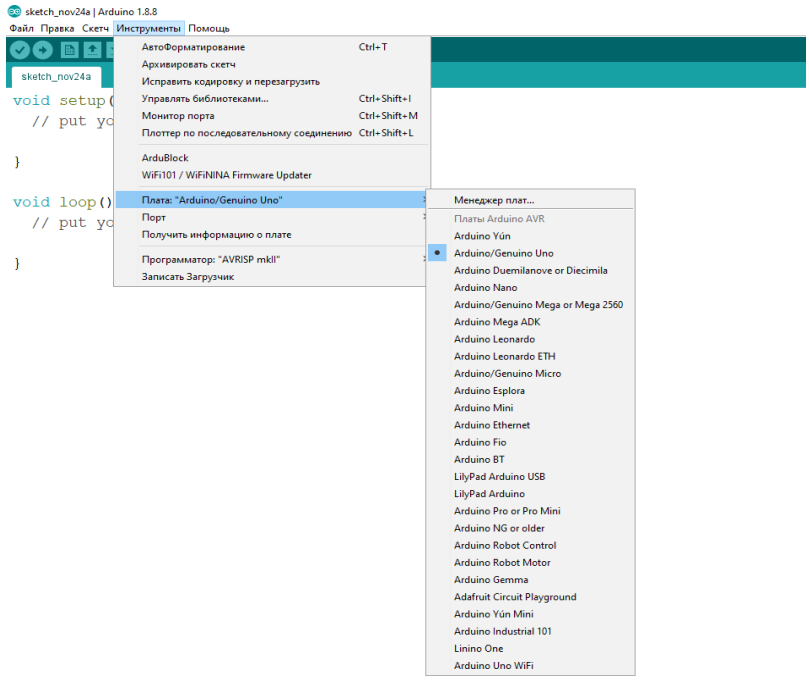


Рис. 34

После выбора платы, необходимо выбрать порт, к которому подключена плата через компьютер.

После проделанных процедур можно загружать программу.

Более подробно и кратко о написании кода можно почерпнуть из справочника https://Arduinoplus.ru/coding-Arduino/

3.2. ArduBlock

Адаптированная среда программирования Arduino ide под scratch подобный язык. Данная вариация использования графико-визуального языка программирования была обусловлена обучением детей от 8 лет.

Более подробная установка дополнения для Arduino ide представлена здесь http://ardublock.ru/index.php?id=rabota-v-ardublock-ustanovka-Arduino-ide-bibliotek-ardublock

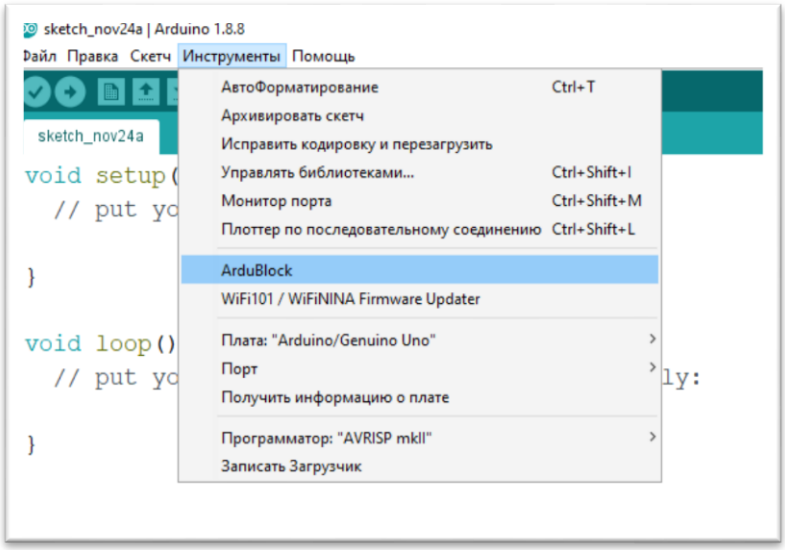


Рис.35 ArduBlock в arduino ide

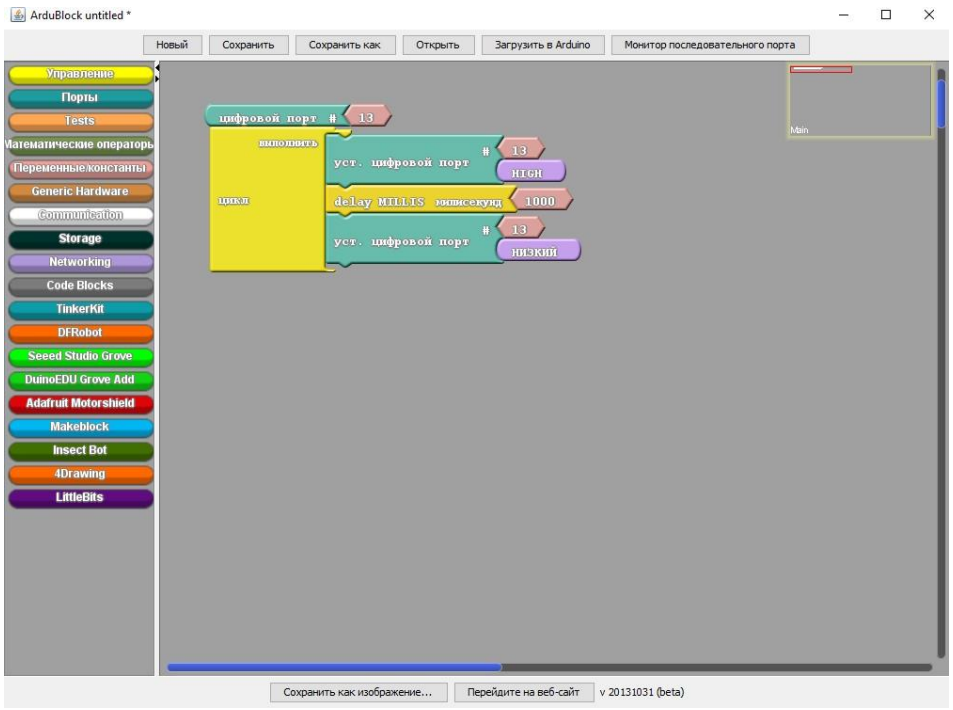


Рис.36 ArduBlock

3.3. MBlock3

Очень хорошая и продвинутая среда для программирования плат Arduino. Здесь можно писать код как на текстовом языке C++ (Arduino ide), так и на scratch подобном языке.

Ссылка для скачивание программного обеспечения

https://www.mblock.cc/en-us/download

Чтобы непосредственно программировать для КЛИК в среде MBlock3, необходимо установить расширение, разработанной компанией CyberTechnic.

Инструкция по установке представлена ниже.

3.3.1. Инструкция по установке mBlock и расширения для НикиРобот

Разделы:

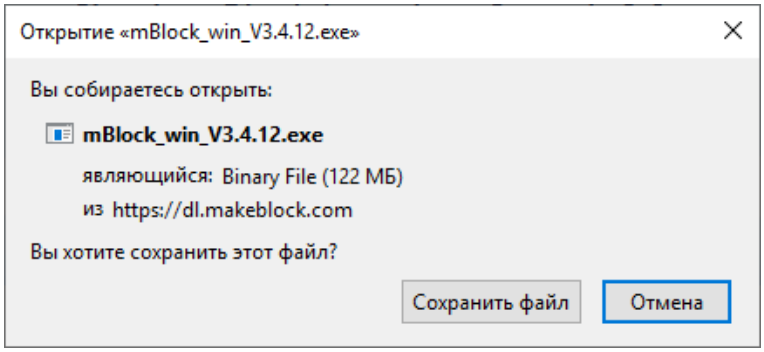
- Установка приложения mBlock3 с помощью инсталлятора компании MakeBlock;
- Изменение языка элементов управления mBlock3 с английского на русский;
- Добавление расширения КиберБот компании ООО БСКомп в приложение mBlock3

Установка приложения mBlock3 с помощью инсталлятора компании MakeBlock

Необходимо скачать последнюю версию инсталлятора приложения mBlock для Windows версии 7 или более поздней по ссылке:

https://www.mblock.cc/en-us/download/

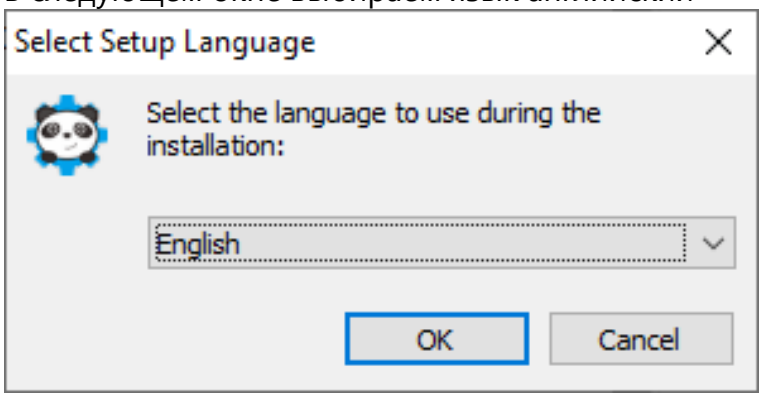
На этой странице найти раздел с версией «mBlock3 for Win 7+» и нажать на кнопку «Download». Появится окно:



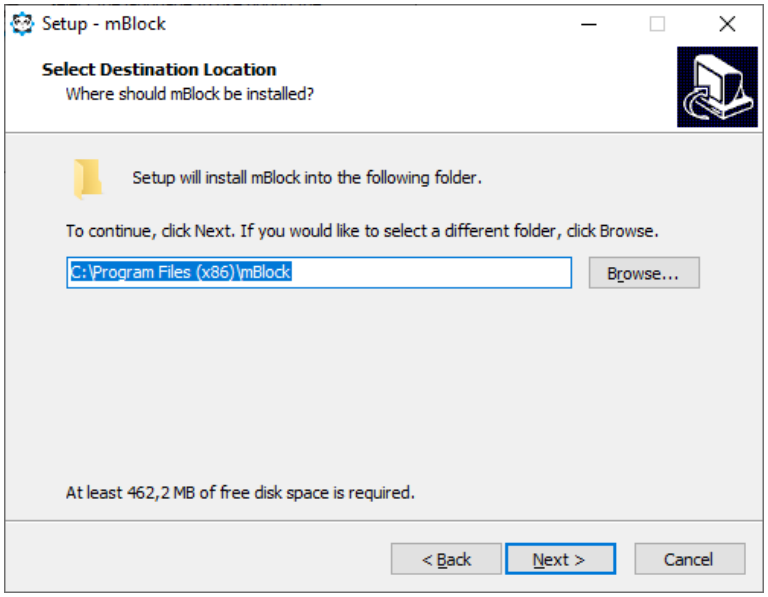
Нажать кнопку “Сохранить файл”, чтобы сохранить файл компьютер в расположение по умолчанию.

Далее, сохранённый файл надо запустить.

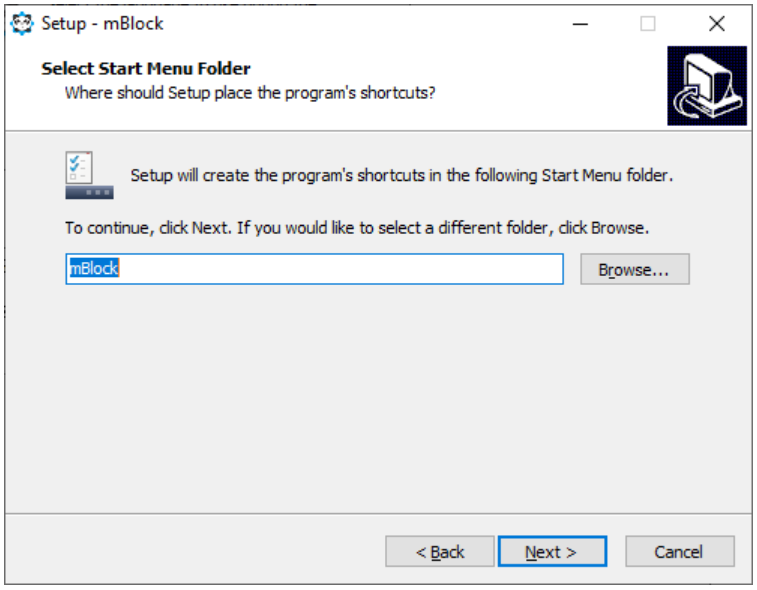
В следующем окне выбираем язык английский



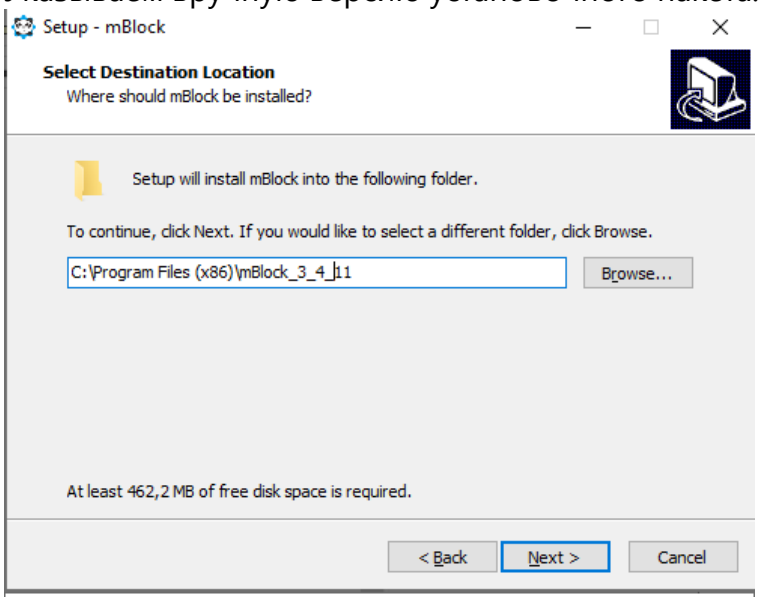
В следующем окне оставить без изменений путь к папке, где будет установлена система:

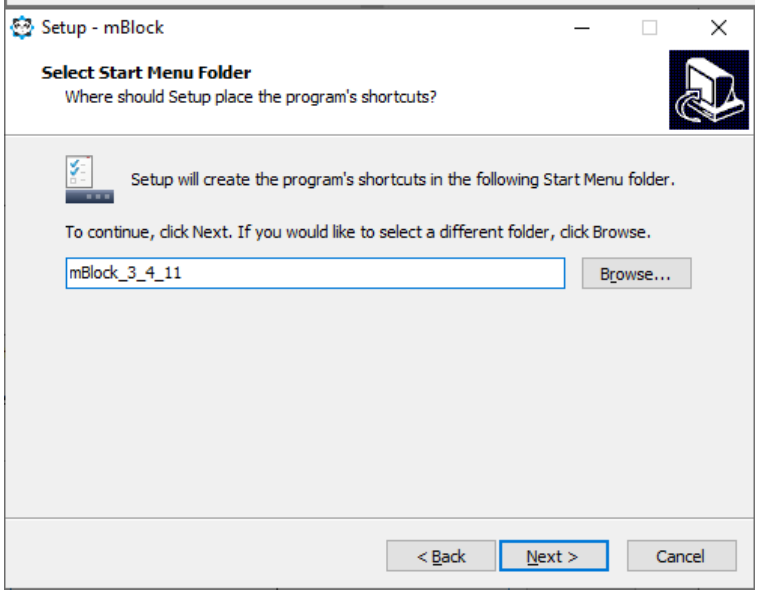


Оставляем без изменений, и нажимаем на кнопку «Next»:

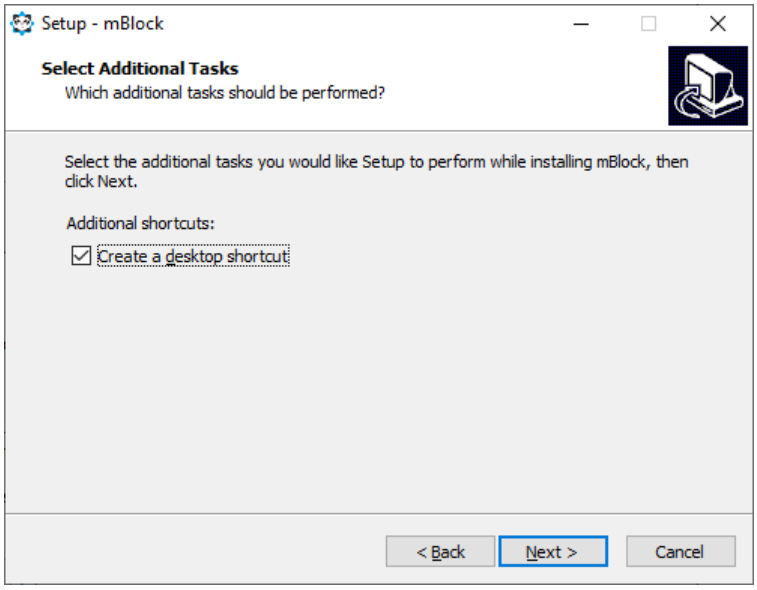


Указываем вручную версию установочного пакета:

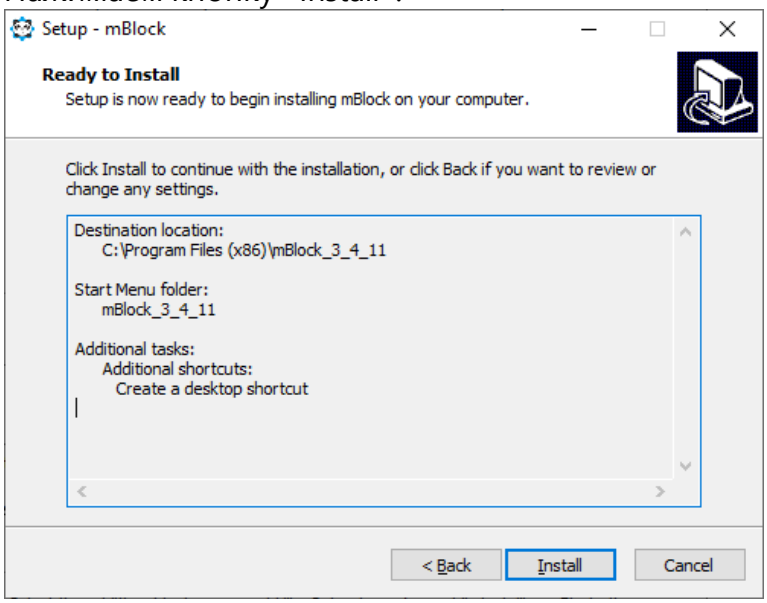




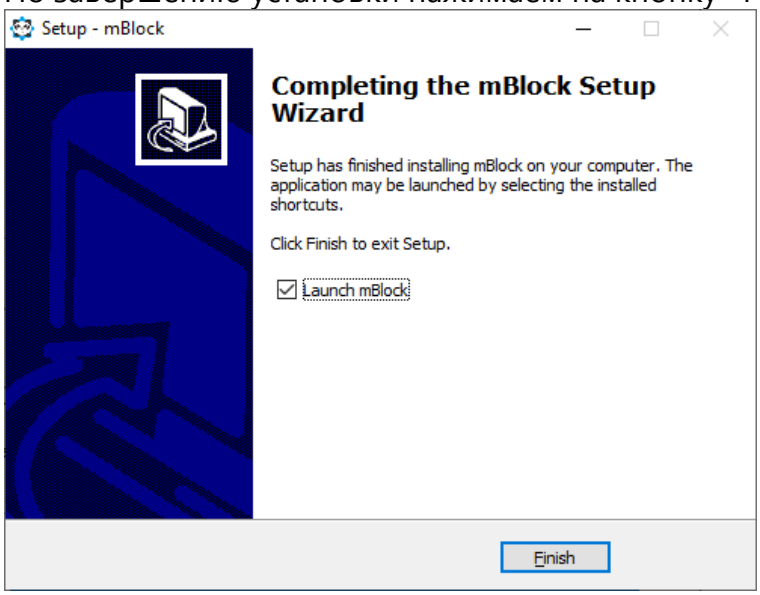
Соглашаемся с параметрами по умолчанию:



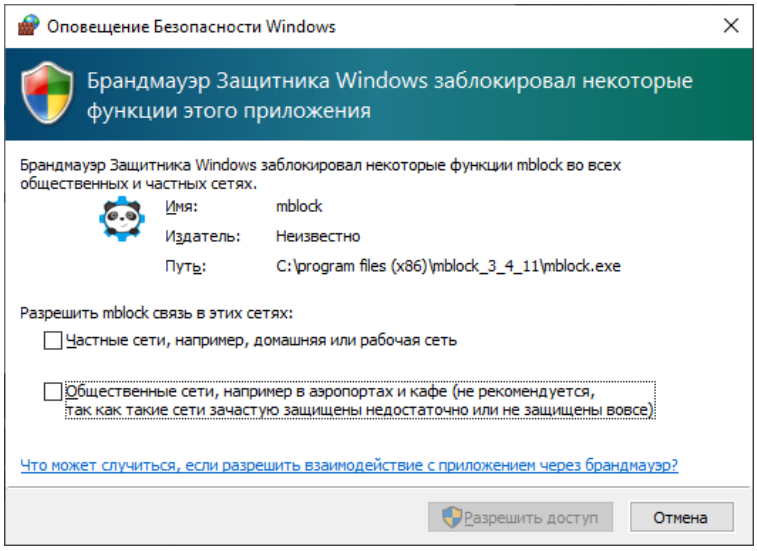
Нажимаем кнопку «Install»:



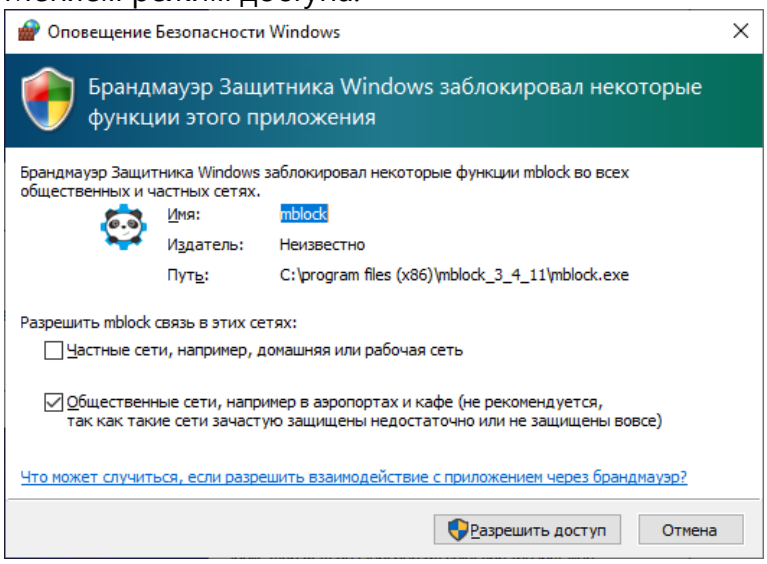
По завершению установки нажимаем на кнопку «Finish»:



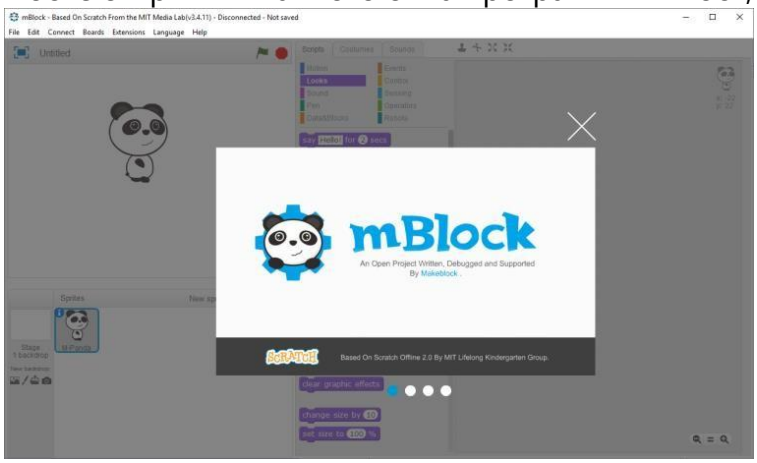
При первом запуске установленной программы может появиться «Оповещение Безопасности Windows»:



Меняем режим доступа:



После открытия главного окна программы mBlock, приложение необходимо закрыть.



На этом установка приложения mBlock3 с помощью инсталлятора компании MakeBlock завершена.

Изменение языка элементов управления mBlock3

Дистрибутив mBlock не содержит полного перевода на русский язык всех своих элементов управления (названий кнопок, пунктов меню и т.п.), для этого необходимо вручную заменить файл локализации «locale.xlsx» в папке : C:\Program Files (x86)\mBlock_3_4_11\locale. В дальнейшем, эта замена будет производиться автоматически.

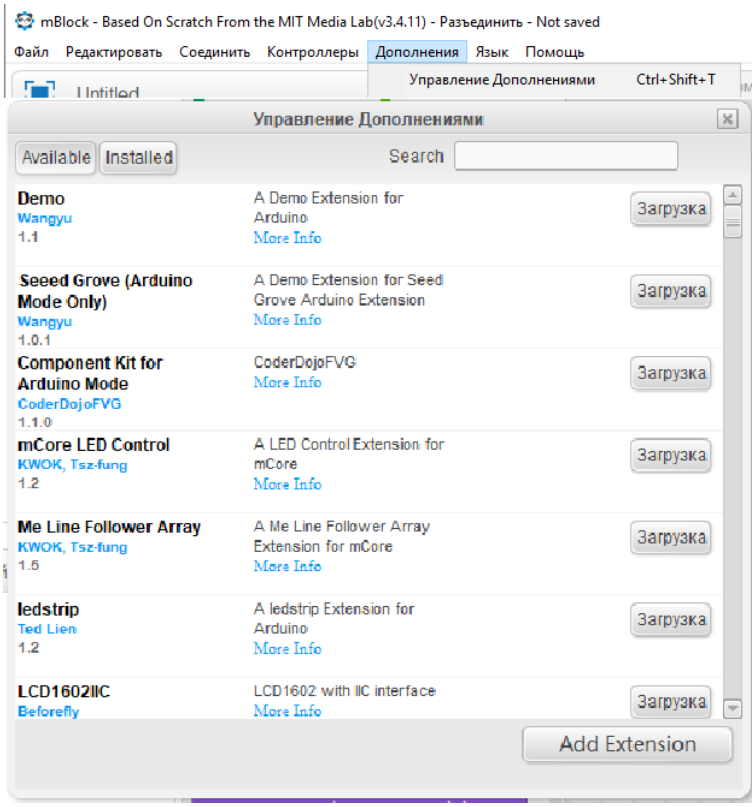
Добавление расширения КиберБот в приложение mBlock3

Программа mBlock имеет возможность добавления сторонних компонентов для управления роботизированными комплексами. Эти компоненты называются «Extensions» и находятся они во вкладке «Дополнение». Эта возможность используется и для добавления функциональности КЛИК.

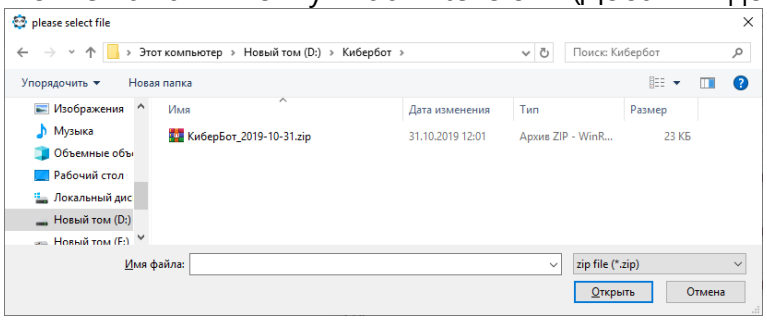
Само дополнение должно быть представлено в ZIP формате, например, КиберБот_2019-10-31.zip.

Для установки дополнения запустить приложение mBlock.

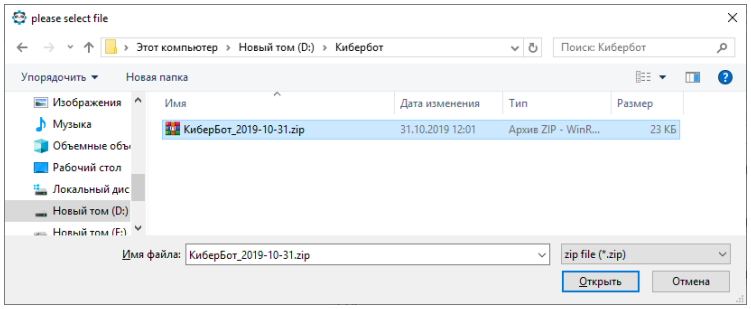
В меню главного окна выбрать раздел “Дополнения” в нём “Управление дополнениями”.



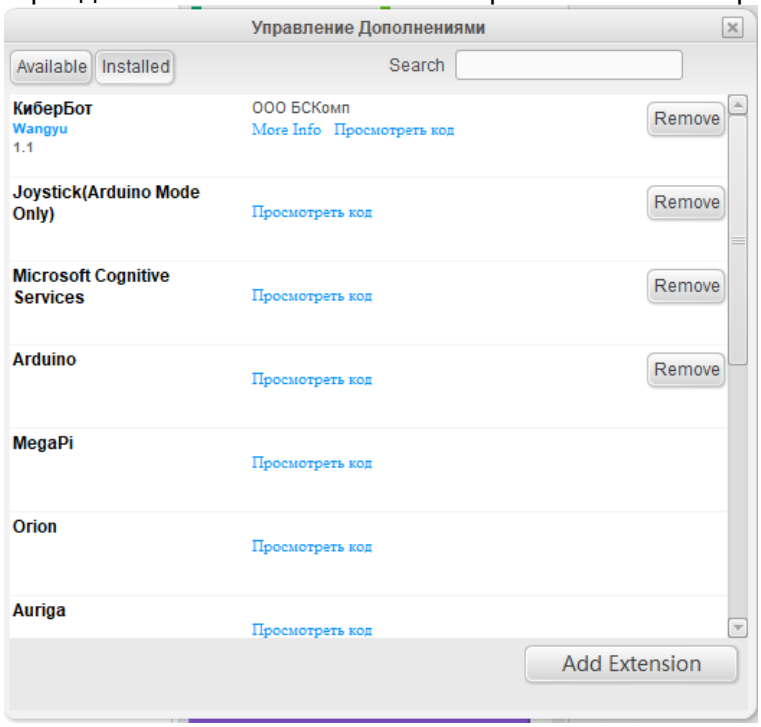
В окне нажать кнопку “Add Extension» (Добавить дополнение)



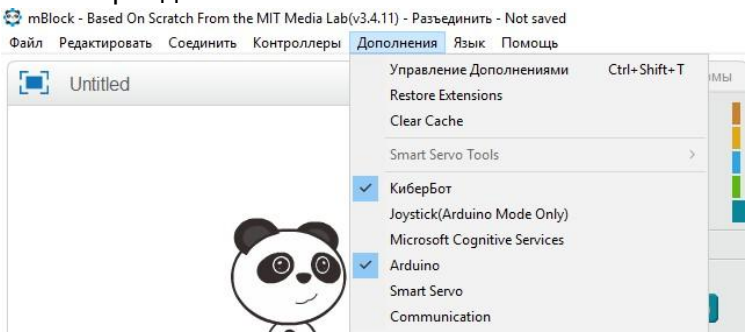
Выбрать ZIP файл и нажать на кнопку «Открыть»

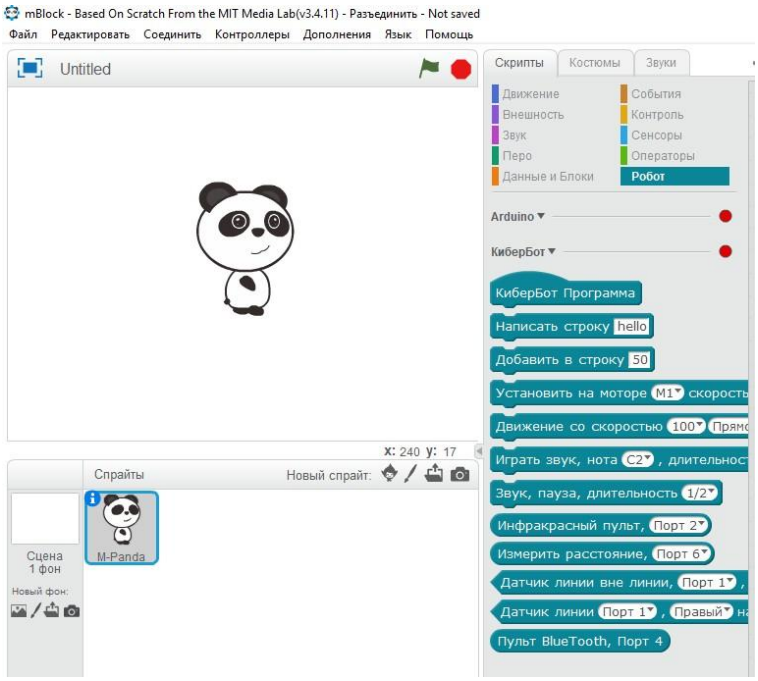


В разделе “Installed” появилось приложение КиберБот



При этом “КиберБот” появится и в меню главного окна mBlock и на вкладке “Скрипты” в разделе “Робот





Установка дополнения КиберБот завершена.

3.3.2. Пример реализации кода в MBlock3 для НикиРобот

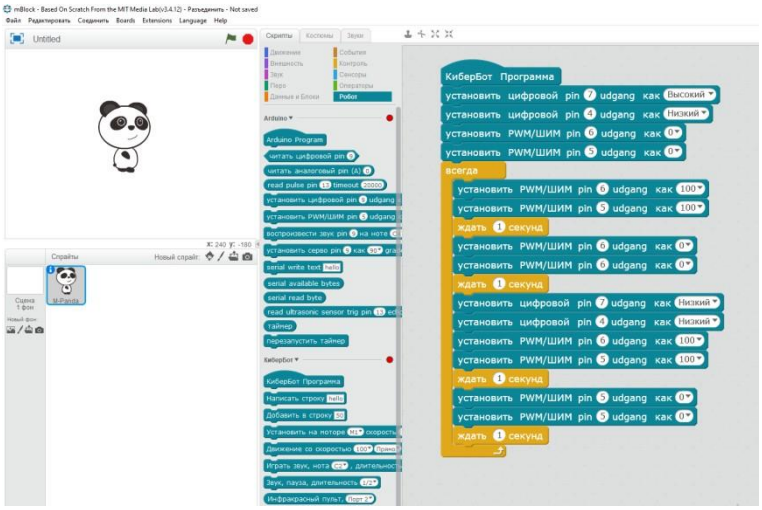
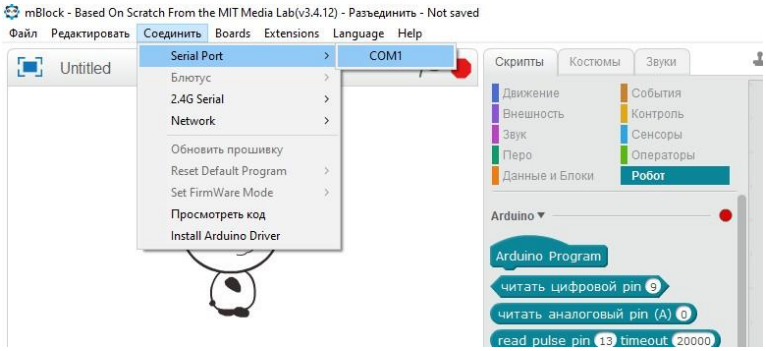


Рис 37 Пример реализации программы в MBlock3

Для загрузки программы в КЛИК, необходимо установить порт подключения, рис. 38.

Рис 38 Установка порта подключения в MBlock3



Убеждаемся, что выбран КиберБот и Arduino.

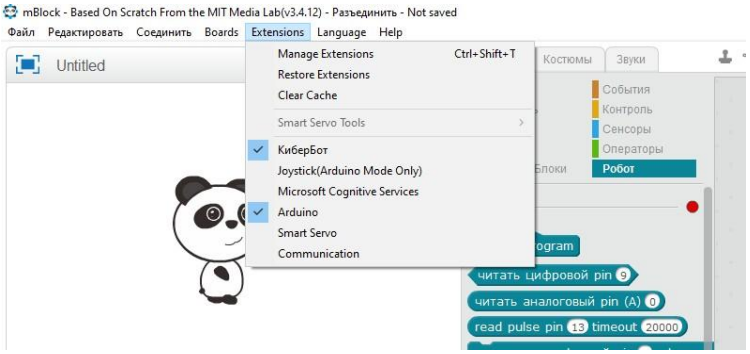


Рис 39 Отмечаем расширение, в котором загружаем программу в MBlock3

Нажимаем двойным щелчком левой кнопкой мыши по ярлыку «Кибербот программа». Открывается окно с переведённым кодом для Arduino ide. Нажимаем левой кнопкой мыши на значок «Upload to Arduino» и обновляем загрузку платы.

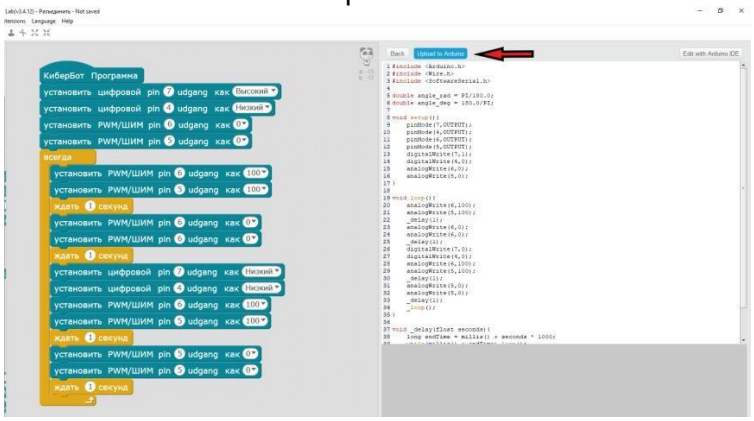


Рис 40 Процедура загрузки программы в MBlock3 на КЛИК

Если всё сделано правильно, то через 1 - 2 мин программа зальётся на плату и ваш робот начнёт выполнять действия согласно прописанному алгоритму.

3.4. MBlock5

Данная среда аналогична среде MBlock3, но с существенными дополнениями и расширениями.

Страница для скачивания MBlock5 https://www.mblock.cc/en-us/download

Рис.41 Среда MBlock5

Как видно из рисунка, есть множество вариаций для установки на разные операционные системы. Кроме локальной программы, есть онлайн редактор MBlock5.

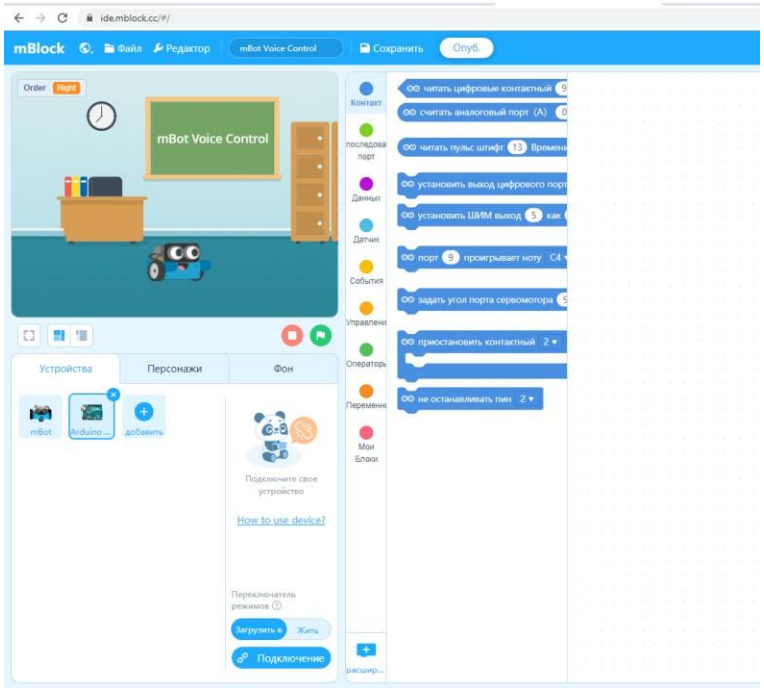


Рис 42 Онлайн редактор MBlock5

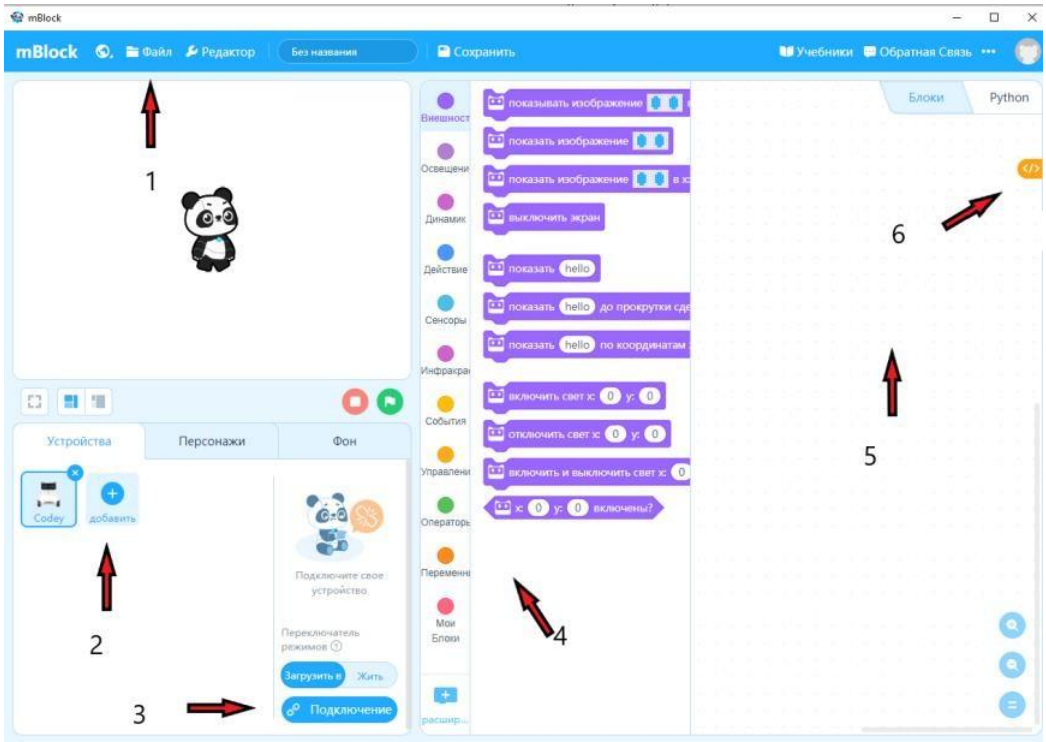


Рис.43 Навигация по MBlock5

Навигация по MBlock5

1. Во вкладке «Файл» создаем, открываем или сохраняем файл;
2. Значок «+» позволяет выбрать устройство, для которого создается программа (библиотека устройств);
3. После выбора устройства, подключите к его компьютеру, выбрав пункт «Подключение», для настройки загрузки программы через COM порт;
4. Набор инструментов (команд) для создания программы;
5. Рабочее поле, где создаётся программа;
6. Альтернативная запись программы на языке C++для Arduino ide.

Для каждого выбранного устройства есть готовый инструментарий команд для создания программ. Кроме этого для устройства есть дополнительные расширения, которые может создавать любой желающий, предварительно, зарегистрировавшись в качестве разработчика.

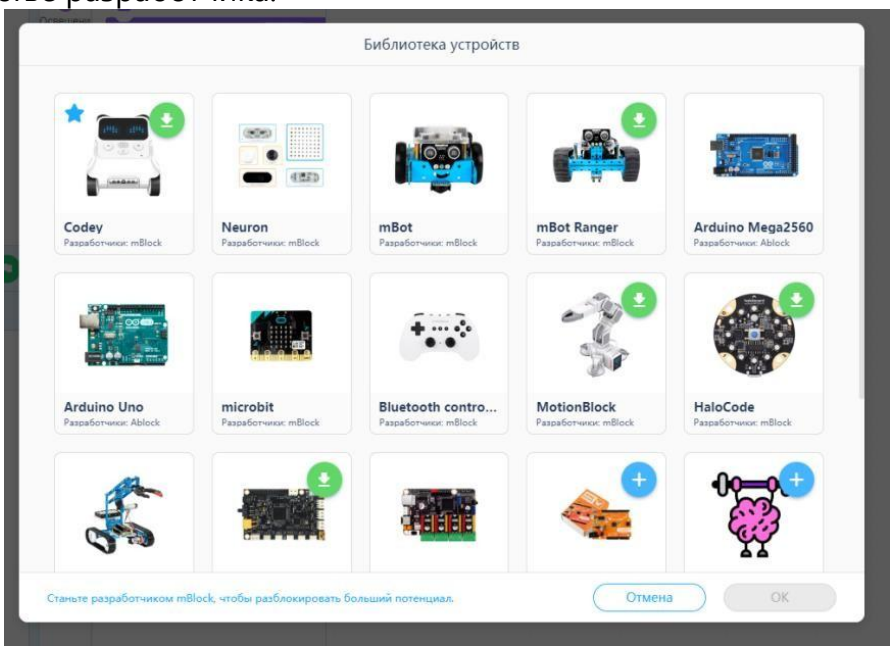


Рис.44 Библиотека устройств на MBlock5

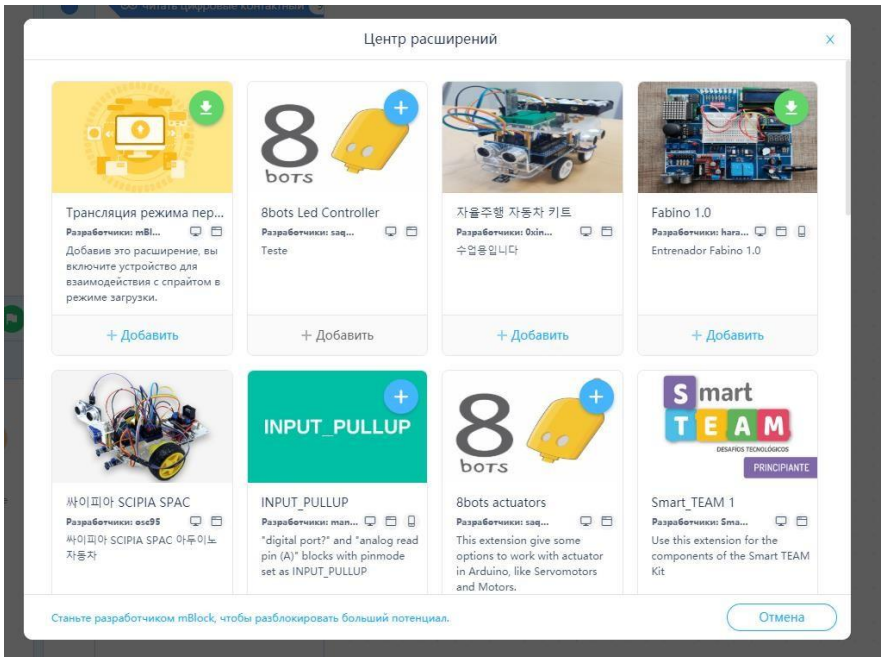


Рис.45 Расширение для устройства на MBlock5

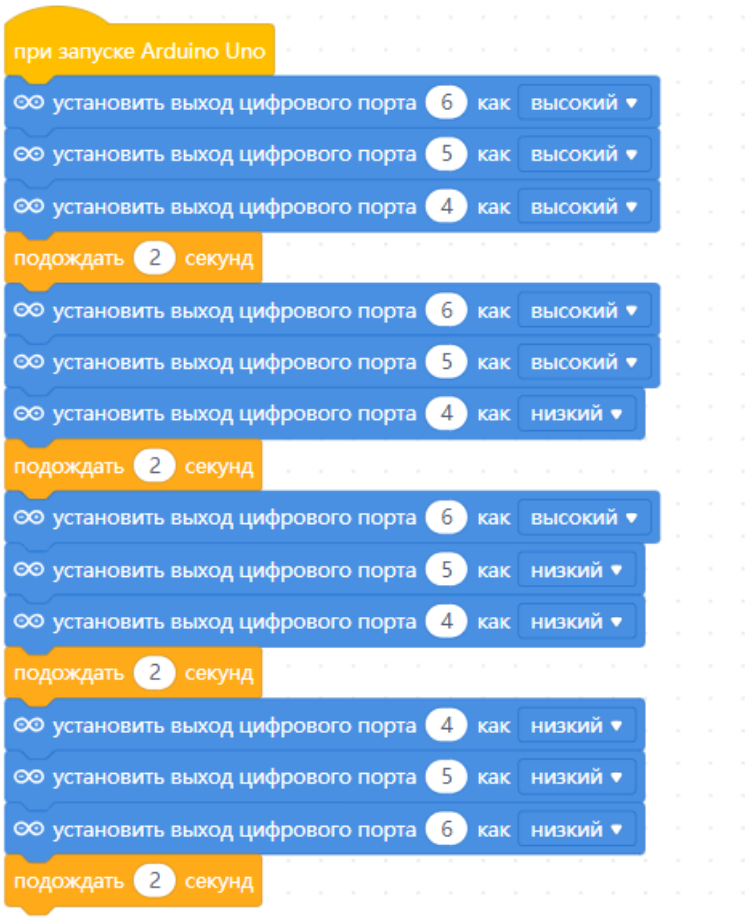


Рис.46 Пример программы на MBlock5

Для загрузки программы подключаемся к устройству по COM порту.

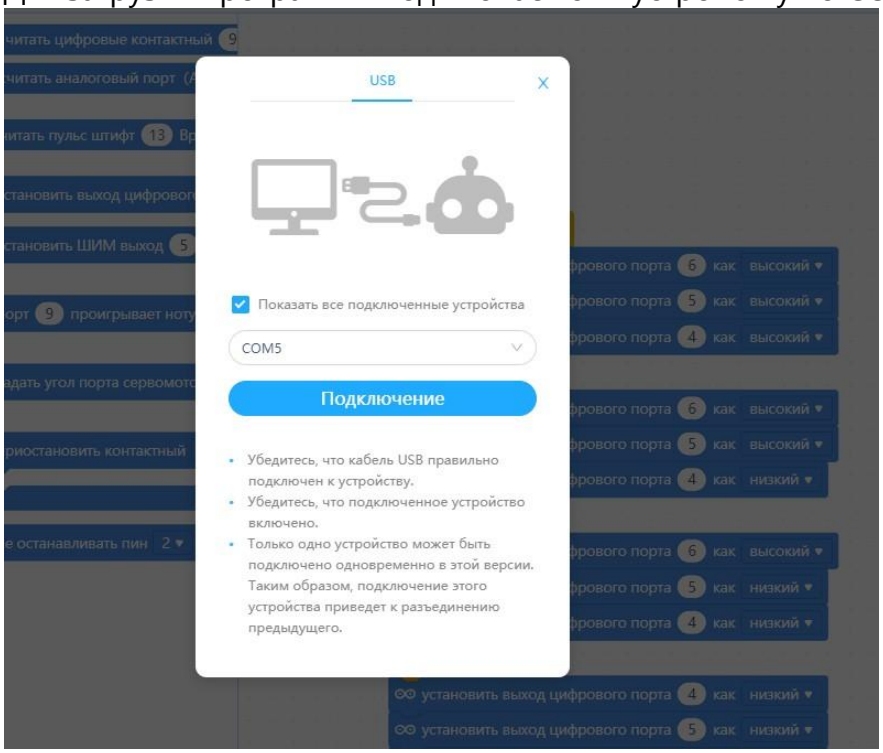


Рис.47 Подключение к устройству на MBlock5

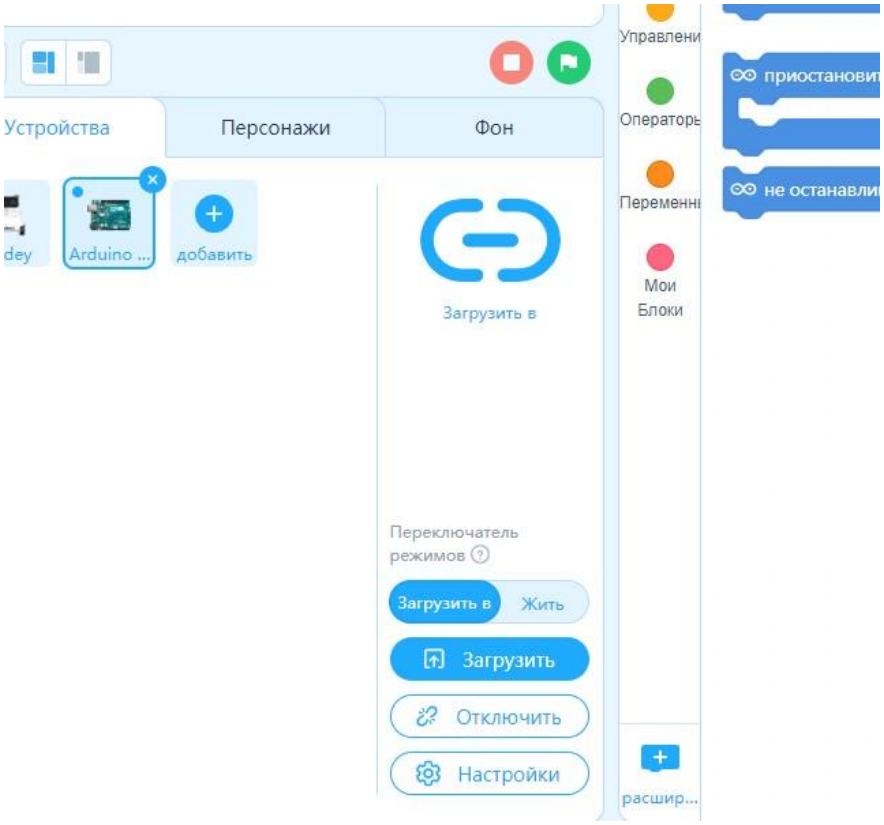


Рис.48 «Загрузить» - загрузка программы на arduino

В данной главе были рассмотрены основные варианты программных сред для создания и загрузки кода в КЛИК. В дальнейшем во всех заданиях будут представлены примеры программ, созданных в Arduino ide и MBlock5.

4. Программирование в среде MBlock5

4.1. Панель инструментов: возможности и функции

Перед тем, как приступить к созданию программ в среде mBlock5, нам необходимо изучить её возможности и познакомиться с инструментарием.

Запускаем программу:

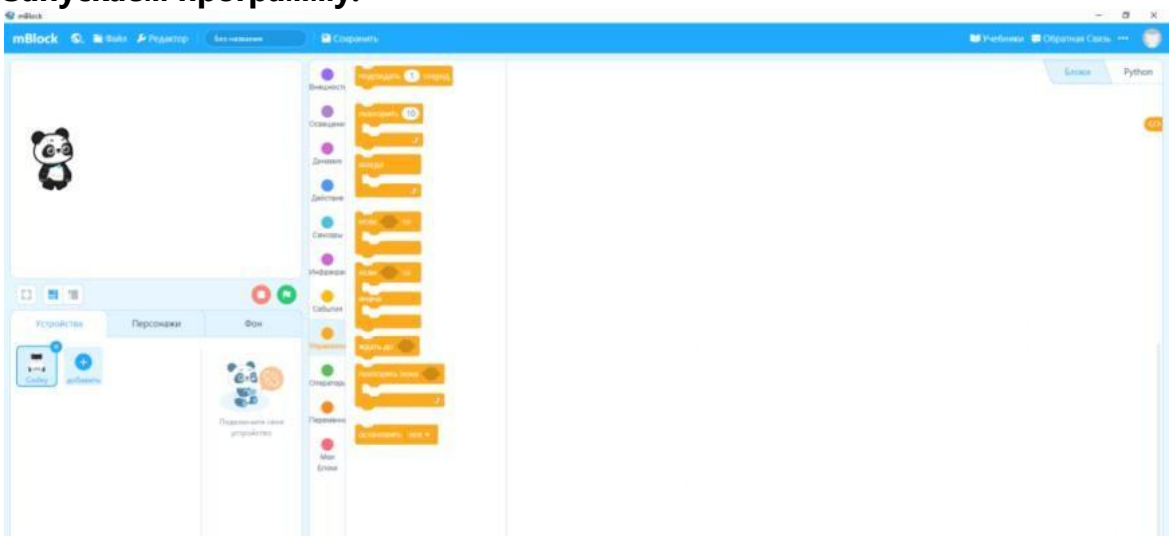


Рис.1

Диалоговое окно разбито на множество секторов. Изучим каждый сектор, рис. 2.

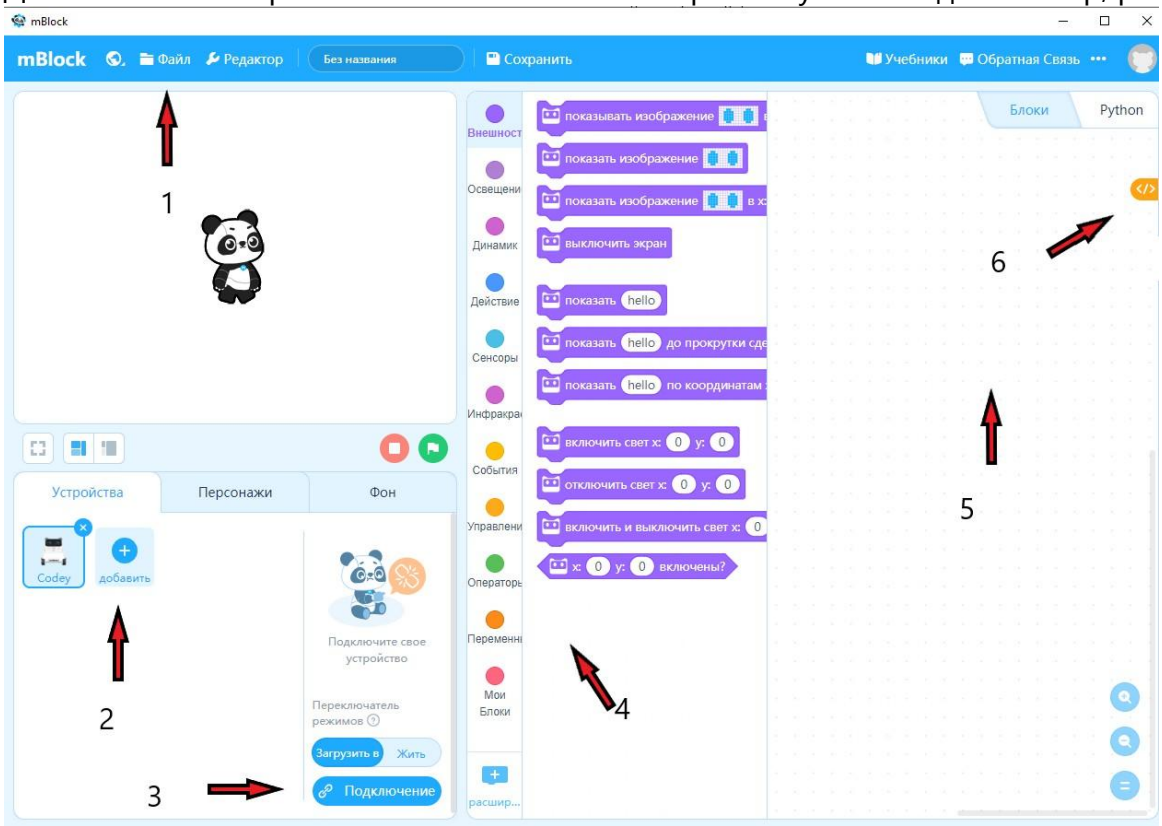


Рис.2

1. Во вкладке «Файл» можно создать, открыть или сохранить файл;
2. Значок «+» позволяет выбрать устройство, для которого создаем программу (библиотека устройств);
3. После выбора устройства, подключите его к компьютеру, выбрав пункт «Подключение», для настройки загрузки программы через COM порт;

4. Набор инструментов (команд) для создания программы;
5. Рабочее поле, где создаётся программа;
6. Альтернативная запись программы на языке C++для Arduino ide.

Открываем вкладку «Файл» и рассмотрим шесть представленных функций, рис.3:

- «Новый» подразумевает создания нового файла (нового окна);
- «Открыть» - открывает, созданный в mBlock5, файл из облака, доступен, если вы зарегистрировались на сайте mBlock5, рис.4;
- «Сохранить как» - сохраняет файл в облаке, если есть учётная запись на сайте, рис.4;
- «Открыть с вашего компьютера» - открывает созданные файлы , которые хранятся на вашем компьютере;
- «Сохранить на вашем компьютере» - сохраняет созданную программу на компьютер;
- «Поделиться с...» - можно поделиться своим проектов в облако «Google classroom», если есть аккаунт на сайте mBlock, рис.4 и рис. 5

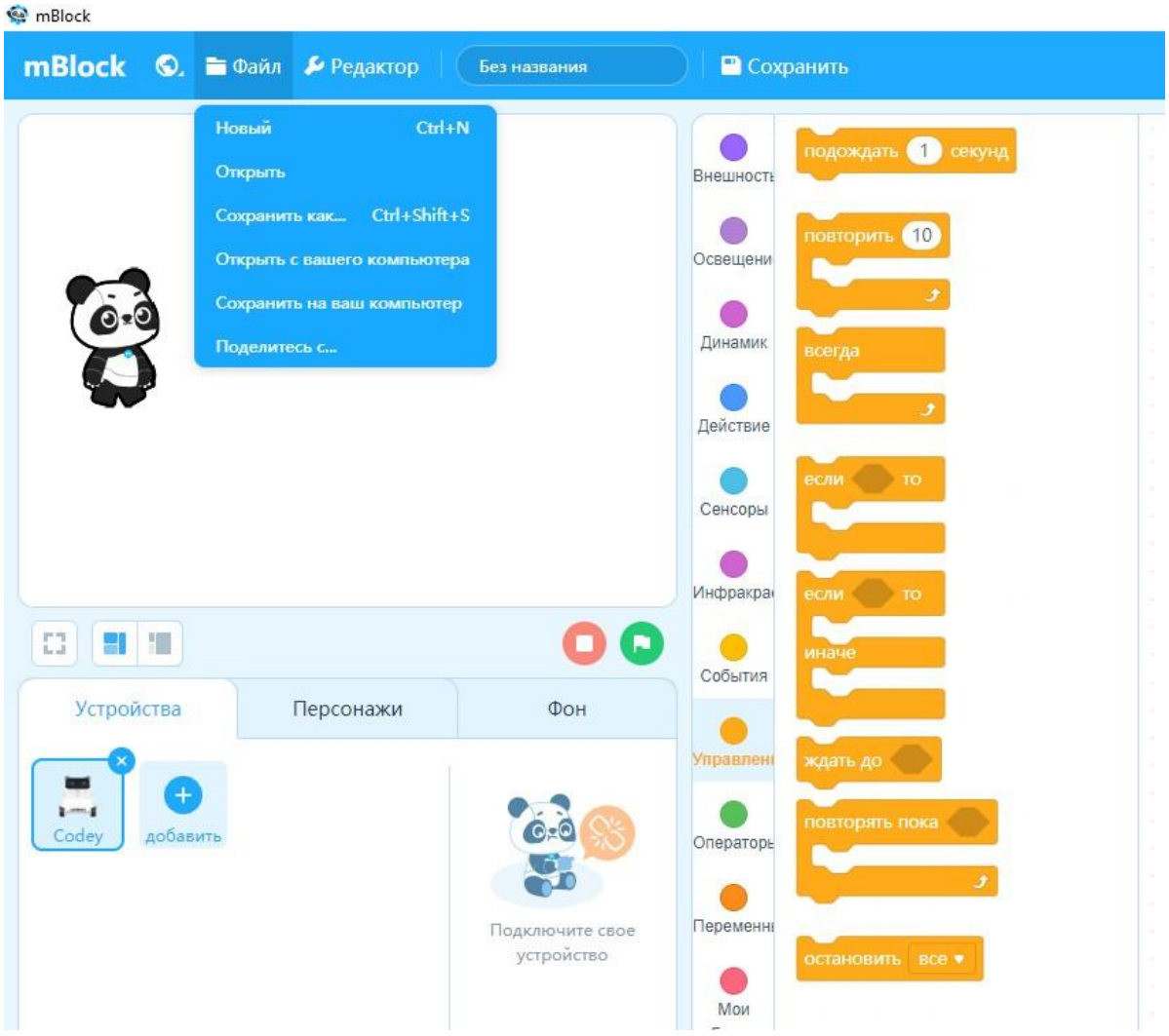


Рис.3

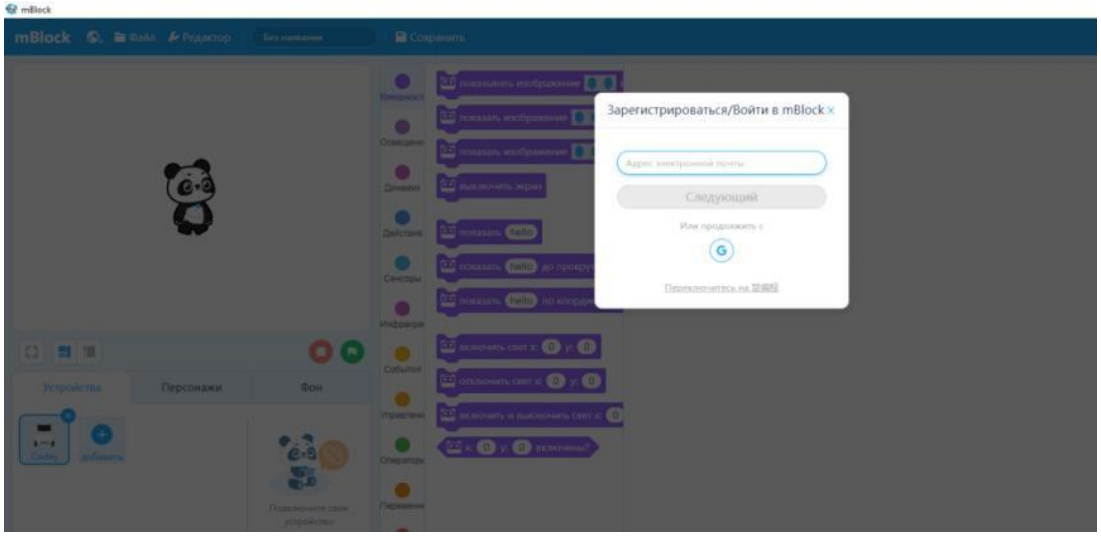


Рис.4

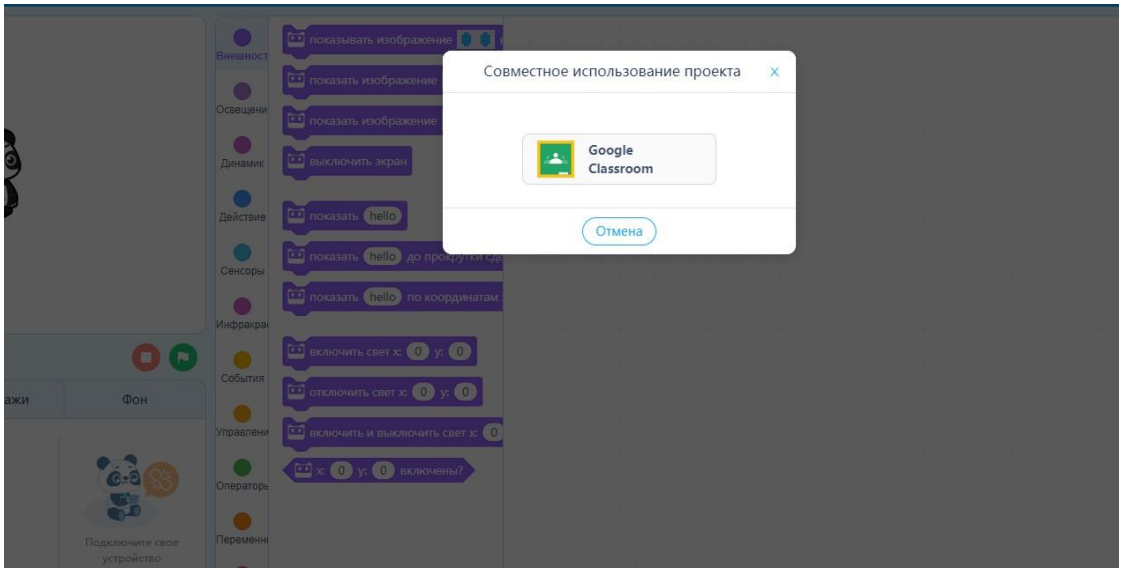


Рис.5

В предыдущей главе была рассмотрена процедура выбора устройства и подключение к нему. Поэтому мы рассмотрим процесс создания программ:

Для этого перейдём во вкладку «Персонажи». В данном разделе можно создавать программу, непосредственно для персонажа, который представлен в левом верхнем углу, т.е. работа как со стандартной программой Scratch, рис.6

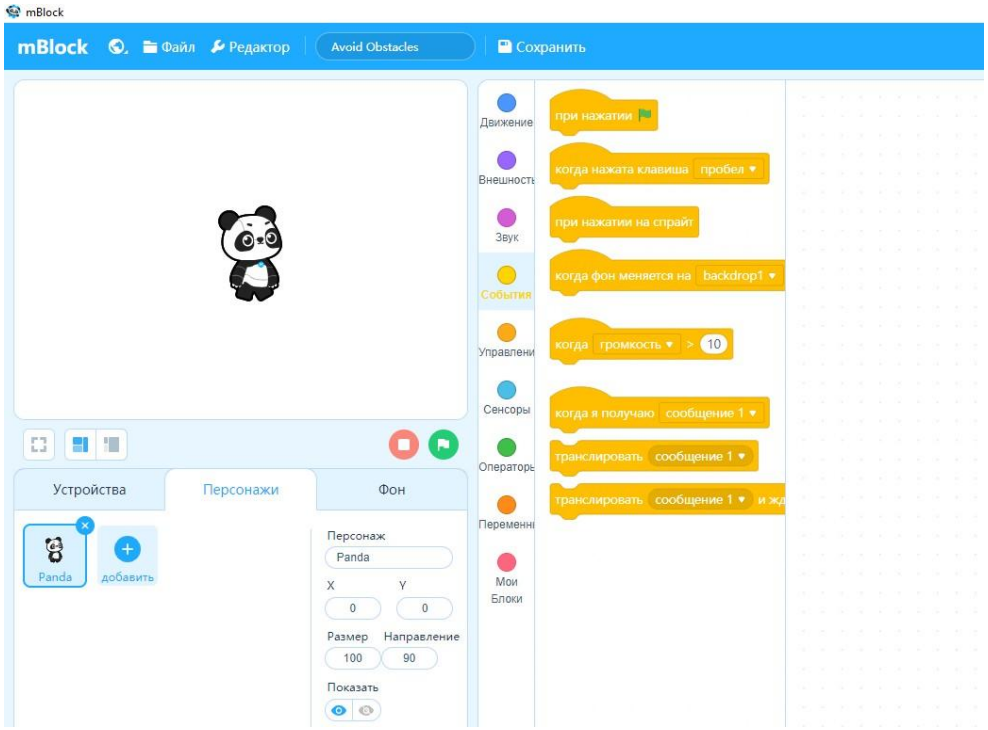


Рис.6

Во вкладке «Персонажи», можно выбрать различных персонажей. Достаточно для этого нажать на значок «+», рис. 7.

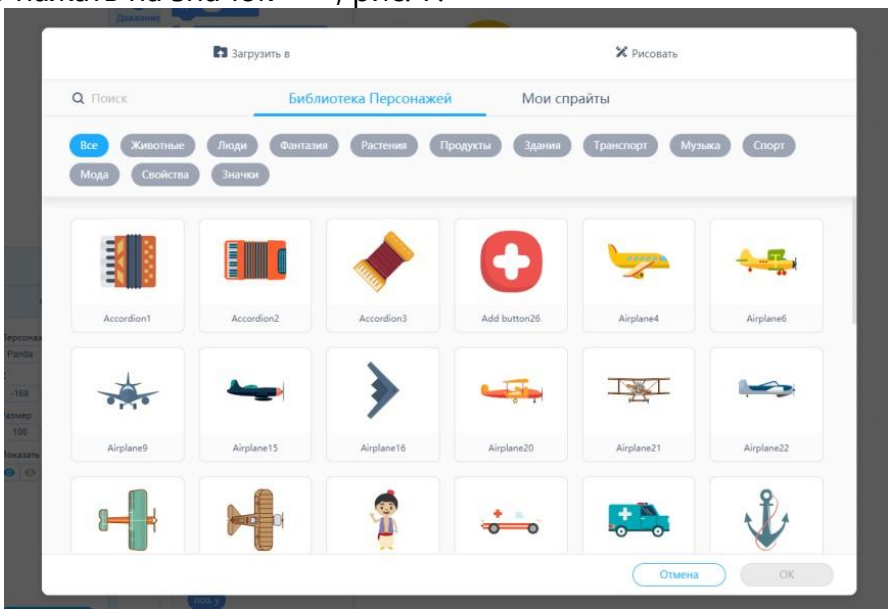


Рис.7

В «Библиотеке персонажей» присутствуют множество различных элементов, при этом они разбиты по категориям, для облегчения поиска. Добавляются персонажи простым нажатием левой кнопки мыши по выбранному значку и нажатием кнопки «ОК».Выбранные персонажи можно редактировать. Если нажать правой кнопки мыши по персонажу, то появится контекстное меню: копировать, экспортировать и удалить (рис.8).

Касательно «копировать и удалить» всё просто – эти функции копируют и вставляют или удаляют персонажа. Функция «Экспортировать» - сохраняет файл на вашем компьютере в формате **.sprite3**.

Кроме этого, персонажа можно перемещать по области экрана, как с использованием левой кнопки мыши, так и непосредственно указывая его координаты X и Y. Центром

координатной сетки, с координатами (0, 0) будет центр экрана, в котором располагается персонаж.

Персонажа можно уменьшать или увеличивать, меняя значение параметра «Размер» (стрелка 1). Также персонажа можно вращать, меняя значение градусов параметра «Направление».

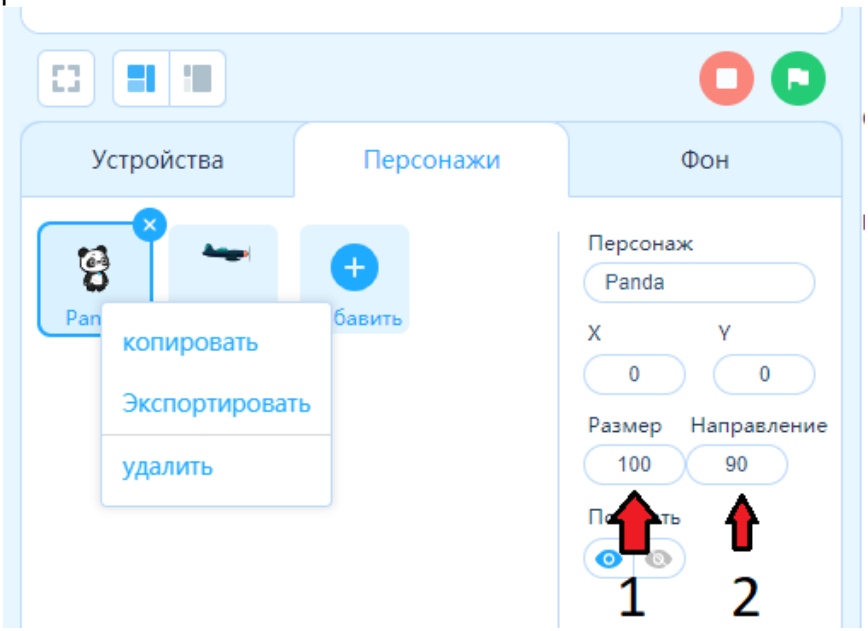


Рис.8

Если нет персонажа из «Библиотеки персонажей», то можно нарисовать своего. Для этого необходимо перейти во вкладку «+» и выбрать вкладку «Рисовать», рис. 9.

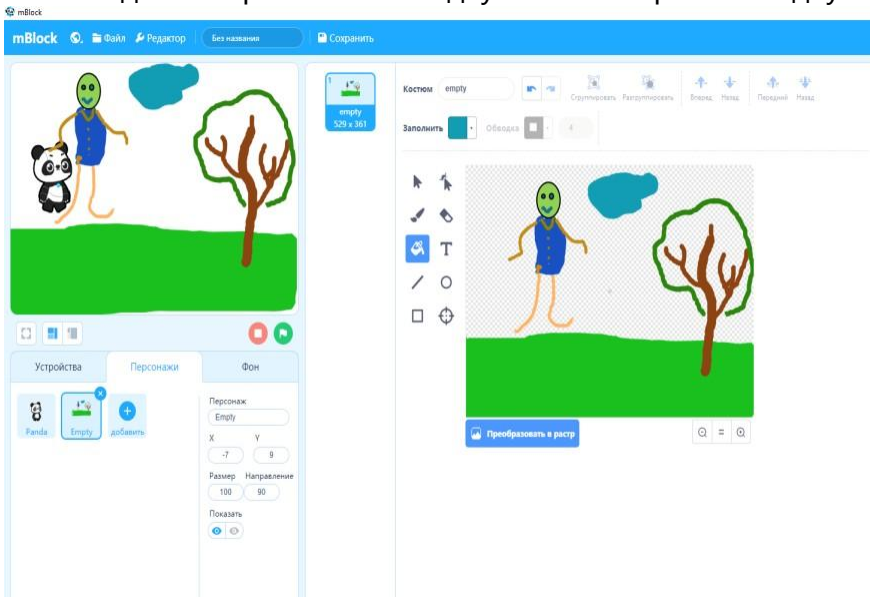


Рис.9

Рисуя персонажа, можно сразу видеть конечную его реализацию во вкладке «Персонажи».

Кроме этого к каждому персонажу можно добавить звук. Для этого необходимо выбрать нужные звуки, которые можно редактировать, рис.10. Нажимаем на вкладку «Звуки».

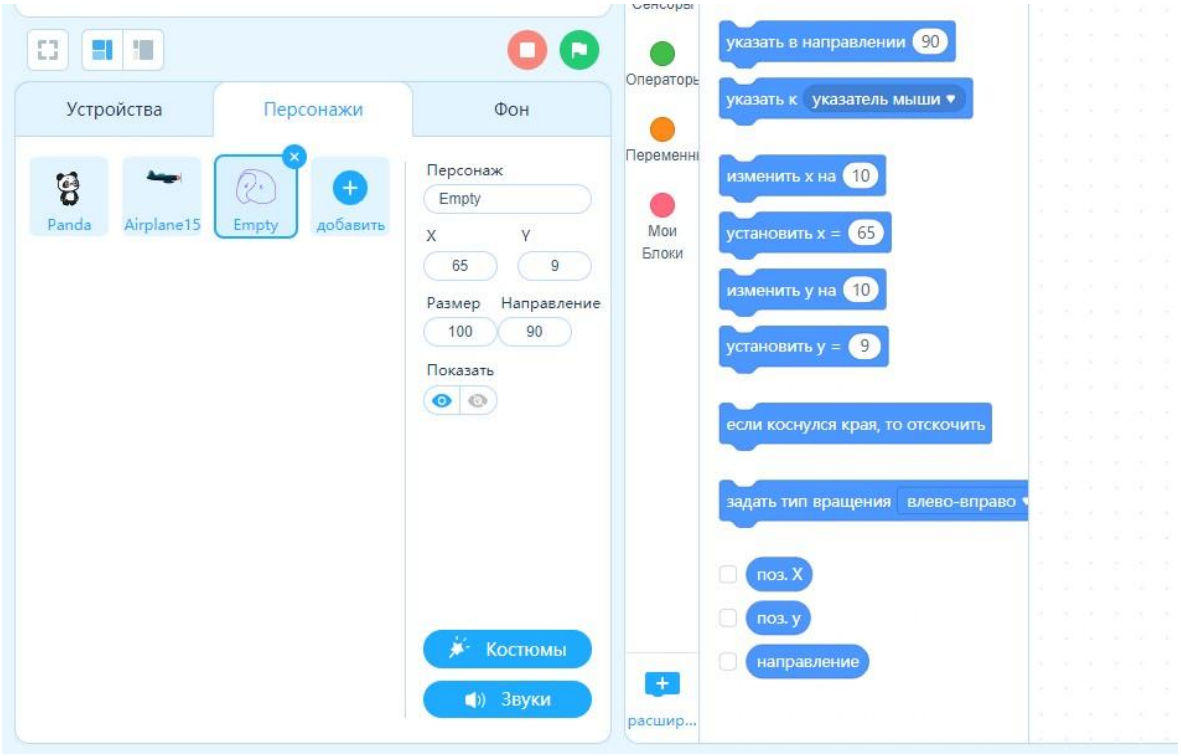


Рис. 10

В окне выбор звука, можно добавить звук, нажав «+» и редактировать звук. Инструменты редактирования находятся справа, рис.11и рис.12.

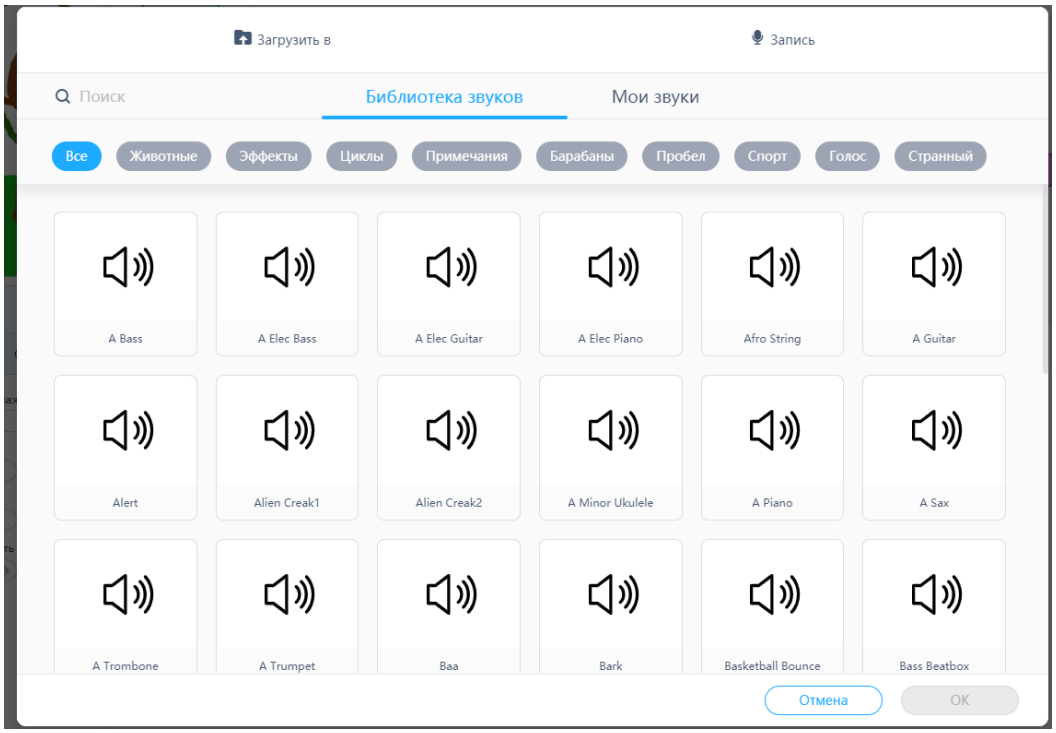


Рис. 11

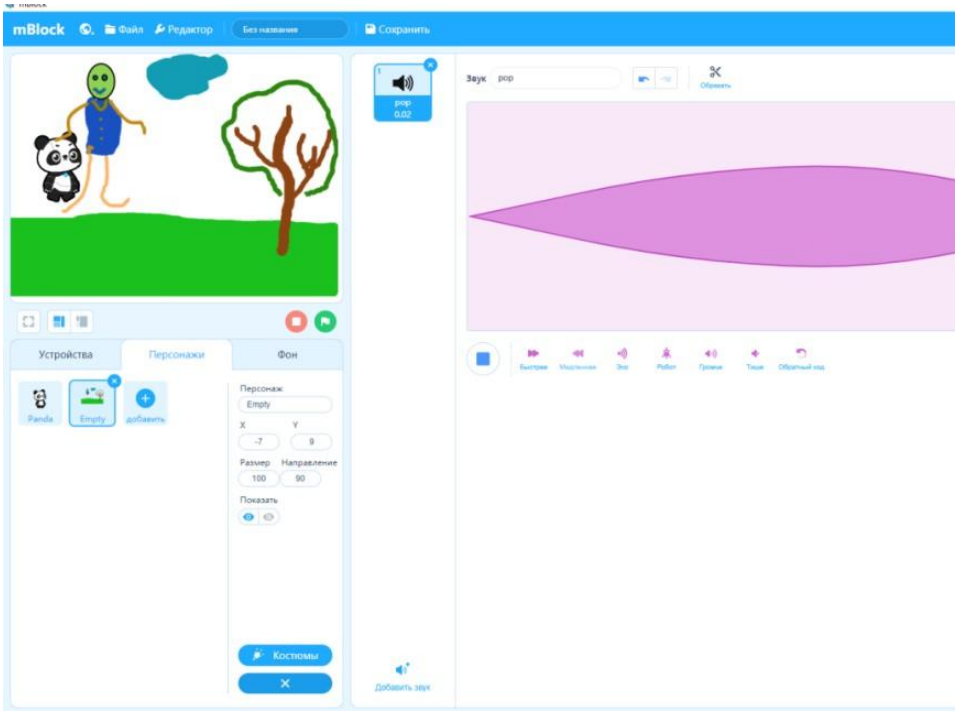


Рис. 12

Ниже представлен пример использование звука в программе, рис. 13.

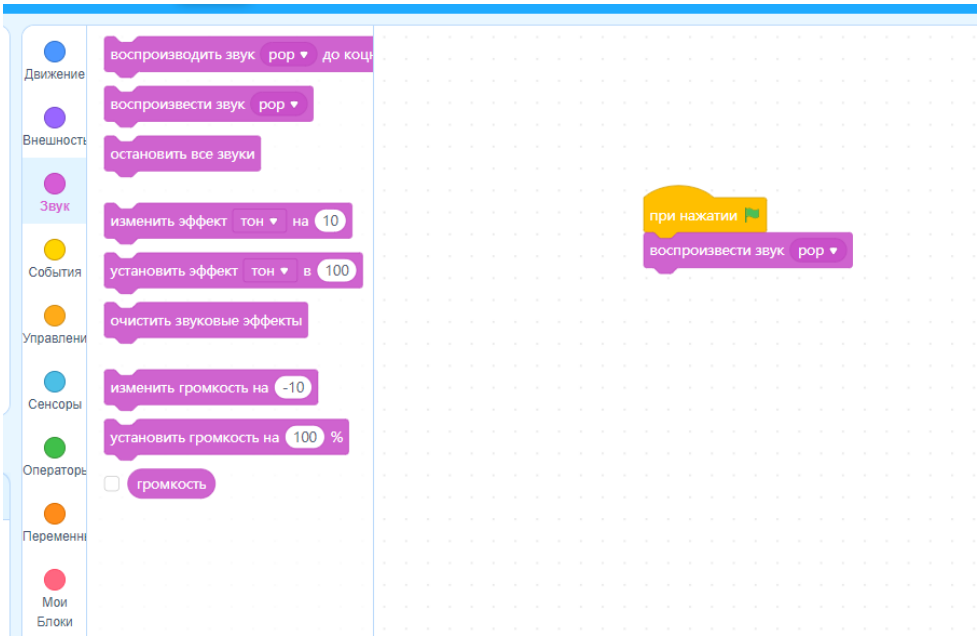


Рис. 13

При нажатии зелёного флажка в окне с персонажем, будет издаваться звук, который мы выбрали.

Интересна вкладка «Учебники» в правом верхнем углу, рис.14.

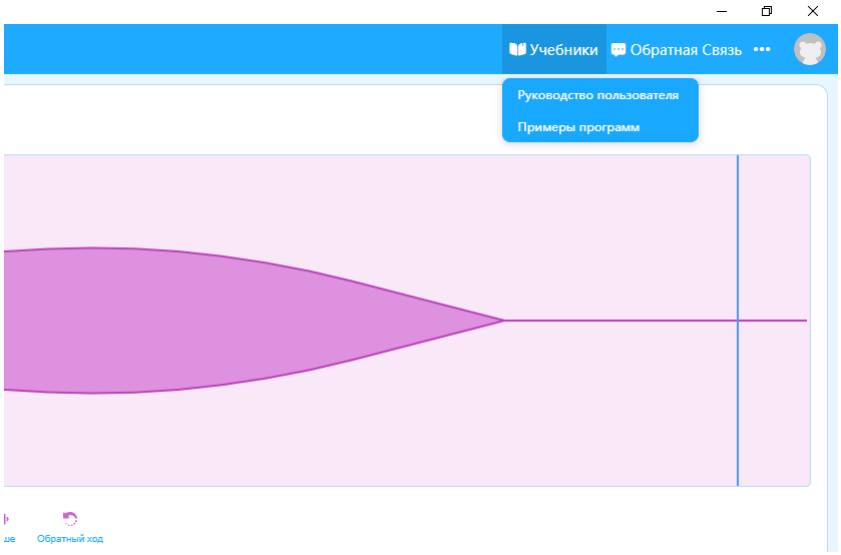


Рис. 14

Вкладка «Руководство пользователя» отправит нас к документации программы mBlock5, рис.15.

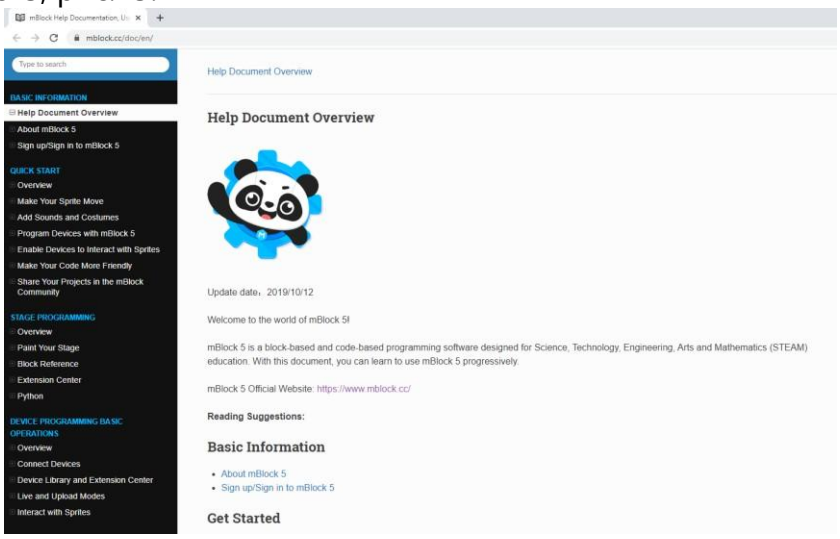


Рис. 15

Вкладка «Примеры программ» представлены различные интересные примеры программы, которые реализованы как для персонажей, так и для робототехнических наборов, рис.16 - 20.

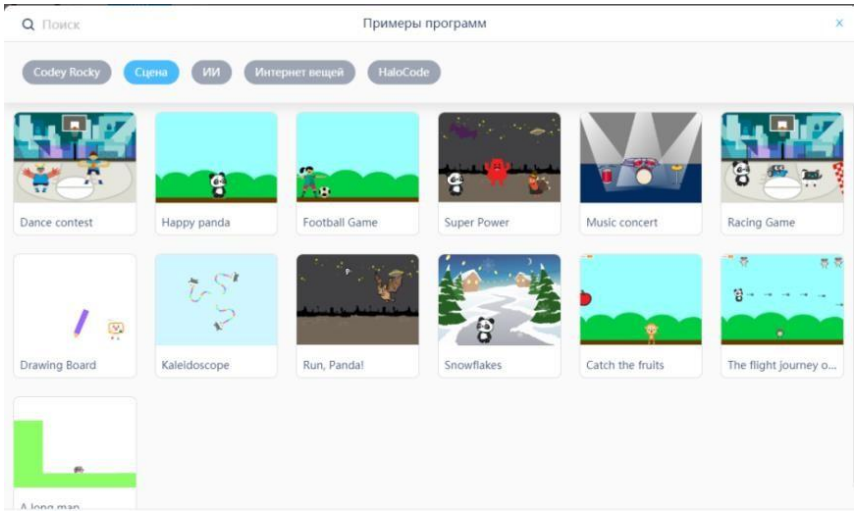


Рис. 16

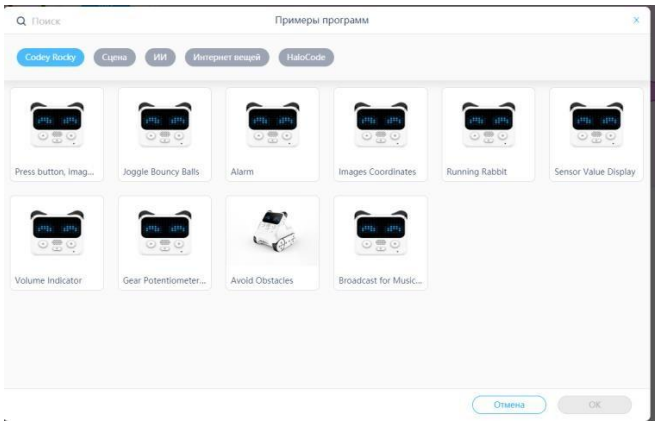


Рис. 17

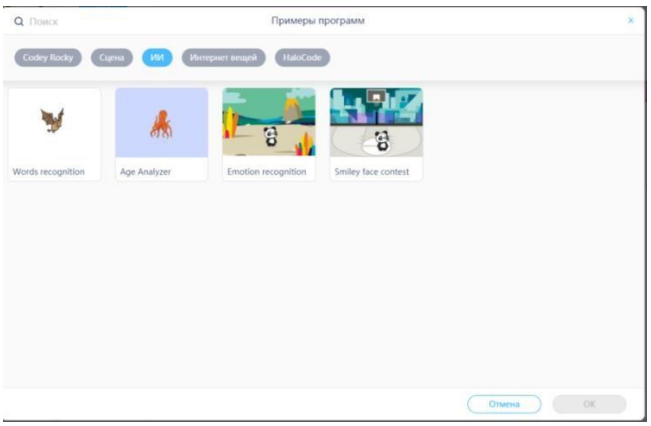


Рис. 18

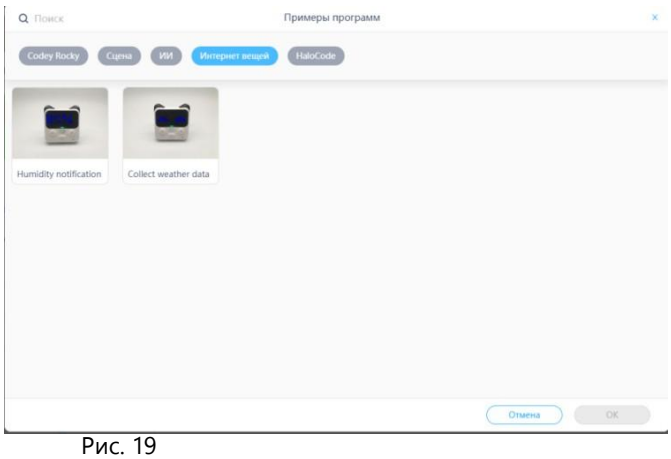


Рис. 19

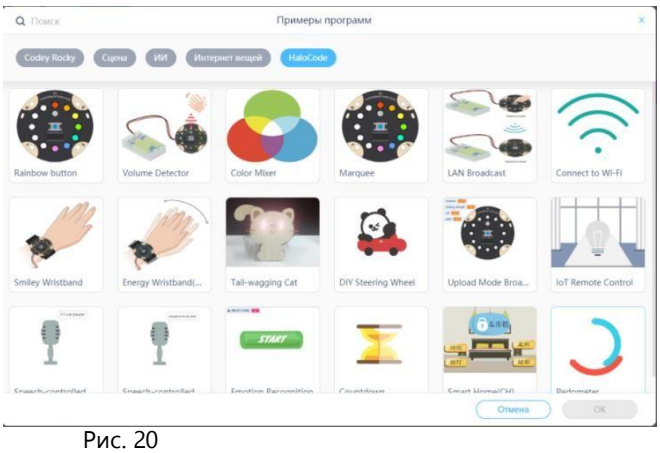


Рис. 20

Рекомендую, использовать некоторые примеры, особенно для персонажей, чтобы понять некоторые аспекты программирования.

Кроме вкладок «Устройства» и «Персонажи», есть вкладка «Фон», где можно настроить фон, на котором будет проводить свои действия персонаж. Фон можно выбрать из «Библиотеки фонов» или же нарисовать свой, рис. 21 и рис. 22.

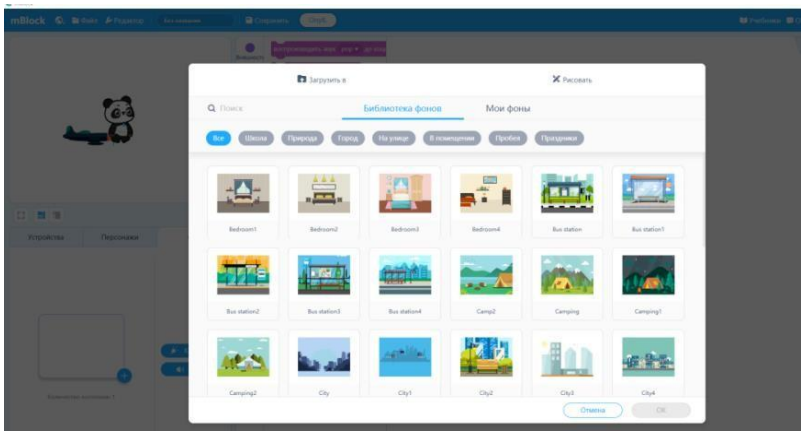


Рис. 21

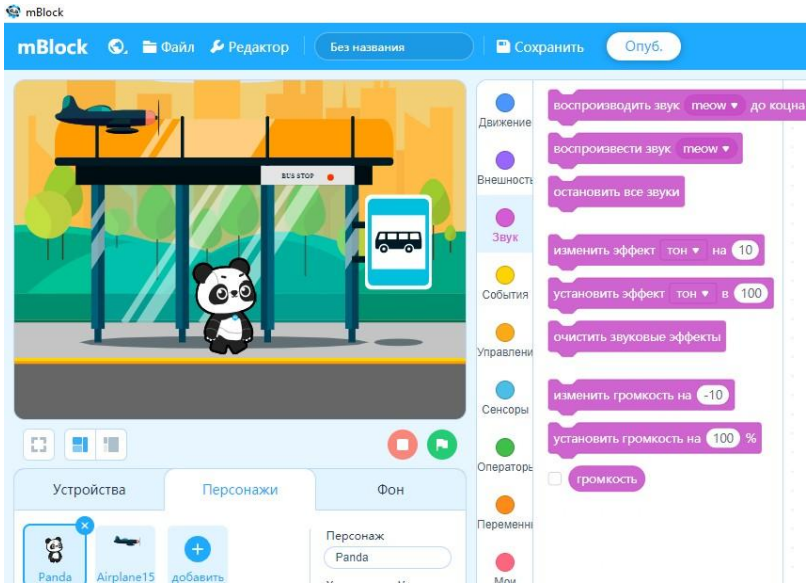


Рис. 22

Основные моменты работы с устройствами и персонажами мы рассмотрели, теперь переходим к процессу создания программ.

Задания:

- Выберите персонаж «самолёт», разместите его в левом крайнем углу экрана и уменьшите до размера 10;
- Нарисуйте своего персонажа: человечек, животное, бабочка, машина, фантастическое существо;
- Наложите фон, где персонаж «панда» на природе;
- Нарисуйте фон для персонажа «кораблик»;
- Сохраните ваш файл на компьютер, а затем откройте его.

4.2. Линейный алгоритм

Для создания программы, необходимо разобраться в основных алгоритмах. Алгоритм – это набор простых действий, который универсален для многих схожих процессов.

Первым алгоритм, который мы должны изучить – это линейный алгоритм.

Линейный алгоритм – это алгоритм из последовательных действий, идущих один за другим.

Пример: Создадим программу, которая заставит персонажа сдвинуться на один шаг «вперёд», рис.23.

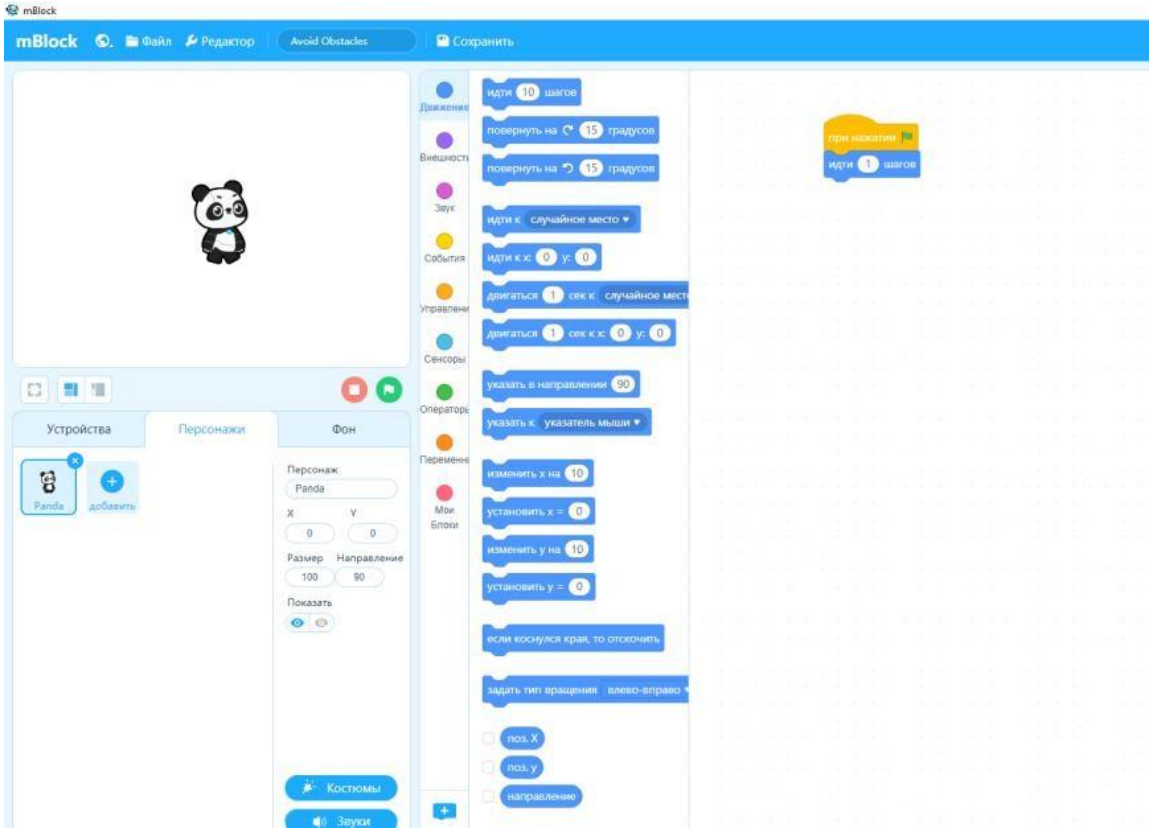


Рис. 23

Всё что связано с перемещением находится во вкладке «Движение».

Представленная ниже программа рис.24, содержит набор действий для персонажа, которые не только перемещают его вперёд, но и поворачивают на 90^0, а каждый шаг имеет задержку в 1 секунду. Тем самым, благодаря данной программе, мы можем наблюдать движение персонажа с течением времени.

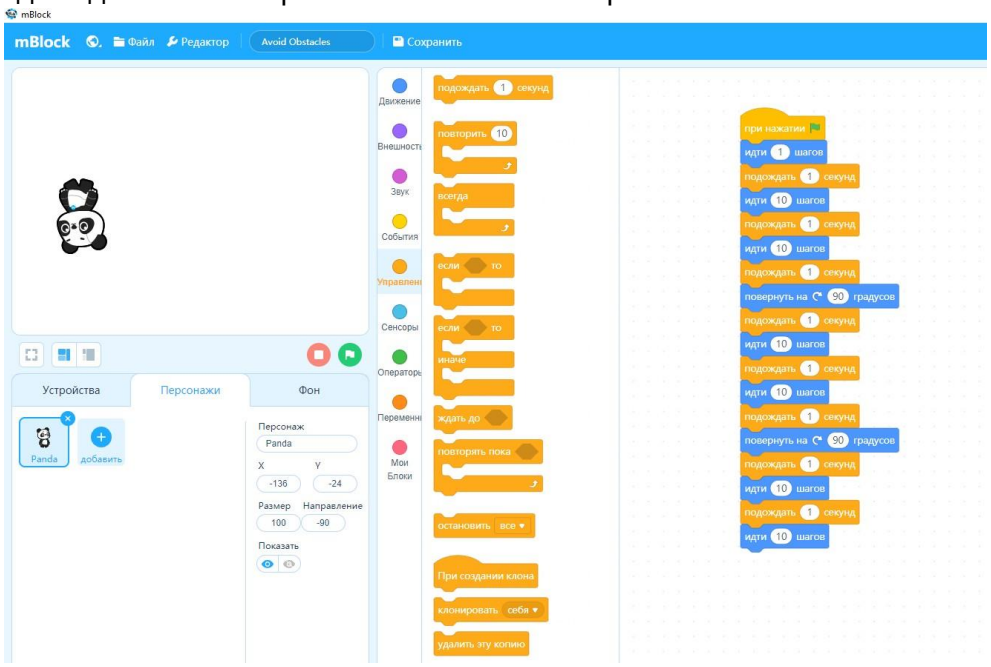


Рис. 24

Мы видим движение персонажа, но не видим траекторию движения. Для записи его траектории можно воспользоваться модулем «Перо», которое содержится в Центре расширений, рис. 25.

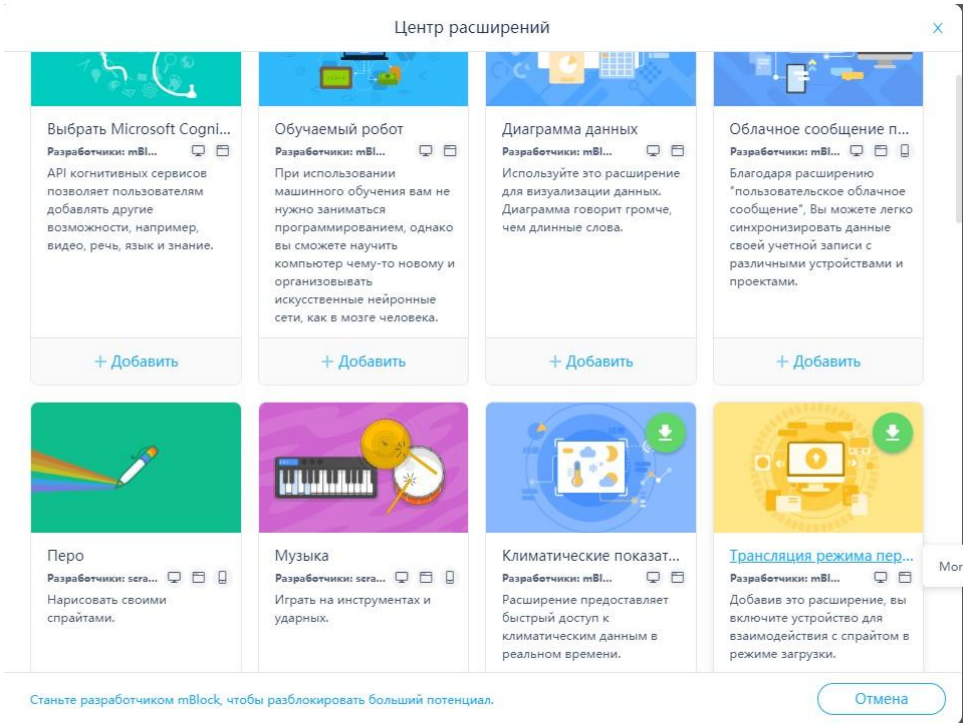


Рис. 25

Как только добавляем модуль «Перо» в панели инструментов появляется большой выбор команд для работы с пером, рис. 26.

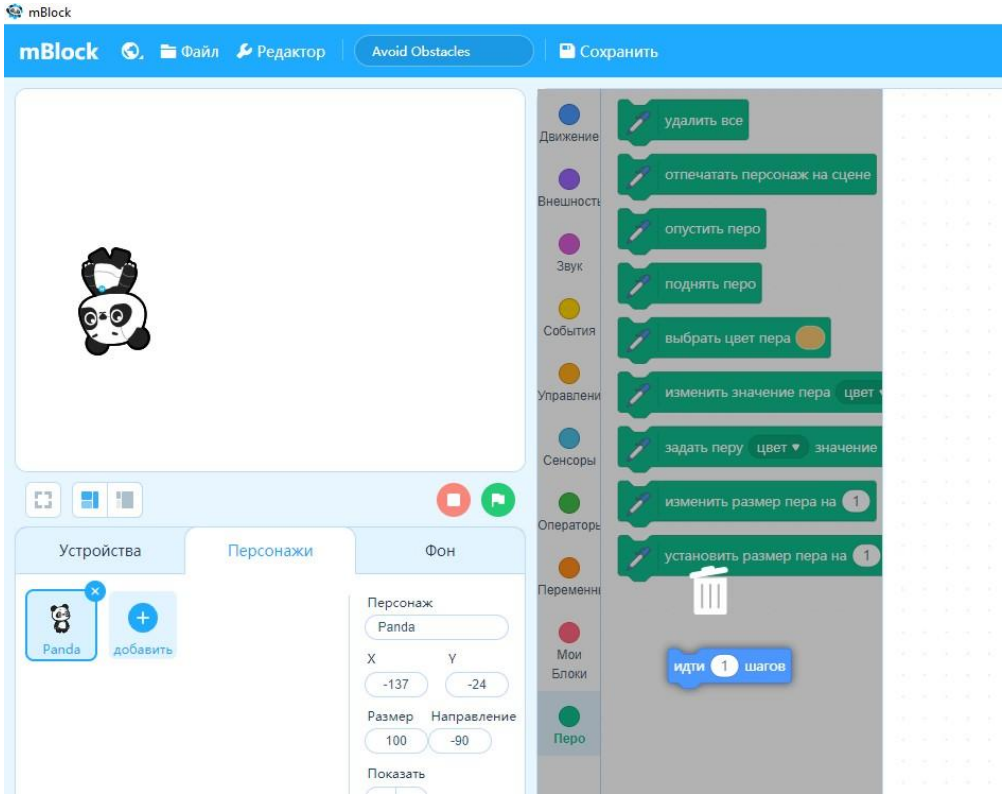


Рис. 26

Добавим в предыдущую программу две команды: опустить перо, выбрать цвет пера. Смотрите рис.27.

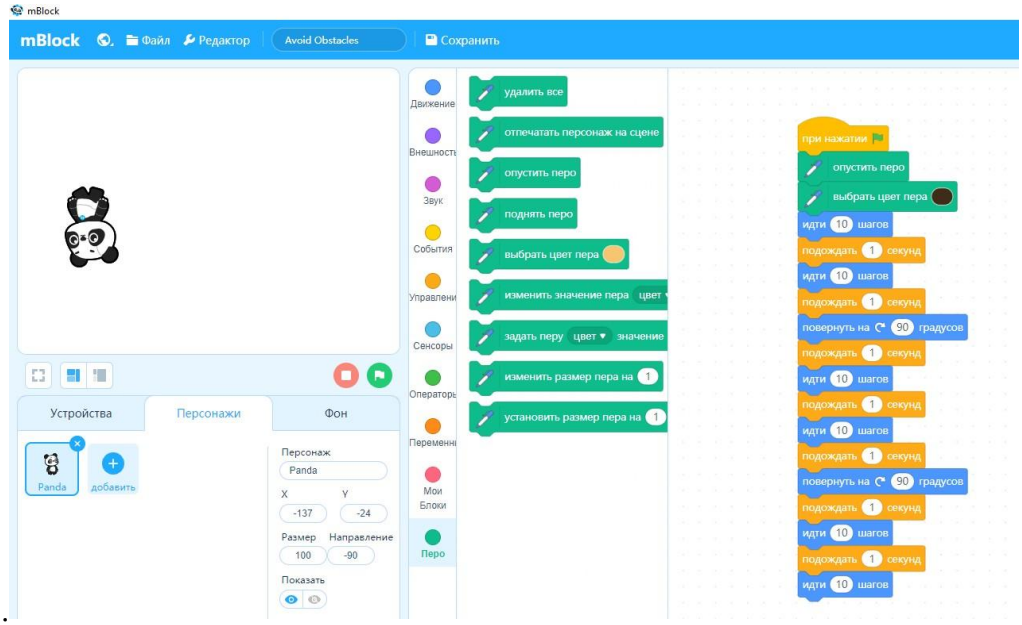


Рис. 27

Если запустим нашу программу, то мы не заметим никаких изменений, хотя они есть. При движении персонаж будет рисовать линию, но так как длина перемещения меньше размера персонажа, то траектория будет не видна. Необходимо уменьшить размер персонажа и даже, лучше, заменить его.

Создадим программу с линейным алгоритмом, с помощью которой персонаж в виде стрелочки рисует квадрат. Сам персонаж уменьшен до сопоставимых размеров, рис.28.

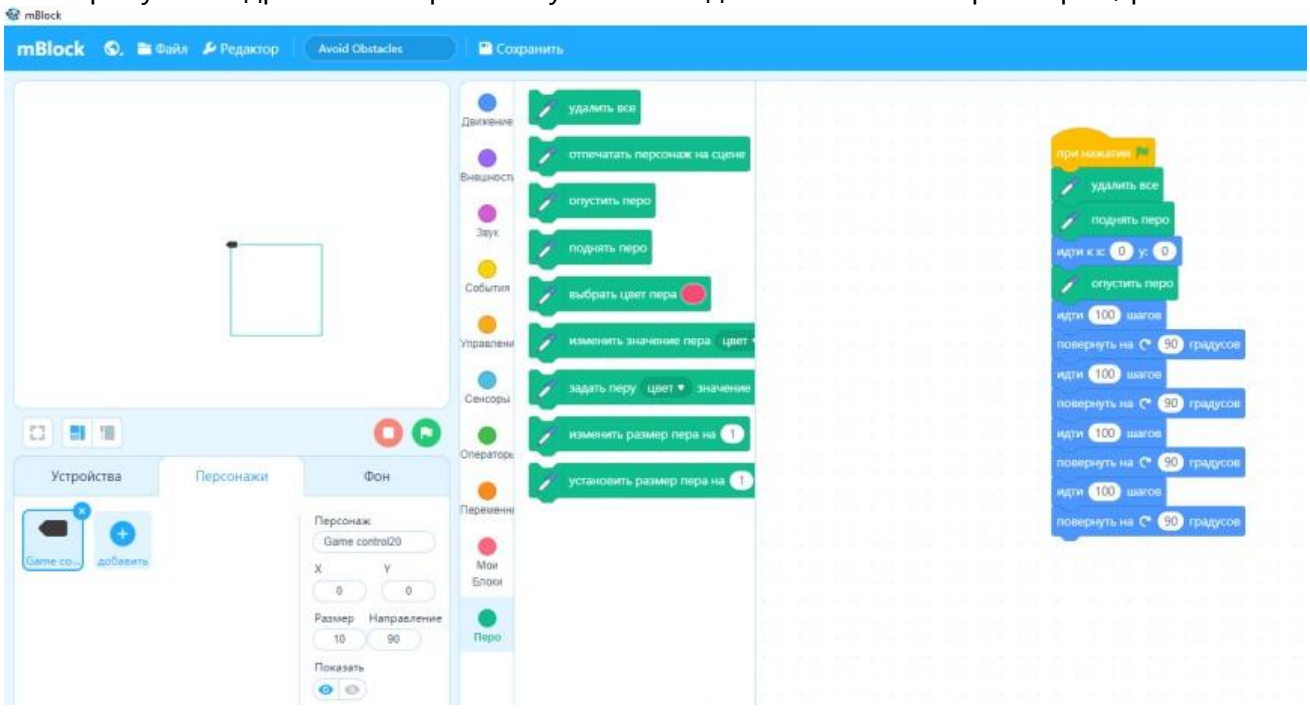


Рис. 28

Как видно, квадрат получился. Сам алгоритм построения квадрата прост – это повторение поочерёдно движения вперёд и поворота на 90^0. Можно также заметить, что добавились две новые команды из модуля «Перо»: «**удалить всё**», «**поднять перо**». Эти две команды позволяют очистить предыдущие рисунки с помощью пера и поднять перо, чтобы не рисовать ничего, но при этом перемещать перо. Команда «**идти к x:0 y:0**» позволяет переместить персонажа на поле с координатой (0, 0) – центр поля.

Дополним нашу программу командой на установку цвета, рис. 29.

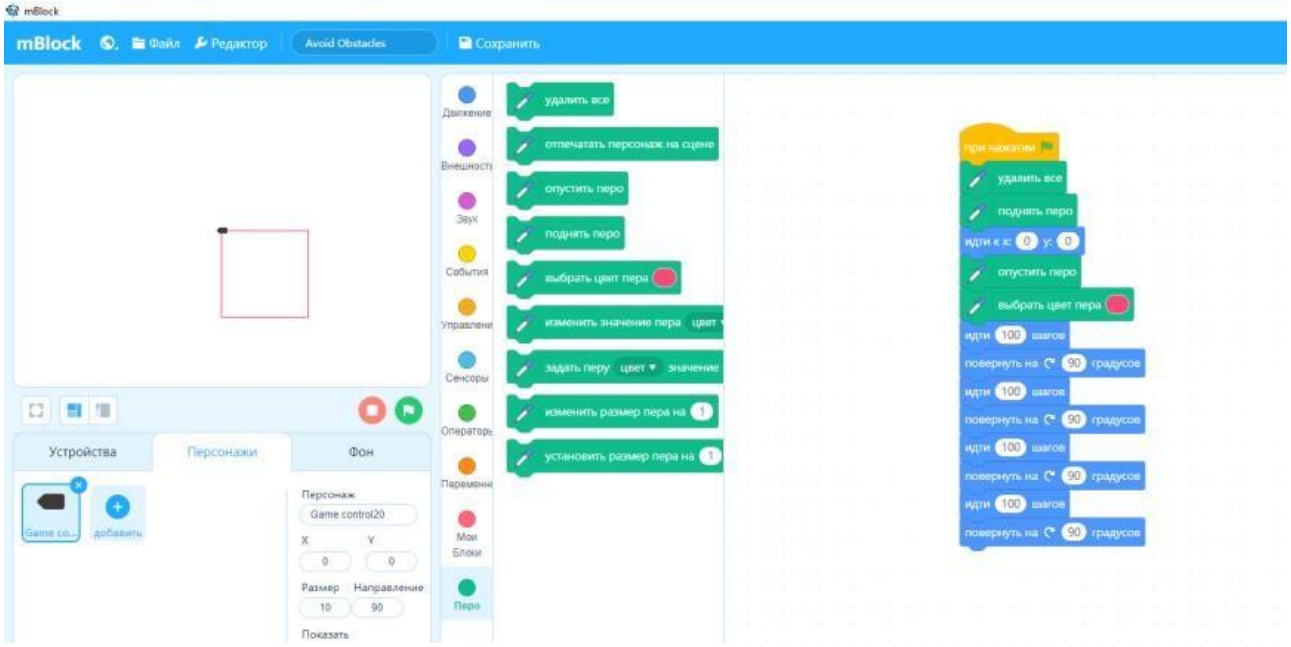


Рис. 29

Кроме этого, можно установить и толщину линии, рис. 30.

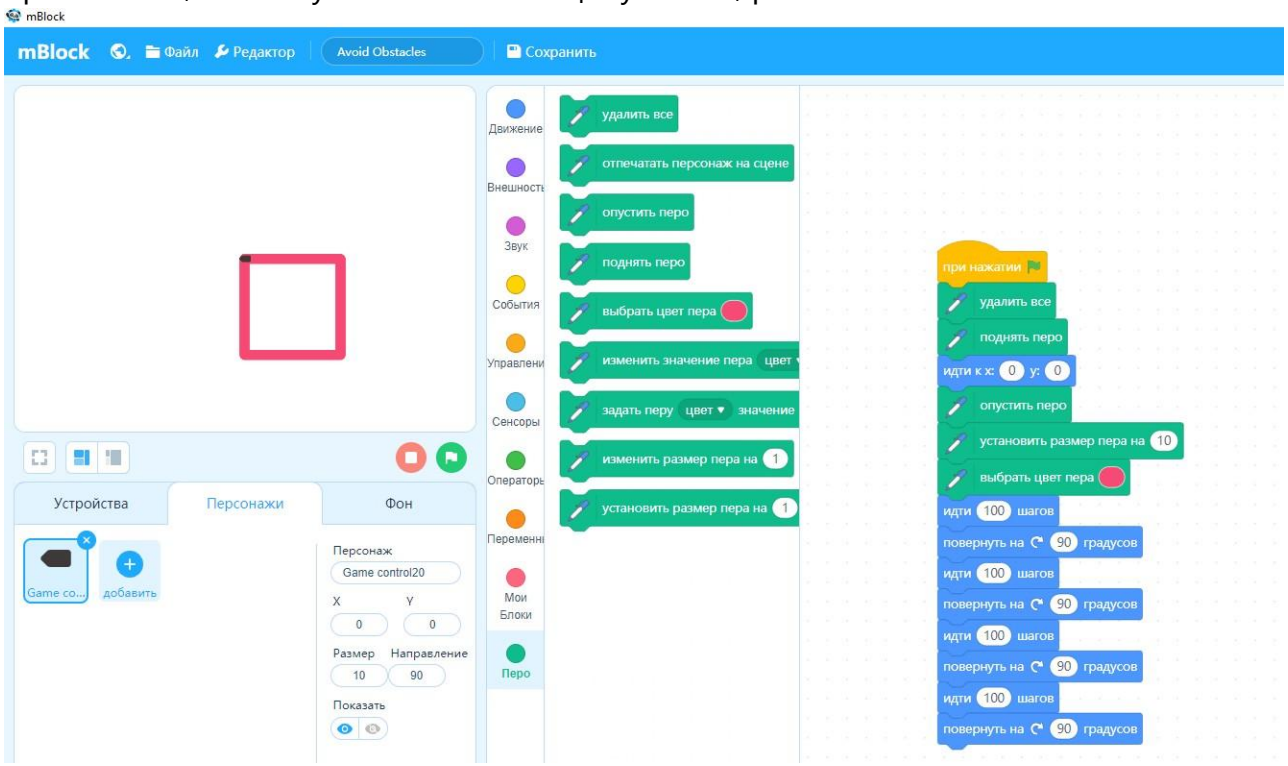


Рис. 30

Задание:

- Создать программы для построения фигур: окружности, треугольника (прямоугольного и равностороннего), пятиугольника, шестиугольника, восьмиугольника, пятиконечной звезды.

4.3. Ветвления и вложенные ветвления

Ветвление – это вид алгоритма, который содержит, как минимум два действия на проверяемые условия. Из определения следует, что в данном алгоритме нужно проверять

условие на вхождение данных и в зависимости от их истинности запускать определённые действия.

Пример.

Программа будет определять по своей шкале число на значения: большое или маленькое.

Если введённое число больше 50, то данное число можно считать большим, если меньше – то маленьким. Программа создаётся для персонажа.

На рис. 31 представлен пример такой программы.

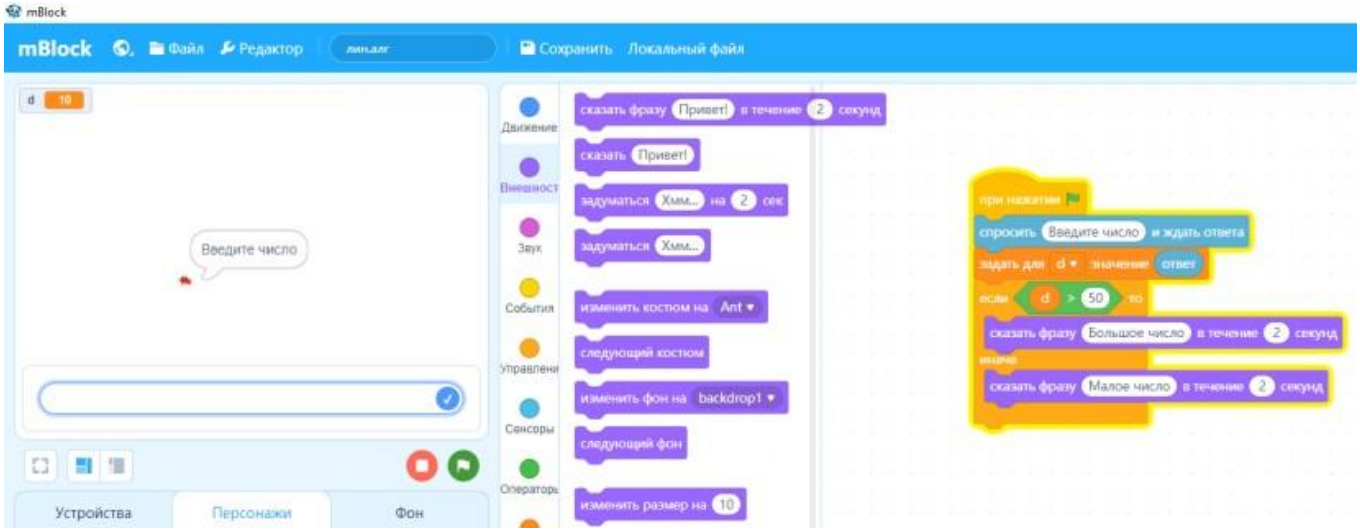


Рис. 31

Для введения числа используется вкладка «**Сенсоры**» и выбирается команда «**Спросить. Как Вас зовут? И ждать ответа**», рис.32.

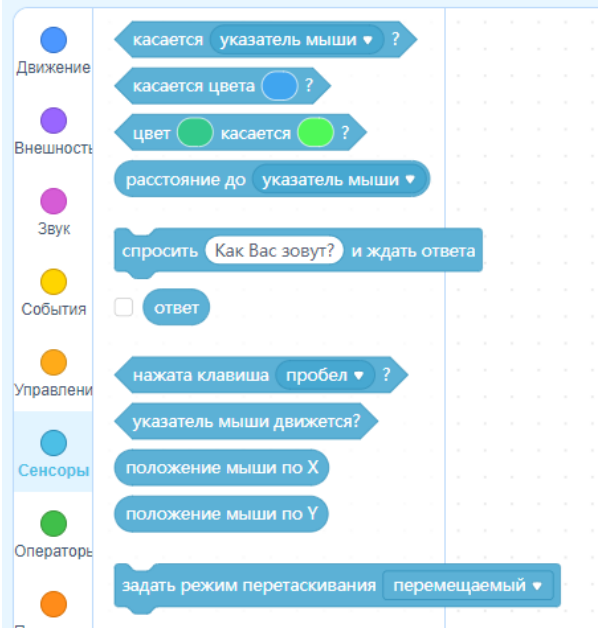


Рис. 32

Для того чтобы можно было вводить числовые значения, создадим переменную, рис. 33.

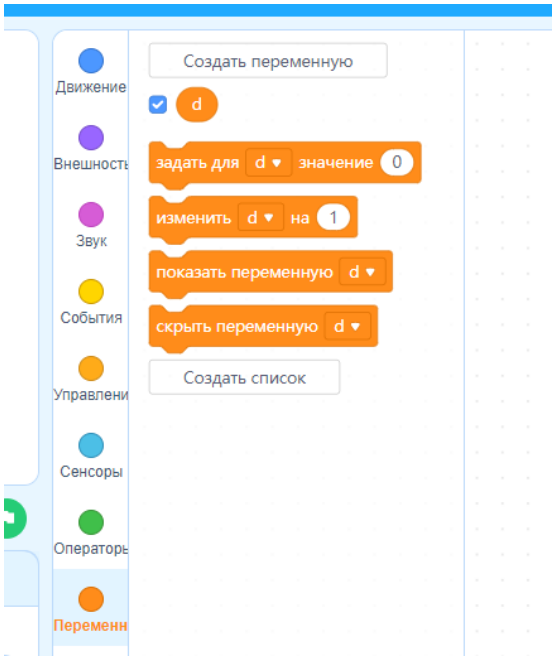


Рис. 33

Оператор условия находится во вкладке «Управление», рис. 34.

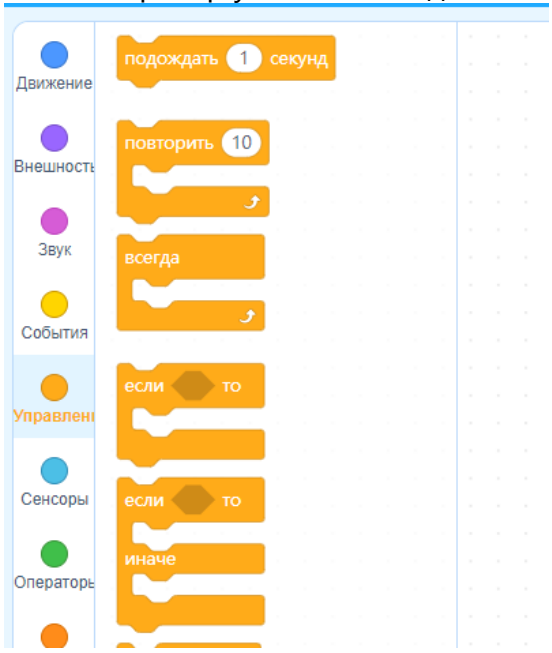


Рис. 34

Для того чтобы проверить число нужно применить оператор сравнения, который находиться во вкладке «Операторы», рис. 35.

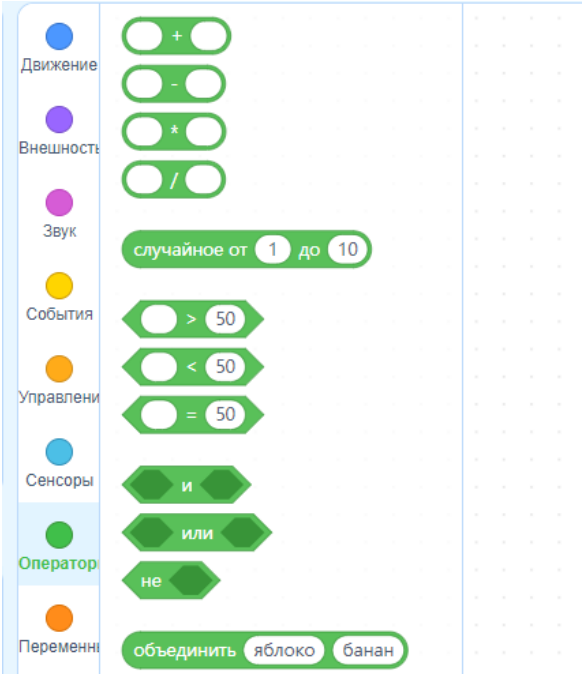


Рис. 35

Для вывода текста воспользуемся действием «Сказать фразу ___в течении __секунд», которое находится во вкладке «Внешность», рис. 36.

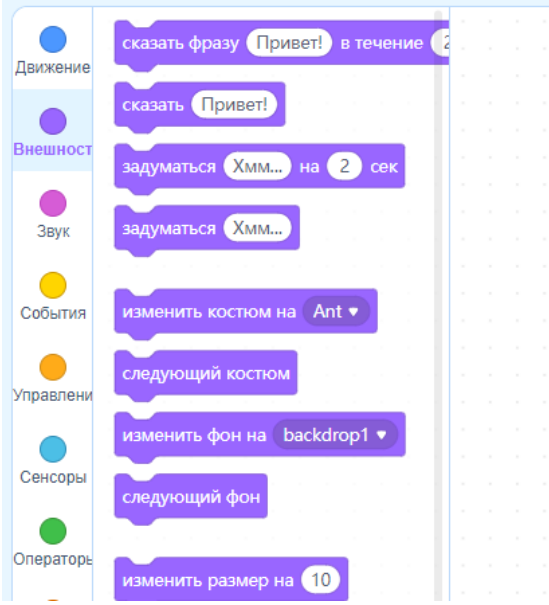


Рис. 36

Для закрепления работы с алгоритмом «ветвление» выполните задание.

ЗаданиеNo1

Составьте программу, которая на вопрос «Какой операционной системой пользуетесь?» :

- Windows
- Linux
- Android
- macOS
- Other

выдаст ответ в зависимости от выбора операционной системы.

Ответ можете придумать сами, главное чтобы он был разным, для разного выбора.

Задание№2

Составьте программу, которая на вопрос «Сколько тебе лет», будет выдавать ответ:

- Если возраст до 7 лет, то ответ: «Ты ходишь в детский садик».
- Если возраст от 7 до 18 лет, то ответ : «Ты школьник».
- Если возраст от 18 до 25 лет, то ответ: «Ты студент».
- Если возраст от 25 лет и более, то ответ: «Ты взрослый человек».

4.4. Циклы: конечные и бесконечные

Циклы – это вид алгоритма, который повторят множество раз определённую часть кода программы. Циклы бывают конечными и бесконечными.

Пример конечного цикла, рис.37.

Рис. 37

Из данной программы видно, что переменная **d** с числовым значением ноль, постепенно увеличивает своё значение на единицу сто раз. Интервал времени изменения значения равняется 0.2 секунды. В конце программа выходит из цикла и значения переменной обнуляется.

Используемый цикл берётся из вкладки «Управление». Оставшиеся элементы мы уже знаем откуда брать.

Рассмотрим пример бесконечного цикла, рис. 38.

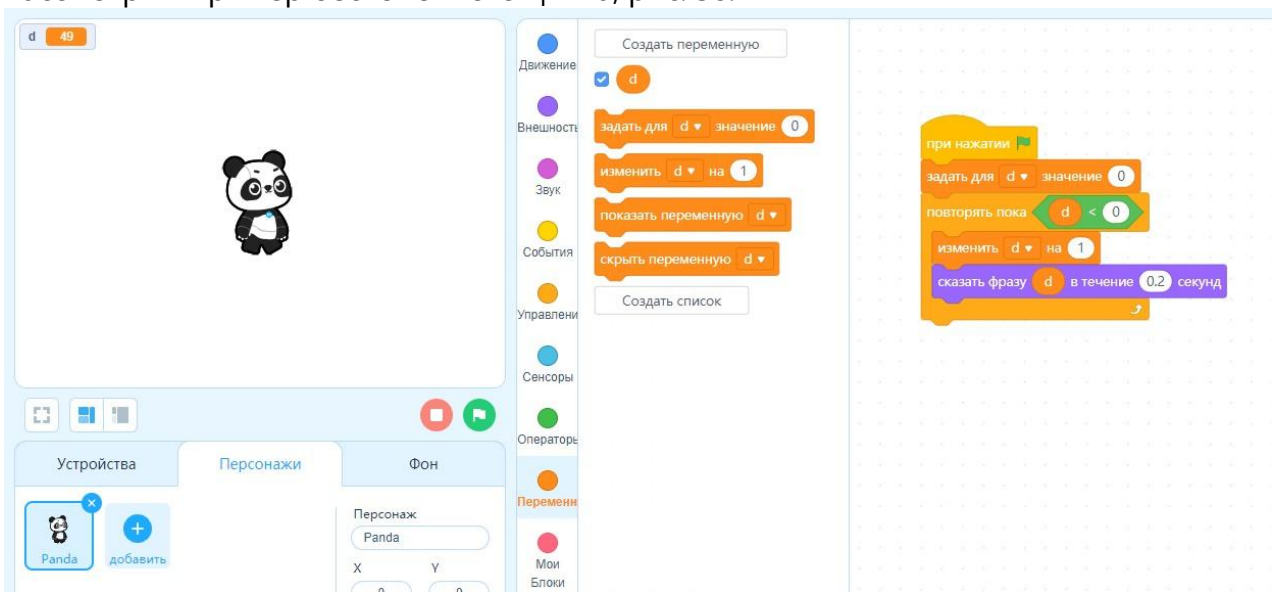


Рис. 40

В данном случае здесь представлен цикл с постусловием. Значение переменной проверяется, после срабатывания цикла первый раз.

Рассмотрим практическое применение цикла – нарисовать с помощью персонажа и пера квадрат.

Мы уже выполняли подобное задание в линейном алгоритме. Теперь с циклом этот процесс можно облегчить. Ниже представлен пример, рис.39.

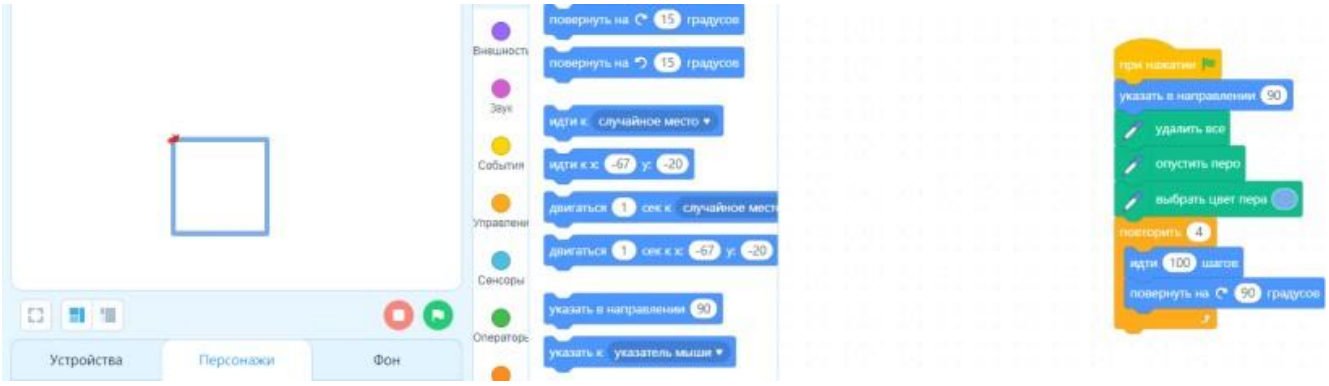


Рис. 39

Задание№1

- Постройте равносторонний треугольник, шестиугольник и восьмиугольник, используя цикл.

4.5. Вложенные циклы

Вложенные циклы – это когда в программе есть два и более циклов, одни из которых находятся внутри других. Часто такие циклы применяются для работы с несколькими параметрами, которые необходимо изменять.

Конечно, можно было бы обойтись без данного подхода и просто писать несколько конечных циклов множество раз друг под другом. В таком случае код оказался бы громоздким, а в программировании всегда есть стремление к упрощению процесса написания программы.

Рассмотрим пример, когда нам нужно нарисовать квадрат и при этом не один раз, а сто, да ещё и поворачивать его на 15^0. Ниже представлен код такой программы, рис.40.

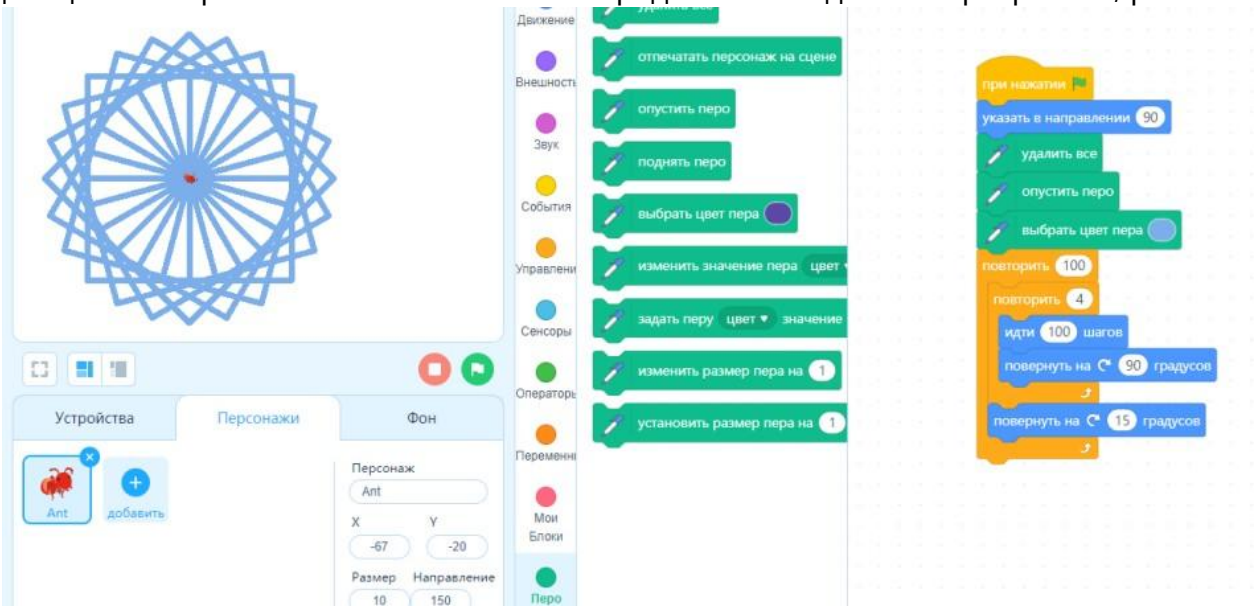


Рис. 40

Как видно здесь применили вложенные циклы и получили интересный рисунок. Кроме простого поворота фигуры, можно её ещё и смещать. Ниже представлен пример с подобным видом операций: поворот и сдвиг. Смотри рис. 41.

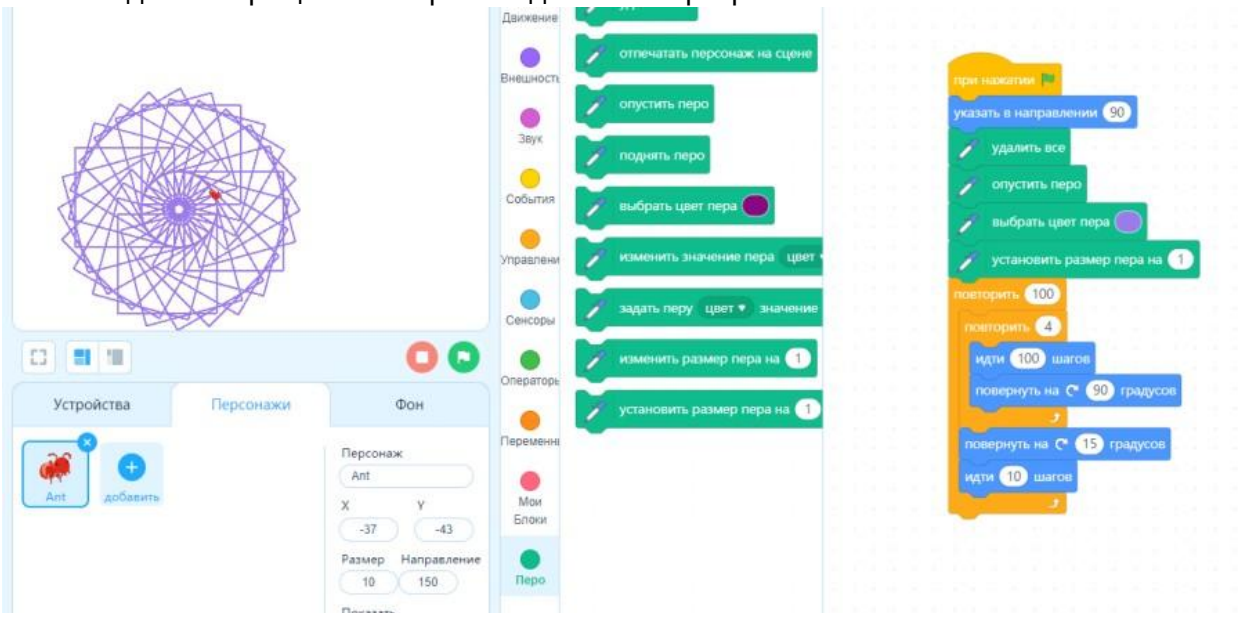


Рис. 41

Подобные операции над фигурами носит название – «Аффинные преобразования».

Задание№2

- Создать программы со вложенными циклами для создания фигур из поворотов равносторонних треугольников, шестиугольников и восьмиугольников. Угол поворота: 5^0, 15^0, 30^0, 45^0, 60^0, 90^0;

- Создать программы со вложенными циклами для создания фигур из поворота и сдвига равносторонних треугольников, шестиугольников и восьмиугольников. Угол поворота: 5^0, 15^0, 30^0, 45^0, 60^0, 90^0 ; при сдвиге 10 px (пикселей).

4.6. Комбинированные алгоритмы

Под комбинированными алгоритмами понимается сочетание уже изложенных алгоритмов в программе.

Рассмотрим пример, где необходимо сочетать ветвление и цикл.

Пример:

- Программа рисует одну из трёх фигур в зависимости от полученной числовой команды: 1,2 или 3. Фигуры: квадрат, шестиугольник и пятиугольник.

- Пример подобной программы представлен на рис. 42.

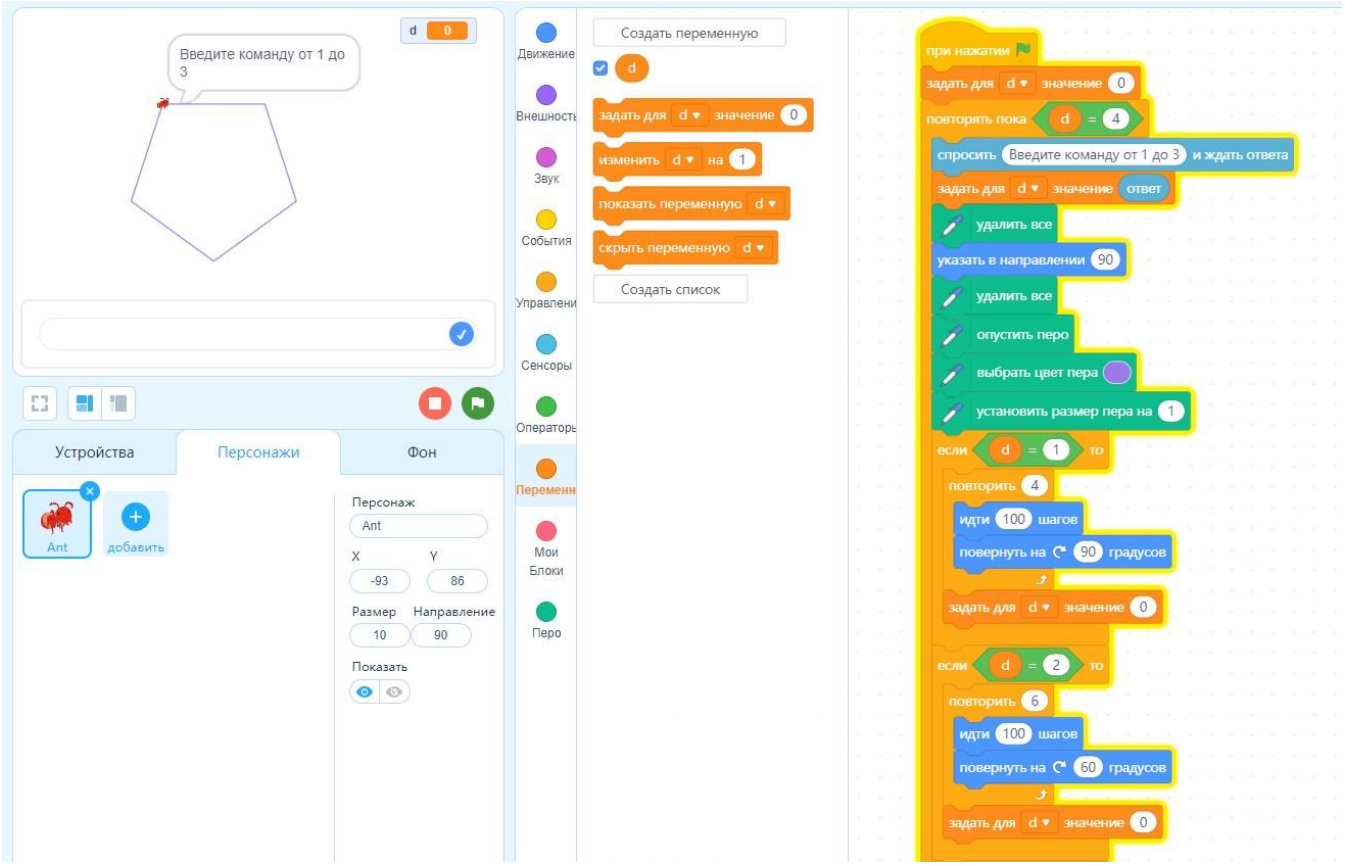


Рис. 42

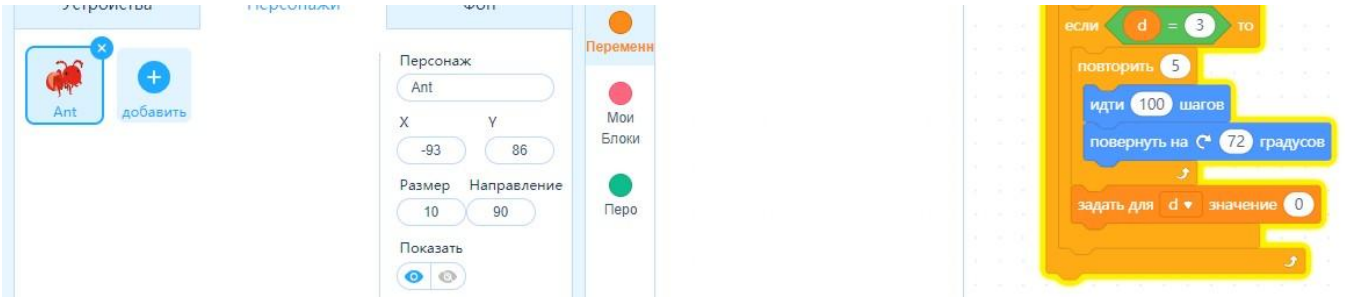


Рис. 43

После каждой выполненной команды, программа рисует фигуру и обнуляет значение переменной, которая отвечает за команду. Процесс повторяется бесконечно, пока не будет введена команда 4.

Задание

Составьте программу, которая рисует одну из четырёх фигур, в зависимости от команды: **1,2,3,4**. Кроме построения фигуры она её закрашивает с помощью одной из команд второго набора: **r, b, g, y**. Фигуры: пятиконечная звезда, прямоугольный треугольник, равнобедренная трапеция, параллелограмм. Цвета: красный, синий, зелёный и жёлтый.

5. Плата Arduino Uno

5.1. Arduino Uno

Одна из самых востребованных плат. Данная плата очень популярна среди любителей робототехники. В самом названии есть слово **Uno**, которая означает «универсальная».

Есть официальная разработка, созданная в Италии, а есть её клоны, созданные в Китае. С официальной платой можно работать сразу без предварительных настроек. У китайских плат есть особенность, они собраны на своей микросхеме CH401 и поэтому сначала необходимо установить драйвер.

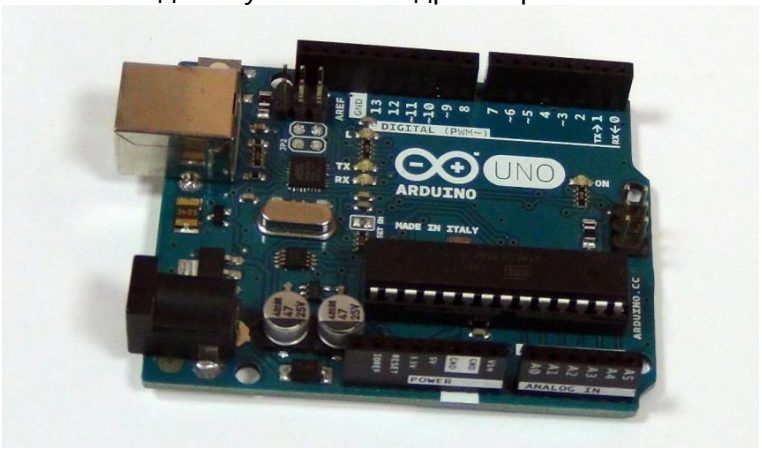


Рис. 1

Первое наше знакомство, начнётся с правильного подключения электронных компонентов и работы с ними. Рекомендую изучить эту главу и отработать все варианты подключения и настройки электрокомпонентов. Рассмотрим плату Arduino Uno.

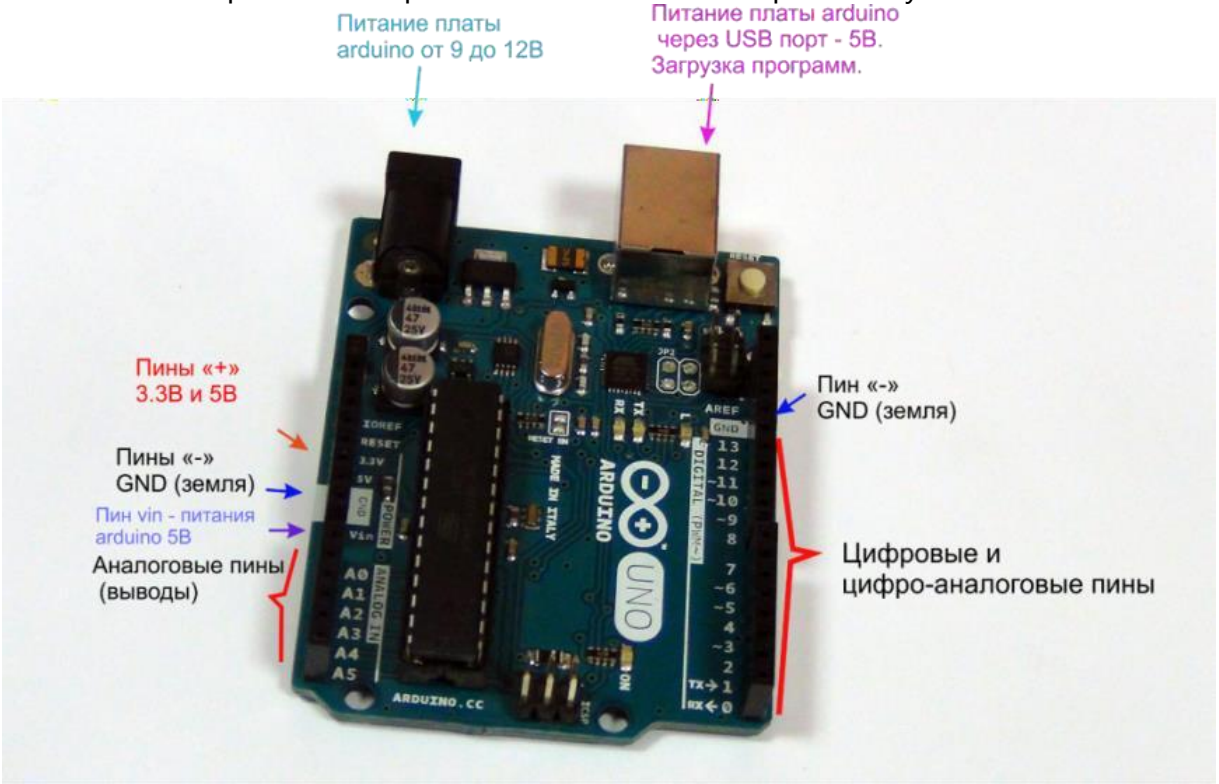


Рис. 2

На рисунке представлено краткое описание ключевых элементов платы Arduino Uno. Все части других плат с данным обозначением выполняют те же функции.

Включить плату можно как минимум тремя способами:

- Через USB –порт подключить плату к компьютеру по специальному шнуру, максимальный ток 0.5А, а напряжение 5В.

- Через разъём для штекеров. Можно подать напряжение от 9 В до 12 В без ущерба для платы.

- Через порты GND и VIN – подача питания напрямую к плате минуя стабилизатор. Максимальное значение тока 0.3А и напряжение 5В.

Чаще всего применяют первый способ подключения – он самый безопасный для платы. Так же через данный порт загружают программы на плату из компьютера или передают данные с платы на компьютер. Вторым способом пользуются, когда программа «залита» в плату, а устройство протестировано и работает исправно. Третий способ используют в редких случаях, либо, когда нет возможности использовать первые два, либо для решения специфических задач. Все входы и выходы принято называть пинами (Pin).

Пины GND – их здесь целых три, отвечают за питание цепи от платы и замыкания цепи – это знак «-» или его ещё называют земля. Пины 3.3В и 5В – это питание цепи платы – знак «+».

Пины с 0 до 13 – являются цифровыми и цифро-аналоговыми выводами. Пины 13, 12, 8, 7,4, 2,1,0 – являются цифровыми, а пины 11, 10, 6,7,5,3 - цифро-аналоговыми. В основном их задача передавать сигналы «команды» на подключённые электронные элементы устройства, но так же они могут выполнять функцию приёма данных с этих элементов.

Пины A0 – A5 – аналоговые выводы их задача получать данные с аналоговых датчиков.

5.2. Особенности конструкции кода

5.2.1. Основные функции и операторы

int, float, pinMode, digitalWrite(), analogRead(), analogWrite(), Serial(), delay()

Мы рассмотрели основные инструменты программы **Arduino ide**. Теперь пришло время перейти к особенностям программирования в данной среде. Программы в среде **Arduino ide** написаны на языке С. Данный язык относится к языкам программирования высокого уровня со статической типизацией.

Статическая типизация – это, если своими словами, обозначение типов данных для переменных, которые будут неизменны уже в начале программы. Данный способ облегчает работу компилятора, избавляя его от диагностики и определения типов данных.

Типы данных – это один из самых важных элементов программирования. В зависимости от типа применяются различные операции или получают разные ответы при одной и той же операции.

Типы данных:

int – возвращает целочисленное значение в диапазоне от -32768 до 32767

float – возвращает дробное значение, записанное в виде десятичной дроби в диапазоне от 3.4028235E+38 до -3.4028235E+38

byte – возвращает восьмибитовое числовое значение без десятичной точки в диапазоне от 0 до 255.

bool – логический тип данных. Возвращает значение **true** (истина) или **false** (ложь).

Как уже было упомянуто, сданными типами данных можно проводить ряд операций. Рассмотрим их. **Математические операторы**: **+** сложение	**x --** уменьшение на 1
- вычитание	**x+=y** увеличение на значение **y**
/ деление	**x- =y** уменьшение на значение **y**
***** умножение	**x*=y** последующее умножение на **y**
x++ увеличение на 1	**x/=y** последующее деление на **y**

С помощью математических операторов можно проводить простые арифметические действия с числами.

5.2.2. Операторы сравнения

x==y-равенство	x<y-меньше
x!=y-не равно	x>=-больше или равно
x>y- больше	x<=y-меньше или равно

С помощью операторов сравнения можно проверять свои результаты с условием на истинность, т.е. подходят ли ваши результаты под условие или же нет.

5.2.3. Логические операторы

Логические операторы необходимы для вычисления логических операций, т.е. для определения истинности или лжи при взаимодействии двух и более логических значений.

Ниже представлена «Таблица истинности », которая проверяет истинность выражения из двух значений A и B. В таблице представлены три логических оператора: AND, OR и NOT. В скобочках представлены альтернативные обозначения логических операторов.

A	**B**	**AND (&&)**	**OR (||)**	**NOT (!A)**
true	true	true	true	false
true	false	false	true	false
false	true	false	true	true
false	false	false	false	true

5.2.4. Переменные

Переменные – это именованные ячейки памяти. Переменными выступают обычно буквы латинского алфавита или английские слова, которые несут смысловую нагрузку принимаемого им значения, так, чтобы его мог понять любой программист.

Перед указанием некой переменной необходимо указать тип данных, который она будет записывать.

Пример записи переменных в программу представлен на рис.1

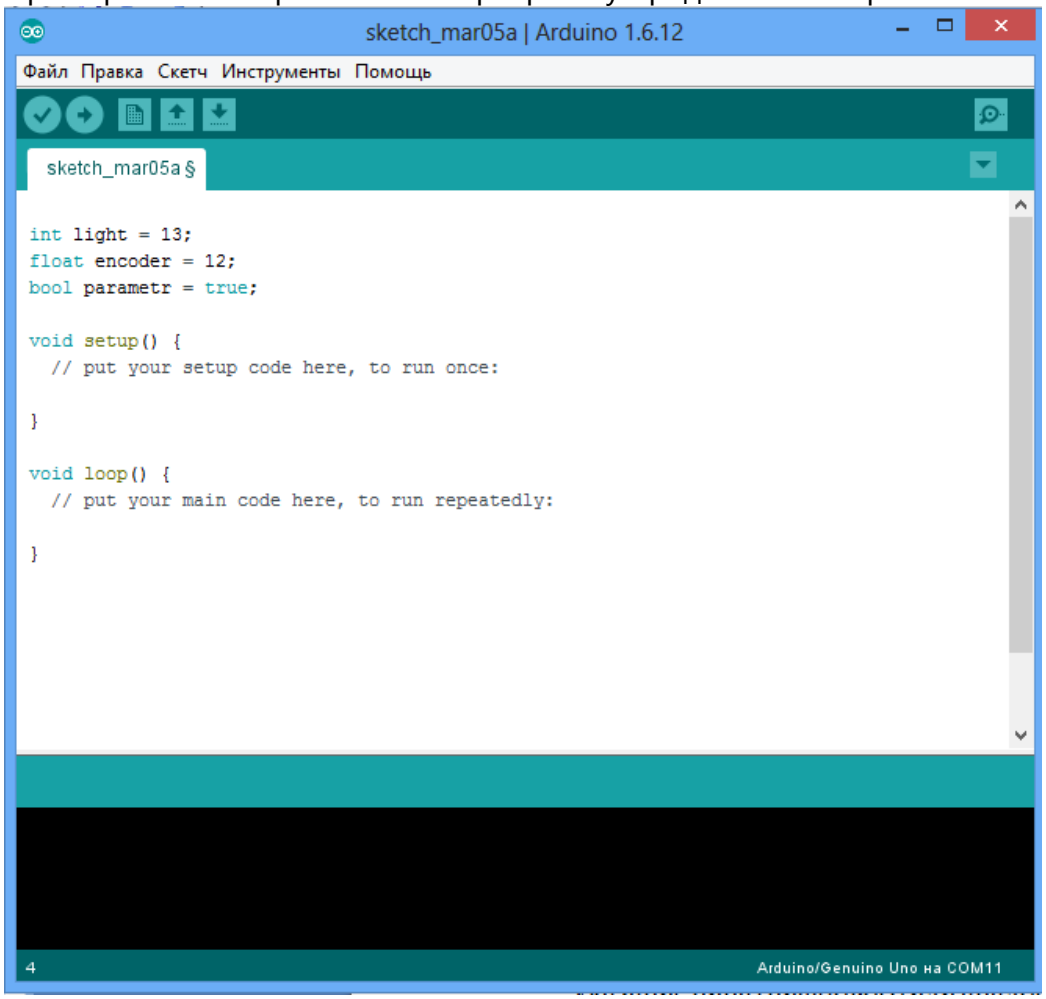


Рис. 3

Как видно, обычно переменные вводятся в самом начале. Здесь представлены три переменных с разными типами данных. Нетрудно заметить, что напротив переменных указаны числа.

Данные числа могут играть две роли:

- начальное значение переменной

- указание пина (цифрового или аналогового), на который будут подавать или от которого будут получать числовые значения.

В нашем случае – это указание пина, но не что не мешает рассматривать это как число, до тех пор, пока не будет указана некая функция. Отметим то, что после каждого определения переменной стоит точка с запятой – это одна из главных особенностей языка **Си**, как и фигурные скобки.

Функции void setup () и void loop().

Как видно, при открытии нового файла в **Arduino ide** всегда присутствуют две некие функции: **void setup () и void loop().** Для большинства задач возможности этих функций достаточно.

В функции **void setup ()** необходимо определить пины, к которым подключены датчики, модули или актуаторы (моторы). Не только их определить, но и указать – получаем или передаём цифровые или аналоговые сигналы. Также здесь настраивается связь с последовательным портом.

Функция **pinMode()** – определяет номер пина на плате Arduino и указывает действие на приём или передачу значений:

- pinMode (13, OUTPUT); - отправляем числовые значения на пин 13.
- pinMode (13, INPUT); - принимаем числовые значения с пина 13.

Функция **Serial.begin(9600); -** открывает последовательный порт и задаём скорость 9600 для последовательной передачи данных.

Ниже представлен пример записи пинов и указания действий с ними.

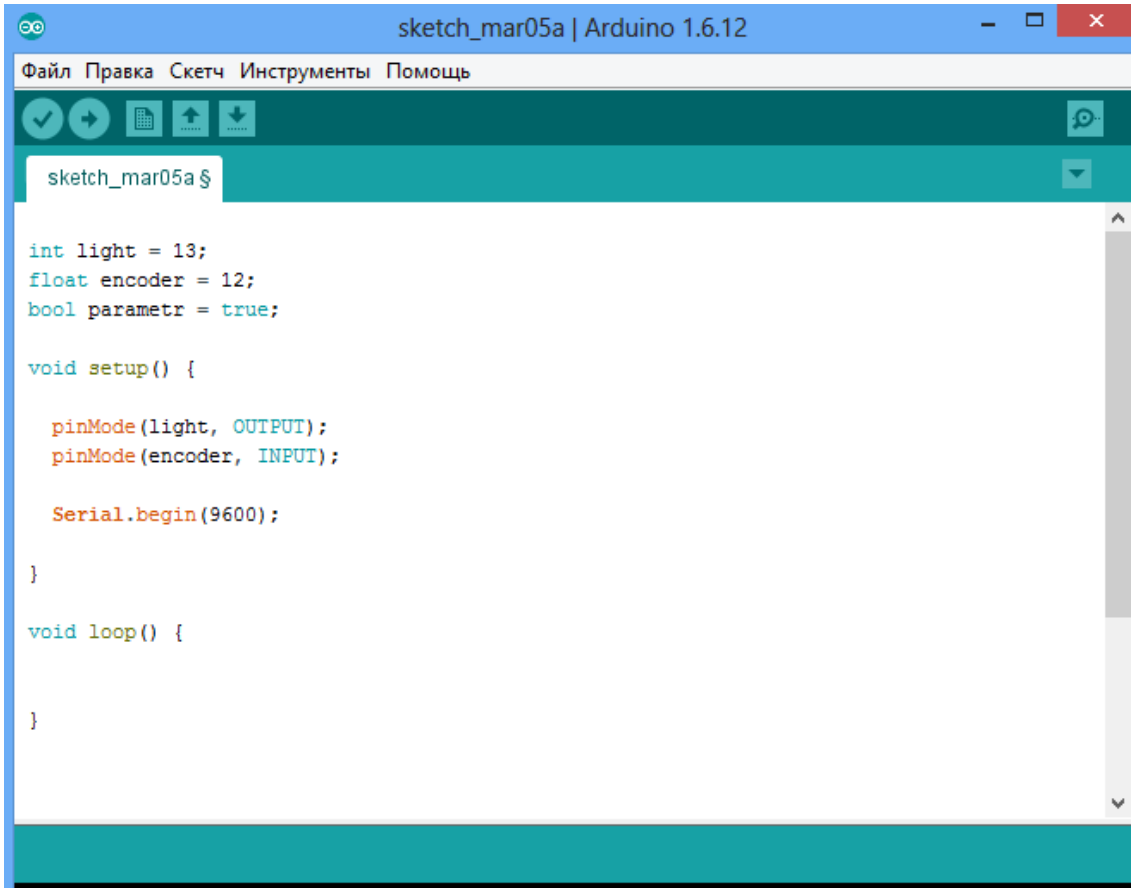


Рис. 4

Как видно, кроме указания номера пина, можно указать имя переменной, которая принимает значение номера пина в функции **pinMode()**.

Функция **void loop()** нужна для запуска бесконечного цикла. В данном цикле можно указать действия, которые будут многократно повторяться. Например, вывод значений через последовательный порт пина номер 12 – encoder.

За вывод информации отвечает функция **Serial.print ()** или **Serial.println()**. По форме записи эти функции отличаются только на две последние буквы. Первая функция будет выводить информацию непрерывно друг за другом в строчку. Значения при таком выводе будут писаться слитно (конечно можно придумать раздельное написание). Вторая функция

будет выводить информацию непрерывно друг за другом построчно (в столбик). Пример представлен ниже.

```
void loop() {

Serial.print(encoder);
}
```

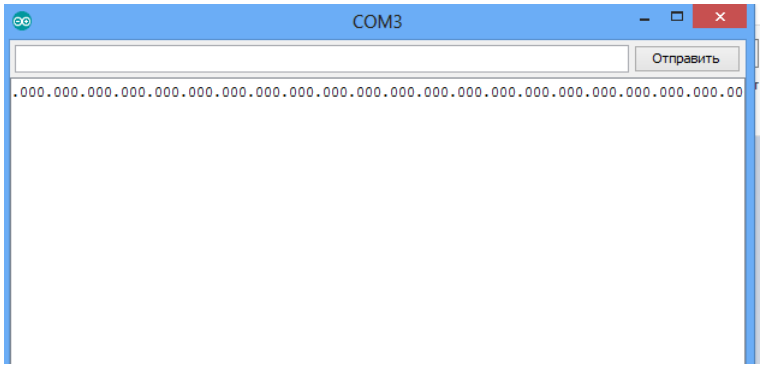


Рис. 5

```
void loop() {

Serial.println(encoder);
}
```

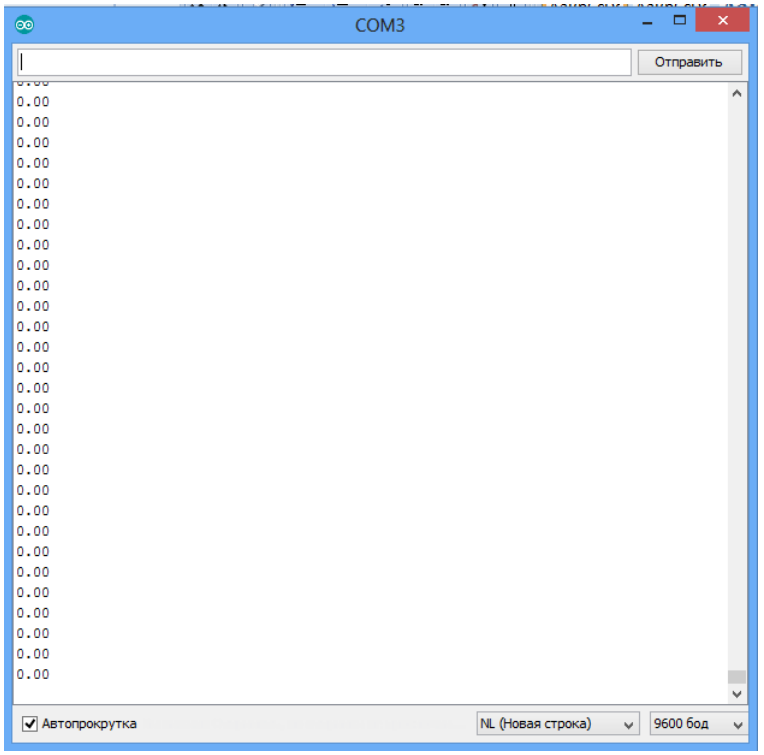


Рис. 6

Как видно из рисунков, в зависимости от функций языка, будут разные варианты вывода, а выводимая числовая информация, будет записана в виде дробного значения, так как тип данных **float**.

Информация выводится через монитор «порта», **смотри 3.3.1** – монитор порта.

Если не указывать в каком формате получать числовые значения, то они выводятся, по умолчанию, подключённого пина (смотри понятие цифровые и аналоговые пины платы Arduino).

5.2.5. Задержка по времени

Выводить информацию плата может со скоростью в 1/1000 секунды. При такой скорости очень трудно сфокусировать взгляд. Поэтому применяют функцию **delay()**. Данная функция проводит задержку выполнения программы. В скобочках функции указываем значение времени в миллисекундах.

Пример.

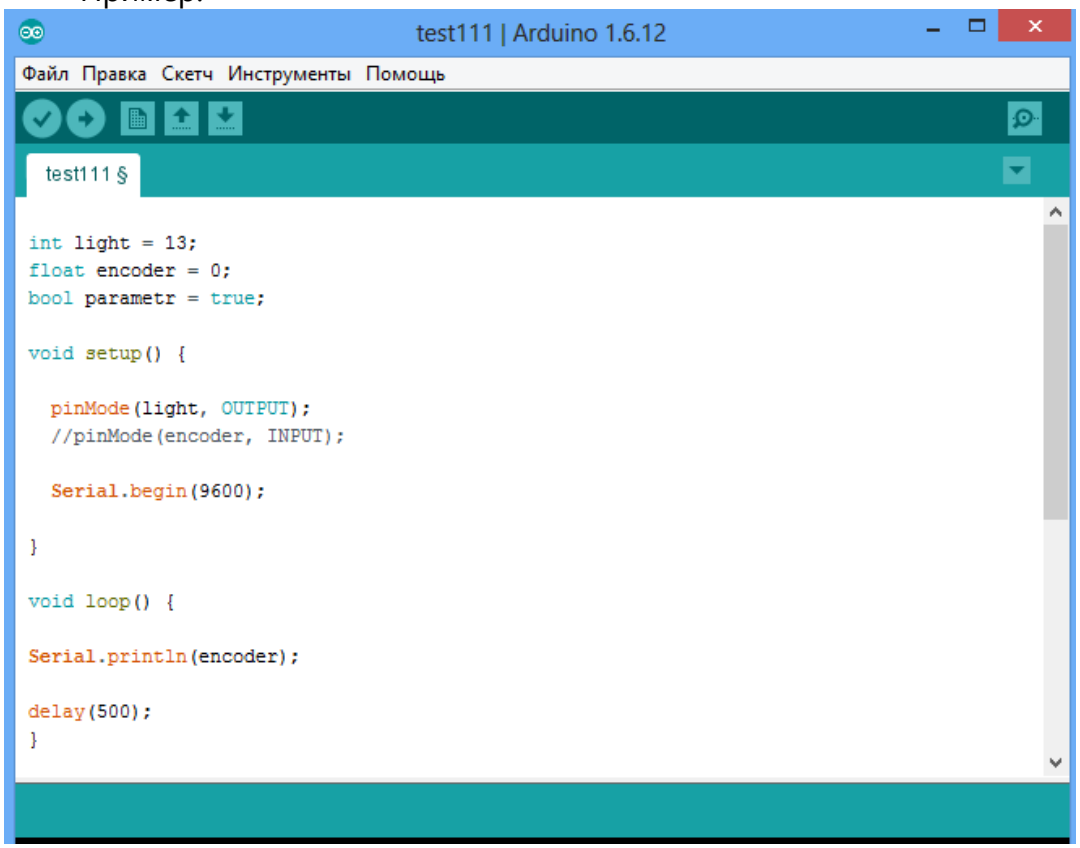


Рис. 7

В примере указано, что значение переменной **encoder** будет выводиться через каждые 0,5 секунды.

Функции **digitalRead()** и **analogRead().**

Как уже упоминалось, на плате Arduino присутствуют цифровые и аналоговые пины, которые по умолчанию выводят либо значения 0 или 1 (цифровые пины), либо от 0 до 1023 (аналоговые пины). Не которые пины могут выводить как цифровые, так и аналоговые значения (они помечены значком ~).

Для чтения с одного пина либо цифровые значения либо аналоговые применяют функции **digitalRead()** и **analogRead()** . Если нам нужно получить значение в виде 0 и 1, то используем функцию **digitalRead()** .

Если нам нужно получить значения от 0 до 1023, то используем функцию **analogRead().**

Ниже представлены примеры работы двух функций.


test111 | Arduino 1.6.12
Файл Правка Скетч Инструменты Помощь
test111 §
int light = 13;
int encoder = 0;
bool parametr = true;
void setup() {
pinMode(light, OUTPUT);
//pinMode(encoder, INPUT);
Serial.begin(9600);
}
void loop() {
Serial.println(digitalRead(encoder));
delay(500);
}

Рис. 8

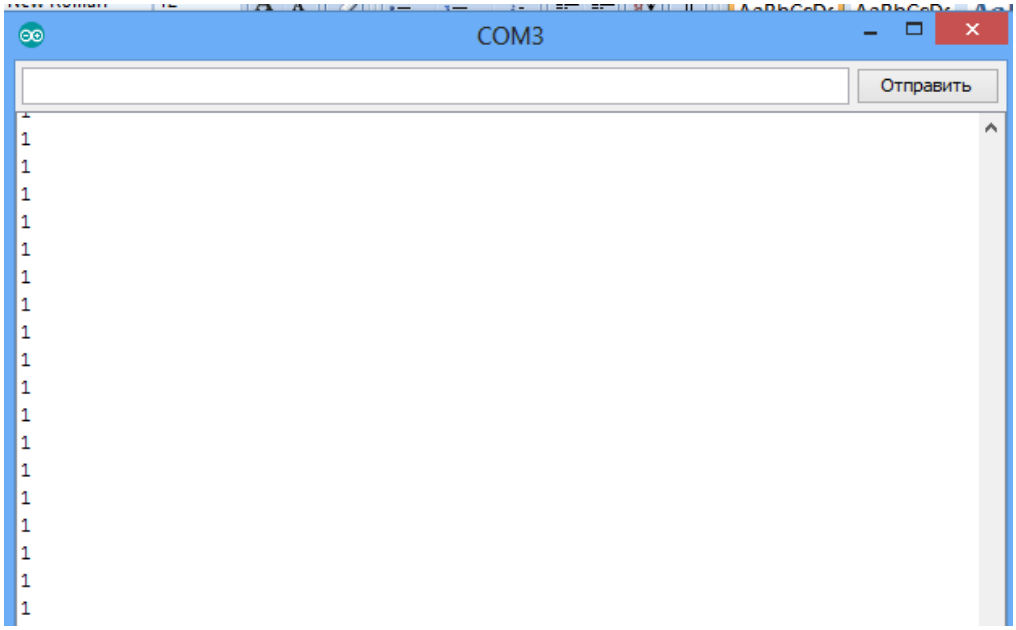

COM3
Отправить
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1
1

Рис. 9

Рис. 10

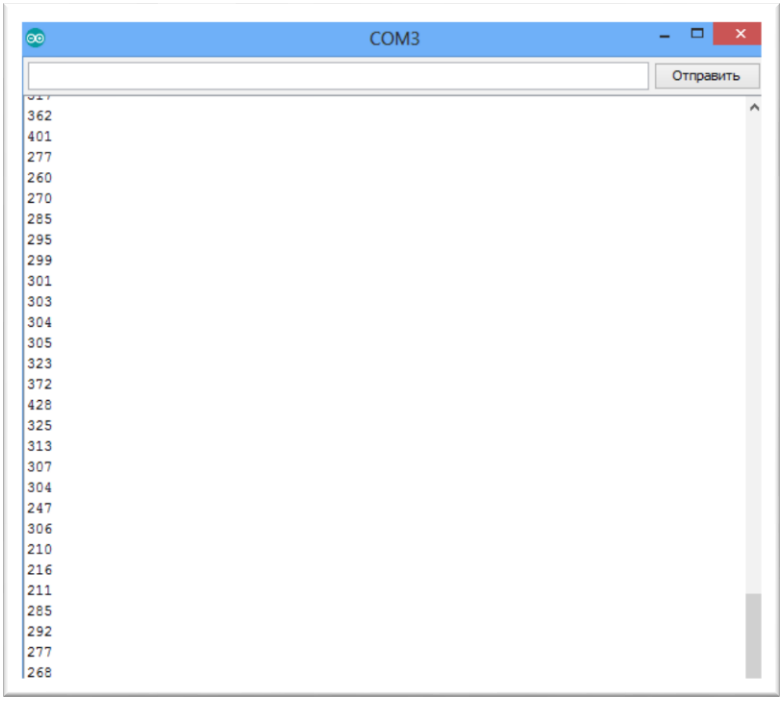


Рис. 11

Для удобства заменим тип данных для переменной encoder с float на int.

Кроме чтения данных с пинов, можно отправлять значения на них. За данную процедуру отвечают функции digitalWrite () и analogWrite().

Функция **digitalWrite ()** позволяет посылать значения на пин: 0 (**LOW**) или 1 (**HIGH**). Функция **analogWrite ()** позволяет посылать значения на пин от **0 до 255**.

Пример представлен ниже.

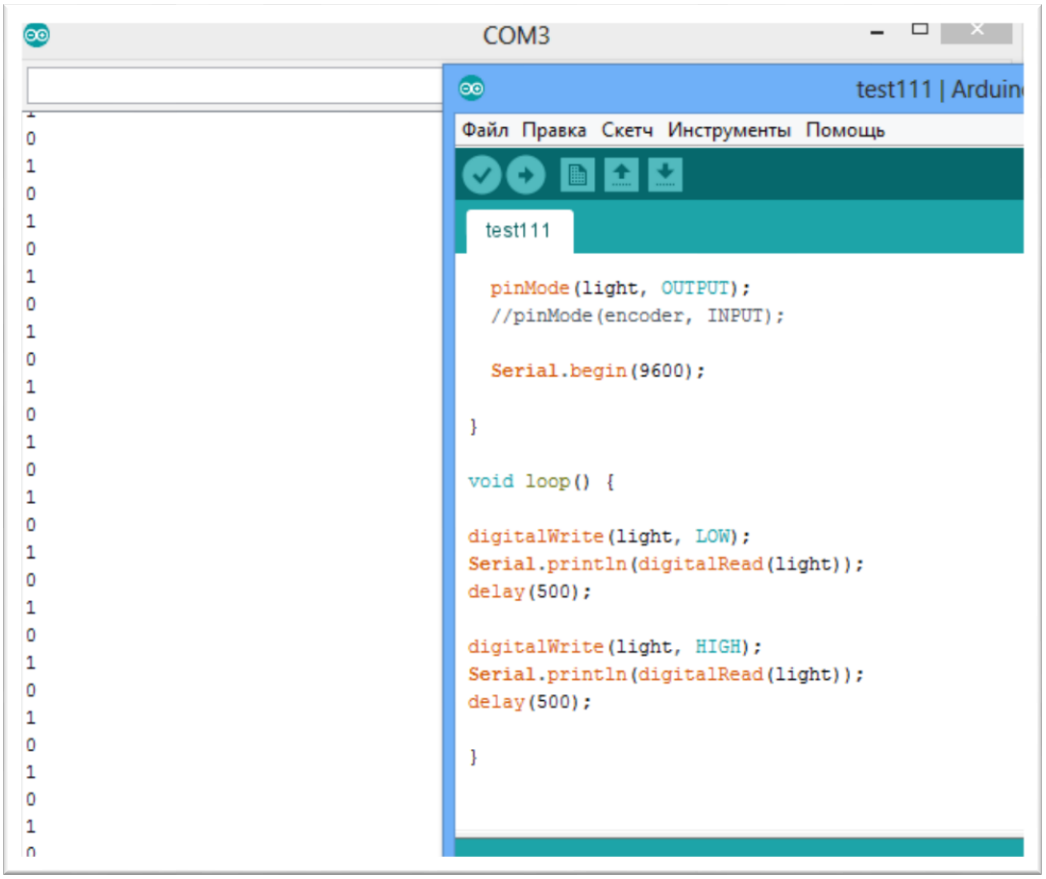


Рис. 12

Рис. 13

5.2.6. Ветвление и вложенные ветвления

Мы рассмотрели основные элементы Arduino api. Теперь необходимо рассмотреть реализацию основных видов алгоритмов. Первым таким алгоритмом является – ветвление (условие).

С данным алгоритмом мы познакомились в параграфе 3.2.3 в среде mBlock. Теперь необходимо рассмотреть его структуру в среде Arduino ide.

Рассмотрим пример реализации условия в задаче управление включением светодиода с помощью кнопки.

Конструкция условия для Arduino ide будет выглядеть так:

if (проверка условия){ действие при истинности выполнения условия } Ниже представлен алгоритм с условием для вышеуказанной задачи.

```
void loop() {


 if (digitalRead(button1) == HIGH) {

digitalWrite(svet, HIGH);

}
else {

digitalWrite(svet, LOW);
}
```

Как только будет нажата кнопка, то пойдёт команда на включение светодиода, если кнопка не будет нажата, светодиод не включится.

В конструкции ***else*** { } прописывается альтернативное действие.

Рассмотрим задачу, когда необходимы вложенные циклы. Например, когда у нас есть два фоторезистора, которые снимают показания освещённости, на основе которых установка вращает дополнительный источник света с помощью сервопривода.

```
if( analogRead(foto1)>analogRead(foto3))
{
   if (analogRead(foto1)-analogRead(foto3)>40){
   i=i+1;
   myservo1.write(i);
   delay(20);
          }

      }

 else{

    if (analogRead(foto3)-analogRead(foto1)>40){
   i=i-1  ;
   myservo1.write(i);
   delay(20);
         }
      }
```

Как видно, внутреннее условие заключено в {} внешнего условия, при этом альтернативного действия может не быть.

Каждое действие, что в одном ветвлении, что во вложенных, всегда в конце заканчивается на знаке « **;** » .

5.2.7. Циклы и вложенные циклы

С циклами мы уже познакомились в параграфах 3.2.4 и 3.2.5 в среде mBlock5.

Общая конструкция конечного цикла представлена ниже.

$$for\ (int\ i = 0; i < 100, i++)\ \{\ действия\ \}$$

Оператор **for** определяет начало цикла. В цикле необходимо указать количество шагов и значения диапазона. Для это чаще применяют локальные переменные (переменные, которые были созданы непосредственно в каком-то месте программы, смотри пример выше).

Согласно примеру, создаётся локальная переменная **i** со значением ноль, которая принимает целочисленные значения. Данные значения увеличиваются на единицу, пока не достигнут значения 99, после этого цикл прервётся. Подобная реализация алгоритма приводиться в примере по управлению сервоприводом:

```
void loop() {
  for (pos = 0; pos <= 180; pos += 1) {
    // in steps of 1 degree
    myservo.write(pos);              //
    delay(15);                       //
  }
```

Конструкция вложенных циклов похожа на конструкцию вложенных условий. Например, необходимо синхронное управление двумя сервоприводами. Первый сервопривод поворачивается на 180^0 и обратно, после этого поворачивается на один градус второй сервопривод.

И так повторяется 180 раз.

```
void loop() {
  for (pos = 0; pos <= 180; pos += 1) {
    // in steps of 1 degree
    myservo.write(pos);
     for (pos1 = 0; pos1 <= 180; pos1 += 1) {

      myservo1.write(pos1);
      delay(10);

      for (pos1 = 180; pos1 >= 0; pos1 -= 1) {
    myservo1.write(pos);
    delay(10);
  }
      }
    delay(15);
  }
```

Бесконечные циклы мы не будем рассматривать, так как достаточно самой функции **void loop()**.

6. Основы управления

6.1. DC моторы

Данные моторы имеют ряд характеристик:

- количество оборотов в секунду
- момент силы
- потребление тока

В блоке управления, для моторов, выделены два порта под цилиндрические штекеры, смотри рис. 5. Максимально возможное количество подключений моторов составляет две штуки. Для управления двигателями, зарезервированы пины на плате Arduino:

Порт	Пины на плате Arduino	Назначение
M1	D6 – скорость мотора D7 - направление	DC моторы
M2	D5 – скорость мотора D4 - направление	DC моторы

Скорость мотора варьируется от 0 до 255. Направление мотора можно менять подавая на пины значение LOW или HIGH.

Методические рекомендации

Тема: «Подключение и управление dc моторами» является одной из первых и ключевых тем в изучении робототехники детьми, после изучения среды программирования.

Цель: изучить процесс подключения dc моторов и получить знания и опыт в области управления ими с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс подключения и крепления моторов
- изучить возможности dc моторов (направления вращение оси, скорость вращения, ограничение вращение)
- получить и закрепить на практике знания, умения и навыки по основам кинематики
- получить и закрепить на практике знания, умения и навыки в области создания программ

Примеры заданий по данной теме.

Задание 1. (Уровень А)

1. Соберите установку согласно представленной инструкции «DC моторы»

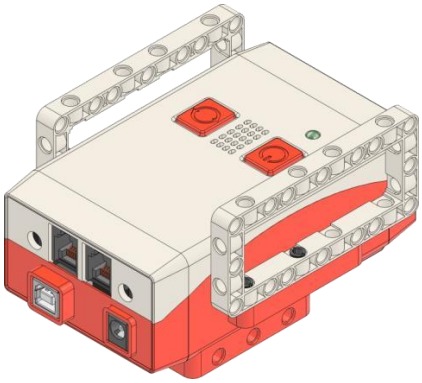


Рис. 1 Блок управления CyberBot с рамочными балками

Рис. 2 DC мотор со штифтами

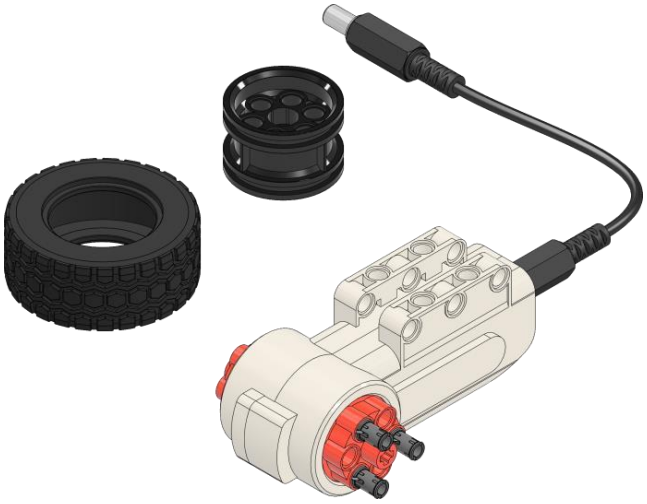


Рис. 3 Колесо для DC мотора

Рис. 4 Крепление колеса к DC мотору

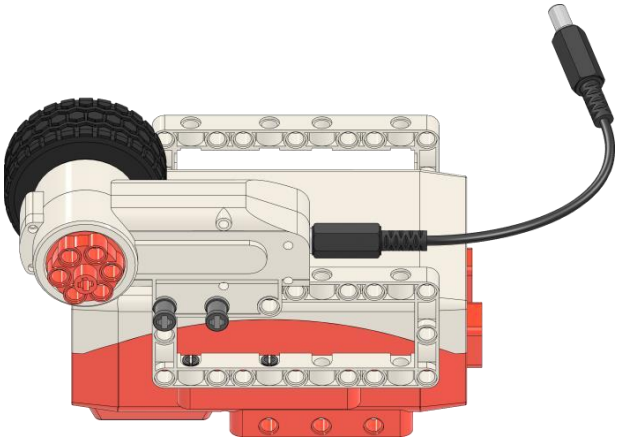


Рис. 5 Крепление DC мотора к рамке с помощью двойных штифтов

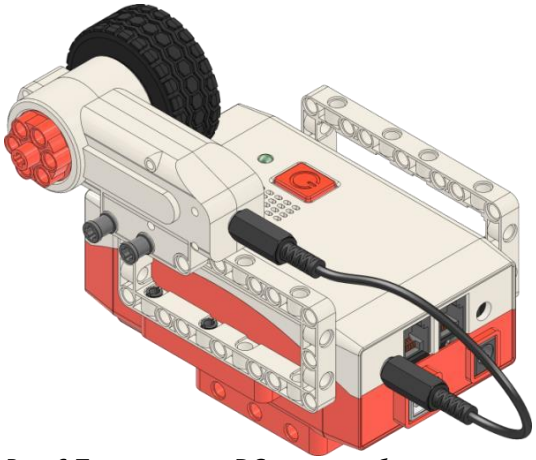


Рис. 6 Подключение DC мотора к блоку управления через порт M1

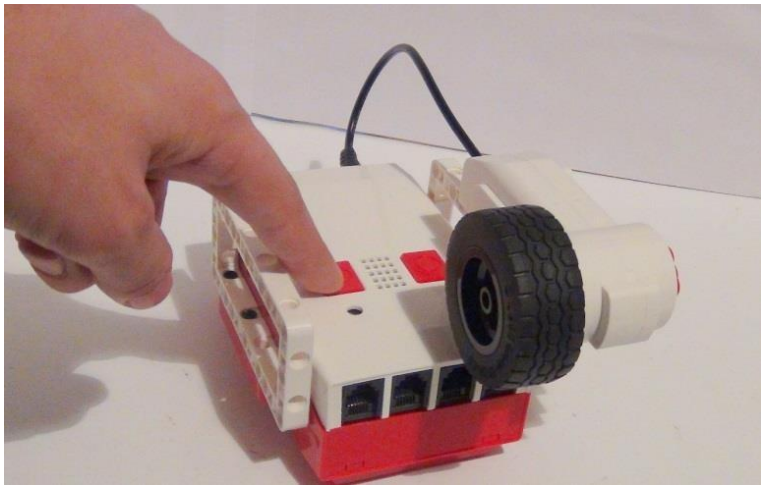


Рис. 7 Включение блока управления

Рис. 8 Крепление метки на колесо

Метка для колеса предназначена для наглядного наблюдения процесса работы мотора.

2. Откройте программную среду Arduino ide (MBlock). Составьте программу, которая включит мотор на вращение.

Пример программы представлен на рис. 9 и рис. 10

```
int ml = 7;

int vl = 6;

void setup()
{
  pinMode(ml,OUTPUT);
  pinMode(vl,OUTPUT);
  analogWrite(vl, 255);
  digitalWrite(ml, LOW);
}

void loop() {


    }
```

Рис. 9 Программа на arduino ide

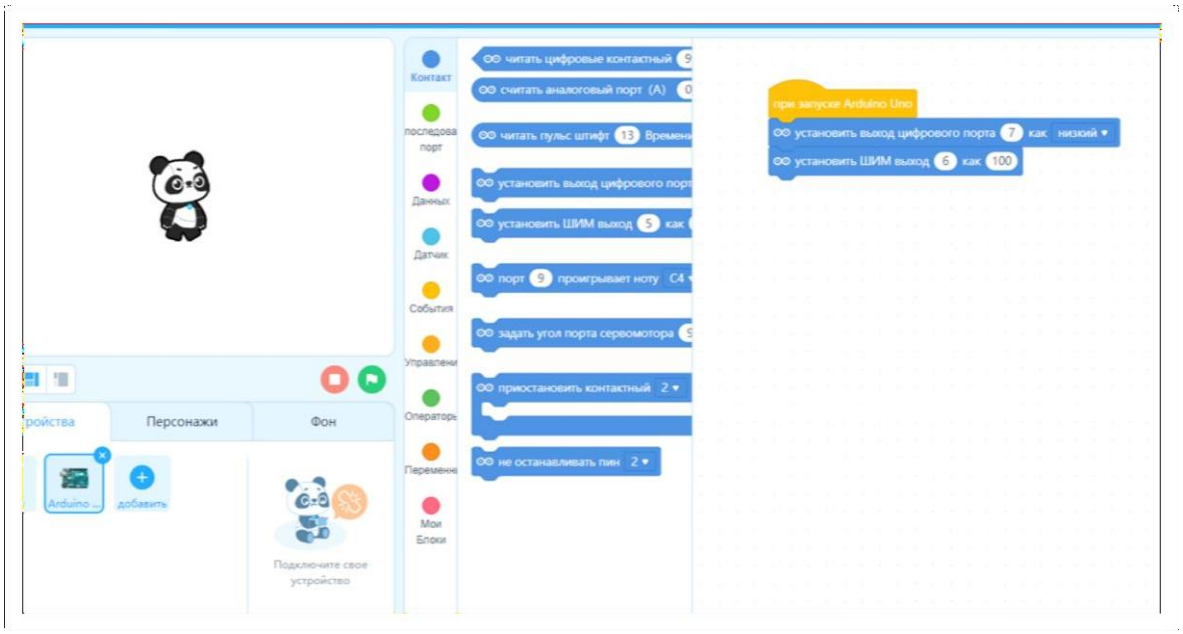


Рис. 10 Программа на MBlock 5

3. Составьте программу, которая запустит вращение мотора в противоположное направление.

Пример программы представлен на рис. 11 и рис. 12

```
int m1 = 7;

int v1 = 6;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  analogWrite(v1, 255);
  digitalWrite(m1, HIGH);

}

void loop() {


    }
```

Рис. 11 Программа на arduino ide

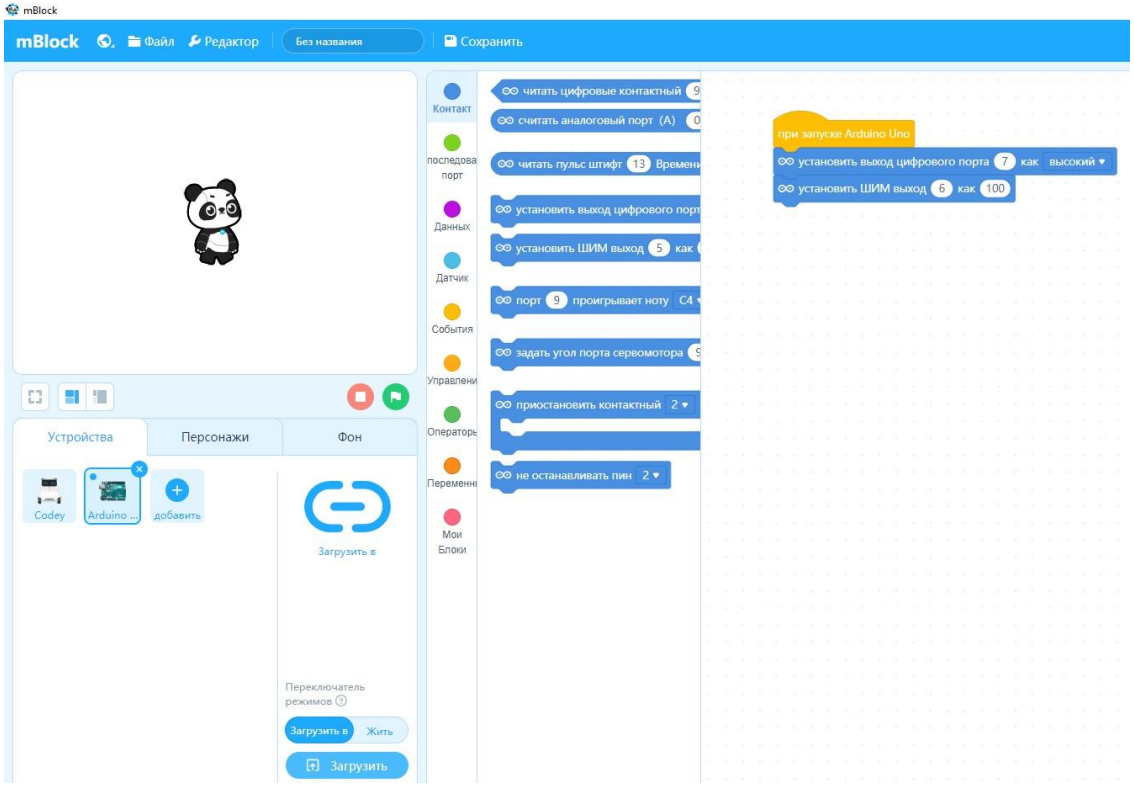


Рис. 12 Программа на MBlock 5

4. Составьте программу, которая включит мотор на вращение в интервале времени 5 секунд

Пример программы представлен на рис. 13 и рис. 14

```
int ml = 7;

int vl = 6;

void setup()
{
  pinMode(ml,OUTPUT);
  pinMode(vl,OUTPUT);
  analogWrite(vl, 255);
  digitalWrite(ml, LOW);
  delay(5000);
  analogWrite(vl, 0);
}

void loop() {


    }
```

Рис. 13 Программа на arduino ide

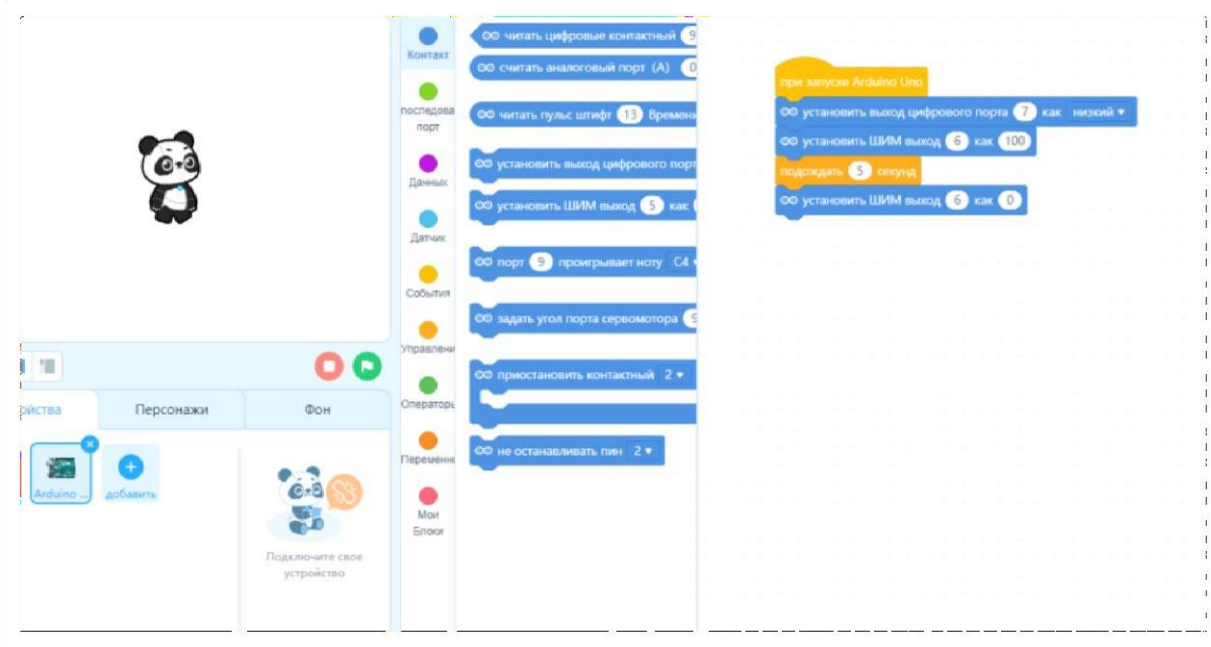


Рис. 14 Программа на MBlock 5

5. Составьте программу, которая будет изменять направление вращения колеса на моторе с интервалом в 5 секунд.

Пример программы представлен на рис. 15 и рис. 16 и рис. 15_a и рис. 16_a

```
int ml = 7;

int vl = 6;

void setup()
{
  pinMode(ml,OUTPUT);
  pinMode(vl,OUTPUT);
  analogWrite(vl, 255);
  digitalWrite(ml, HIGH);
  delay(5000);
  digitalWrite(ml, LOW);
  delay(5000);
  analogWrite(vl, 0);
}

void loop() {


    }
```

Рис. 15 Программа на arduino ide линейная

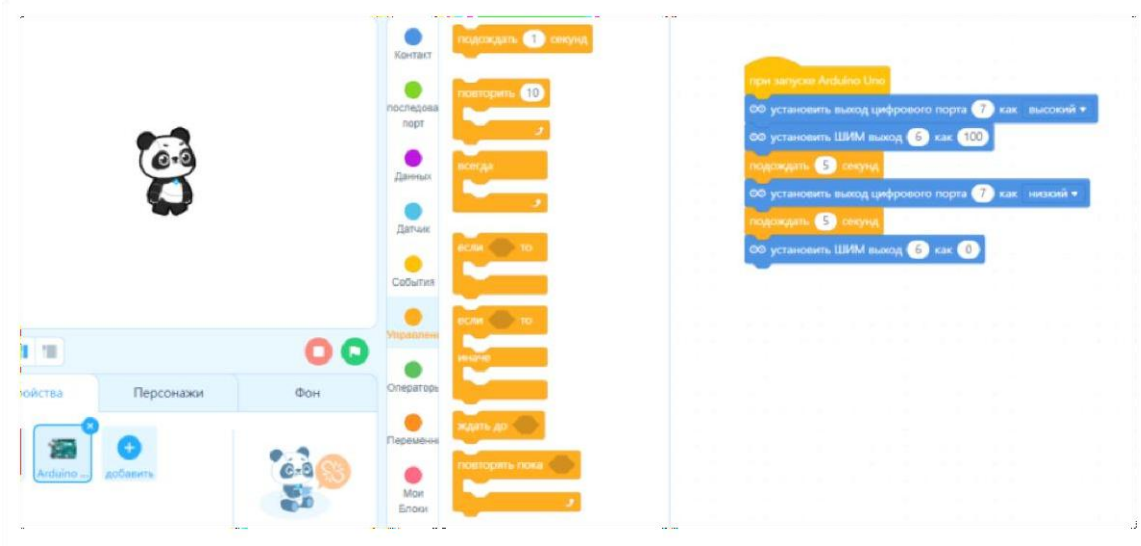


Рис. 16 Программа на MBlock 5

```
int ml = 7;

int vl = 6;

void setup()
{
  pinMode(ml,OUTPUT);
  pinMode(vl,OUTPUT);

  analogWrite(vl, 0);
}

void loop() {

analogWrite(vl, 255);
  digitalWrite(ml, HIGH);
  delay(5000);
  digitalWrite(ml, LOW);
  delay(5000);
  analogWrite(vl, 0);
    }
```

Рис. 15_а Программа на arduino ide циклическая

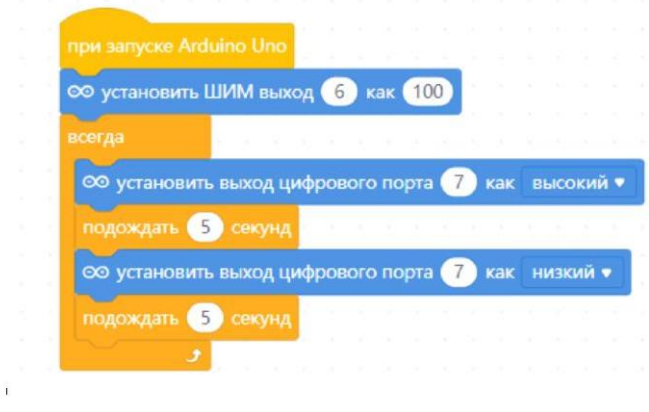


Рис. 16_а Программа на MBlock 5

6. Подключите второй dc мотор согласно инструкции.

Рис. 17 Второй dc мотор со штифтами

Рис. 18 Крепление колеса ко второму dc мотору

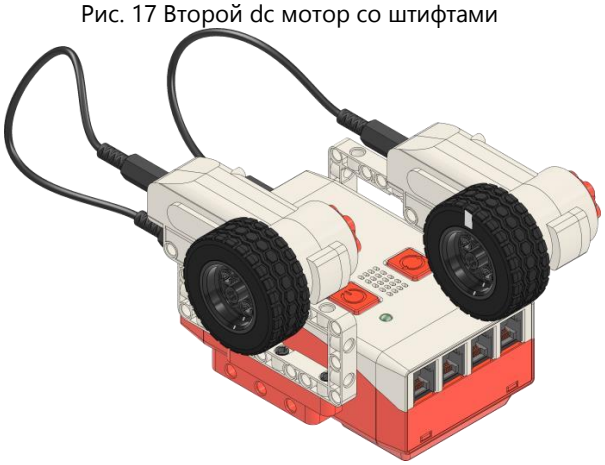


Рис. 19 Крепление второго dc мотора к блоку управления

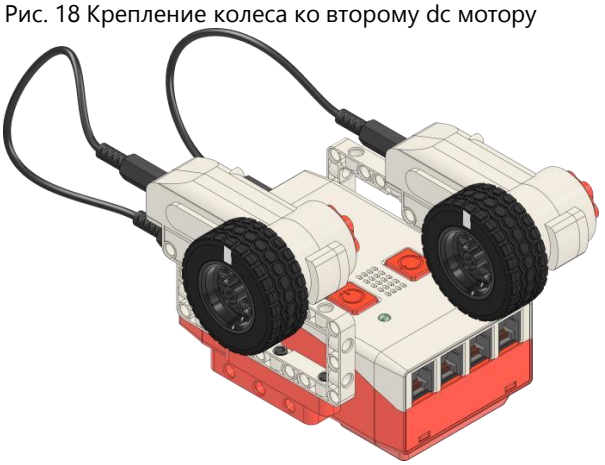


Рис. 20 Крепление метки ко второму dc мотору

7. Составьте программу, которая, запускает два мотора в одно направление

Пример программы представлен на рис. 21 и рис. 22

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 255);
  analogWrite(v2, 255);
  digitalWrite(m1, LOW);
  digitalWrite(m2, HIGH);
}

void loop() {


    }
```

Рис. 21 Программу управление моторами в одну сторону

Рисунок 22 Программа на MBlock 5

8. Составьте программу, которая, запускает два мотора вращать колёса в разные направления.

Пример программы представлен на рис. 23 и рис. 24

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 255);
  analogWrite(v2, 255);
  digitalWrite(m1, LOW);
  digitalWrite(m2, LOW);
}

void loop() {


    }
```

Рис. 23 Программу управление вращением моторов в противоположные стороны

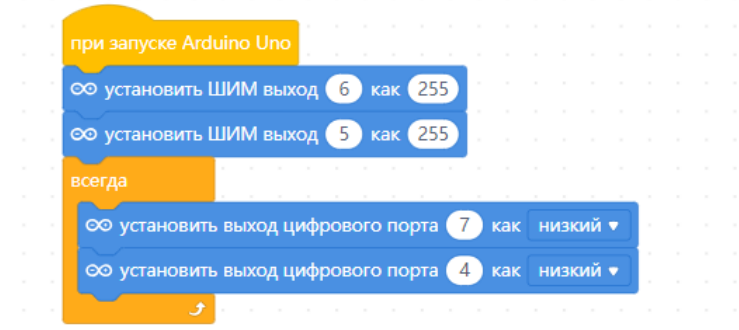


Рис. 24 Программа на MBlock 5

9. Составьте программу, которая, запускает два мотора, так, что скорость вращения колеса второго мотора в два раза меньше скорости первого.

Пример программы представлен на рис. 25 и рис. 26

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 255);
  analogWrite(v2, 127);
  digitalWrite(m1, LOW);
  digitalWrite(m2, HIGH);
}

void loop() {


    }
```

Рис. 25 Программу управление вращением моторов с разной скоростью

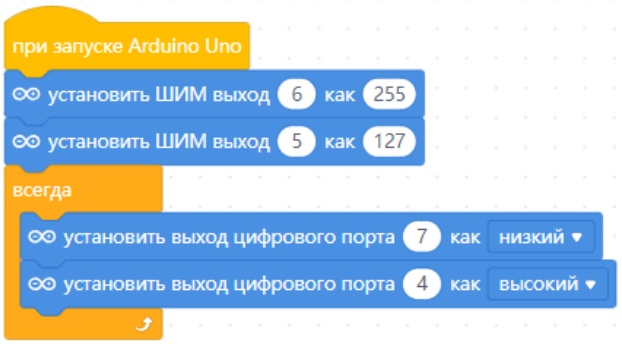


Рис. 26 Программа на MBlock 5

10. Посчитайте количество оборотов колеса на втором моторе за интервал времени в пять секунд. Если данный мотор работал в половину мощности, то сколько оборотов в одну секунду совершает мотор в полную мощность?

(Примерный ответ 4 оборота в секунду)

Задание 2. (Уровень А, В)

1. Соберите установку согласно представленной инструкции «Мобильный робот»(А)

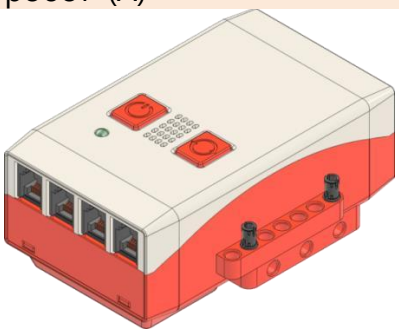


Рис. 27 Мобильный робот_1

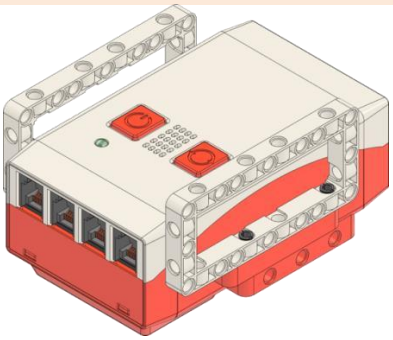


Рис. 28 Мобильный робот_1

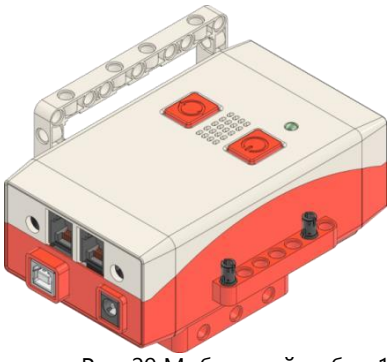


Рис. 29 Мобильный робот_1

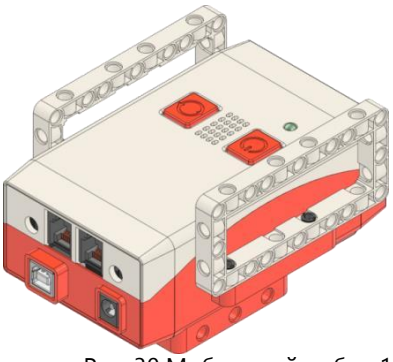


Рис. 30 Мобильный робот_1

Рис. 31 Мобильный робот_1

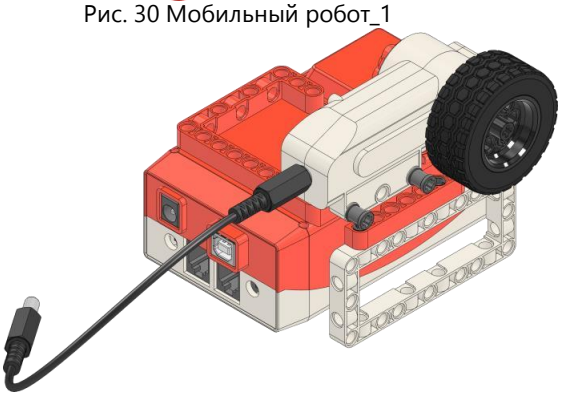


Рис. 32 Мобильный робот_1

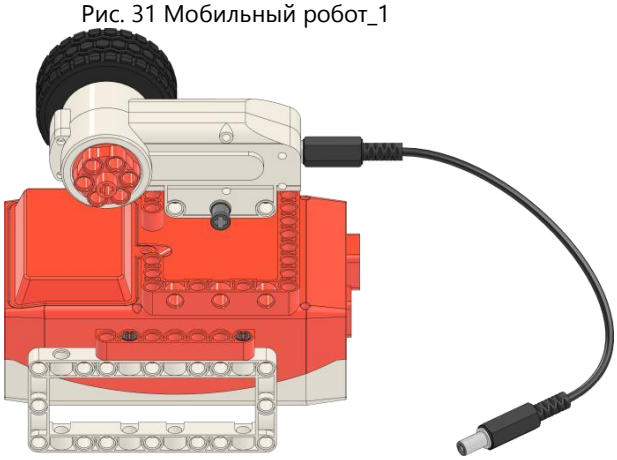


Рис. 33 Мобильный робот_1

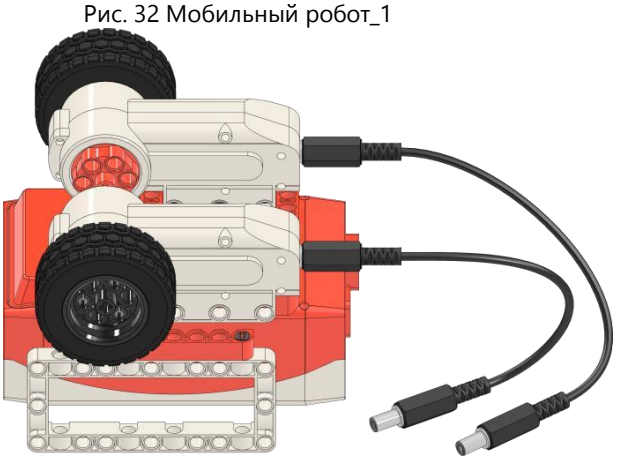


Рис. 34 Мобильный робот_1

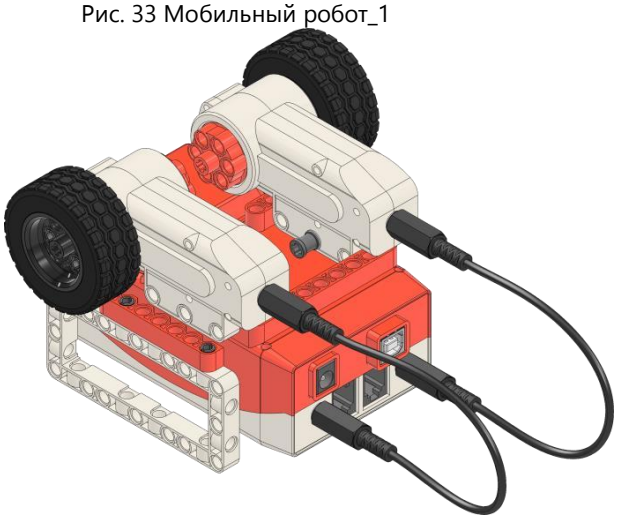


Рис. 35 Мобильный робот_1

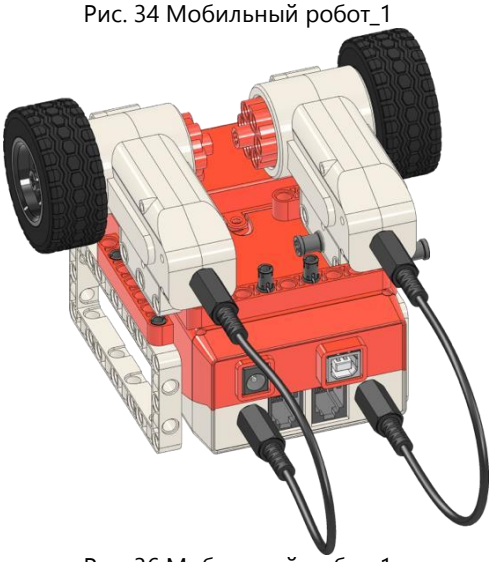


Рис. 36 Мобильный робот_1

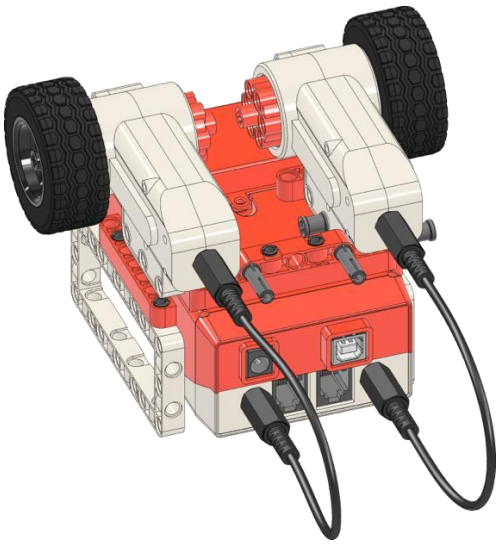


Рис. 37 Мобильный робот_1

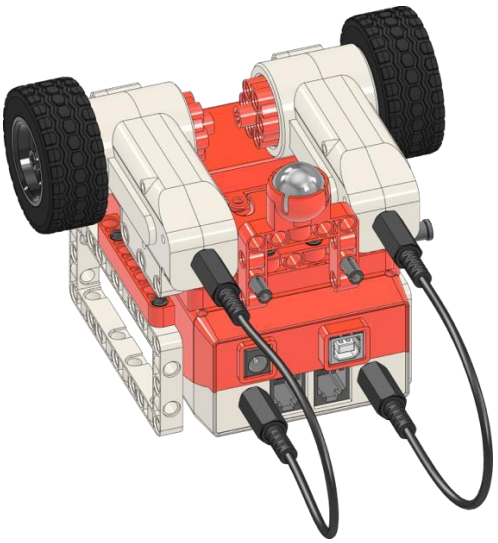


Рис. 38 Мобильный робот_1

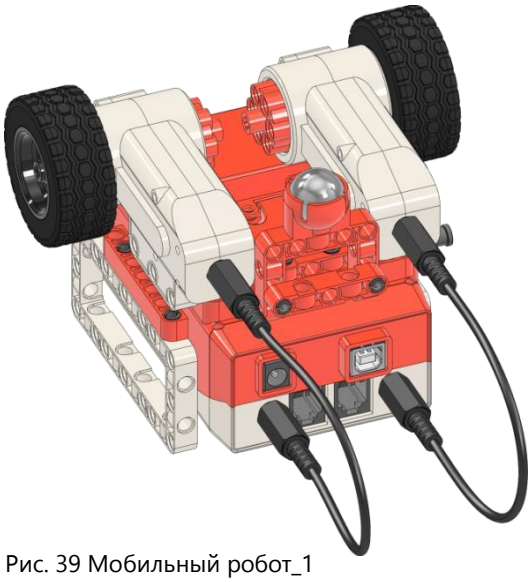


Рис. 39 Мобильный робот_1

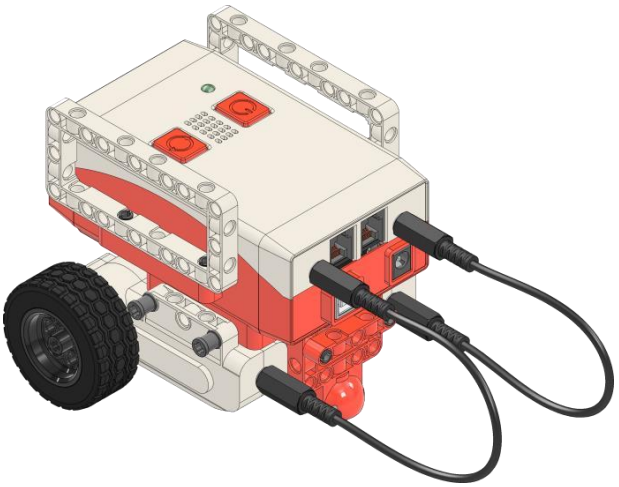


Рис. 40 Мобильный робот_1

2. Создайте программу, с помощью которой робот сможет проехать по прямой три секунды с мощностью 100 (А).

Пример программы рис. 41 и 42

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 100);
  analogWrite(v2, 100);
  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);

  delay(3000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
}

void loop() {


    }
```

Рис. 41 Программа управление движением робота три секунды в одном направлении, arduino ide

Рис. 42 Программа на MBlock 5

3. Создайте программу, с помощью которой робот сможет развернуться на $90^0, 180^0, 270^0, 360^0$ вокруг своей оси (В)

Пример программы для 90^0, рис. 43 и рис.44

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 100);
  analogWrite(v2, 100);
  digitalWrite(m1, LOW);
  digitalWrite(m2, LOW);

  delay(425);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
}

void loop() {


    }
```

Рис. 43 Программа управление поворотом робота вокруг своей оси на $\mathbf{90^0}$ arduino ide

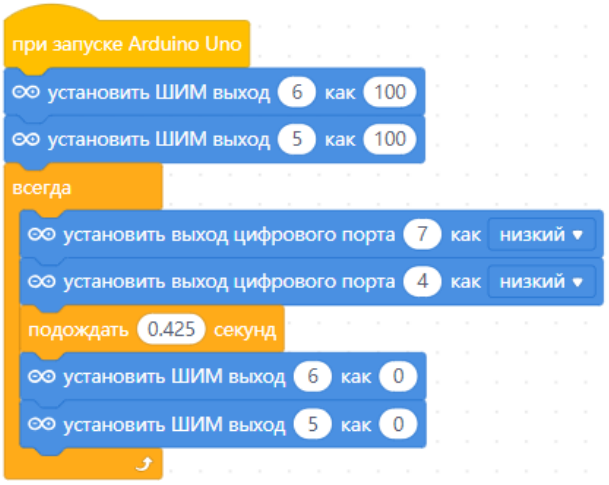


Рис. 44 Программа на MBlock 5

4. Решите задачу, прохождения роботом пути в 1 метр. (Погрешность до 5 сантиметров допускается) (В)

Пример решения

Так как на моторах не установлен энкодеры, то для точного позиционирования робота нам нужно знать:

- радиус колеса
- количество оборотов в секунду при заданной мощности

В наборе, на шине колеса написан его диаметр $D = 2 * R = 49{,}5$ мм, отсюда следует, что радиус колеса равен 24,75 мм.

Вычислим длину окружности, т.е. путь, который пройдёт робот за один оборот.

$L = 2 * \pi * R$, где $\pi \approx 3.14$.

Нетрудно вычислить $L = 155{,}43$ мм.

Зная значение пути пройденного за один оборот, можно вычислить, сколько таких оборотов нужно совершить колесу, чтобы робот прошёл путь в 1 метр или 1000 мм.

$$N = \frac{s}{L} = \frac{1000}{155{,}43} = 6{,}43$$

N – количество оборотов. Из задания 1 пункт 11 следует, что при максимальной мощности в 255 колесо совершает 4 оборота в секунду, а при мощности в 127 соответственно 2 оборота в секунду.

При мощности в 255, $t = \frac{N}{n} = \frac{6{,}43}{4} \approx 1{,}61$ секунды, а при мощности в 127 время составит $t = \frac{N}{n} = \frac{6{,}43}{2} \approx 3{,}215$ секунды.

Опираясь на данные результаты, составим программу, рис. 45

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 127);
  analogWrite(v2, 127);
  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);

  delay(3215);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
}

void loop() {

}
```

Рис. 45 Программа для преодоления роботом пути в 1 метр, arduino ide

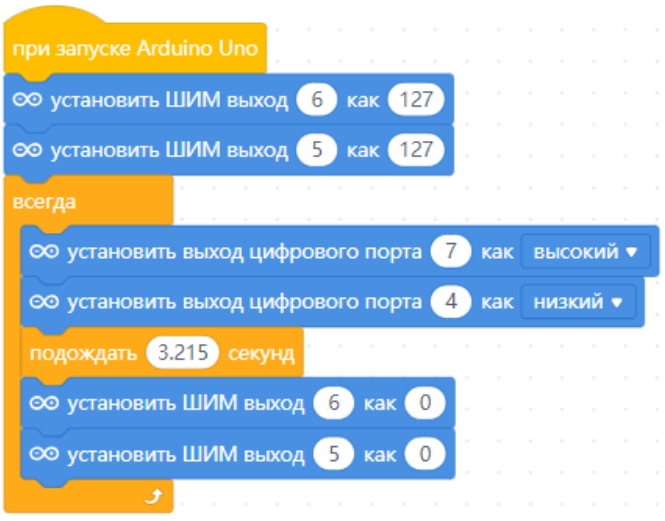


Рис. 46 Программа на MBlock 5

Альтернативное решение.

Мы брали значение количества оборотов, опираясь на наблюдения. Можно рассчитать путь другим решением. Составим программу рис.47

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 127);
  analogWrite(v2, 127);
  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);

  delay(1000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
}

void loop() {


    }
```

Рис. 47 Программа управления роботом одну секунду, arduino ide

Рис. 48 Программа на MBlock 5

- Загрузив программу в блок управления, отметим начальное положение робота.
- Запустим блок и отметим точку, до которой доехало колесо робота, именно точка касания колеса с поверхностью.
- Измерим расстояние между отмеченными точками.

Расстояние между точками в нашем случае будет, примерно 284 мм (всё будет зависеть от характеристик двигателей, всегда существует погрешность измерения).

Из этого значения вытекает, что количество оборотов в одну секунду при мощности в 127 будет составлять

$$N = \frac{s}{L} = \frac{284}{155{,}43} \approx 1{,}83$$

Следовательно, при мощности в 255, количество оборотов увеличится вдвое 3,65

Отсюда вытекает, что более точное время будет $t = \frac{N}{n} = \frac{6{,}43}{3{,}65} \approx 1{,}76$ секунды, при мощности в 255, и $t = \frac{N}{n} = \frac{6{,}43}{1{,}83} \approx 3{,}51$ секунды.

Подкорректируем программу и проверим на практике.

Проведите сравнительный анализ первого и второго варианта вычисления. Какой из результатов оказался точнее.

6.1.1. Одометрия робота

Мобильный робот или транспортное средство имеет шесть степеней свободы, выраженных в изменении положения в пространстве по **x**, **y**, **z**, а также крен, тангаж и рыскание. Крен относится к боковому вращению, тангаж относится к переднему и заднему вращению(повороту) и рыскание (называется курс или направление) означает направление, в котором робот движется в плоскости xy.

Робот движется в плоскости XY, следовательно в 2D формате представляют его в основном по X, Y и θ, где θ- угол поворота робота. Этой информации достаточно, чтобы описать положение мобильного робота данной конструкции.

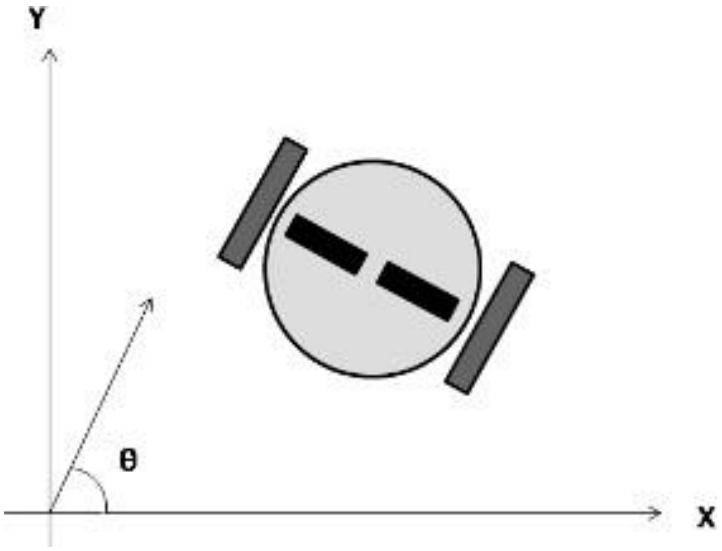


Рис. 49 Кинематика робота. Взят из книги «Изучение робототехники с помощью Python»

Движение робота можно контролировать путем регулирования скорости двух независимо управляемых двигателей с левой и правой стороны платформы.

Уравнения кинематики передней части робота с дифференциально-приводной системой используется для решения следующей проблемы:

если робот стоит в положении (X, Y, θ) в момент времени t, то можно определить положение (X', Y', θ') за промежуток времени t + δt с учетом контроля параметров ϑl - *скорость левого* и ϑr- *скорость правого колеса*.

Эту технику можно использовать в роботе для прохода по определенной траектории.

Объяснение уравнений кинематики для переднего привода.

Мы можем начать с формулирования решения для движения вперед. Следующий рисунок является иллюстрацией кинематики одного из колес робота:

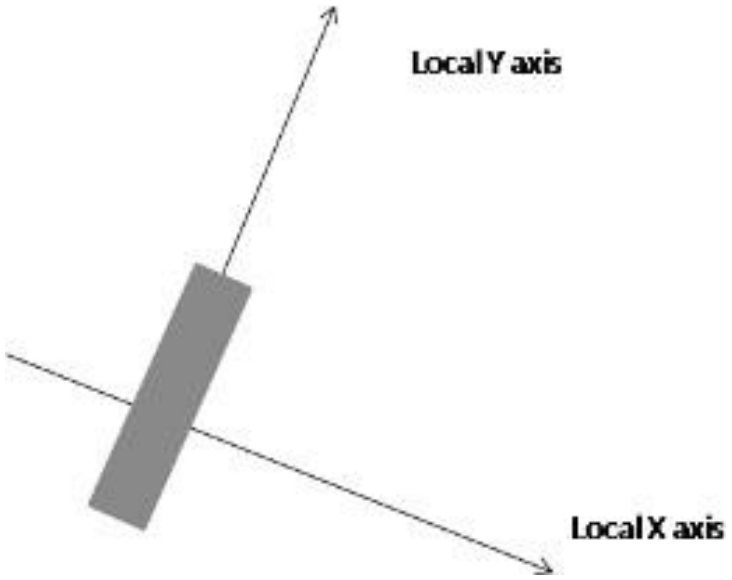


Рис. 50 Кинематика робота. Взят из книги «Изучение робототехники с помощью Python»

Одно колесо робота вращается вдоль локальной оси Y.

Движение вокруг оси Y называется креном; все остальное можно рассматривать как скольжение. Предположим, что проскальзывания в данном случае не происходит. Когда колесо совершает один полный поворот, обод колеса перемещается на расстоянии 2πR, где R-радиус колеса. Будем считать, что движение двухмерное. Это означает, что поверхность плоская и ровная. Когда робот собирается выполнить движение качения, робот должен вращаться вокруг точки, лежащей на продолжении осей левого и правого колеса. Дело в том, что точка, вокруг которой вращается робот, имеет название как ICC - мгновенный центр кривизны (мгновенный центр скоростей). На следующей схеме показана конфигурация дифференциальной передачи колеса с условиями ICC:

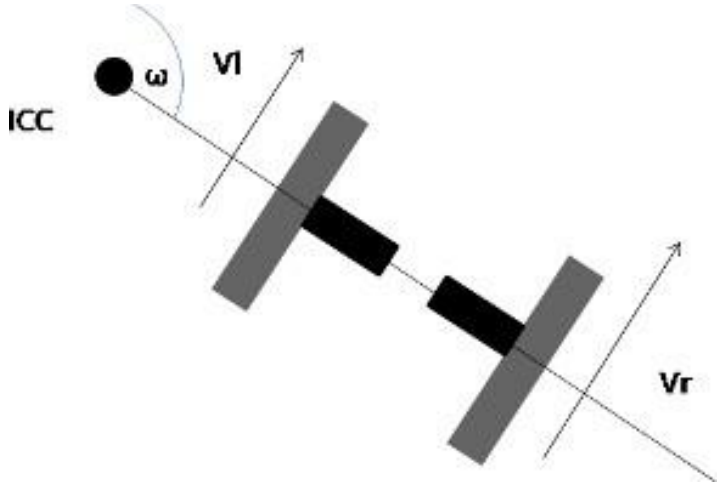


Рис. 51 Кинематика робота. Взят из книги «Изучение робототехники с помощью Python»

Центральным понятием для вывода кинематических уравнений – это **ω** угловая скорость робота. Каждое колесо робота вращается вокруг ICC по кругу

с радиусом колеса **r**. Скорость колеса **v = 2 πr / T**, где **T** - время, затраченное на завершение полного поворот вокруг ICC. А **ω** угловая скорость определяется как **2 π / T** и,

как правило, измеряется в радианах (градусы) в секунду. Объединение уравнений для **v** и ω дает **ω= 2 π / T**.

$$\nu = r\omega$$

Модель кинематики робота показана на следующем рисунке:

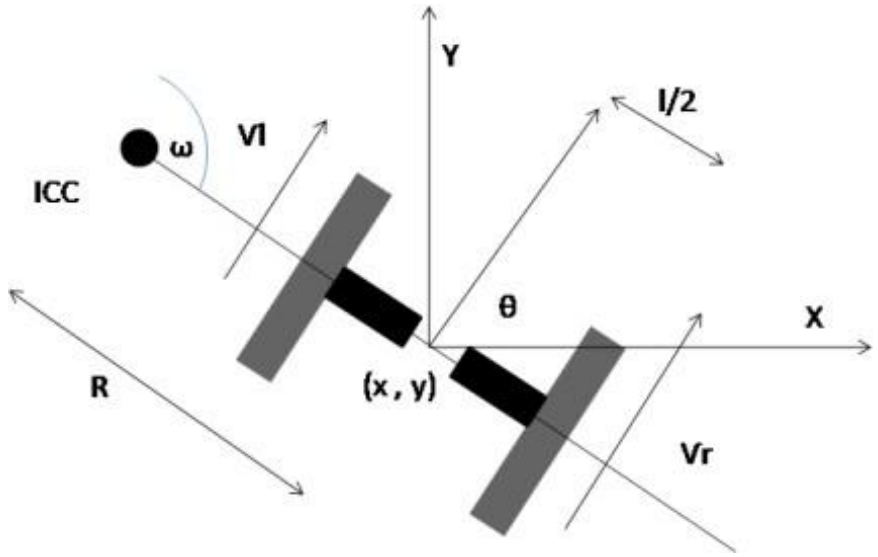


Рис. 52 Кинематика робота. Взят из книги «Изучение робототехники с помощью Python»

Если применить предыдущее уравнение на оба колеса, результат будет одинаковым, то есть ω:

Где R-расстояние между ICC и средней точкой оси, проходящей через колеса, а l - длина оси колеса. После выражения ω и R, мы получим следующий вид записи:

Предыдущие уравнения полезны для решения поставленной задачи кинематики. Предположим, что робот двигается с угловой скоростью в течение δt секунд, тогда может изменяться траектория движения робота или его направление:

$$\theta' = \omega \delta t + \theta \qquad (6)$$

Где центр вращения ICC определяется базовой тригонометрией, как:

$$ICC = [ICC_x, ICC_y] = [x - R\sin\theta, y + R\cos\theta] \qquad (7)$$

Вращение робота на угол ωδt в градусах вокруг ICC

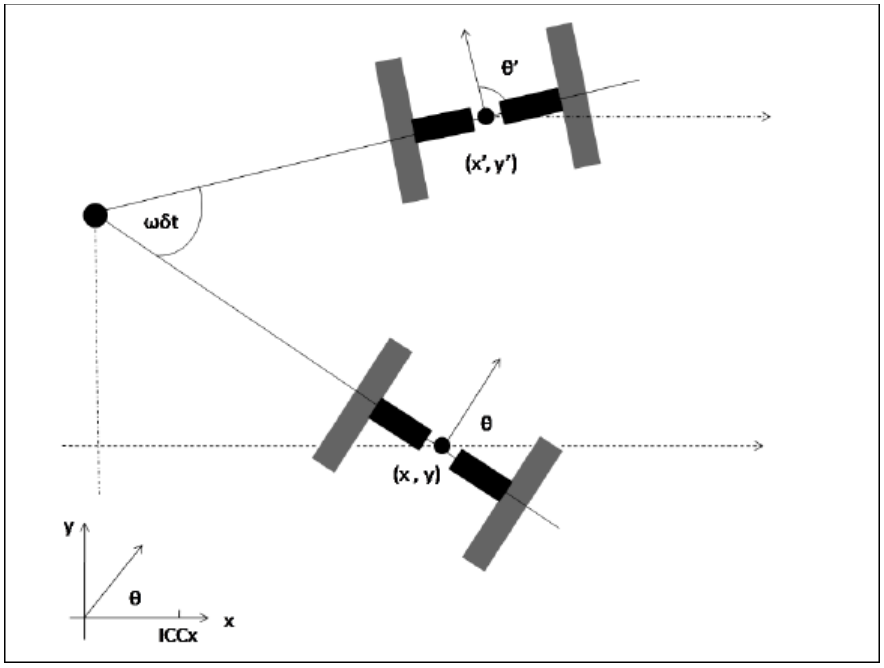


Рис. 53 Кинематика робота. Взят из книги «Изучение робототехники с помощью Python»

С учетом стартовой позиции (x, y), то новое положение (x', y') может быть вычислено с помощью 2D матрицы вращения. Вращение вокруг ICC с угловой скоростью в течение δt секунд выводит следующую позицию для времени t + δt:

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} cos(\omega\delta t) & -sin(\omega\delta t) \\ sin(\omega\delta t) & cos(\omega\delta t) \end{pmatrix} \begin{pmatrix} x - ICC_x \\ y - ICC_y \end{pmatrix} + \begin{pmatrix} ICC_x \\ ICC_y \end{pmatrix} \quad (8)$$

Новое положение (x', y' и θ') можно вычислить из уравнений (6) и (8),

учитывая **ω**, **δt**, и **R**. **ω** может быть вычислена из уравнения (5); скорости Vr и Vl часто трудно измерить точно.

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} \cos(\omega\delta t) & -\sin(\omega\delta t) & 0 \\ \sin(\omega\delta t) & \cos(\omega\delta t) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x\text{-}ICC_x \\ y\text{-}ICC_y \\ \theta \end{pmatrix} + \begin{pmatrix} ICC_x \\ ICC_y \\ \omega\delta t \end{pmatrix} \quad (12)$$

Полученное кинематическое уравнение в основном зависит от конструкции и геометрии робота. Различные конструкции могут привести к различным уравнениям.

Существует закономерность

$$\frac{R_l}{R_r} = \frac{\vartheta_l}{\vartheta_r}$$

Где $R_l -$ расстояние от ICC до левого колеса,

$R_r -$ расстояние от ICC до правого колеса

$\vartheta_l -$ скорость левого колеса $\vartheta_r -$ скорость правого колеса

6.1.2. Инверсная кинематика

Уравнение обычной кинематики определяет новое положение робота на заданной скорости колеса при поступательном их вращении. Теперь мы можем подумать об обратной проблеме.

Поставим робота в момент времени t в положение (x, y, θ) и определим параметры V левого и V правого колеса так, что положение во время t + δt есть (x', y', θ').

В дифференциальном приводе, эта проблема не может иметь решения, потому что этот вид робота не может быть перемещен в любую позу, простой установкой скорости колеса. Это потому что робот имеет ограничения и поэтому он называется неголономным роботом, но эта проблема может быть решена,

потому что такие виды роботов могут перемещаться в любое положение.

У неголономных роботов есть несколько способов увеличить ограниченную подвижность если мы позволим неодинаковую последовательность работы V-левого, V-правого колеса. Если вставить значения из уравнений (12) в (15), мы можем определить некоторые особые случаи управления:

- Если V-левого = V-правого, => R = ∞, в свою очередь это значит, что робот движется прямолинейно и θ остаётся неизменной.
- Если V-левого = - V-правого, => R = 0 и

ICC = [ICCx, ICCy] = [x,y] => x' = x, y' = y, θ' = θ + ωδt =>: это означает, что робот вращается вокруг ICC, т.е. ∀θ, при постоянных [x, y]

Для более подробной информации по кинематике, смотрите:
http://www8.cs.umu.se/~thomash/reports/KinematicsEquationsForDifferentialDriveAndArticulatedSteeringUMINF-11.19.pdf

Задание 1. (Уровень С)

Согласно теории одометрии робота, создайте программу, благодаря которой робот будет совершать движение по кругу. Поясните своими словами от чего будет зависеть радиус данного круга.

Пример решения.

Из теории по одометрии робота, следует, то что изменяя значение скорости моторов, мы изменяем траекторию движения робота. Чем больше разница в значениях скоростей, тем более «круче» будет поворот.

Создадим программу, в которой будем запускать моторы с разной мощностью длительностью в три секунды. Соотношение скоростей моторов поставим так:

а) $V_1 = 2V_2$

б) $V_1 = 3V_2$

в) $V_1 = 4V_2$

г) $V_1 = 10V_2$

д) $V_1 = -V_2$

```
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
}

void loop() {
  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);
  analogWrite(v1, 255);
  analogWrite(v2, 127);
  delay(3000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(1000);

  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);
  analogWrite(v1, 240);
  analogWrite(v2, 80);
  delay(1000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(1000);

  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);
  analogWrite(v1, 240);
  analogWrite(v2, 60);
  delay(3000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(1000);

  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);
  analogWrite(v1, 250);
  analogWrite(v2, 50);
  delay(3000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(1000);


  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);
  analogWrite(v1, 250);
  analogWrite(v2, 25);
  delay(3000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(1000);


digitalWrite(m1, LOW);
  digitalWrite(m2, LOW);
  analogWrite(v1, 100);
  analogWrite(v2, 100);
  delay(3000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(1000);
}
```

Рис. 54 Программа управления роботом для движения по кругу, arduino ide

Рис. 55 Программа на MBlock 5

Если запустить робота с данной программой, то наглядно видно, как изменяется мгновенный центр скоростей.

Задание 2. (Уровень С)

Согласно теории одометрии, составьте программу, которая при заданных скоростях моторов 255 и 127 запустит движение робота ровно на один оборот вокруг ICC**.**

Пример решения.

Согласно теории, при заданных скоростях ICC будет находиться на расстоянии l – расстояние между центрами колёс . При данных мощностях одно колесо будет совершать примерно четыре оборота в секунду, а другое два оборота в секунду. Зная путь, которое проходит колесо при полном обороте, нетрудно вычислить скорость колёс

$v_2 = 155{,}43 * 2 = 310{,}86$ мм/с , $v_1 = 155{,}43 * 4 = 621{,}72$ мм/с

Вычислим, какой путь должно пройти колесо с меньшей скоростью.

$R = 120$ мм - расстояние от колеса с меньшей скоростью до ICC, согласно теории. Данное значение взято согласно рисунку. 32.

Из этого следует

$L = 2 * \pi * R = 2 * 3.14 * 120 = 753{,}6$ мм

Зная скорость и путь, нетрудно вычислить время движения.

$$t = 2{,}424\ c.$$

Осталось воспользоваться результатом для программы и проверить правильность вычислений.

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);

  analogWrite(v1, 0);
  analogWrite(v2, 0);
}

void loop() {
  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);
  analogWrite(v1, 255);
  analogWrite(v2, 127);
  delay(2424);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(1000);


}
```

Рис. 56 Программа управления роботом для движения по кругу, arduino ide

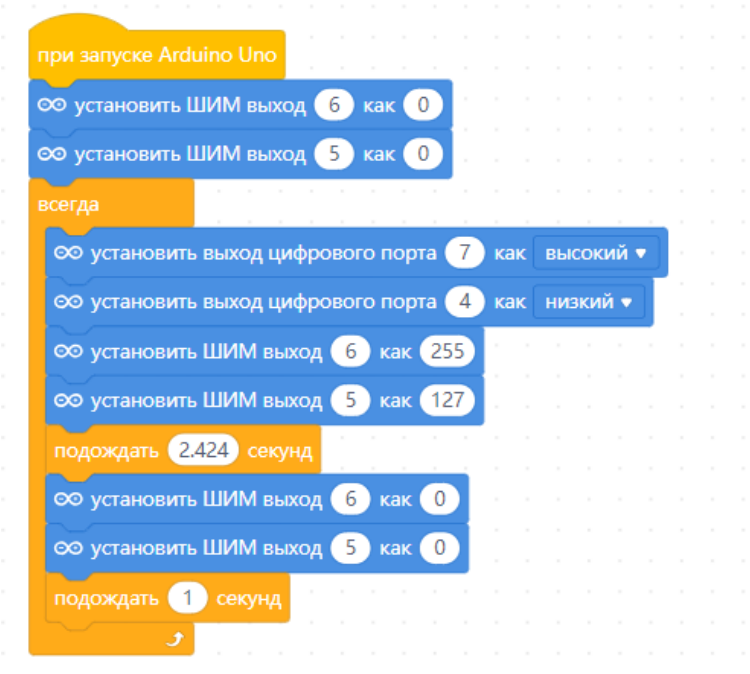


Рис. 57 Программа на MBlock 5

6.2. Сервопривод

Данные моторы имеют ряд характеристик:

- градус поворота
- момент силы
- потребление тока

Сервопривод можно подключить к первому, второму, четвёртому, пятому и третьему портам блока управления. Для управления приводом, зарезервированы пины на плате Arduino:

Порт	Пин
1	D11
2	D3
4	D1
5	D12
6	D8

Вращение данного привода варьируется от 0 до 180^0.

Методические рекомендации.

Тема: «Подключение и управление сервопривода»

Цель: изучить процесс подключения сервопривода и получить знания и опыт в области управления им с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс подключения и крепления сервомотора
- изучить возможности сервопривода (поворотные возможности, скорость вращения, точность поворота)
- получить и закрепить на практике знания, умения и навыки по основам геометрии.
- получить и закрепить на практике знания, умения и навыки в области создания программ

Примеры заданий по данной теме.

Сервопривод необходим для точного перемещения.

Задание 1 (Уровень А)

1. Соберите конструкцию согласно инструкции

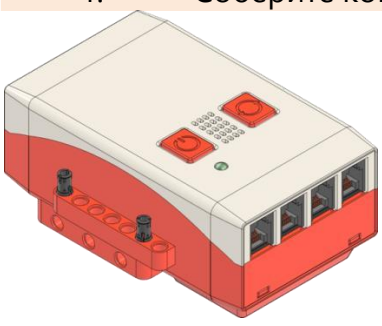


Рис. 58 Управление сервоприводом

Рис. 59 Управление сервоприводом

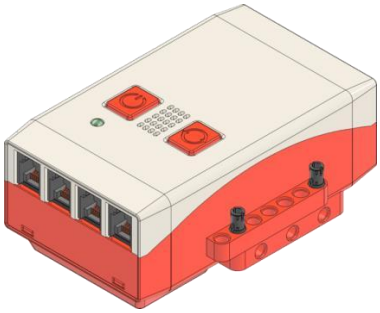


Рис. 60 Управление сервоприводом

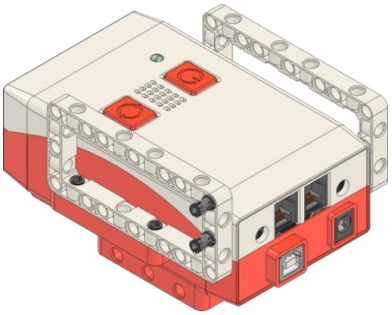


Рис. 61 Управление сервоприводом

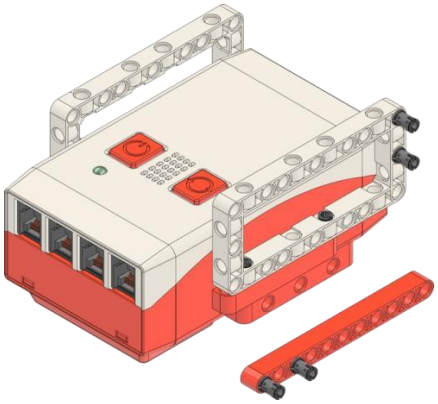


Рис. 62 Управление сервоприводом

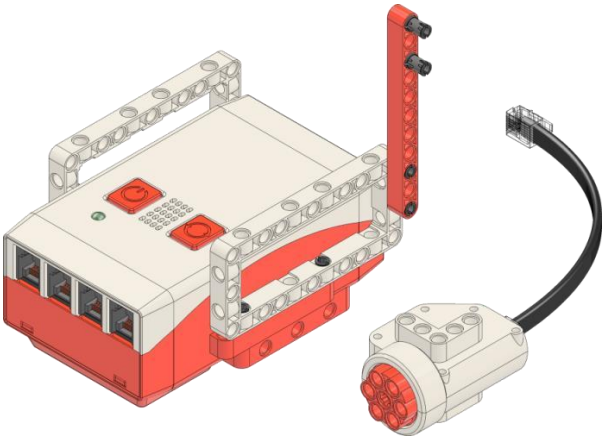


Рис. 63 Управление сервоприводом

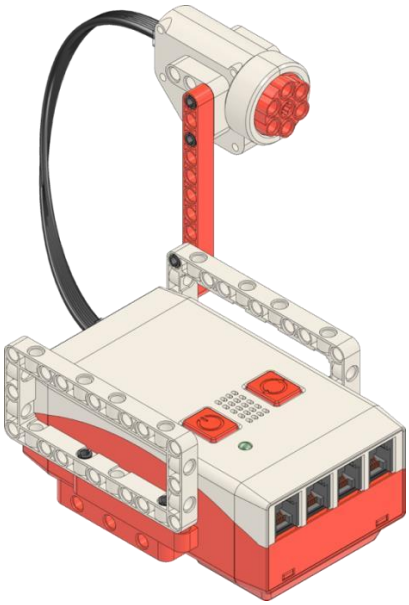


Рис. 64 Управление сервоприводом

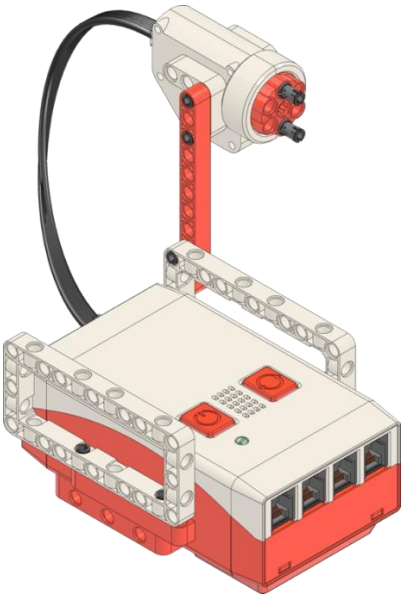


Рис. 65 Управление сервоприводом

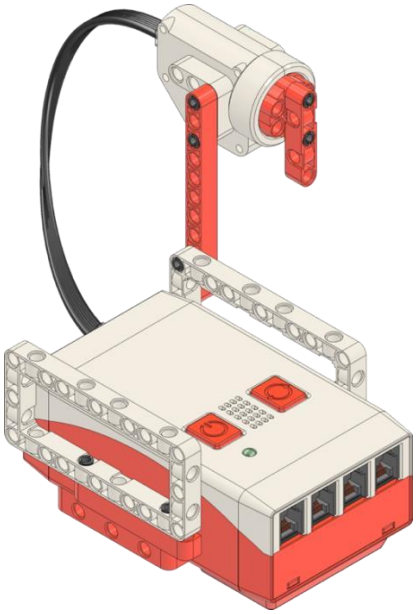


Рис. 66 Управление сервоприводом

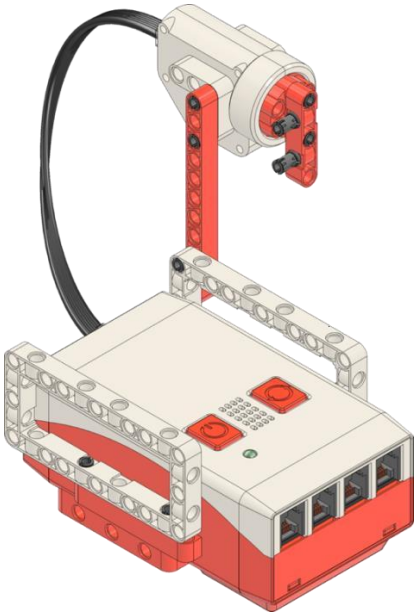


Рис. 67 Управление сервоприводом

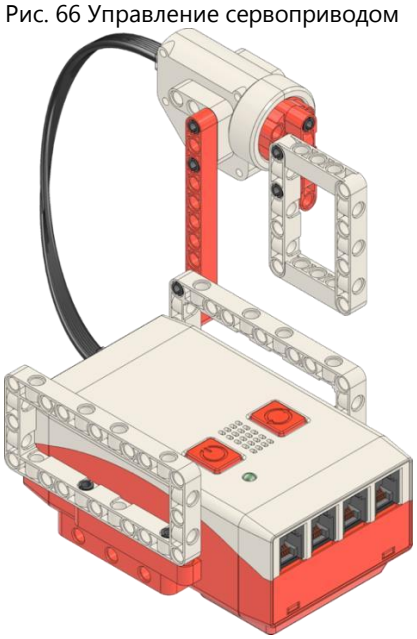


Рис. 68 Управление сервоприводом

2. Составьте программу, которая позволит изменять положение оси сервопривода от 0 до 180^0

Пример решения.

Для создания программы, необходимо воспользоваться встроенной библиотекой Servo.h

```
#include <Servo.h>

Servo myservo;

void setup() {
  myservo.attach(11);

}

void loop() {

    myservo.write(0);
    delay(1000);


    myservo.write(180);
    delay(1000);


}
```

Рис. 69 Программа управления сервоприводом

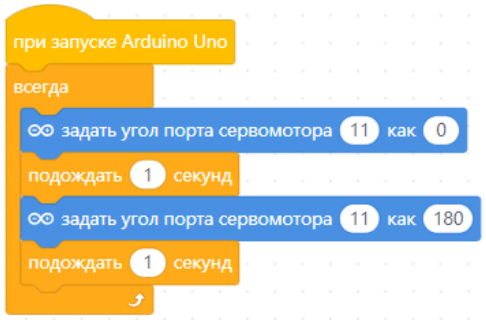


Рис. 70 Программа на MBlock 5

3. Составьте программу, которая управляет плавным поворотом оси сервопривода.

Пример решения.

```
#include <Servo.h>

Servo myservo;

void setup() {
  myservo.attach(11);

}

void loop() {
  for (int i=0; i<180; i++ ){
    //myservol.write(i);
    myservo.write(i);
    delay(10); }

  for (int i=180; i>0; i-- ){
   // myservol.write(i);
    myservo.write(i);
    delay(10); }

 }
```

Рис. 71 Программа управления сервоприводом

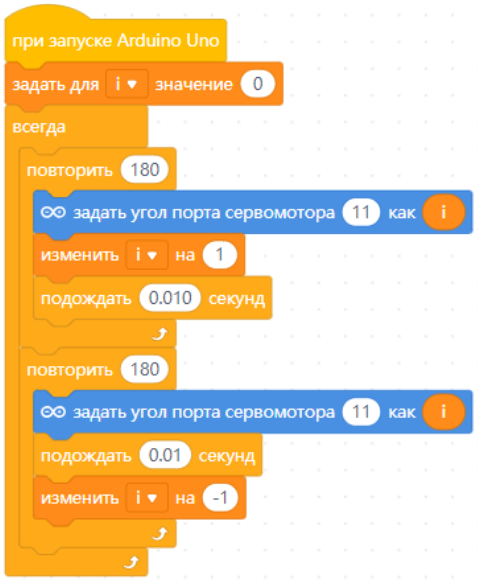


Рис. 72 Программа на MBlock 5

Задание 2 (Уровень В)

Используя знания из тригонометрии, вычислите угол поворота, согласно представленному рисунку 50. Проверьте ваш результат в программе по управлению сервоприводом.

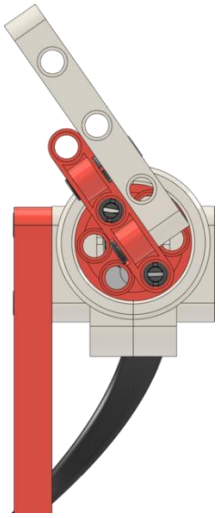
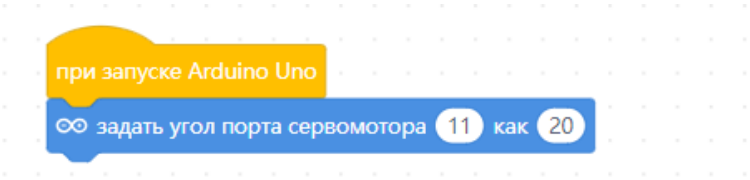


Рис. 73 Вычисление угла поворота

Пример решения.

Указания для учителя

Первым делом необходимо откалибровать положение сервопривода, так, чтобы поворотный элемент смотрел ровно перпендикулярно. Для этого может потребоваться указать угол поворота сервопривода, пример на рис. 50



На рис. 74 представлено изначальное положение детали.

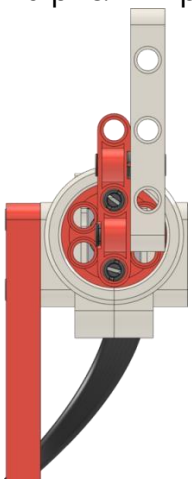


Рис. 74 Начальное положение детали

Примите во внимание угол, с помощью которого вам удалось откалибровать положение.

Решение для ученика

С помощью измерительного прибора (линейки или штангельциркуля), измерьте длину поворотной части и расстояние конца этой детали от перпендикуляра, опущенного к нижней точки детали, рис. 75.

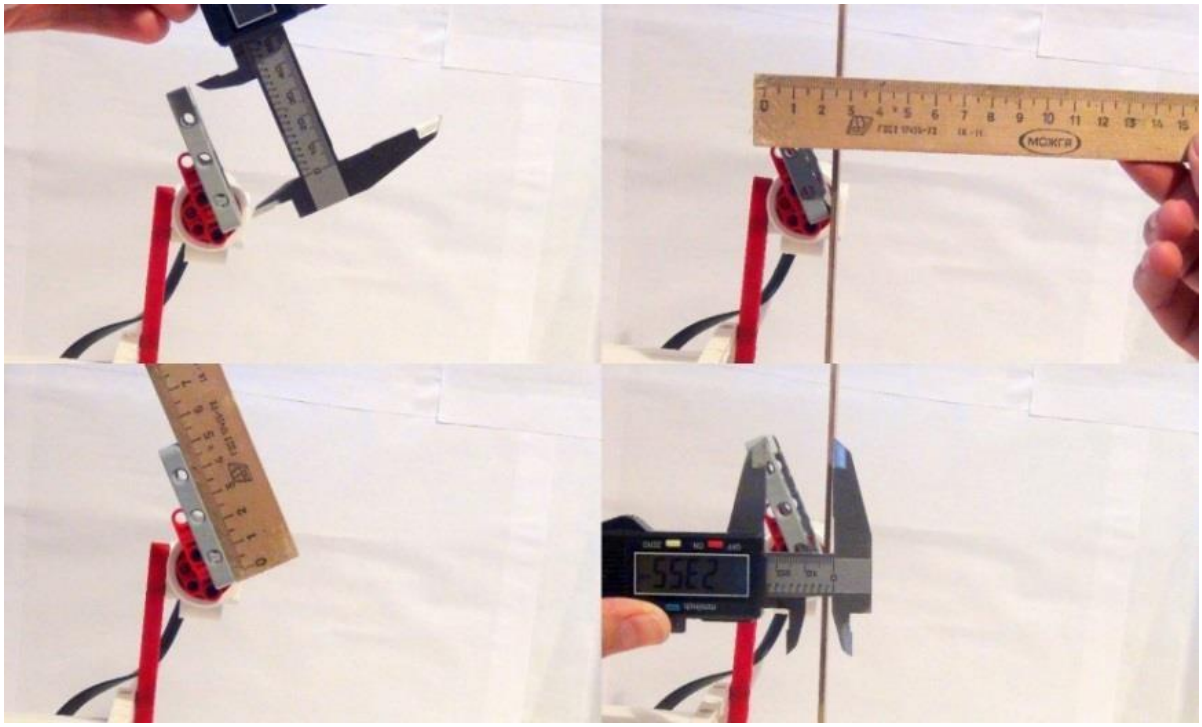


Рис. 75 Измерение сторон прямоугольного треугольника.

Угол можно вычислить через $\sin(A) = \frac{\text{противолежащий катет}}{\text{гипотенуза}}$. Согласно измерению, представленному на рис. 78, гипотенуза равна 5 см, а катет 2,5 см. Из этого можно сделать вывод, что $\sin(A) = \frac{\text{противолежащий катет}}{\text{гипотенуза}} = \frac{2.5}{5} = 0.5$. Тогда не трудно найти угол, используя таблицу Брадиса или калькулятор, $< A = 30^0$.

Впишем это значение угла с учётом угла калибровки в программу.

```
#include <Servo.h>

Servo myservo;

void setup() {
  myservo.attach(11);
  myservo.write(20);
  delay(1000);
  myservo.write(50);
}

void loop() {

}
```

Рис. 76 Программа позиционирования сервопривода

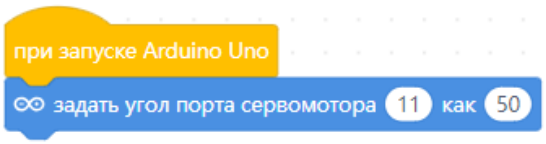


Рис. 77 Программа на MBlock 5

1. Зная угол поворота и длину поворотного механизма, найдите его путь.

Пример решения. Обратная задача, зная угол поворота и гипотенузу, определить противолежащий катет и длину окружности.

Для вычисления, нам понадобятся два выражения:

$$\sin(\text{A}) = \frac{\text{противолежащий катет}}{\text{гипотенуза}}$$

$$L = \frac{R * \pi * \theta}{180^0}$$

Первая математическая запись нам известна из предыдущего задания, а вторая – это вычисление длины дуги окружности, зная её угол и радиус.

Значение гипотенузы возьмём из предыдущей задачи – 5 сантиметров. Угол поворота назначим 60^0 (Не забудьте про угол калибровки).

$$\text{катет} = \sin(60^0) * 5 \text{ см} = 4.33 \text{ см}$$

$$L = \frac{3.14 * 5 \text{ см} * 60^0}{180^0} = 5.23 \text{ см}$$

Результат проверьте с помощью измерительных приборов (линейка, транспортир) и циркуля.

Задание 3 (Уровень С)

Проведите опыт по определению момента силы сервопривода.

Пример решения.

Есть два варианта проведения опыта:

А) у нас есть грузы с определённой массой, которые крепятся на определённом расстоянии от центра вращения оси сервопривода, т.е. смотрим при каком значении плеча, данный привод не способен поднять груз данной массы.

Б) У нас есть электронные весы, которые крепятся к разным точкам плеча и замеряется нагрузка, которую создаёт сервопривод.

Для примера воспользуемся вторым решением. На рис. 81 представлено плечо для сервопривода с заготовленными петлями из шпагата. Петли расположены на расстоянии от центра: 1 см, 2 см и 5 см.

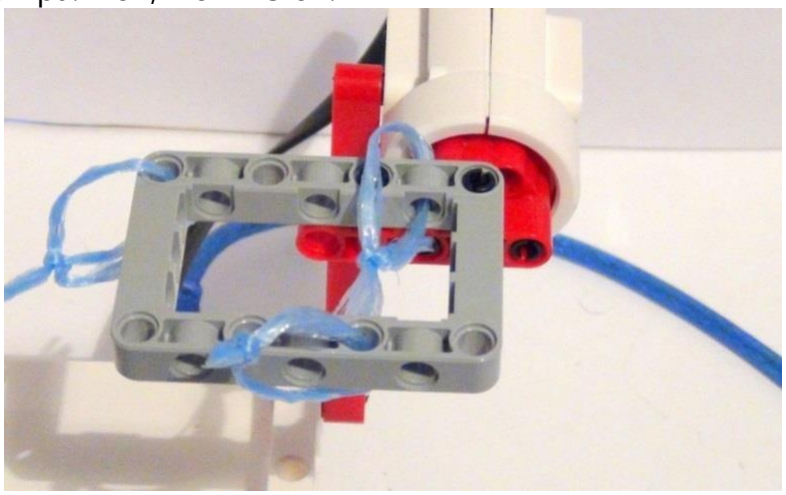


Рис.78 Конструкция для опытов с сервоприводом

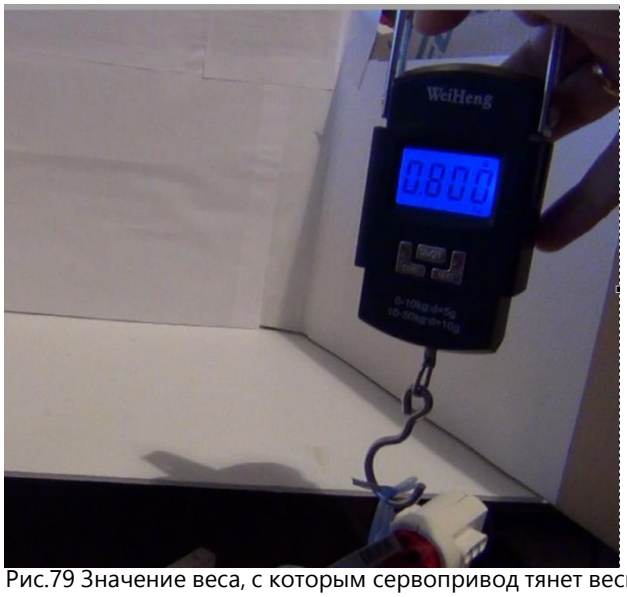


Рис.79 Значение веса, с которым сервопривод тянет весы при плече в 2 см. Примерно это 1 кг нагрузки.

Рис.80 Значение веса, с которым сервопривод тянет весы при плече в 2 см. Примерно значение веса 0.415 кг.

Рис.81 Значение веса, с которым сервопривод тянет весы при плече в 5 см. Вес примерно равен 0.260 кг.

Опираясь на эти данные составим таблицу и вычислим момент силы.

Плечо, L метры	Вес, F, Ньютоны	Момент силы $M = F * L$, *Н*м*
0.01	8	0.08
0.02	4.15	0.083
0.05	2.6	0.13
Среднее значение		0.098

В характеристиках к сервоприводу момент силы пишут в кг *см. Нетрудно перевести значение.

6.3. Ультразвуковой датчик расстояния

Данные имеет ряд характеристик:

- сектор обзора
- дальность обзора
- погрешность измерения

Ультразвуковой датчик расстояния рекомендуется подключить к первому, четвёртому, пятому и шестому портам блока управления. Для управления датчиком, зарезервированы пины на плате Arduino:

Порт	TRIG	ECHO
1	D11	D10
4	D1	D0
5	D12	D13
6	D8	D2

Модель ультразвукового датчика расстояния, в большинстве случаях, используется HC-SR04.

Методические рекомендации

Тема: «Подключение и управление HC-SR04»

Цель: изучить процесс подключения ультразвукового датчика и получить знания и опыт в области управления им с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс подключения и крепления ультразвукового датчика
- изучить возможности ультразвукового датчика
- получить и закрепить на практике знания, умения и навыки по основам геометрии.
- получить и закрепить на практике знания, умения и навыки в области создания программ

Примеры заданий по данной теме.

Ультразвуковой датчик расстояния, как писалось ранее, необходим для определения расстояния до объекта.

Задание 1 (Уровень А)

1. Соберите конструкцию согласно инструкции

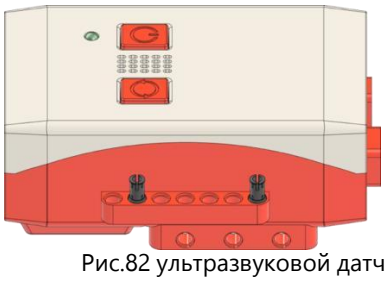


Рис.82 ультразвуковой датчик HC-SR04

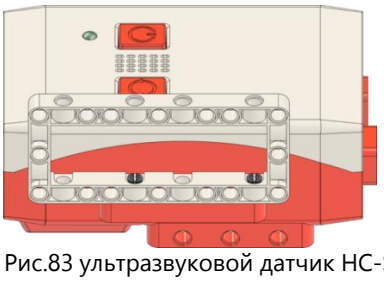


Рис.83 ультразвуковой датчик HC-SR04

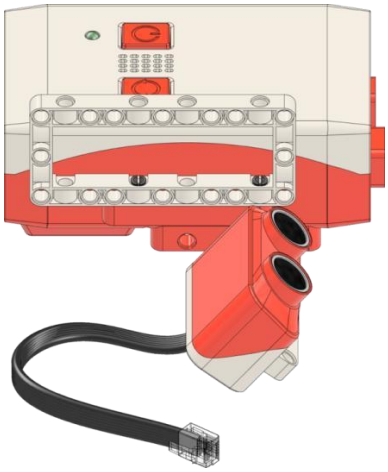


Рис.84 ультразвуковой датчик HC-SR04

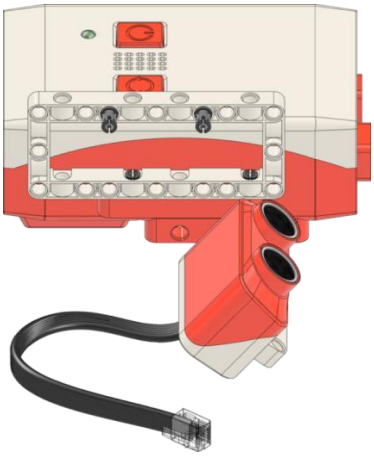


Рис.85 ультразвуковой датчик HC-SR04

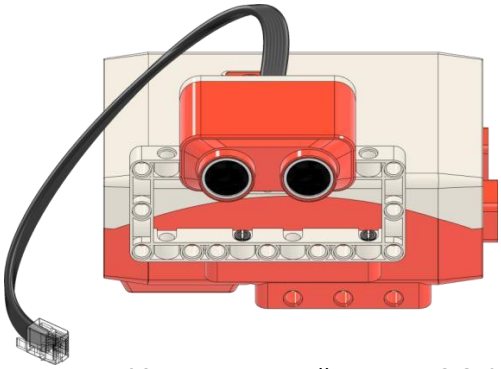


Рис.86 ультразвуковой датчик HC-SR04

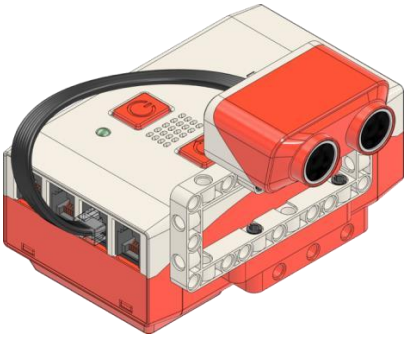


Рис.87 ультразвуковой датчик HC-SR04

2. Составьте программу, которая позволит проводить измерения с помощью ультразвукового датчика.

Пример решения для пятого порт. В программе будем использовать выражение для перевода значений в сантиметры.

```
int trig = 12;
int echo = 13;
long dur, cm;

void setup() {

Serial.begin (9600);

pinMode(trig, OUTPUT);

pinMode(echo, INPUT);

}
void loop(){
// Датчик генерирует импульсы шириной 10 мкс
// Генерируем короткий LOW импульс, чтобы обеспечить хороший импульс HIGH:
```

```
digitalWrite(trig, LOW);

delayMicroseconds(5);

digitalWrite(trig, HIGH);

delayMicroseconds(10);

digitalWrite(trig, LOW);

// Считываем данные с ультразвукового датчика: значение HIGH, которое

// зависит от длительности (в микросекундах) между отправкой

// акустической волны и ее обратном приеме на эхолокаторе.

pinMode(echo, INPUT);

dur = pulseIn(echo, HIGH);

cm = (dur/2) / 29.1;

Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(300);
}
```

Рис.88 Программа управления ультразвуковым датчиком

Задание 2 (Уровень В)

2. Собрать радар

Примерная инструкция сборки.

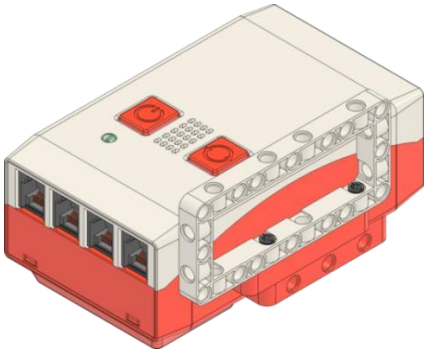


Рис.89 Радар

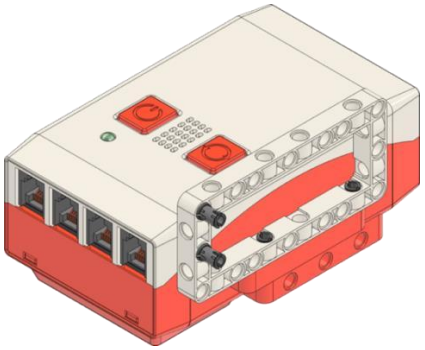


Рис.90 Радар

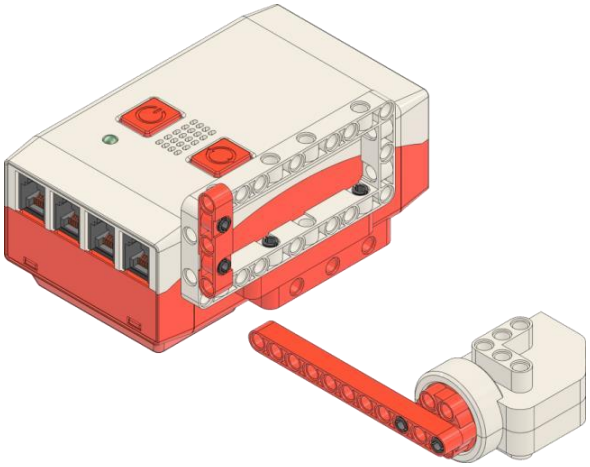


Рис.91 Радар

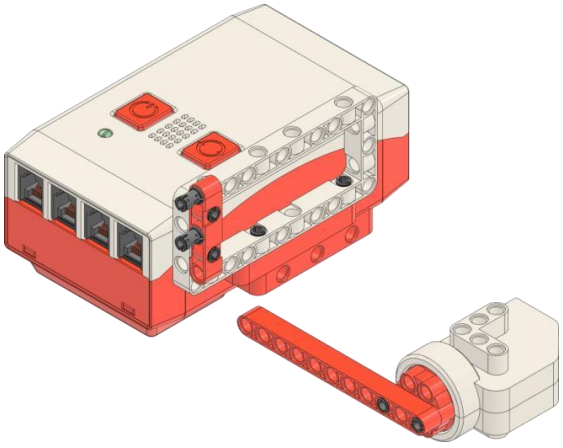


Рис.92 Радар

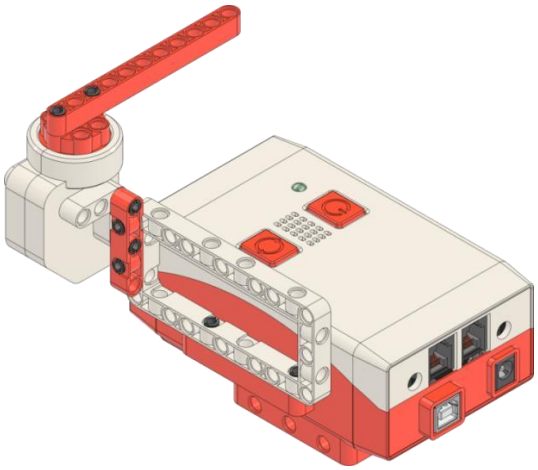


Рис.93 Радар

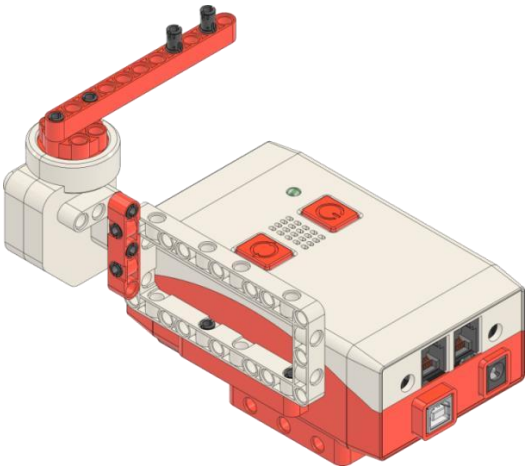


Рис.94 Радар

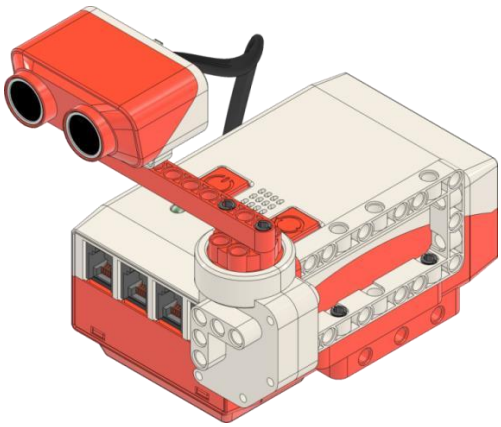


Рис.95 Радар

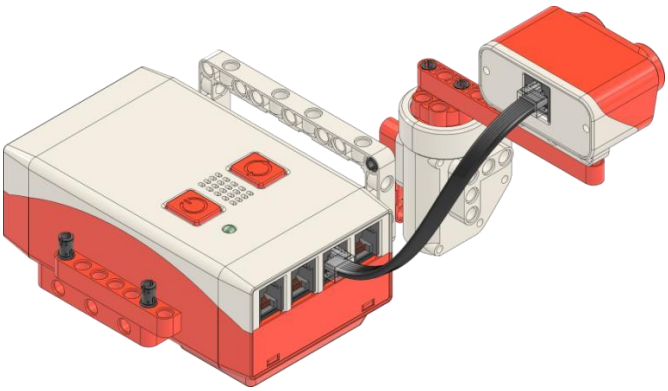


Рис.96 Радар

Для данной установки создадим программу, скомбинировав предыдущие программы для сервопривода и ультразвукового датчика расстояния.

```
#include <Servo.h>
Servo myservo;
int trig = 12;
int echo = 13;
long dur, cm;

void setup() {
myservo.attach(11);
Serial.begin (9600);
pinMode(trig, OUTPUT);
pinMode(echo, INPUT);
}
void loop(){

for (int i=0; i<180; i++ ){
      myservo.write(i);
digitalWrite(trig, LOW);
delayMicroseconds(5);
digitalWrite(trig, HIGH);
delayMicroseconds(10);
digitalWrite(trig, LOW);
pinMode(echo, INPUT);
dur = pulseIn(echo, HIGH);
cm = (dur/2) / 29.1;
Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(10);
}
 for (int i=180; i>0; i-- ){
   // myservo1.write(i);
    myservo.write(i);
    delay(10);
   digitalWrite(trig, LOW);
delayMicroseconds(5);
digitalWrite(trig, HIGH);
delayMicroseconds(10);
digitalWrite(trig, LOW);
pinMode(echo, INPUT);
dur = pulseIn(echo, HIGH);
cm = (dur/2) / 29.1;
Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(10);
    }
}
```

Рис.97 Программа для управления радаром

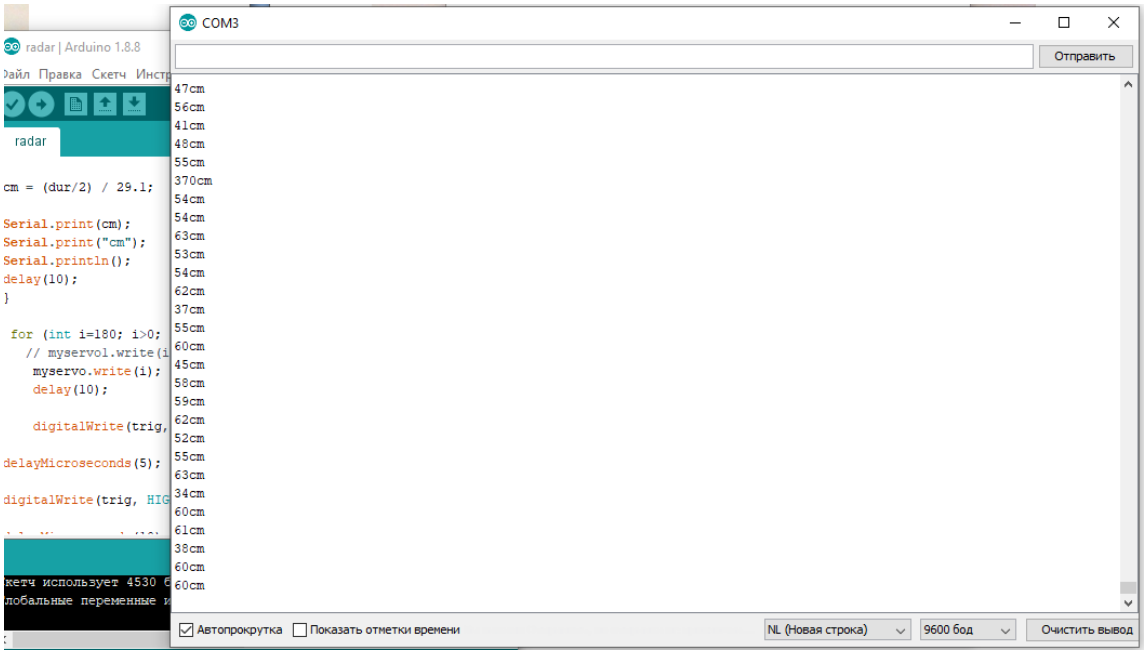


Рис.98 Результат работы радара

6.4. Датчик линии

Данные имеет ряд характеристик:

- дальность обзора
- погрешность измерения по степени освещённости

Датчик линии рекомендуется подключить к первому, четвёртому, пятому и шестому портам блока управления. Для управления датчиком, зарезервированы пины на плате Arduino:

Порт	Левый (канал S1)	Правый (канал S2)
1	D11	D10
4	D1	D0
5	D12	D13
6	D8	D2

Методические рекомендации.

Тема: «Подключение и управление датчиком линии»

Цель: изучить процесс подключения датчика линии и получить знания и опыт в области управления им с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс подключения и крепления датчика линии
- изучить возможности датчика;
- получить и закрепить на практике знания, умения и навыки по основам физических аспектов, которые могут повлиять на работы датчика;
- получить и закрепить на практике знания, умения и навыки в области создания программ.

Примеры заданий по данной теме.

Датчик линии необходим для определение границы между чёрной и светлой полосой на поверхности. Датчики снабжены потенциометрами, которые нужны для отрегулирования чувствительности датчика к границе между чёрным и белым.

Задание 1 (Уровень А)

1. Соберите конструкцию согласно инструкции

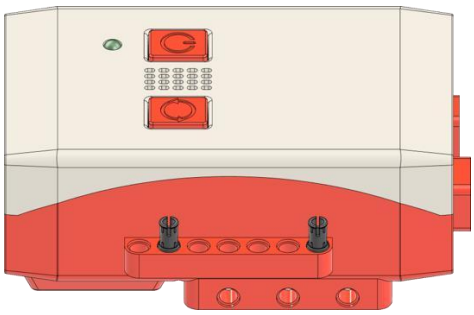


Рис.99 Датчик линии

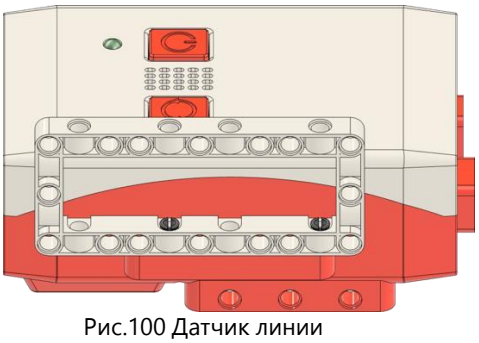


Рис.100 Датчик линии

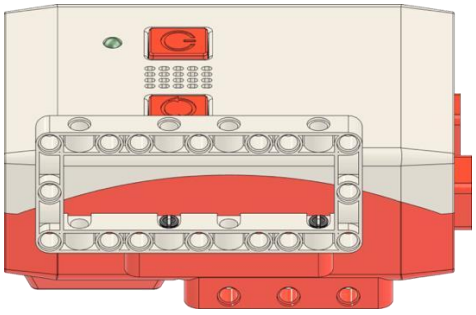


Рис.101 Датчик линии

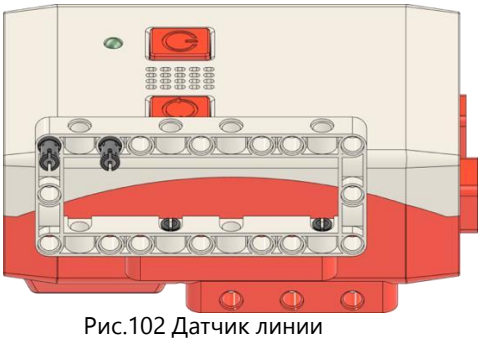


Рис.102 Датчик линии

Рис.103 Датчик линии

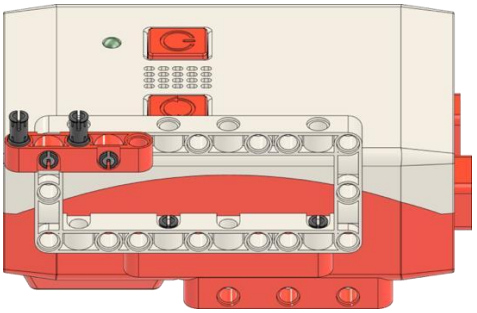


Рис.104 Датчик линии

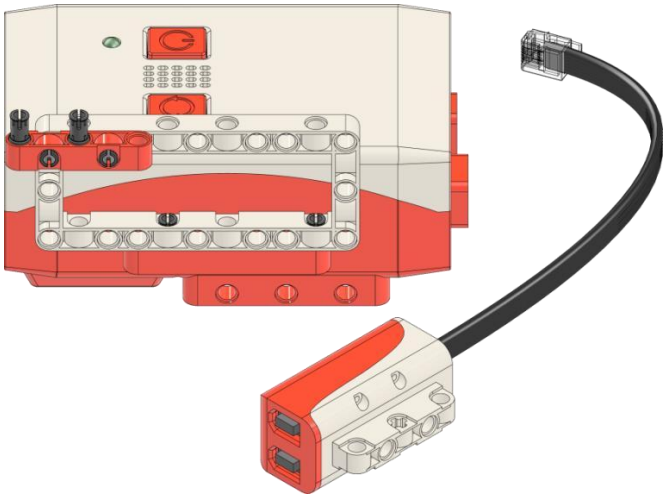


Рис.105 Датчик линии

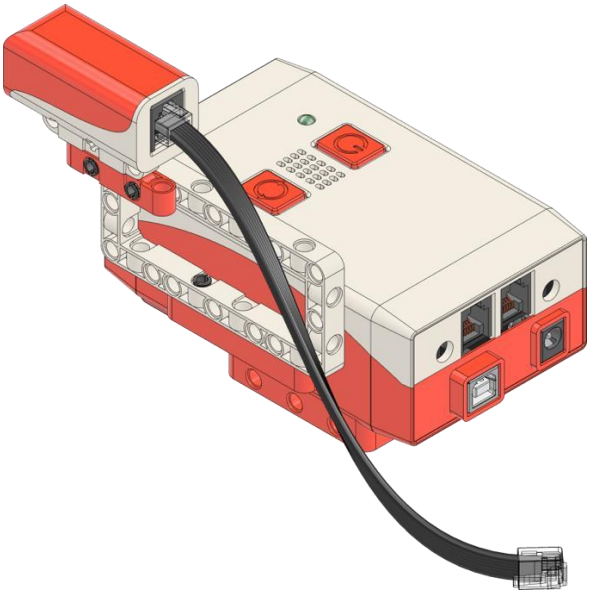


Рис.106 Датчик линии

2. Составьте программу, которая позволит находить границу между тёмной и светлой линиями с помощью датчика линии.

Пример решения.

Для реализации используются цифровые выводы, если белая сторона, то значение один, если чёрная, то значение ноль.

```
int l1 =11;
int l2 =10;

void setup() {
 pinMode(11,INPUT);
pinMode(12,INPUT);
Serial.begin(9600);
}

void loop() {

  Serial.println("left_line");
  Serial.println(digitalRead(l1));

  Serial.println("right_line");
  Serial.println(digitalRead(l2));

delay(500);
}
```

Рис.107 Программа по управлению датчиком линии

6.5. Датчик цвета

Данные имеет ряд характеристик:

- чувствительность к цвету
- температура цвета
- погрешность измерения

Датчик цвета можно подключать к любому порту блока управления.

Методические рекомендации.

Тема: «Подключение и управление датчиком цвета»

Цель: изучить процесс подключения датчика цвета и получить знания и опыт в области управления им с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс подключения и крепления датчика цвета
- изучить возможности датчика
- получить и закрепить на практике знания, умения и навыки по основам физики света.
- получить и закрепить на практике знания, умения и навыки в области создания программ

Примеры заданий по данной теме.

Датчик цвета необходим для определения цвета объекта.

Задание 1 (Уровень А)

1. Соберите конструкцию согласно инструкции

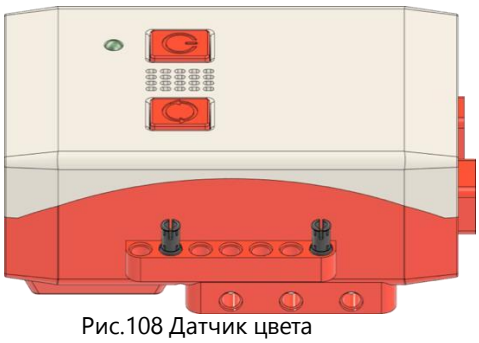


Рис.108 Датчик цвета

Рис.109 Датчик цвета

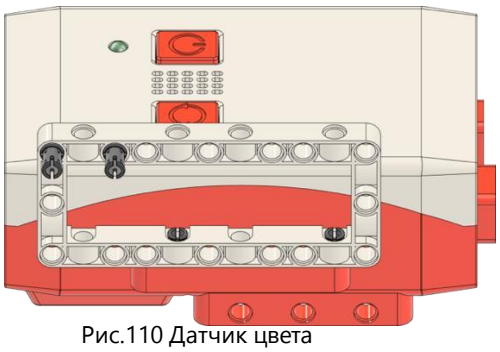


Рис.110 Датчик цвета

Рис.111 Датчик цвета

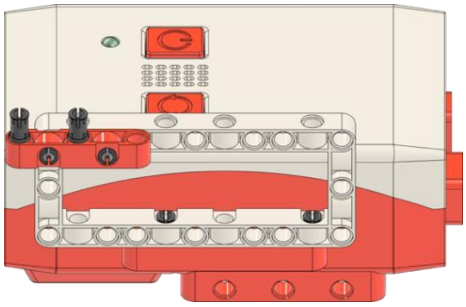


Рис.112 Датчик цвета

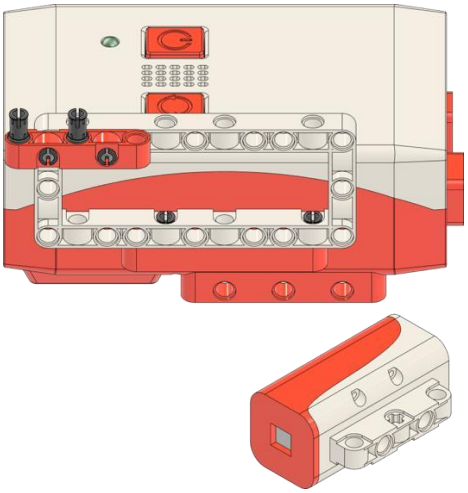


Рис.113 Датчик цвета

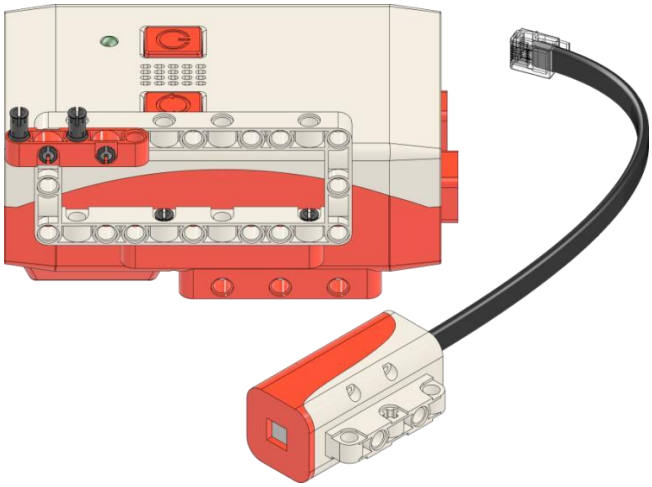


Рис.114 Датчик цвета

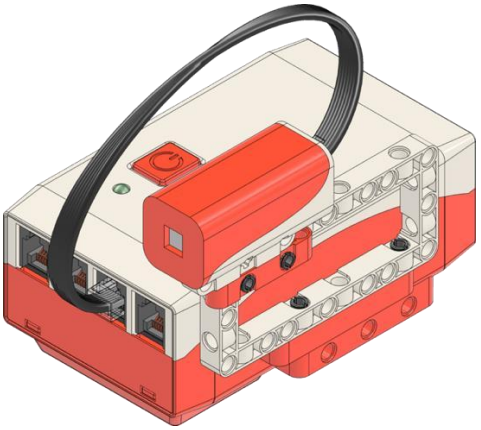


Рис.115 Датчик цвета

2. Составьте программу, которая позволит проводить измерения с помощью датчика цвета

Для подключения к датчику необходимо установить библиотеку для Arduino ide Adafruit_TCS34725.h

Данная библиотека содержит ряд примеров. Рассмотрим пример для выявления состава цвета объекта.

colorview §

```
#include <Wire.h>
#include "Adafruit_TCS34725.h"

// Pick analog outputs, for the UNO these three work well
// use ~560  ohm resistor between Red & Blue, ~1K for green (its brighter)
#define redpin 8
#define greenpin 3
#define bluepin 2
// for a common anode LED, connect the common pin to +5V
// for common cathode, connect the common to ground

// set to false if using a common cathode LED
#define commonAnode true

// our RGB -> eye-recognized gamma color
byte gammatable[256];


Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_50MS, TCS34725_GAIN_4X);

void setup() {
  Serial.begin(9600);
  //Serial.println("Color View Test!");

  if (tcs.begin()) {
    //Serial.println("Found sensor");
  } else {
    Serial.println("No TCS34725 found ... check your connections");
    while (1); // halt!
  }

  // use these three pins to drive an LED
#if defined(ARDUINO_ARCH_ESP32)
  ledcAttachPin(redpin, 1);
  ledcSetup(1, 12000, 8);
  ledcAttachPin(greenpin, 2);
  ledcSetup(2, 12000, 8);
  ledcAttachPin(bluepin, 3);
  ledcSetup(3, 12000, 8);
#else
  pinMode(redpin, OUTPUT);
  pinMode(greenpin, OUTPUT);
  pinMode(bluepin, OUTPUT);
#endif
```

Рис.116 Программа по анализу цвета предмета

```
  // thanks PhilB for this gamma table!
  // it helps convert RGB colors to what humans see
  for (int i=0; i<256; i++) {
    float x = i;
    x /= 255;
    x = pow(x, 2.5);
    x *= 255;

    if (commonAnode) {
      gammatable[i] = 255 - x;
    } else {
      gammatable[i] = x;
    }
    //Serial.println(gammatable[i]);
  }
}

// The commented out code in loop is example of getRawData with clear value.
// Processing example colorview.pde can work with this kind of data too, but It requires manual conversion to
// [0-255] RGB value. You can still uncomments parts of colorview.pde and play with clear value.
void loop() {
  float red, green, blue;

  tcs.setInterrupt(false);  // turn on LED

  delay(60);  // takes 50ms to read

  tcs.getRGB(&red, &green, &blue);

  tcs.setInterrupt(true);  // turn off LED

  Serial.print("R:\t"); Serial.print(int(red));
  Serial.print("\tG:\t"); Serial.print(int(green));
  Serial.print("\tB:\t"); Serial.print(int(blue));

//  Serial.print("\t");
//  Serial.print((int)red, HEX); Serial.print((int)green, HEX); Serial.print((int)blue, HEX);
  Serial.print("\n");

//  uint16_t red, green, blue, clear;
//
//  tcs.setInterrupt(false);  // turn on LED
//
//  delay(60);  // takes 50ms to read
//
//  tcs.getRawData(&red, &green, &blue, &clear);
//  tcs.setInterrupt(true);  // turn off LED
//
//  Serial.print("C:\t"); Serial.print(int(clear));
//  Serial.print("R:\t"); Serial.print(int(red));
//  Serial.print("\tG:\t"); Serial.print(int(green));
//  Serial.print("\tB:\t"); Serial.print(int(blue));
//  Serial.println();


#if defined(ARDUINO_ARCH_ESP32)
  ledcWrite(1, gammatable[(int)red]);
  ledcWrite(2, gammatable[(int)green]);
  ledcWrite(3, gammatable[(int)blue]);
#else
  analogWrite(redpin, gammatable[(int)red]);
  analogWrite(greenpin, gammatable[(int)green]);
  analogWrite(bluepin, gammatable[(int)blue]);
#endif
}
```

Рис.117 Программа по анализу цвета предмета

Задание 2 (Уровень В)

Создать программу, которая будет узнавать определённый цвет и выводить в последовательный порт.

6.6. IR-приёмник

Данные имеет ряд характеристик:

- сектор обзора
- дальность приёма
- погрешность измерения сигналов

IR приёмник рекомендуется подключить к первому, второму, четвёртому, пятому и шестому портам блока управления. Для управления датчиком, зарезервированы пины на плате Arduino:

Порт	Канал
1	D11
2	D3
4	D1
5	D12
6	D8

Приёмник должен идти в комплекте с пультом IR.

Методические рекомендации.

Тема: «Подключение и управление IR приёмником»

Цель: изучить процесс подключения IR приёмника и получить знания и опыт в области управления им с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс подключения и крепления IR приёмника
- изучить возможности приёмника
- получить и закрепить на практике знания, умения и навыки по основам кодирования сигналов.
- получить и закрепить на практике знания, умения и навыки в области создания программ

В память робота можно записать несколько независимых алгоритмов работы робота и для их выбора удобно использовать инфракрасный датчик.

Поэтому алгоритм работы контроллера робота реализован так, что коды команд с пультов управления (инфракрасный) считываются с датчиков в самом начале главной функции управления роботом, вызываемой в неограниченном цикле на протяжении всей его работы. Далее, эти команды обрабатываются по заданным алгоритмам, и в соответствии с ними меняют поведение робота. Таким образом обеспечивается приоритет команд пользователя над внутренними командами робота.

Примеры заданий по данной теме.

IR приёмник, необходим для управлением роботизированным устройством.

Задание 1 (Уровень А)

1. Соберите конструкцию согласно инструкции

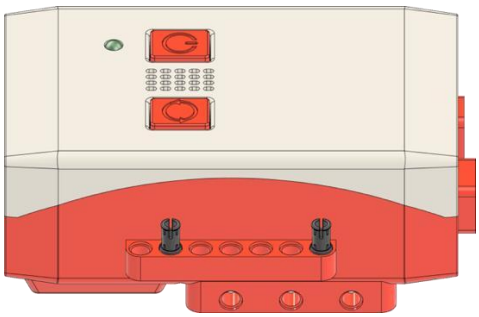


Рис.118 IR приёмник

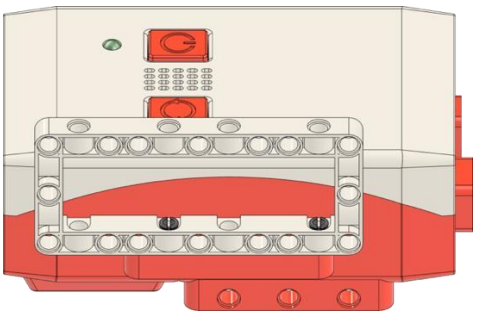


Рис.119 IR приёмник

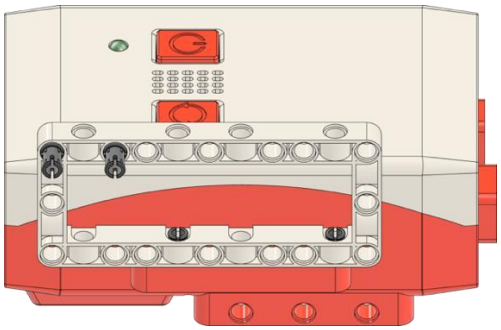


Рис.120 IR приёмник

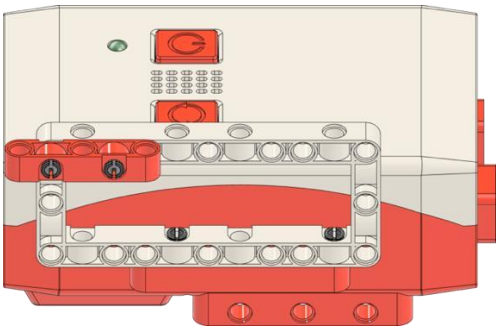


Рис.121 IR приёмник

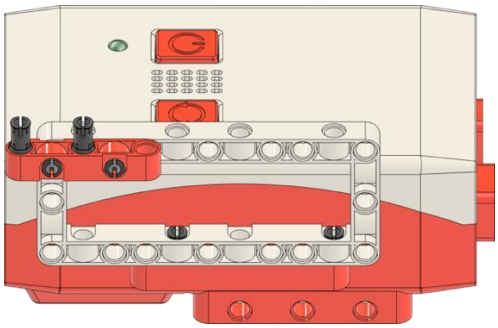


Рис.122 IR приёмник

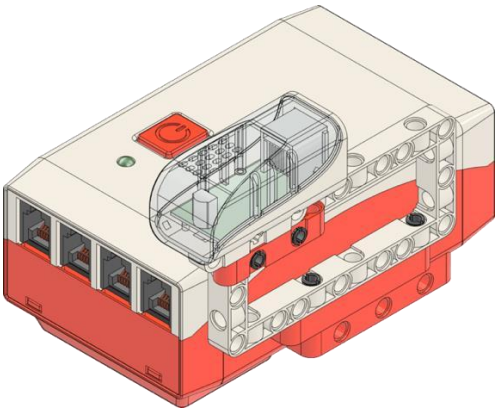


Рис.123 IR приёмник

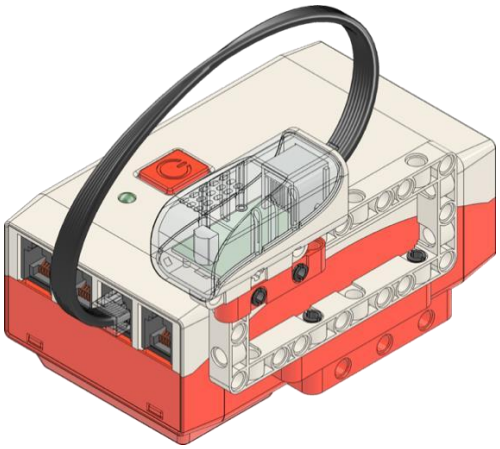


Рис.124 IR приёмник

2. Составьте программу, которая позволит передавать сигналы на IR приёмник. Для корректной работы с приёмником понадобиться установить библиотеку **IRremote.**

```
#include <IRremote.h>

int RECV_PIN = 8;

IRrecv irrecv(RECV_PIN);

decode_results results;

void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
  }

void loop() {
  if (irrecv.decode(&results)) {
    //Serial.println(results.value, HEX);
    Serial.println(results.value);
    irrecv.resume();

  }
}
```

Рис.125 Программа для приёма сигналов IR приёмником

Задание 2 (Уровень В)

Составьте таблицу кодирующего сигнала поступающего на приёмник от IR пульта.

Пример решения:

- IR пультов существует множество вариантов и все они могут взаимодействовать с IR приёмником.

Рассмотрим IR пульт, идущий в комплекте.

Рис.126 IR пульт

Загрузим программу из рис. 126 на блок управления CyberBot. Будем посылать сигналы при нажатии каждой клавиши пульта и выписывать их.

У нас получится таблица по умолчанию, сигналы идут в десятичном формате, но также можно представить в шестнадцатеричной системе счисления. В программе есть команда переводящая в шестнадцатеричный формат **Serial.println(results.value, HEX)**.

BUTTON	DEC	HEX
CH –	16753245	FFA25D
CH	16736925	FF629D
CH+	16769565	FFE21D
<<	16720605	FF22DD
>>	16712445	FF02FD
>‖	16761405	FFC23D
-	16769055	FFE01F
+	16754775	FFA857
EQ	16748655	FF906F
0	16738455	FF6897
100+	16750695	FF9867
200+	16756815	FFB04F
1	16724175	FF30CF
2	16718055	FF18E7
3	16743045	FF7A85
4	16716015	FF10EF
5	16726215	FF38C7
6	16734885	FF5AA5
7	16728765	FF42BD
8	16730805	FF4AB5
9	16732845	FF52AD

Кодировка сигналов может отличаться.

6.7. Bluetooth модуль

Данные имеет ряд характеристик:

- дальность сигнала
- объём передачи данных
- скорость передачи

Bluetooth модуль рекомендуется подключить к четвёртому порту блока управления. Для управления модулем, зарезервированы пины на плате Arduino:

Порт	RX	TX
4	D1	D0

Модели Bluetooth модуля в большинстве случаях, используется HC-05, HC-06.

Методические рекомендации.

Тема: «Подключение и управление Bluetooth модулем»

Цель: изучить процесс подключения Bluetooth модуля и получить знания и опыт в области управления им с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс подключения и крепления Bluetooth модуля
- изучить возможности Bluetooth модуля
- получить и закрепить на практике знания, умения и навыки в области создания программ

Примеры заданий по данной теме.

Bluetooth модуля, как писалось ранее, необходим для дистанционного управления роботом.

Задание 1 (Уровень А)

1. Соберите конструкцию согласно инструкции

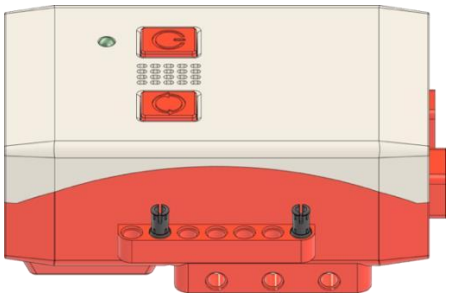


Рис.127 Bluetooth модуль

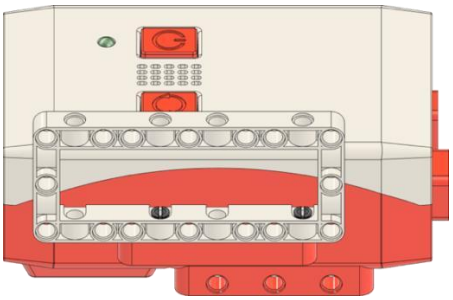


Рис.128 Bluetooth модуль

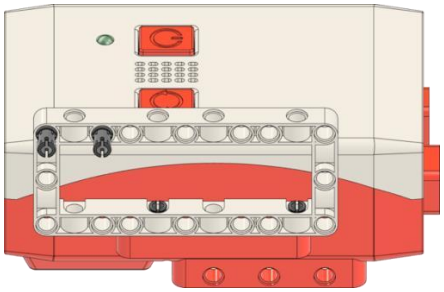


Рис.129 Bluetooth модуль

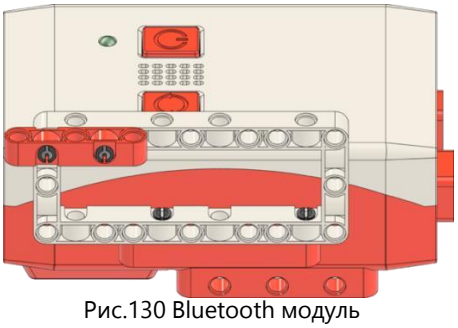


Рис.130 Bluetooth модуль

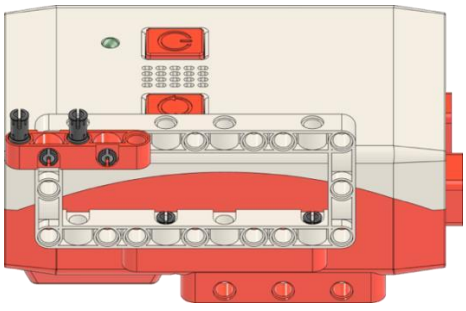


Рис.131 Bluetooth модуль

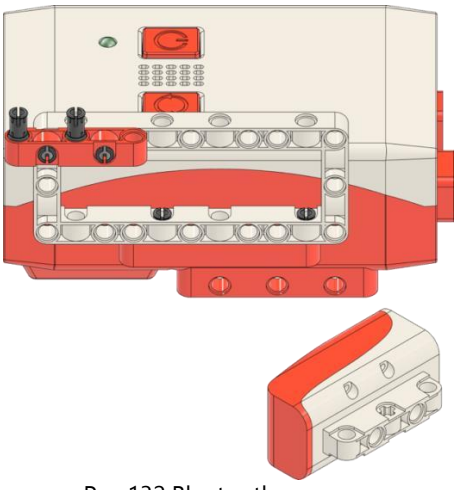


Рис.132 Bluetooth модуль

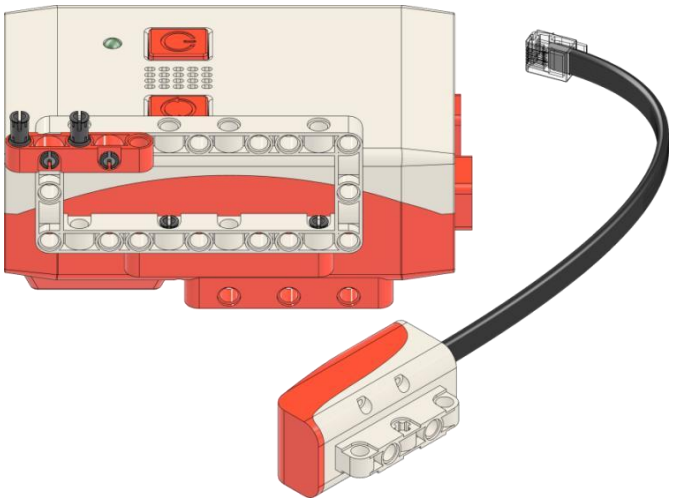


Рис.133 Bluetooth модуль

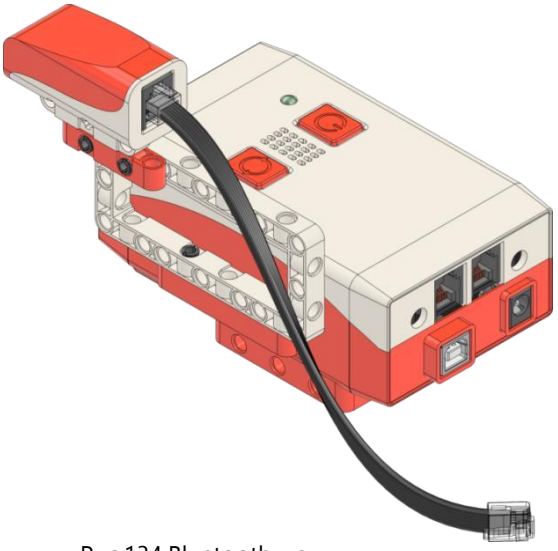


Рис.134 Bluetooth модуль

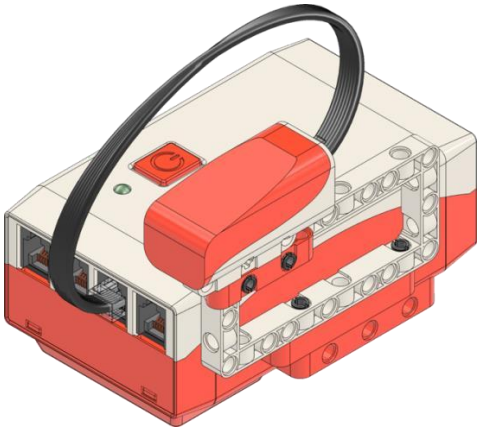


Рис.135 Bluetooth модуль

2. Составьте программу, которая позволит получать данные Bluetooth модуль и передавать их обратно.

Пример решения. Для корректной работы, необходимо провести сопряжение смартфона (планшета, ноутбука) с Bluetooth модулем по радиосвязи. При сопряжении устройство запросит пароль , в большинстве случаев – это 1234.

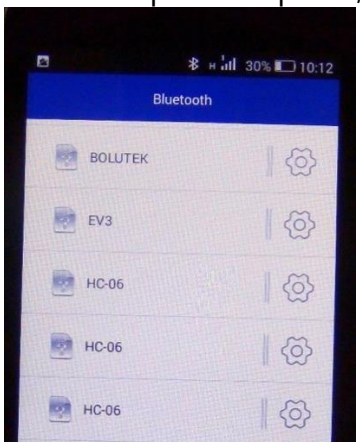


Рис.136 Сопряжение с bluetooth модулем

После этого необходимо установить Bluetooth терминал. С помощью данной программы можно подключаться к сопряжённому устройству и обмениваться данными по беспроводному каналу.

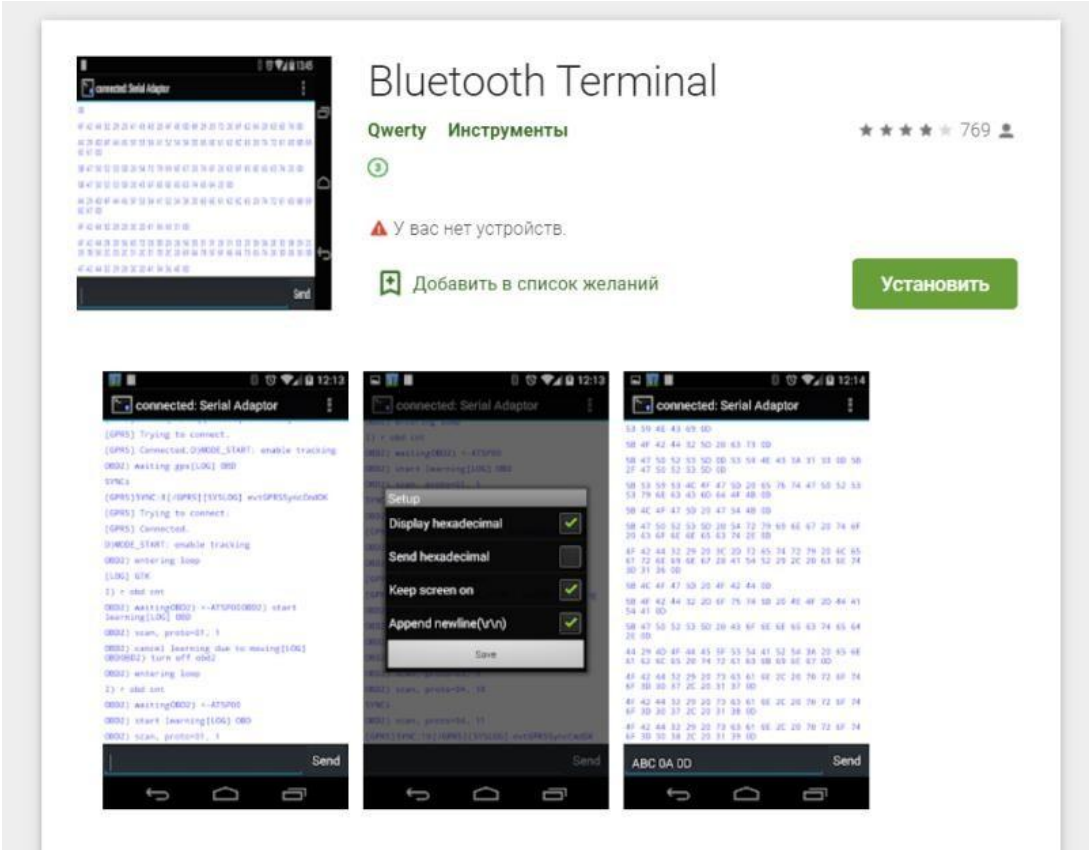


Рис.137 bluetooth терминал

Составим программу для приёма сигналов и передачи их через последовательный порт. Bluetooth соединение рассматривается, как подключение к последовательному порту (COM порт).

```
int x;

void setup() {

Serial.begin(9600);

void loop() {

  if (Serial.available())
  {
     x=Serial.read();

Serial.println(x);
  }

}
```

Рис.138 Программа для взаимодействия с bluetooth модулем

Функция Serial.available() получает количество байт(символов) доступных для чтения из последовательного интерфейса связи. Это те байты которые уже поступили и записаны в буфер последовательного порта. Буфер может хранить до 64 байт

6.8. Пьезоэлемент

Методические рекомендации.

Тема: «Управление пьезоэлементом»

Цель: получить знания и опыт в области управления пьезоэлементом с использованием языков программирования.

Задачи:

- изучить и закрепить на практике процесс программирования пьезоэлемента.
- изучить возможности пьезоэлемента.
- получить и закрепить на практике знания, умения и навыки в области создания программ

Примеры заданий по данной теме.

Пьезоэлемент подключён к пину A0. Функция позволяющая выдать из него звук является tone(). В функции обязательно нужно указать порт подключения и частоту вещания. Дополнительным третьим параметром может выступать время длительности воспроизводимого сигнала в миллисекундах.

Задание 1 (Уровень А)

Создайте программу, которая будет чередовать звуковые сигналы 2000 Гц и 100 Гц с различным интервалом времени, а затем отключаться.

Пример решения (файл **piezo.ino**)

```
int p=A0;


void setup() {
  pinMode(p, OUTPUT);

}

void loop() {

tone (p, 2000);

delay(3000);

tone(p, 100);

delay(2000);

tone(p, -1);

delay(1000);

}
```

Рис.139_а Программа по управлению пьезоэлементом (начало).

Задание 2 (Уровень В)

Создайте программу, которая будет чередовать звуковые сигналы согласно нотам с определённым интервалом времени, а затем отключаться.

Пример решения.

Для решения поставленной задачи нужно найти таблицу частот для нот разной октавы. Выбрать октаву и выписать частоты. Файл **piezo_NOT.ino**.

```
int p=A0;


void setup() {
  pinMode(p, OUTPUT);

}

void loop() {

tone(p, 261);
    delay(500);


    tone(p, 293);
      delay(500);

      tone(p, 329);
      delay(500);

          tone(p,349 );
      delay(500);

        tone(p,392);
      delay(500);

      tone(p, 440);
      delay(500);


     tone(p,466 );
     delay(500);

     tone(p,494 );
     delay(500);

    tone(p, -1);
    delay(1000);
}
```

Рис.140_b Программа по управлению пьезоэлементом по нотам (заключение)

Задание 3 (Уровень С)

Создать программу, с помощью которой пьезоэлемент проигрывает известную мелодию.

Пример решения.

Чтобы проиграть мелодию нужно знать её по нотам. Часто программистами выкладываются коды по проигрыванию различных мелодий.

Ниже представлен пример мелодии имперский марш из фильма «Звёздные войны».

Сохраним файл под именем **muz.ino**.

```
int S = A0;

void setup()
{
}

void loop()

{
tone(S, 392, 350);
delay(350);
tone(S, 392, 350);
delay(350);
tone(S, 392, 350);
delay(350);
tone(S, 311, 250);
delay(250);
tone(S, 466, 100);
delay(100);
tone(S, 392, 350);
delay(350);
tone(S, 311, 250);
delay(250);
tone(S, 466, 100);
delay(100);
tone(S, 392, 700);
delay(700);
tone(S, 587, 350);
delay(350);
tone(S, 587, 350);
delay(350);
tone(S, 587, 350);
delay(350);
tone(S, 622, 250);
delay(250);
tone(S, 466, 100);
delay(100);
tone(S, 369, 350);
delay(350);
tone(S, 311, 250);
delay(250);
tone(S, 466, 100);
delay(100);
tone(S, 392, 700);
delay(700);
```

Рис.141_c Программа по управлению пьезоэлементом - мелодия (начало)

```
tone(S, 784, 350);
delay(350);
tone(S, 392, 250);
delay(250);
tone(S, 392, 100);
delay(100);
tone(S, 784, 350);
delay(350);
tone(S, 739, 250);
delay(250);
tone(S, 698, 100);
delay(100);
tone(S, 659, 100);
delay(100);
tone(S, 622, 100);
delay(100);
tone(S, 659, 450);
delay(450);
tone(S, 415, 150);
delay(150);
tone(S, 554, 350);
delay(350);
tone(S, 523, 250);
delay(250);
tone(S, 493, 100);
delay(100);
tone(S, 466, 100);
delay(100);
tone(S, 440, 100);
delay(100);
tone(S, 466, 450);
delay(450);

tone(S, 311, 150);
delay(150);
tone(S, 369, 350);
delay(350);
tone(S, 311, 250);
delay(250);
tone(S, 466, 100);
delay(100);
tone(S, 392, 750);
delay(750);
delay(5000);

}
```

Рис.142_d Программа по управлению пьезоэлементом - мелодия (заключение)

7. Механика конструкции

В предыдущих главах мы познакомились с программным обеспечением и сопутствующей электроники, которая необходима для создания роботов. Роботизированное устройство не только состоит из электронных компонентов, но так же из соединительных деталей, которые образуют целостную структуру робота.

Разнообразие креплений деталей расширяет конструкторские возможности, а разнообразие самих деталей расширяет инженерный потенциал набора.

С разнообразием крепления деталей вы познакомитесь при выполнении заданий по созданию роботов. Здесь мы уделим внимание различным видам механических передач, которые мы можем использовать для реализации более сложных конструкций.

7.1. Зубчатая передача

Одной из самых распространённых передач является зубчатая. Зубчатая передача передаёт энергию благодаря множеству зубчатых колёс, которые касаются друг друга зубчиками. Зубчатые колёса могут с разным количеством зубчиков. В основном используют колёса с одинаковой частотой зубчиков, тем самым обеспечивая постоянную скорость вращения на всём участке колеса.

Скорость вращения колёс зависят от их радиуса и соотношения данных колёс друг другу. Самый простой вид зубчатой передачи – это взаимодействие двух зубчатых колёс одинакового диаметра. Рассмотрим схему такой передачи, соберём конструкцию.

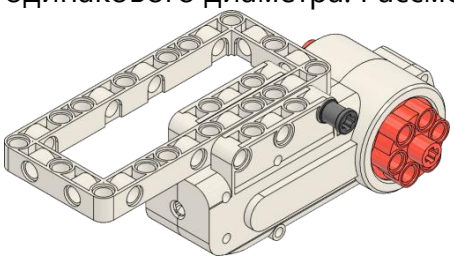


Рис.1 Зубчатая передача

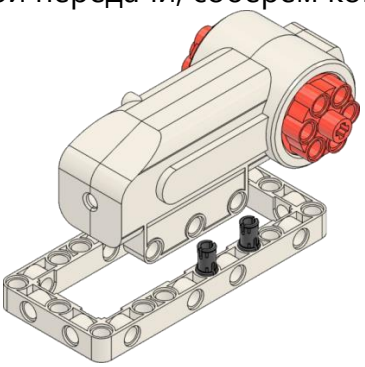


Рис.2 Зубчатая передача

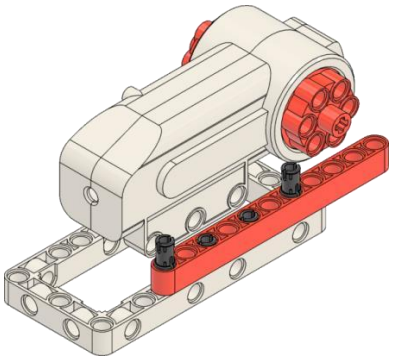


Рис. 3 Зубчатая передача

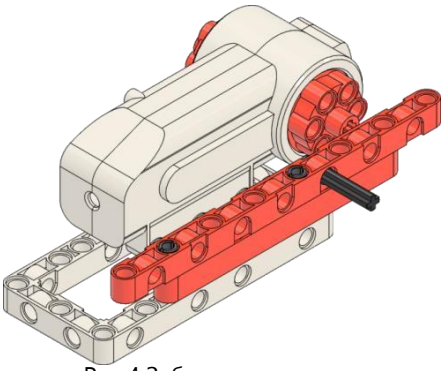


Рис.4 Зубчатая передача

Для запуска конструкции воспользуемся программой по управлению мотором
Загрузите программу и включите блок управления.

Ответьте на ряд вопросов:

- **Сравните скорости вращения зубчатых колёс, есть ли различия?**
- **В каком направлении происходит вращения зубчатых колёс?**

Теперь рассмотрим передачу, когда зубчатые колёса не одного диаметра.

Соберём конструкцию:

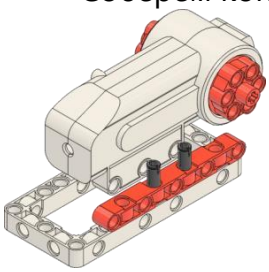


Рис.7 Зубчатая передача

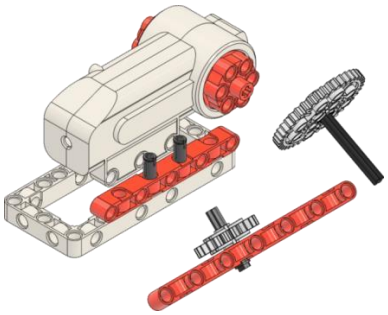


Рис.8 Зубчатая передача

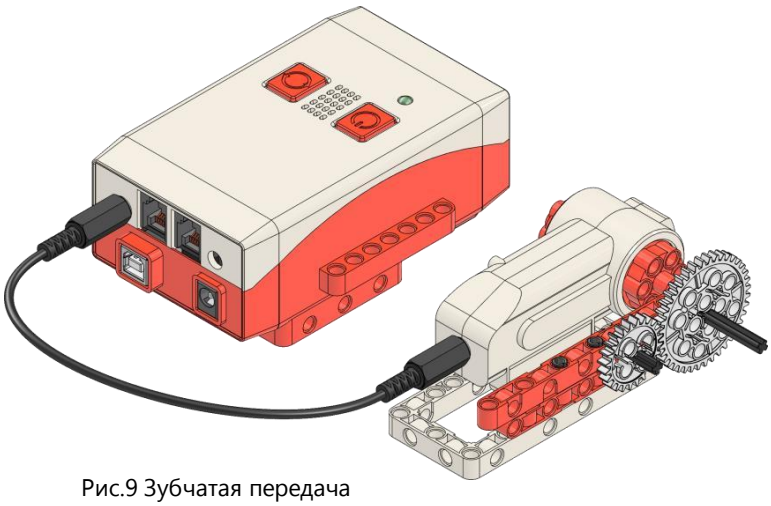


Рис.9 Зубчатая передача

Запустите программу и ответьте на ряд вопросов:

- **Сравните скорости вращения зубчатых колёс, есть ли различия?**
- **В каком направлении происходит вращения зубчатых колёс?**
- **Сравните силу вращения ведомого колеса.**

Рассмотрим другой вид взаимодействия этих зубчатых колёс:

12

Запустите программу и ответьте на те же вопросы:

- **Сравните скорости вращения зубчатых колёс, есть ли различия?**
- **В каком направлении происходит вращения зубчатых колёс?**
- **Сравните силу вращения ведомого колеса.**

Как видно, скорости вращения ведомого колеса будет зависеть от радиуса ведущего колеса.

Итак мы рассмотрели три вида зубчатых передач: простая зубчатая передача, повышающая зубчатая передача и понижающая зубчатая передача.

Интересен вопрос:

- **Насколько понижается или повышается скорость вращения и сила ведомого колеса?**

Для того чтобы ответить на этот вопрос необходимо затронуть тему передаточного отношения.

Передаточное отношение показывает во сколько угловая скорость и момент сил одного колеса больше (меньше) угловой скорости и момента сил второго колеса. Выражается формулой:

$$i = i_{12} = \frac{d_2}{d_1} = \frac{z_2}{z_1} = \frac{M_2}{M_1} = \frac{\omega_1}{\omega_2} = \frac{n_1}{n_2}$$

Где d_1, d_2 – диаметры колёс; z_1, z_2 – количество зубчиков колёс; M_1, M_2 – момент сил; ω_1, ω_2 – угловая скорость; n_1, n_2 – частота;

В первом случае два зубчатых колеса имеют диаметр 14 мм. Следовательно, их передаточное отношение равно 1, поэтому скорость и сила остались неизменны.

Во втором и третьем случаях большое колесо имеет диаметр 37,9 мм, а второе колесо – 22.18 мм. Следовательно, передаточное отношение во втором случае равно 1,7, а в третьем – 0,59.

Во втором случаи угловая скорость ведомого колеса будет больше угловой скорости ведущего в 1,7 раза, а момент силы в меньше во столько же раз. Для третьего варианта верно обратное.

Задание 1 (Уровень В)

Соберите установку, представленную ниже, и вычислите передаточные отношения для каждого из ведомых колёс.

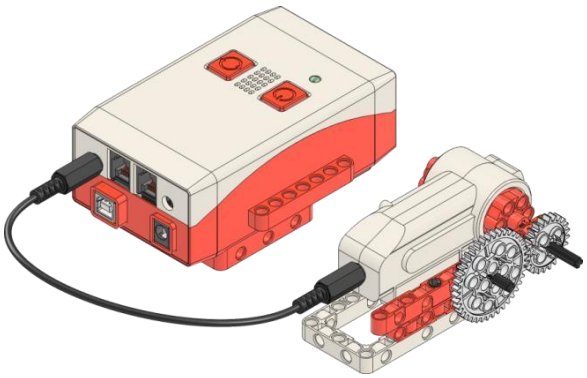


Рис.13 Передаточное отношение. Задание

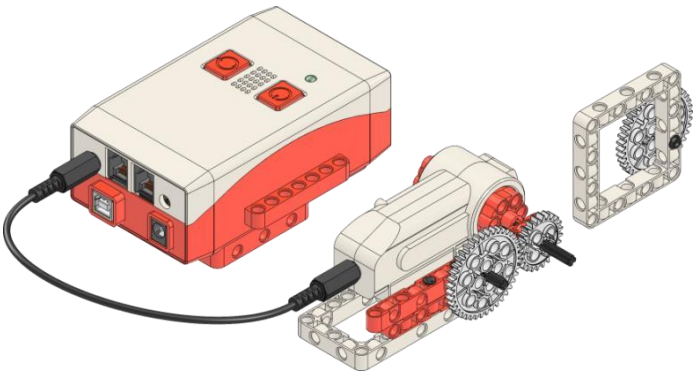


Рис.14 Передаточное отношение. Задание

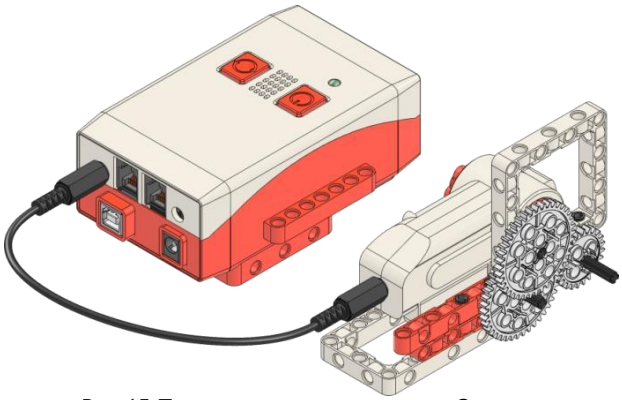


Рис.15 Передаточное отношение. Задание

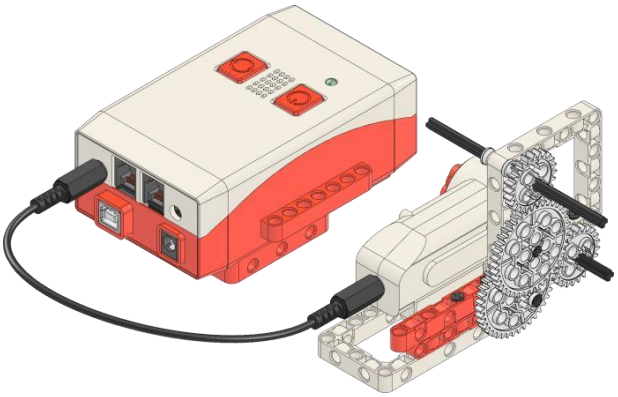


Рис.16 Передаточное отношение. Задание

Диаметр самого маленького зубчатого колеса – 5,7 мм.

7.2. Гусеничная передача

Гусеничная передача одна из частых после зубчатых передач. Для гусеничной передачи не нужно касание колёс, достаточно их соединения с помощью гусеничной ленты.

Гусеницы используют во многих задачах: транспортировка объектов (аэропорт, торговые центры, вокзал, фабрики и т.д.), способ перемещения (гусеничный ход по пересечённой местности).

Соберём простую конструкцию с использованием гусеницы.

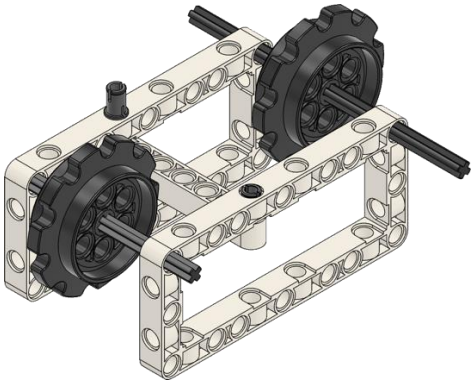


Рис.18 Гусеничная передача

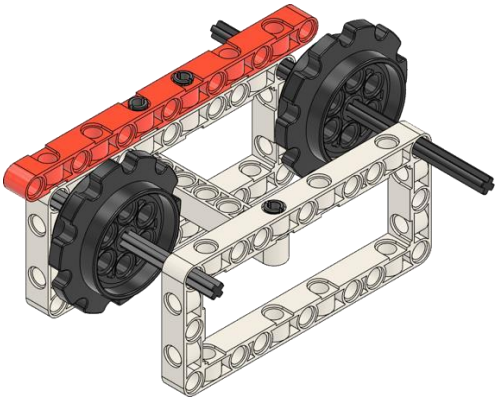


Рис.19 Гусеничная передача

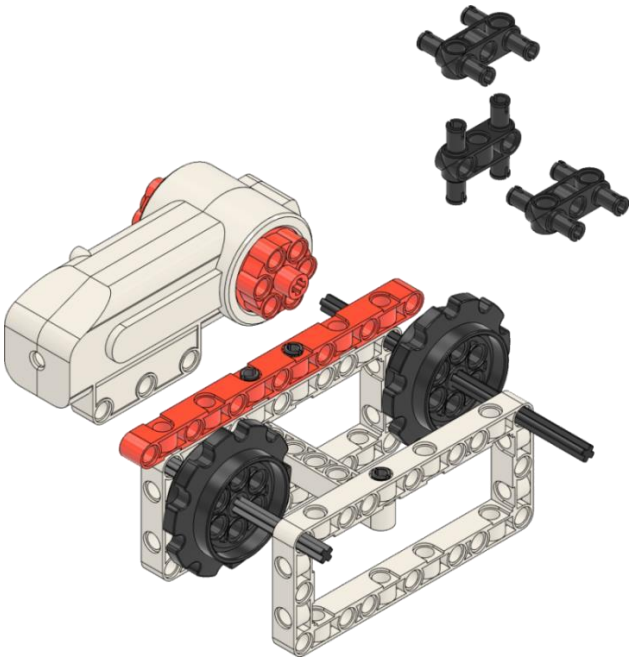


Рис.20 Гусеничная передача

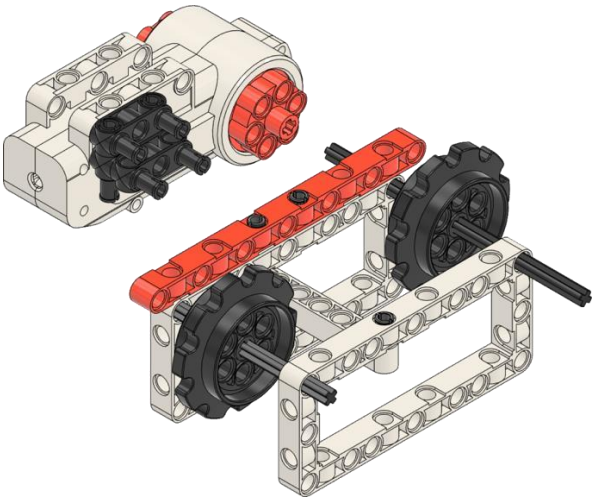


Рис.21 Гусеничная передача

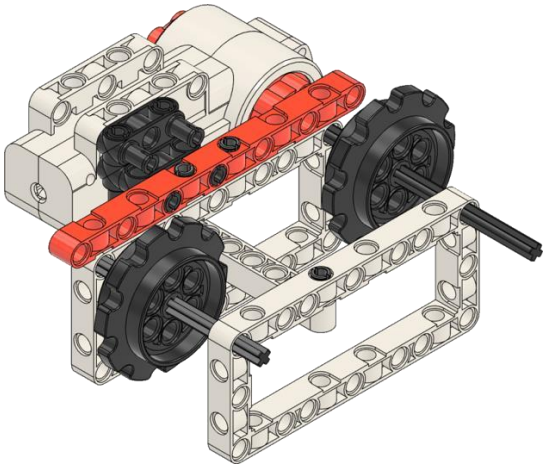


Рис.22 Гусеничная передача

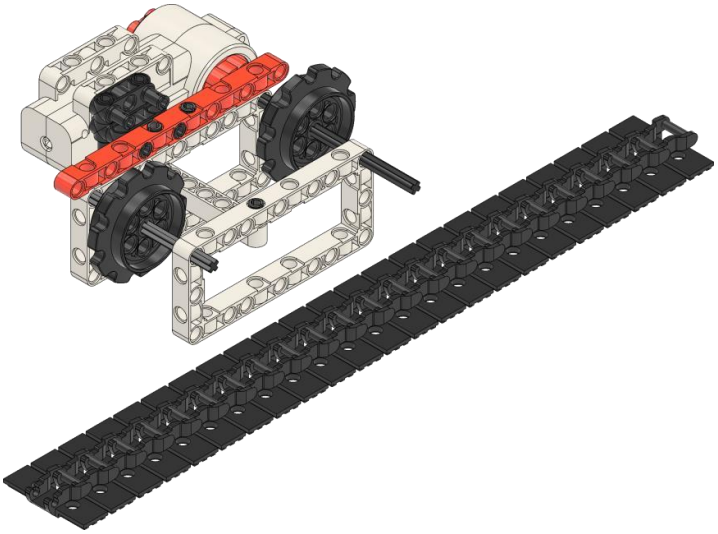


Рис.23 Гусеничная передача

Запустите программу для мотора

Задание 1 (Уровень А)

Создайте вертикальную конвейерную ленту, способную перемещать детали наверх.

Задание 2 (Уровень В)

Создайте две скомбинированные конвейерные ленты: одна горизонтальная, вторая вертикальная. Спроектировать их нужно так, чтобы деталь с первой конвейерной ленты перемещалась бы на вторую, а затем попадала бы в контейнер.

7.3. Кулачковый механизм

Ещё один из видов передач **– кулачковый.** С помощью него можно создавать механизмы с поступательно повторяющимся движением: движение поршня, стопоходящие механизмы, насос или качели и т.д.

Рассмотрим простой вид такой передачи. Соберите конструкцию:

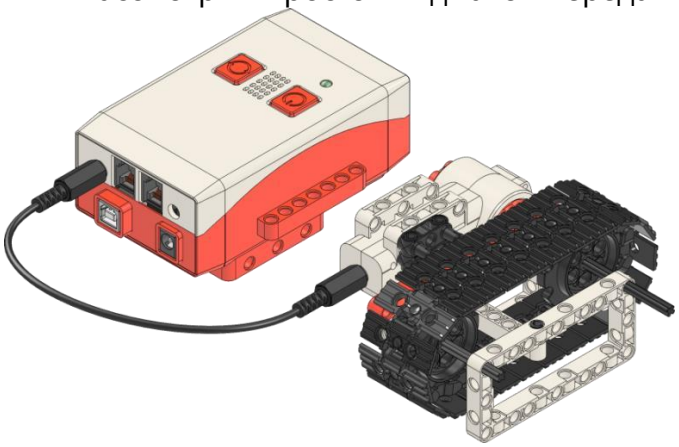


Рис.24 Кулачковый механизм

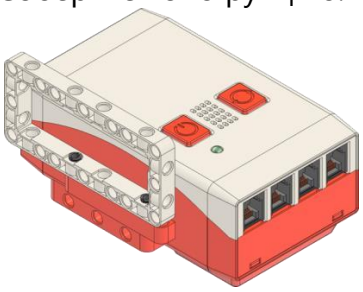


Рис.25 Кулачковый механизм

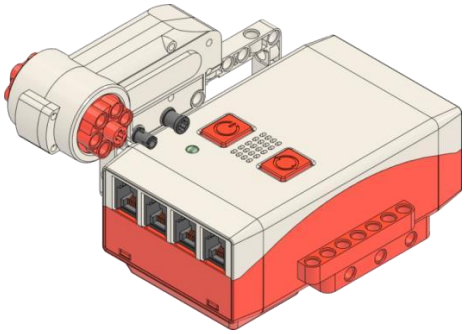


Рис.26 Кулачковый механизм

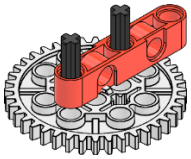


Рис.27 Кулачковый механизм

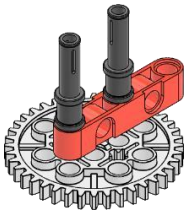


Рис.28 Кулачковый механизм

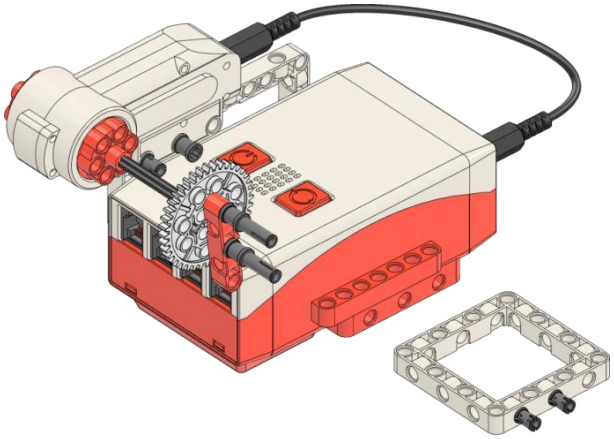


Рис.29 Кулачковый механизм

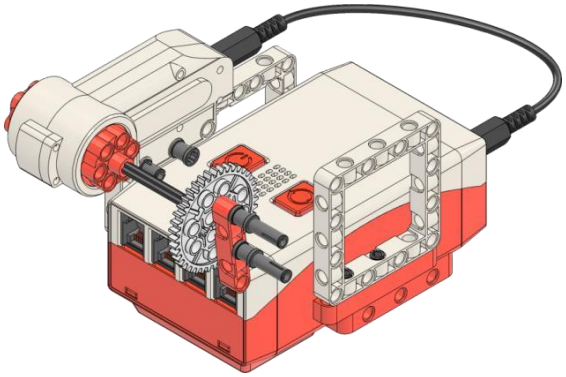


Рис.30 Кулачковый механизм

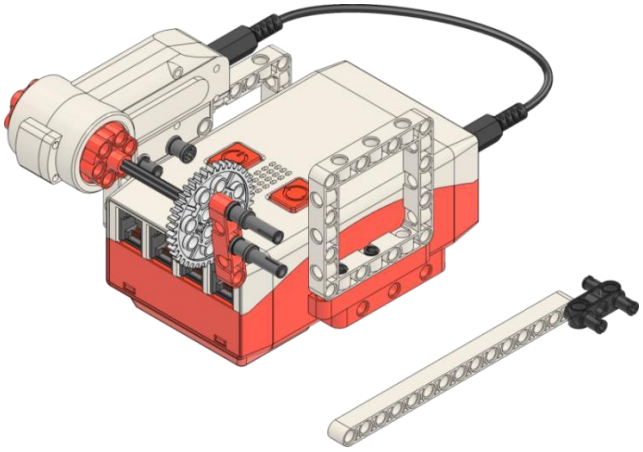


Рис.31 Кулачковый механизм

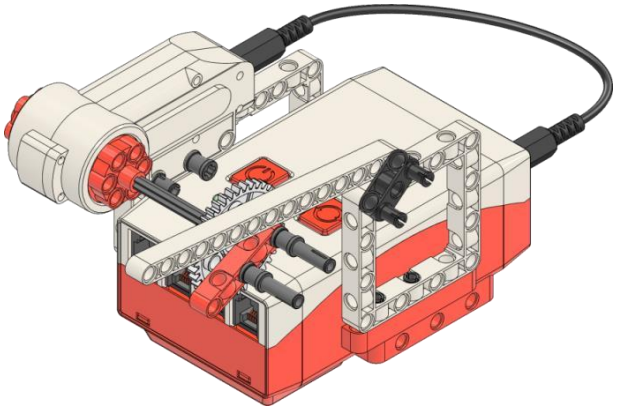


Рис.32 Кулачковый механизм

Запустим программы

Для наглядности эксперимента уменьшите скорость вращения мотора.

Задание 1 (Уровень А)

Используя только кулачковый механизм, соберите мобильного робота.

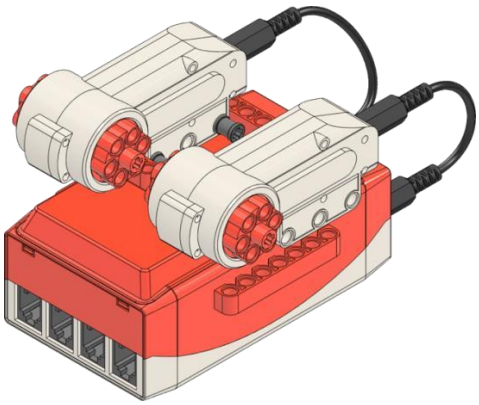


Рис.33 Кулачковый механизм

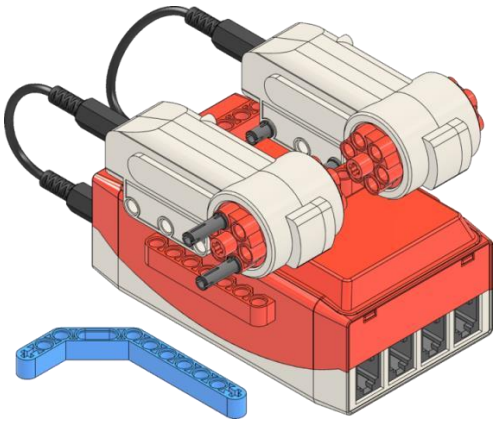


Рис.34 Кулачковый механизм

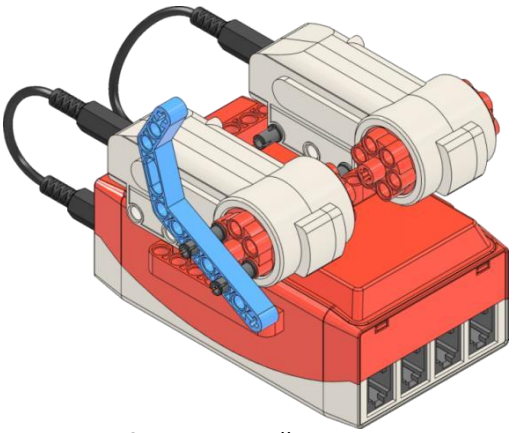


Рис.35 Кулачковый механизм

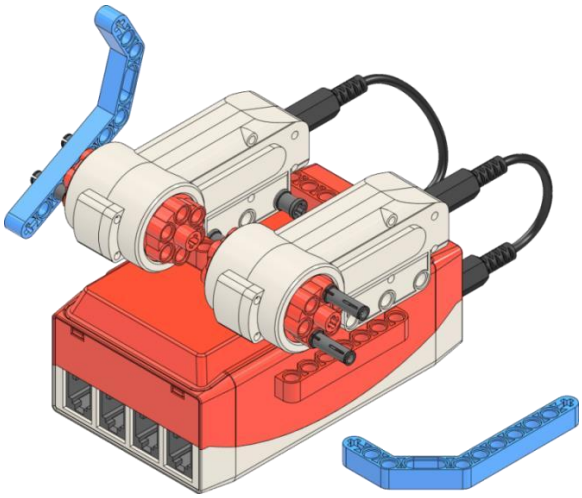


Рис.36 Кулачковый механизм

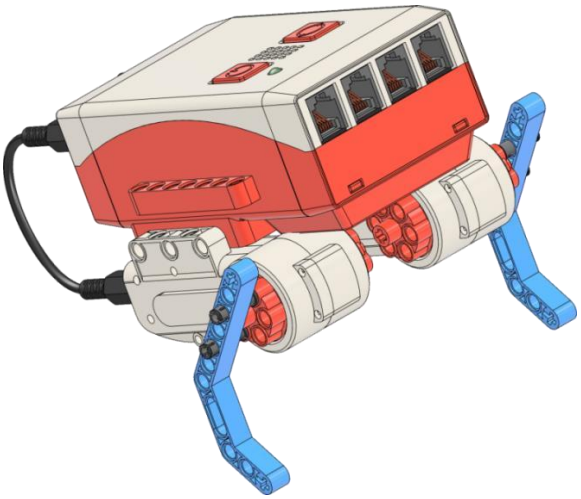


Рис.37 Кулачковый механизм

Составим программу для управления роботом в среде Arduino ide и mBlock5

stopo_hod

```
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 126);
  analogWrite(v2, 126);
  digitalWrite(m1, HIGH);
  digitalWrite(m2, HIGH);

delay(5000);
  analogWrite(v1, 0);
  analogWrite(v2, 0);

}

void loop() {


    }
```

Рис.38 Кулачковый механизм

Рис.39 Кулачковый механизм

Задание 2 (Уровень В)

Используя конструкцию робота из предыдущего задания, создайте робота перемещающегося на четырёх кулачках, используя только два мотора.

8. Мобильная робототехника

Мы рассмотрели основные возможности электронной части набора, теперь необходимо применить их при решении конкретных задач. Рассмотрим вариацию заданий связанных с популярной темой в робототехнике – это мобильные роботы.

Мобильный робот – это конструкция, способная перемещаться в пространстве и взаимодействовать с окружающей средой.

8.1. Робоплатформа НикиРобот

Методические рекомендации.

Тема: «Мобильный робот с дифференциальным приводом»

Цель: изучить процесс создания и программирования робоплатформ с дифференциальным приводом.

Задачи:

- изучить и закрепить на практике процесс создания мобильных трёхколёсных роботов
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий по данной теме.

Задание 1 (Уровень А)

Соберите мобильного робота согласно представленной инструкции.

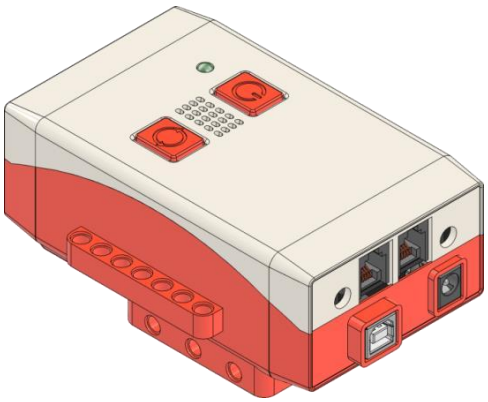


Рис.1 Робоплатформа **КЛИК**

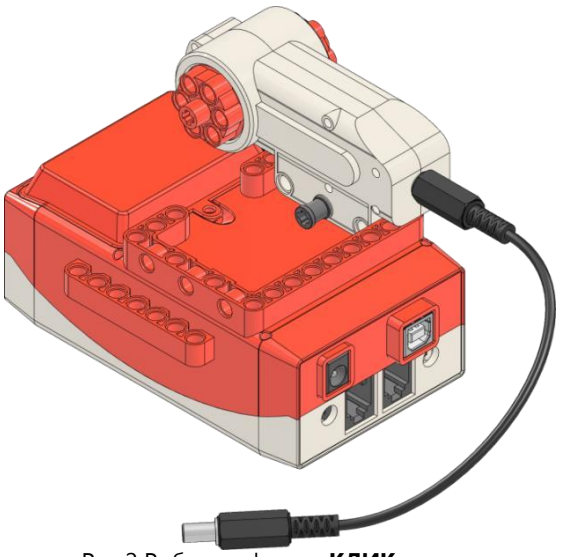


Рис.2 Робоплатформа **КЛИК**

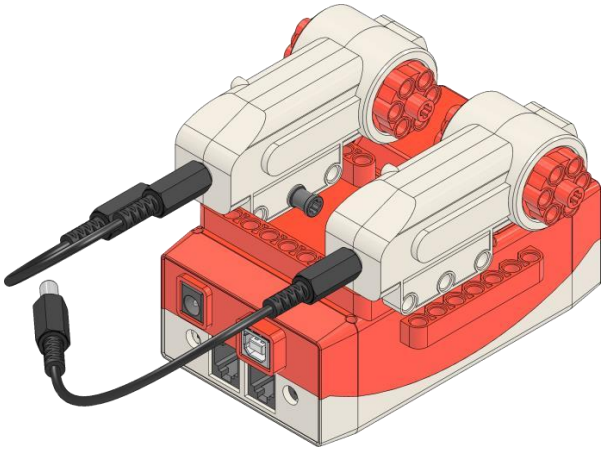


Рис.3 Робоплатформа **КЛИК**

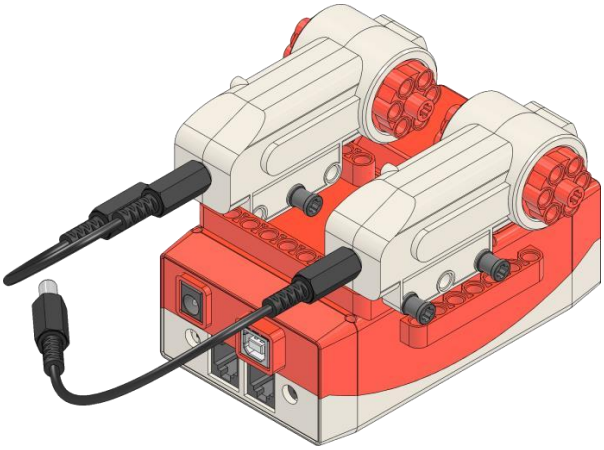


Рис.4 Робоплатформа **КЛИК**

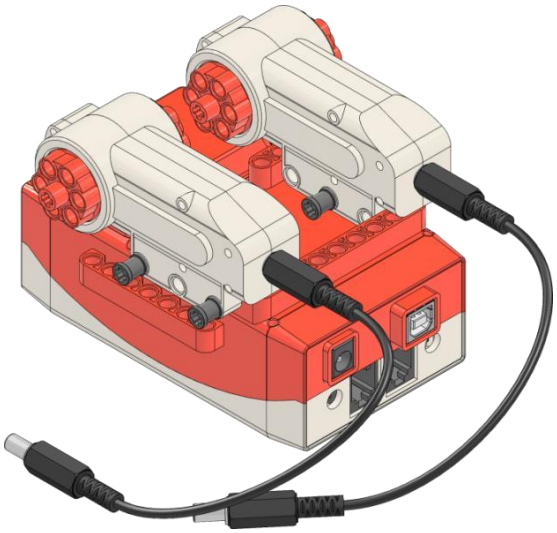


Рис.5 Робоплатформа **КЛИК**

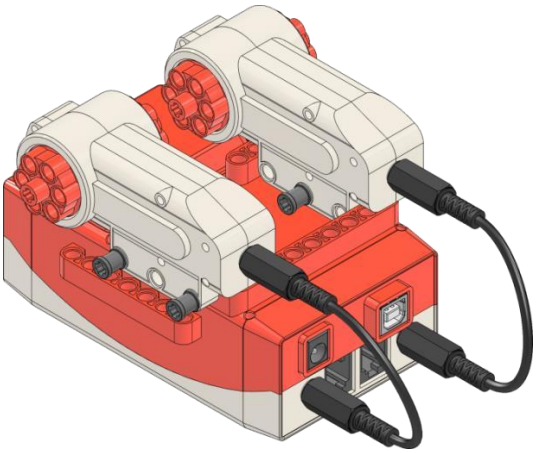


Рис.6 Робоплатформа **КЛИК**

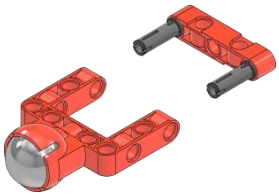


Рис.7 Робоплатформа **КЛИК**

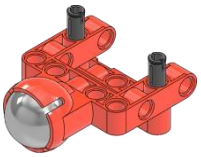


Рис.8 Робоплатформа **КЛИК**

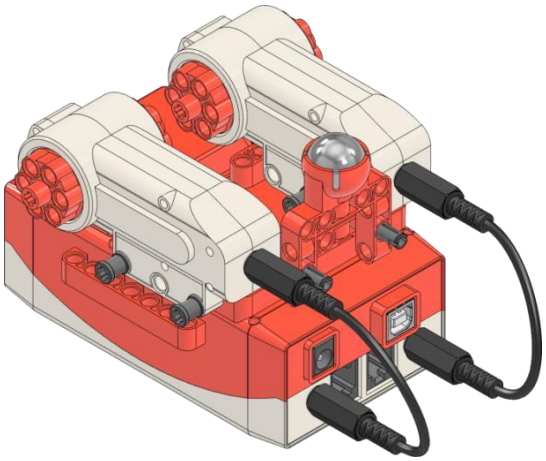


Рис.9 Робоплатформа **КЛИК**

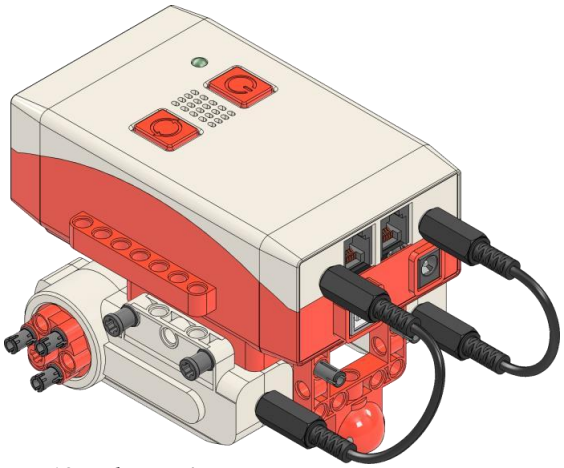


Рис.10 Робоплатформа **КЛИК**

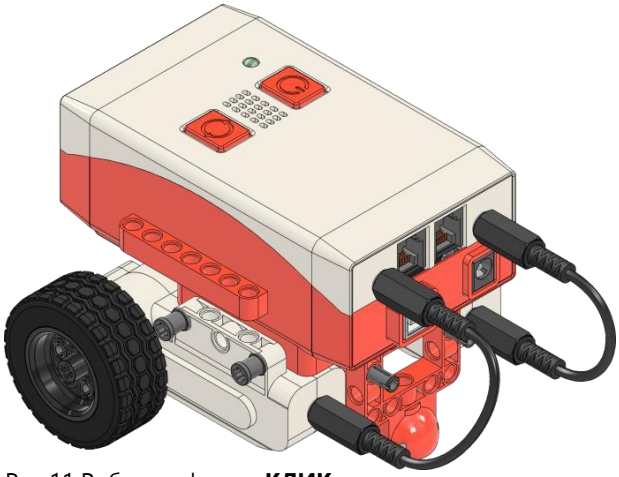


Рис.11 Робоплатформа **КЛИК**

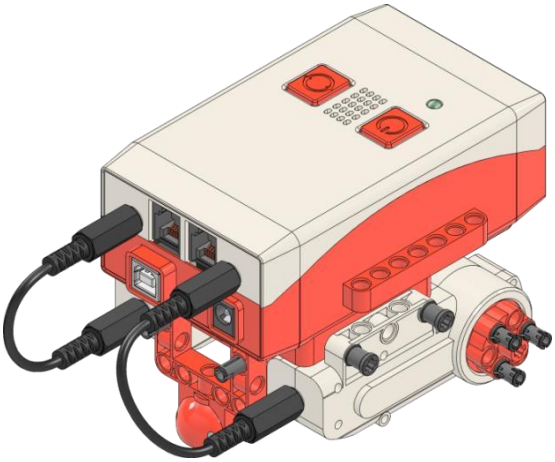


Рис.12 Робоплатформа **КЛИК**

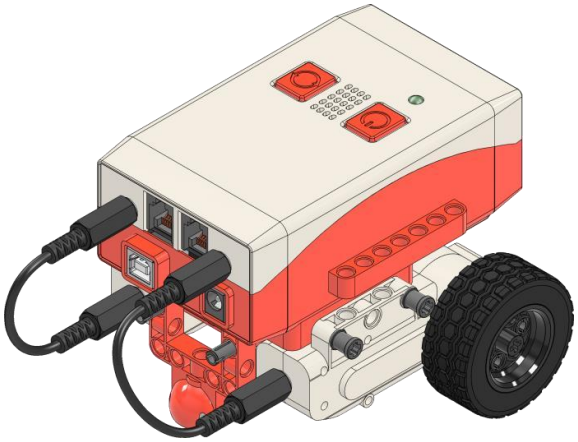


Рис.13 Робоплатформа **КЛИК**

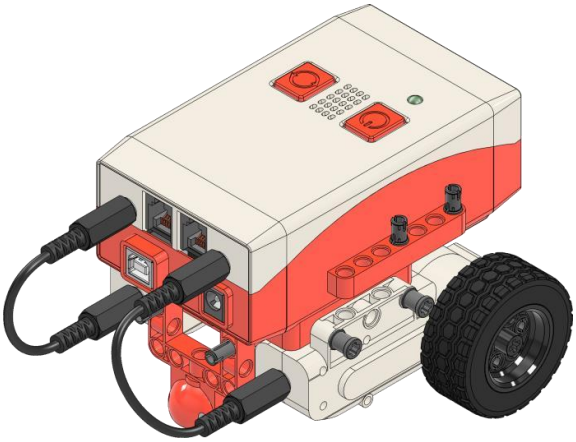


Рис.14 Робоплатформа **КЛИК**

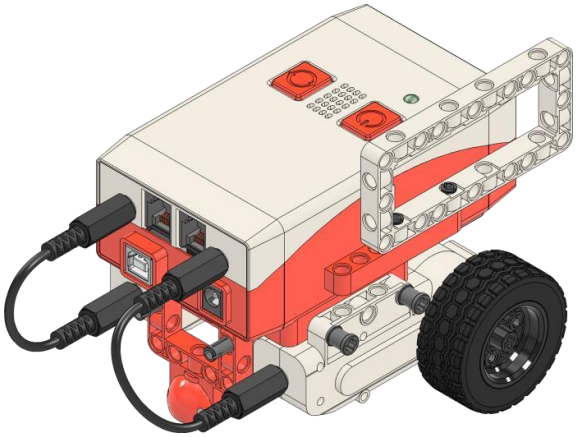


Рис.15 Робоплатформа **КЛИК**

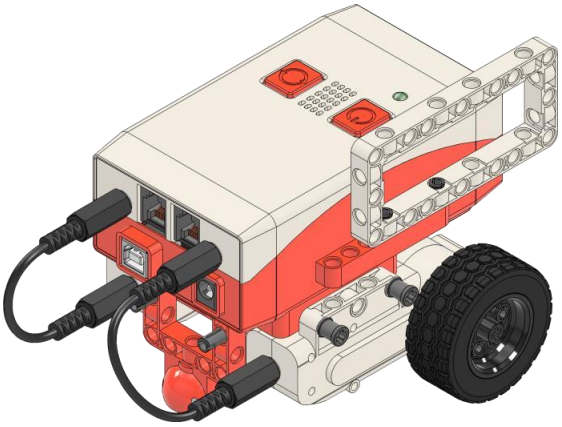


Рис.16 Р Робоплатформа **КЛИК**

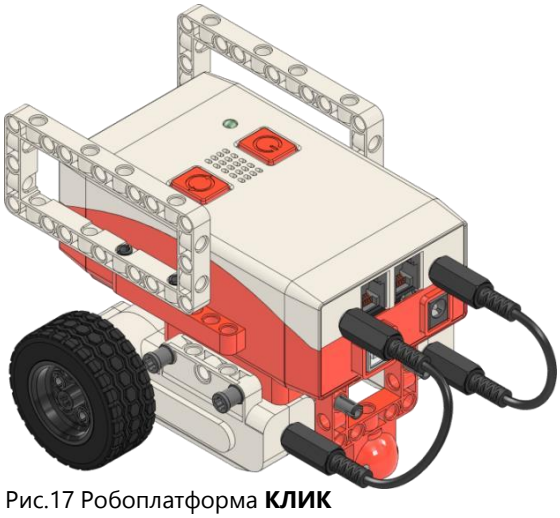


Рис.17 Робоплатформа **КЛИК**

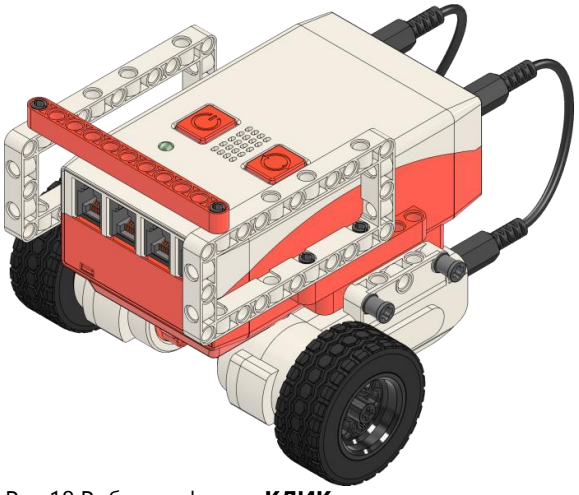


Рис.18 Робоплатформа **КЛИК**

Задание 2 (Уровень А)

Составьте программу, благодаря которой робоплатформа будет перемещаться вперёд две секунды, а затем, разворачиваться на 180^0 и процесс повторяется бесконечно.

Пример решения задания.

Из главы 4.1. мы знаем, как заставить робота двигаться в нужном направлении и знаем, как повернуть его на 90^0 с небольшой погрешностью. Применим полученные знания для данной задачи.

Mobil_robot1

```
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 =  5;


void setup() {
  pinMode(m1, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(v1, OUTPUT);
  pinMode(v2, OUTPUT);

  analogWrite(6, 0);
  analogWrite(5, 0);

}

void loop() {

digitalWrite(m1,LOW);
digitalWrite(m1,LOW);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(2000);

digitalWrite(m1,HIGH);
digitalWrite(m1,LOW);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(900);

}
```

Рис.19 Программа для робоплатформы КЛИК, Arduino ide

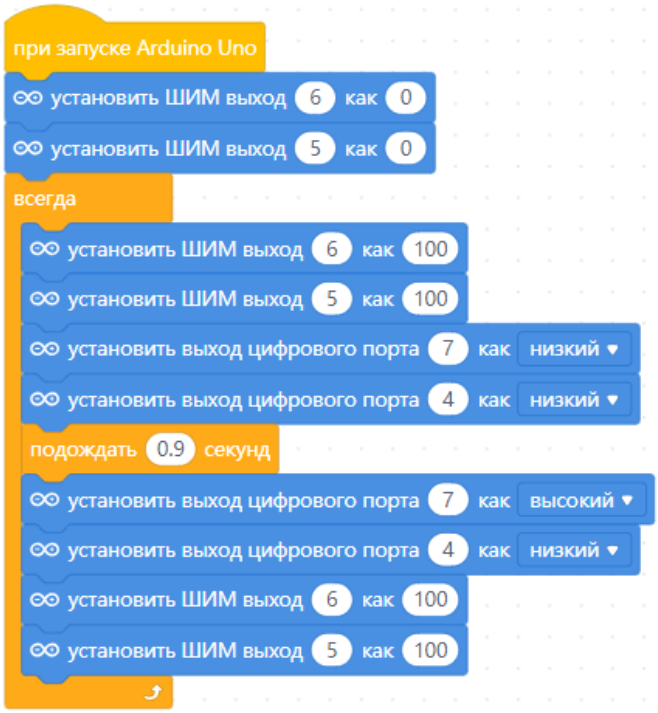


Рис.20 Программа для робоплатформы **КЛИК**, MBlock5

Задание 3 (Уровень В)

Составьте программы, с помощью которых робоплатформа будет перемещаться по замкнутой кривой, образуя:

- квадрат
- шестиугольник
- треугольник

Пример решения.

Зная, как повернуть робота на угол 90^0 и 180^0 , не трудно вычислить и промежуточные углы. Приведём программы по порядку.

Mobil_robot_CVAD

```
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 =  5;


void setup() {
  pinMode(m1, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(v1, OUTPUT);
  pinMode(v2, OUTPUT);

  analogWrite(6, 0);
  analogWrite(5, 0);

for (int i=0; i<4; i++)
{
  digitalWrite(m1,LOW);
digitalWrite(m1,LOW);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(450);

analogWrite(6, 0);
analogWrite(5, 0);
delay(1000);

digitalWrite(m1,LOW);
digitalWrite(m1,HIGH);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(1000);

  analogWrite(6, 0);
  analogWrite(5, 0);
  }

}

void loop() {


}
```

Рис.21 Программа для движения робоплатформы по квадрату Arduino ide

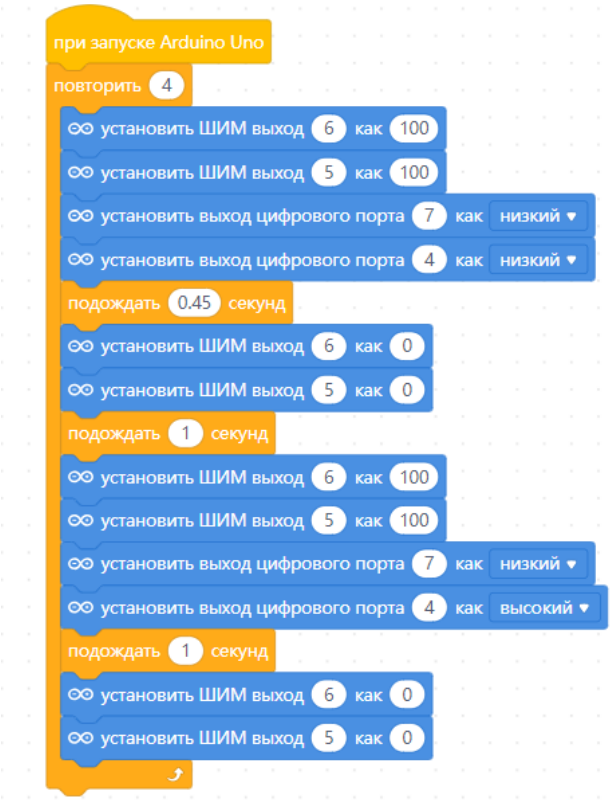


Рис.22 Программа для движения робоплатформы по квадрату MBlock5

Mobil_robot_Shesty

```
int ml = 7;
int m2 = 4;
int vl = 6;
int v2 =  5;


void setup() {
  pinMode(ml, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(vl, OUTPUT);
  pinMode(v2, OUTPUT);

  analogWrite(6, 0);
  analogWrite(5, 0);

for (int i=0; i<6; i++)
{
  digitalWrite(ml,LOW);
digitalWrite(ml,LOW);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(300);

analogWrite(6, 0);
analogWrite(5, 0);
delay(1000);

digitalWrite(ml,LOW);
digitalWrite(ml,HIGH);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(1000);

  analogWrite(6, 0);
  analogWrite(5, 0);
  }

}

void loop() {


}
```

Рис.23 Программа для движения робоплатформы по шестиугольнику Arduino ide

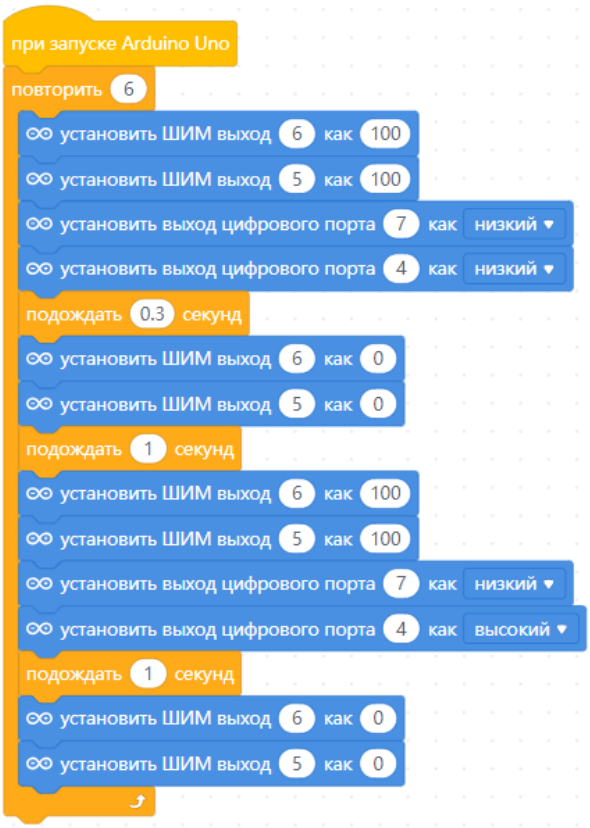


Рис.24 Программа для движения робоплатформы по шестиугольнику MBlock5

Mobil_robot_Treyg

```
int ml = 7;
int m2 = 4;
int vl = 6;
int v2 =  5;


void setup() {
  pinMode(ml, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(vl, OUTPUT);
  pinMode(v2, OUTPUT);

  analogWrite(6, 0);
  analogWrite(5, 0);

for (int i=0; i<3; i++)
{
  digitalWrite(ml,LOW);
digitalWrite(ml,LOW);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(600);

analogWrite(6, 0);
analogWrite(5, 0);
delay(1000);

digitalWrite(ml,LOW);
digitalWrite(ml,HIGH);
analogWrite(6, 100);
 analogWrite(5, 100);
 delay(1000);

  analogWrite(6, 0);
  analogWrite(5, 0);
  }

}

void loop() {


}
```

Рис.25 Программа для движения роботплатформы по треугольнику Arduino ide

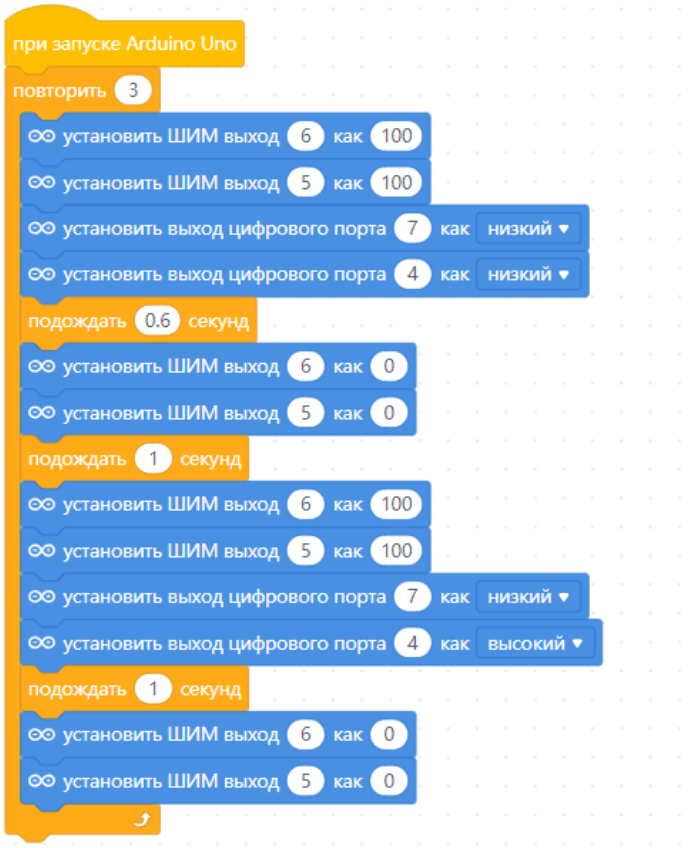


Рис.26 Программа для движения роботплатформы по треугольнику MBlock5

Как видно из программ, отличие только во времени поворота платформы.

8.1.1. Объезд препятствий

Методические рекомендации.

Тема: «Техническое зрение - объезд препятствий»

Цель: изучить процесс создания и программирования робоплатформ, способных, объезжать окружающие их предметы.

Задачи:

- изучить и закрепить на практике процесс создания мобильных трёхколёсных роботов для объезда препятствий.
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий.

Задание 1 (Уровень А)

Соберите платформу согласно главе 5.1 и дополните её конструкцией, инструкция к которой расположена ниже.

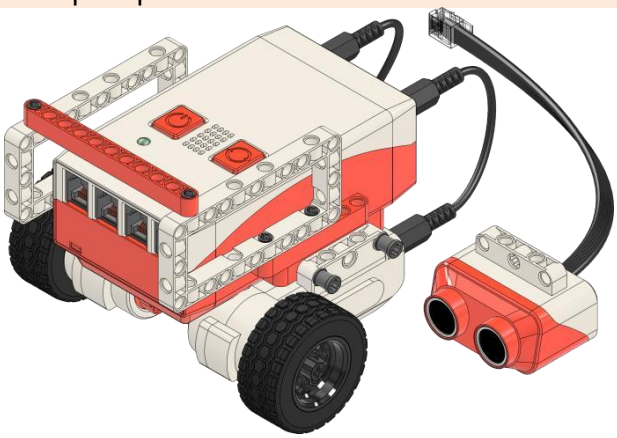


Рис.27 Объезд препятствий

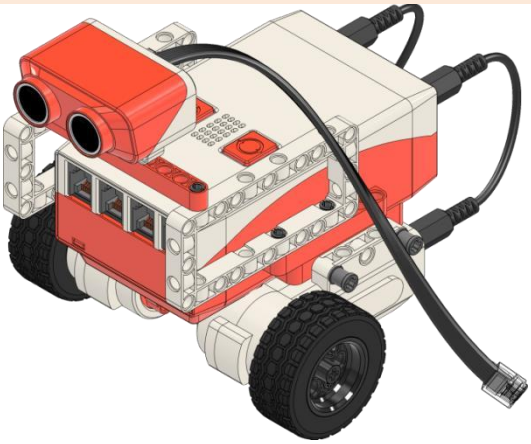


Рис.28 Объезд препятствий

Рис.29 Объезд препятствий

Задание 2 (Уровень В)

Создайте программу, которая позволит роботу объезжать препятствия.

Пример решения.

Опираясь на главы 4.1 и 4.3, создадим программу, с помощью которой, будем получать значение о расстоянии до предмета, и при достижении критического значения (например 30 см) будем подавать на робота команду для его поворота.

ultrason_motor §

```
int ml = 7;
int m2 = 4;
int vl = 6;
int v2 = 5;
int trig = 12;
int echo = 13;
long dur, cm;

void setup() {
Serial.begin (9600);
pinMode(trig, OUTPUT);
pinMode(echo, INPUT);
 pinMode(ml,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(vl,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(vl, 0);
  analogWrite(v2, 0);
}
void loop(){
digitalWrite(trig, LOW);
delayMicroseconds(5);
digitalWrite(trig, HIGH);
delayMicroseconds(10);
digitalWrite(trig, LOW);
pinMode(echo, INPUT);
dur = pulseIn(echo, HIGH);
cm = (dur/2) / 29.1;
Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(300);

if ( cm>=30 ){
      analogWrite(vl, 100);
      analogWrite(v2, 100);
      digitalWrite(ml, LOW);
       digitalWrite(m2, HIGH);
      }
else{
      analogWrite(vl, 100);
      analogWrite(v2, 100);
      digitalWrite(ml, LOW);
       digitalWrite(m2, LOW);
      }
    }
```

Рис.30 Программа - объезд препятствий Arduino ide

8.1.2. Поиск объекта

Методические рекомендации.

Тема: «Техническое зрение – следование за предметом»

Цель: изучить процесс создания и программирования робоплатформ, способных, следить за объектом

Задачи:

- изучить и закрепить на практике процесс создания мобильных трёхколёсных роботов для движения за объектом.
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Задание 1 (Уровень В)

Создайте программу, с помощью которой робот будет двигаться за предметом, который попадает заданную область расстояния.

Пример решения.

Опираясь на главы 4.1 и 4.3, создадим программу, с помощью которой, будем получать значение о расстоянии до предмета, и при достижении критического значения из промежутка (например, 5 - 20 см) будем подавать на робота команду для его движения вперёд, в противном случае поворот.

ultrason_motor §

```
int ml = 7;
int m2 = 4;
int vl = 6;
int v2 = 5;
int trig = 12;
int echo = 13;
long dur, cm;

void setup() {
Serial.begin (9600);
pinMode(trig, OUTPUT);
pinMode(echo, INPUT);
 pinMode(ml,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(vl,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(vl, 0);
  analogWrite(v2, 0);
}
void loop(){
digitalWrite(trig, LOW);
delayMicroseconds(5);
digitalWrite(trig, HIGH);
delayMicroseconds(10);
digitalWrite(trig, LOW);
pinMode(echo, INPUT);
dur = pulseIn(echo, HIGH);
cm = (dur/2) / 29.1;
Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(300);
```

```
if ( cm>=30 ){
       analogWrite(v1, 100);
       analogWrite(v2, 100);
       digitalWrite(m1, LOW);
        digitalWrite(m2, HIGH);
       }
else{
       analogWrite(v1, 100);
       analogWrite(v2, 100);
       digitalWrite(m1, LOW);
        digitalWrite(m2, LOW);
       }
     }
```

Рис.31 Программа – движение за объектом Arduino ide

8.1.3. Захват объекта

Методические рекомендации.

Тема: «Захват и удержание предмета роботом»

Цель: изучить процесс создания и программирования робоплатформ, способных, захватывать и перемещать предметы в окружающей среде.

Задачи:

- изучить и закрепить на практике процесс создания мобильных трёхколёсных роботов для захвата предметов.
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Задание 1 (Уровень А)

Дополните конструкцию из главы 5.1 и 5.2 деталями по инструкции, расположенной ниже.

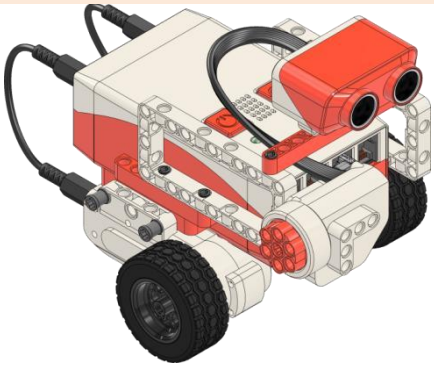


Рис.32 Захват объекта

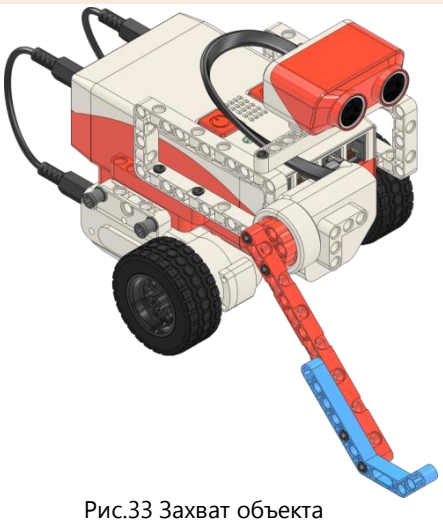


Рис.33 Захват объекта

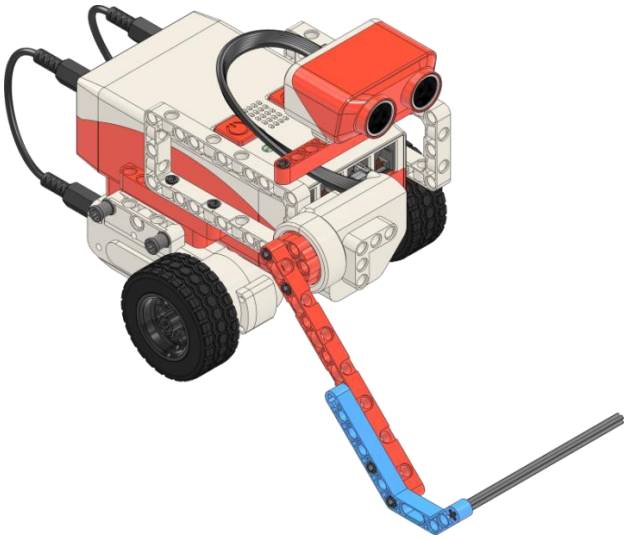


Рис.34 Захват объекта

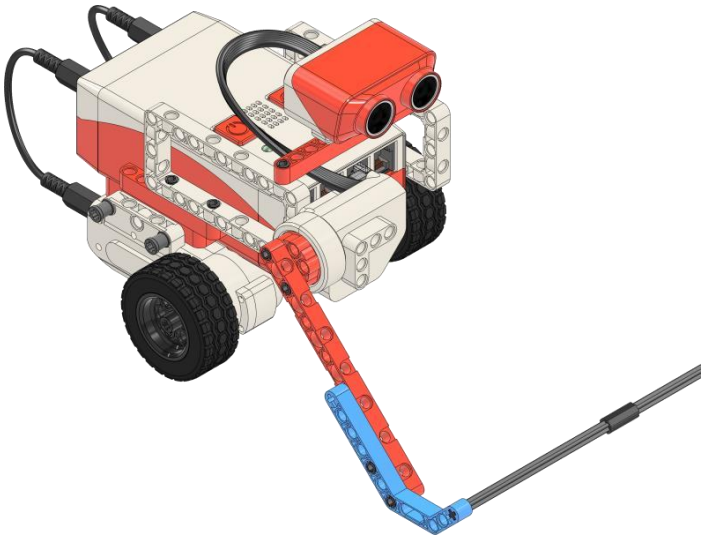


Рис.35 Захват объекта

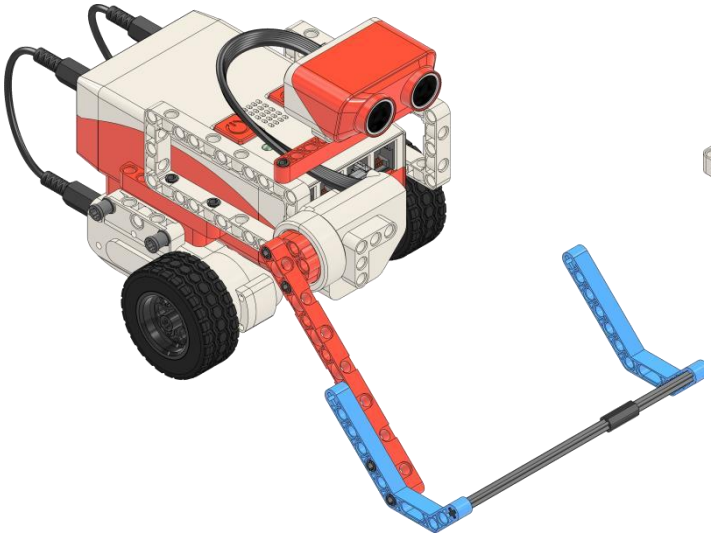


Рис.36 Захват объекта

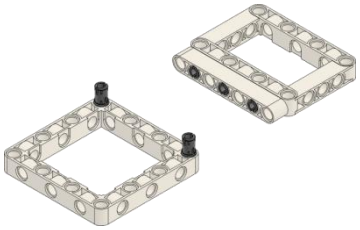


Рис.37 Захват объекта

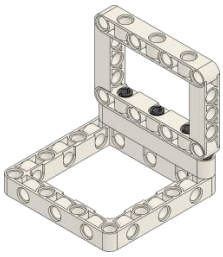


Рис.38 Захват объекта

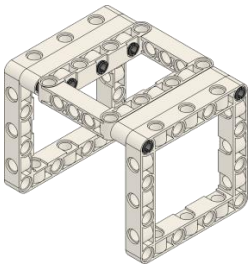


Рис.39 Захват объекта

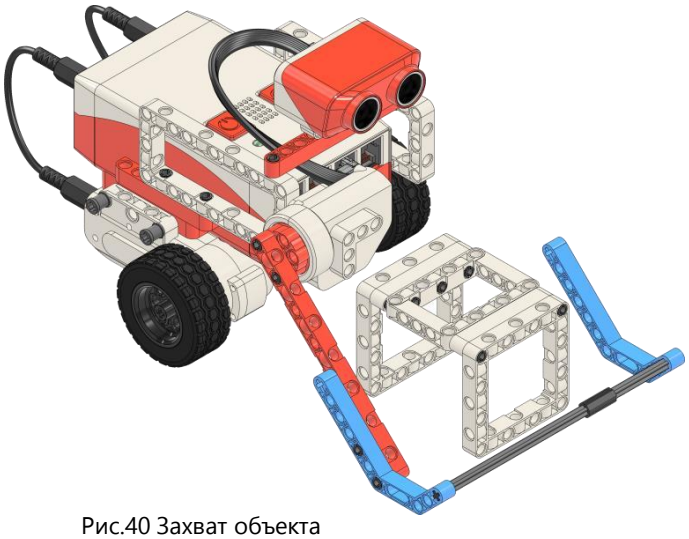


Рис.40 Захват объекта

Задание 2 (Уровень В)

Создайте программу, с помощью которой робот будет захватывать предмет и перемещать его с помощью поворота вокруг себя и отпускать предмет. Можно сделать процесс циклическим.

Пример решения.

Первым делом нам необходимо определить градусы положения сервопривода в точке, где забрало опущено и в точке – где поднято.

В нашем случае это 10^0 и 100^0.

Создадим программу, в которой робот с начало производит захват предмета, затем поворачивается и отпускает предмет. После этого процесс повторяется.

```
#include <Servo.h>
Servo servo_8;
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;

void setup() {
  servo_8.attach(8);
  pinMode(m1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(v2,OUTPUT);
}

void loop() {
servo_8.write(10);
delay(1000);

analogWrite(v1,100);
analogWrite(v2,100);
digitalWrite(m1,1);
digitalWrite(m2,1);
delay(1000);

analogWrite(v1,0);
analogWrite(v2,0);
servo_8.write(100);
delay(1);
  }
```

Рис.41 Программа по захвату объекта на arduino ide

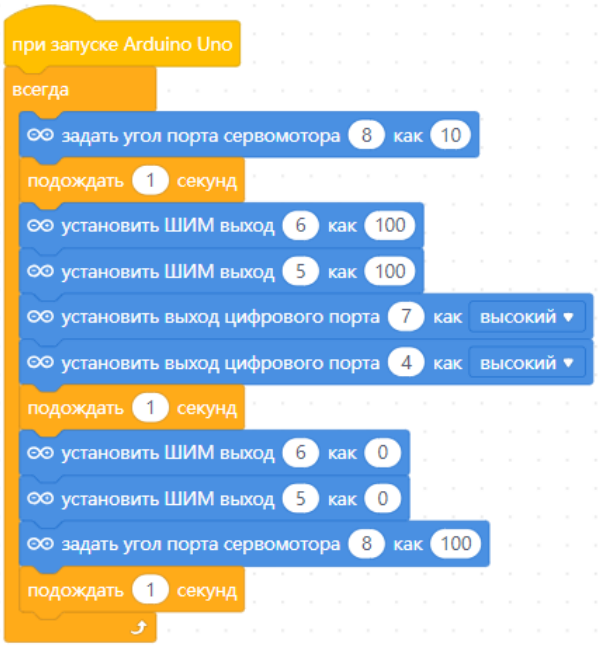


Рис.42 Программа по захвату объекта на MBlock5

8.1.4. Движение по линии

Методические рекомендации.

Тема: «Движение по линии»

Цель: изучить процесс создания и программирования робоплатформ, способных, следовать по кривой линии.

Задачи:

- изучить и закрепить на практике процесс создания мобильных трёхколёсных роботов для движения по маркерам.
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий.

Задание 1 (Уровень А)

Соберите робоплатформу, способную двигаться по линии, согласно представленной инструкции.

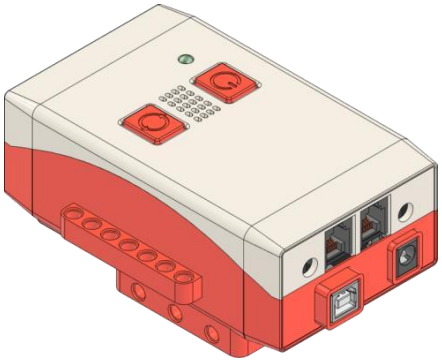


Рис.43 Движение по линии

Рис.44 Движение по линии

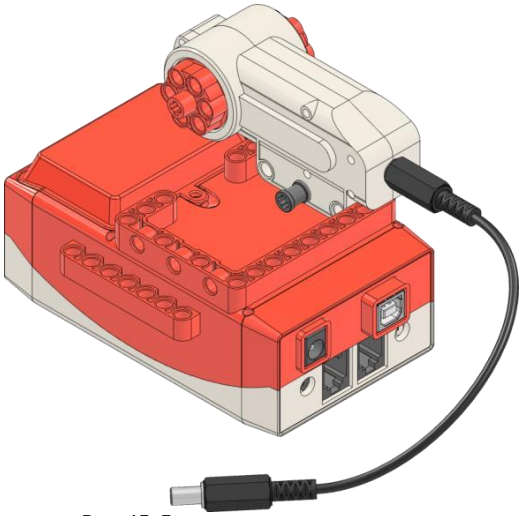


Рис.45 Движение по линии

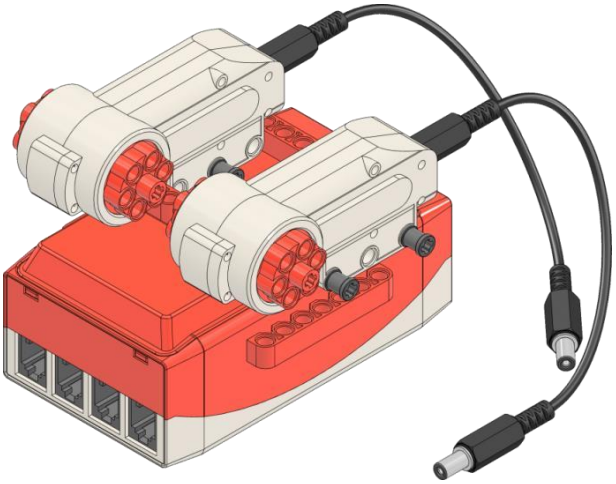


Рис.46 Движение по линии

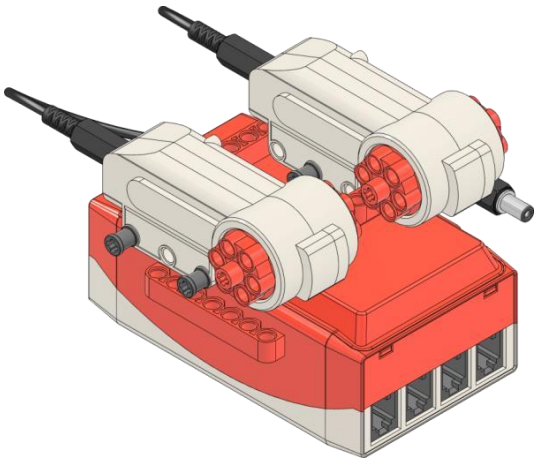


Рис.47 Движение по линии

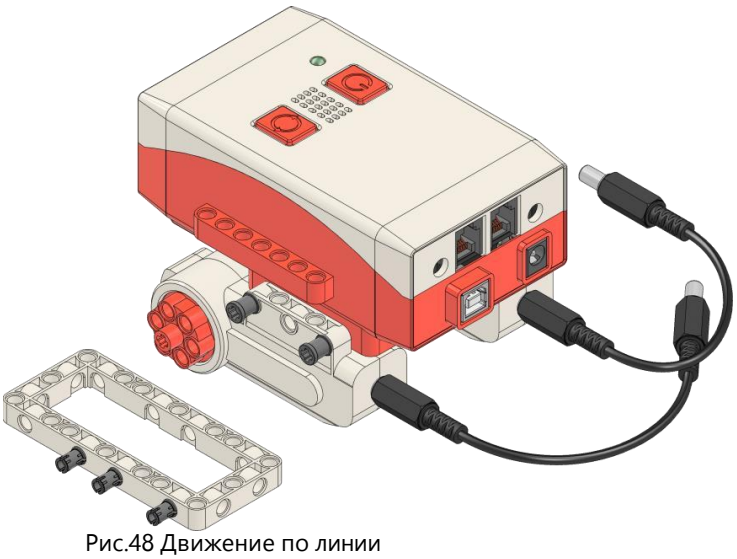


Рис.48 Движение по линии

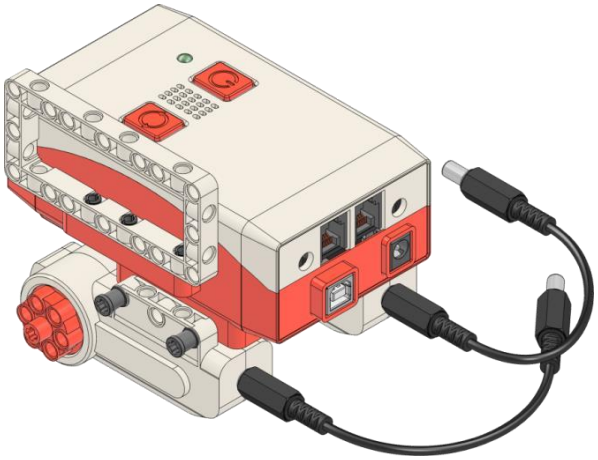


Рис.49 Движение по линии

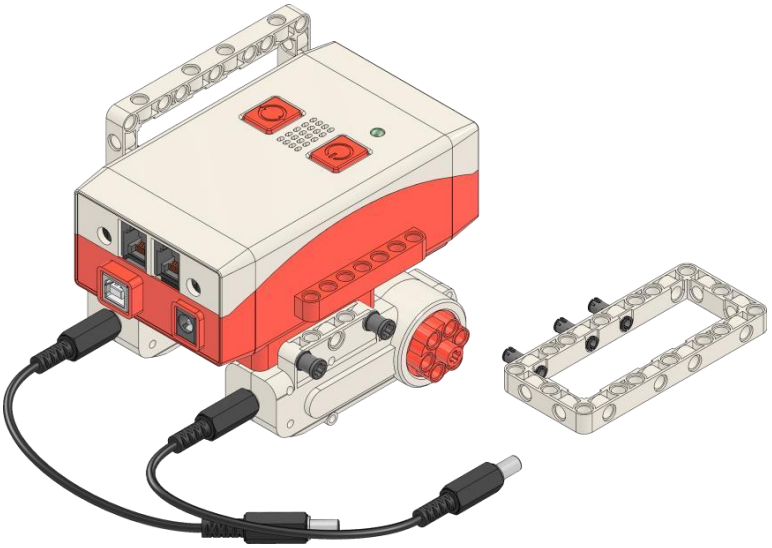


Рис.50 Движение по линии

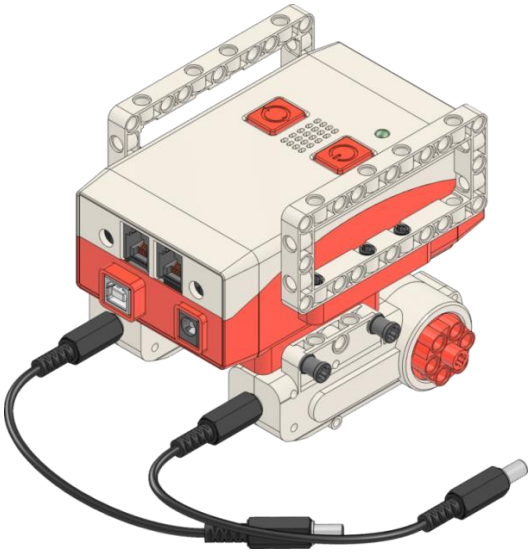


Рис.51 Движение по линии

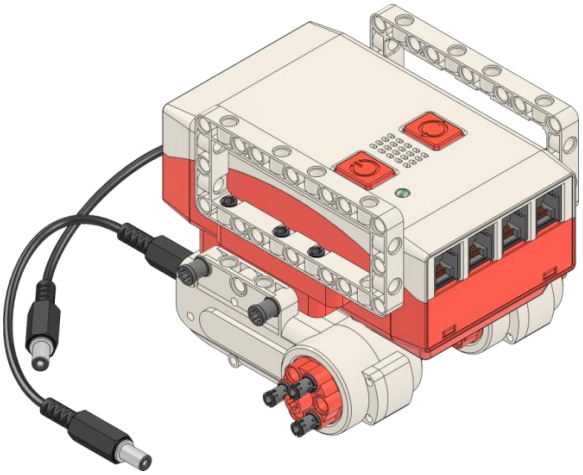


Рис.52 Движение по линии

Рис.53 Движение по линии

Рис.54 Движение по линии

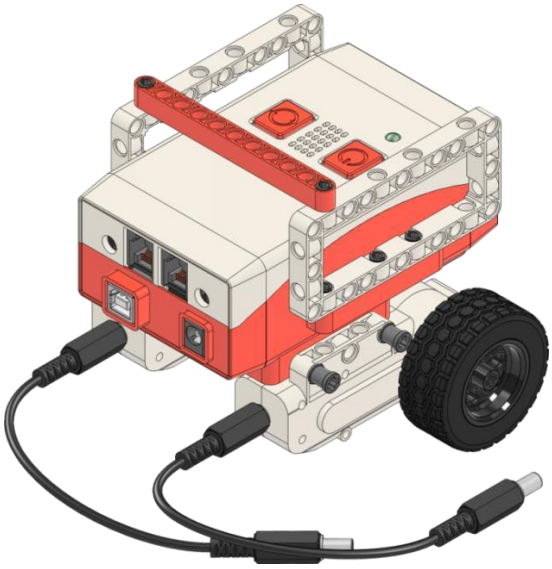


Рис.55 Движение по линии

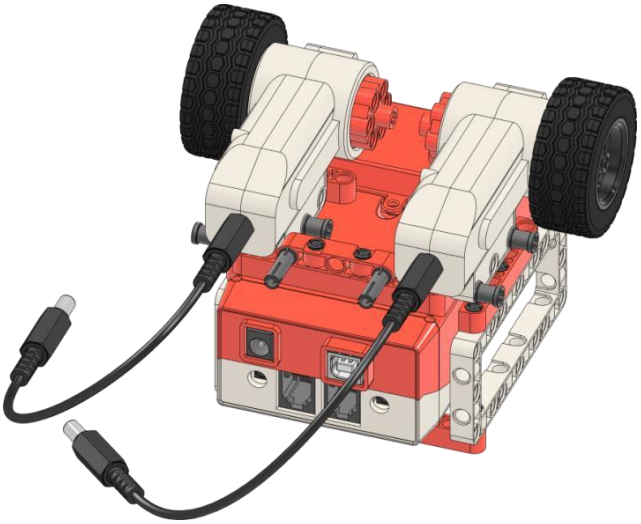


Рис.56 Движение по линии

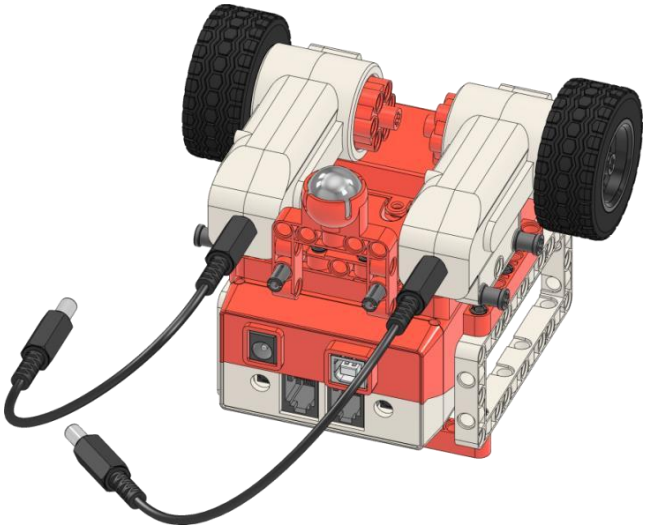


Рис.57 Движение по линии

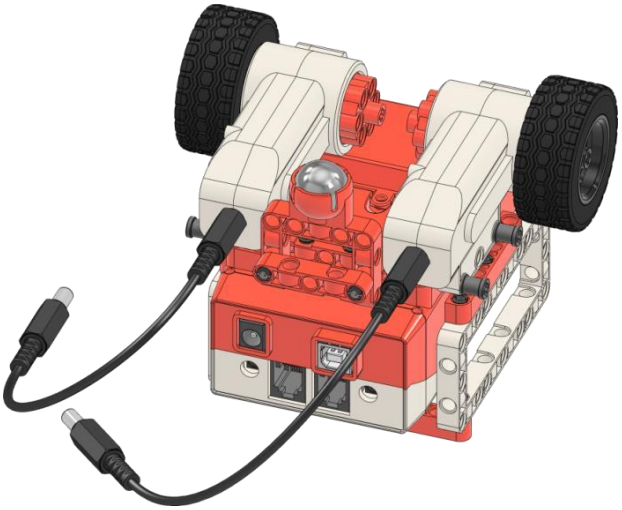


Рис.58 Движение по линии

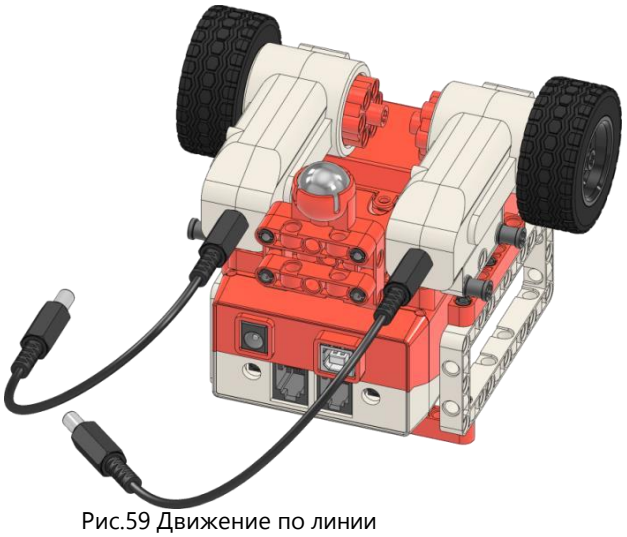


Рис.59 Движение по линии

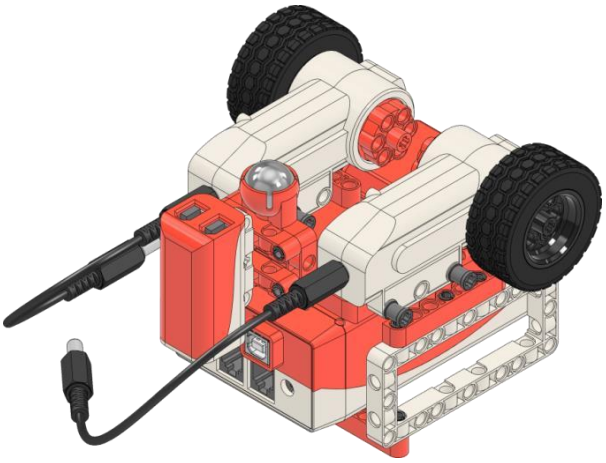


Рис.60 Движение по линии

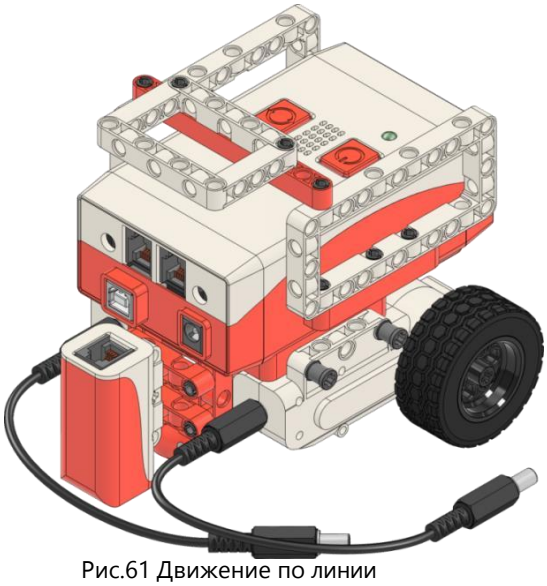


Рис.61 Движение по линии

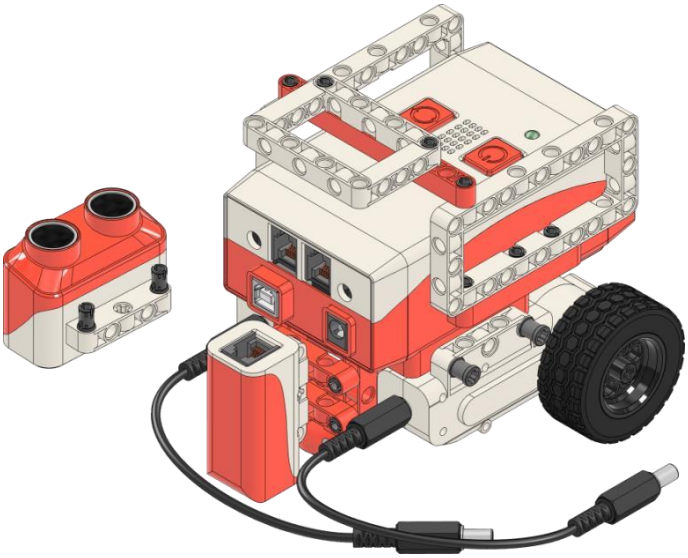


Рис.62 Движение по линии

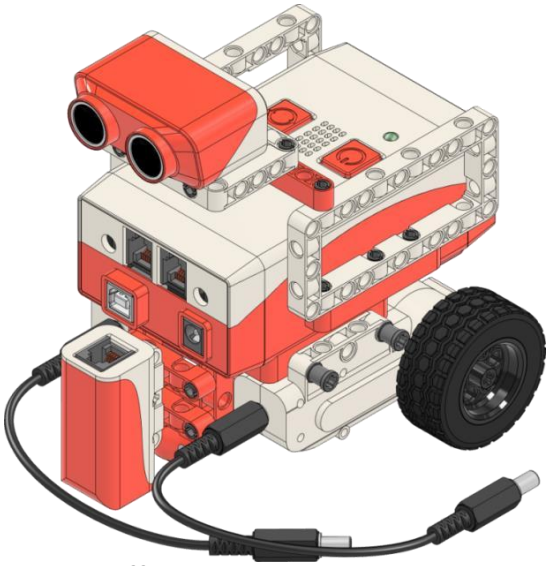


Рис.63 Движение по линии

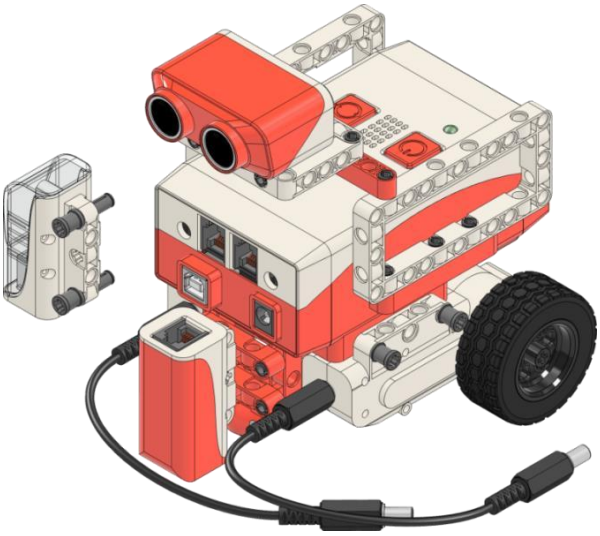


Рис.64 Движение по линии

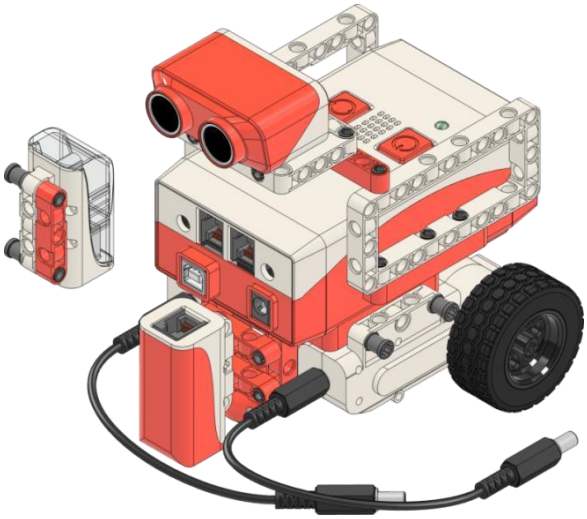


Рис.65 Движение по линии

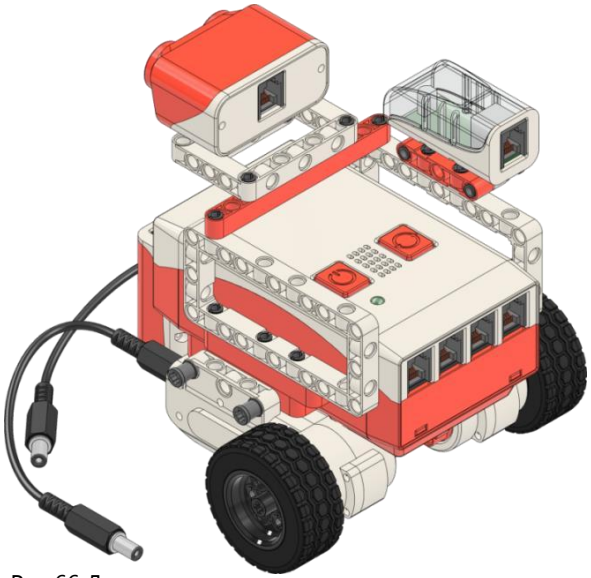


Рис.66 Движение по линии

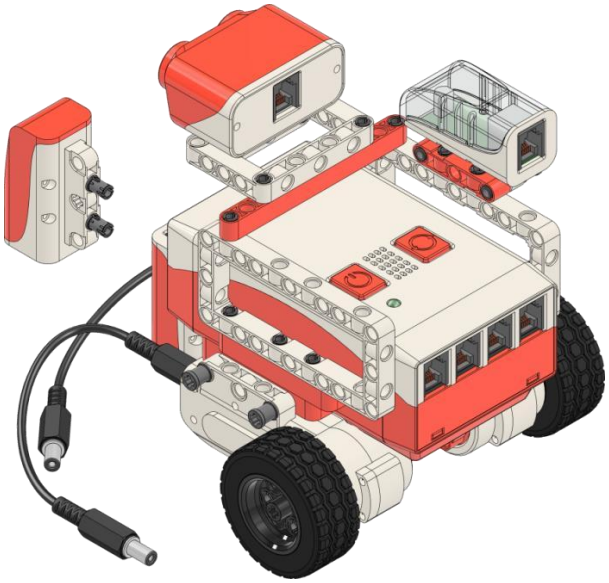


Рис.67 Движение по линии

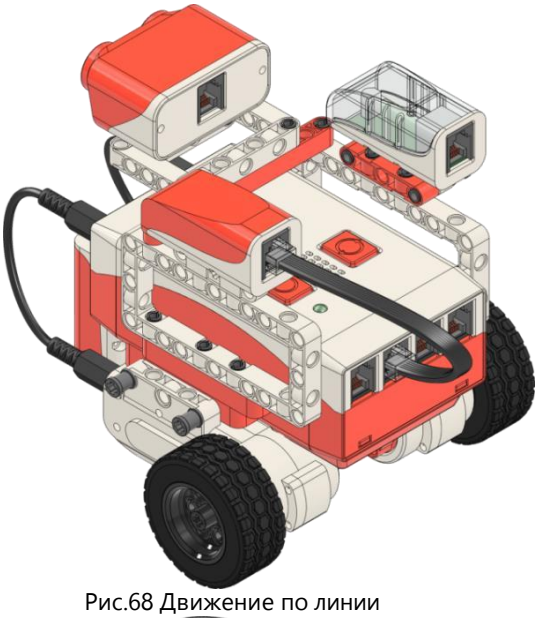


Рис.68 Движение по линии

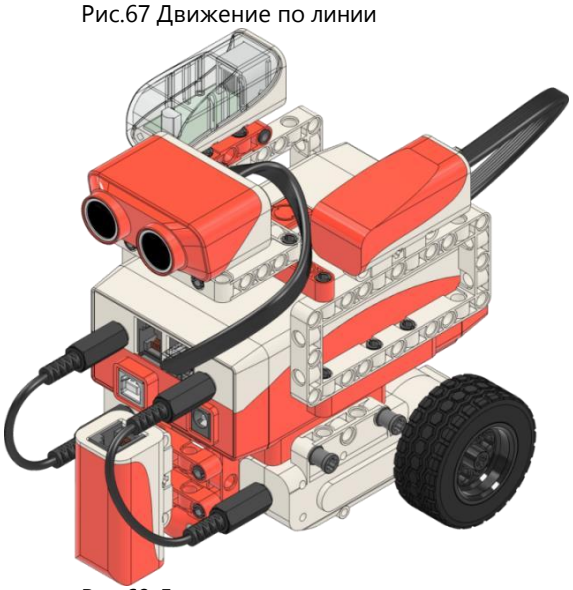


Рис.69 Движение по линии

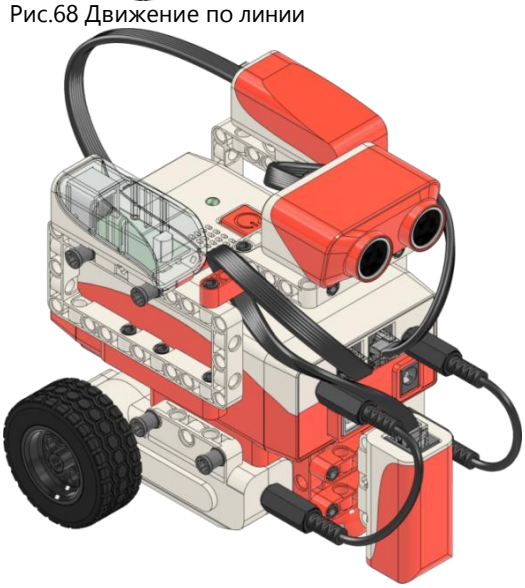


Рис.70 Движение по линии

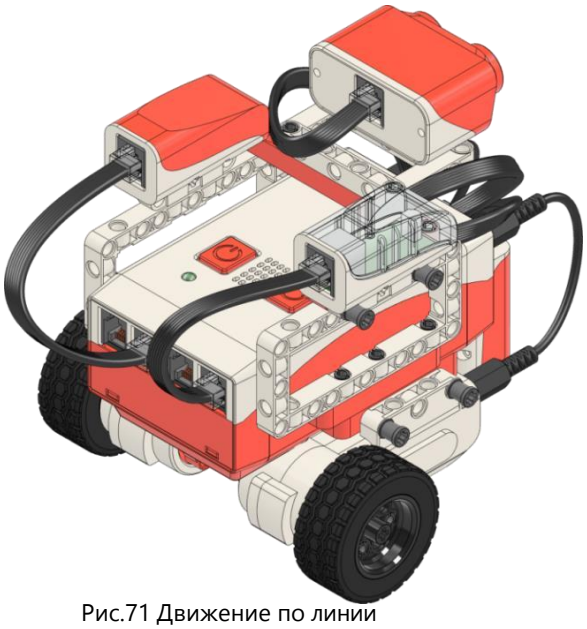


Рис.71 Движение по линии

Задание 2 (Уровень В)

Создайте программу, с помощью которой робот будет двигаться по линии.

Пример решения:

- Создадим программу, где в зависимости от показания левого или правого датчика линии будут переключаться моторы, так чтобы робот поворачивался. Необходимо сделать задержку, чтобы было совершено движение вперёд, иначе робот будет поворачиваться влево или вправо на одном месте.

```
int l1 =8;
int l2 =2;

int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;

void setup() {
 pinMode(l1,INPUT);
pinMode(l2,INPUT);
pinMode(m1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 0);
  analogWrite(v2, 0);

Serial.begin(9600);
}

void loop() {

    if ( digitalRead(l1)== 1 && digitalRead(l2)== 0 ){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, LOW);
       digitalWrite(m2, LOW);
       delay(20);
      }

       else if ( digitalRead(l1)== 0 && digitalRead(l2)== 1 ){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, HIGH);
       delay(20);
      }
      else if( digitalRead(l1)== 0 && digitalRead(l2)== 0 )
      {
       analogWrite(v1, 100);
      analogWrite(v2, 100);
     digitalWrite(m1, LOW);
digitalWrite(m2, HIGH);
delay(20);
      }
  Serial.println("left_line");
  Serial.println(digitalRead(l1));
  Serial.println("right_line");
  Serial.println(digitalRead(l2));
 delay(50);
}
```

Рис.72 Программа для движения по линии

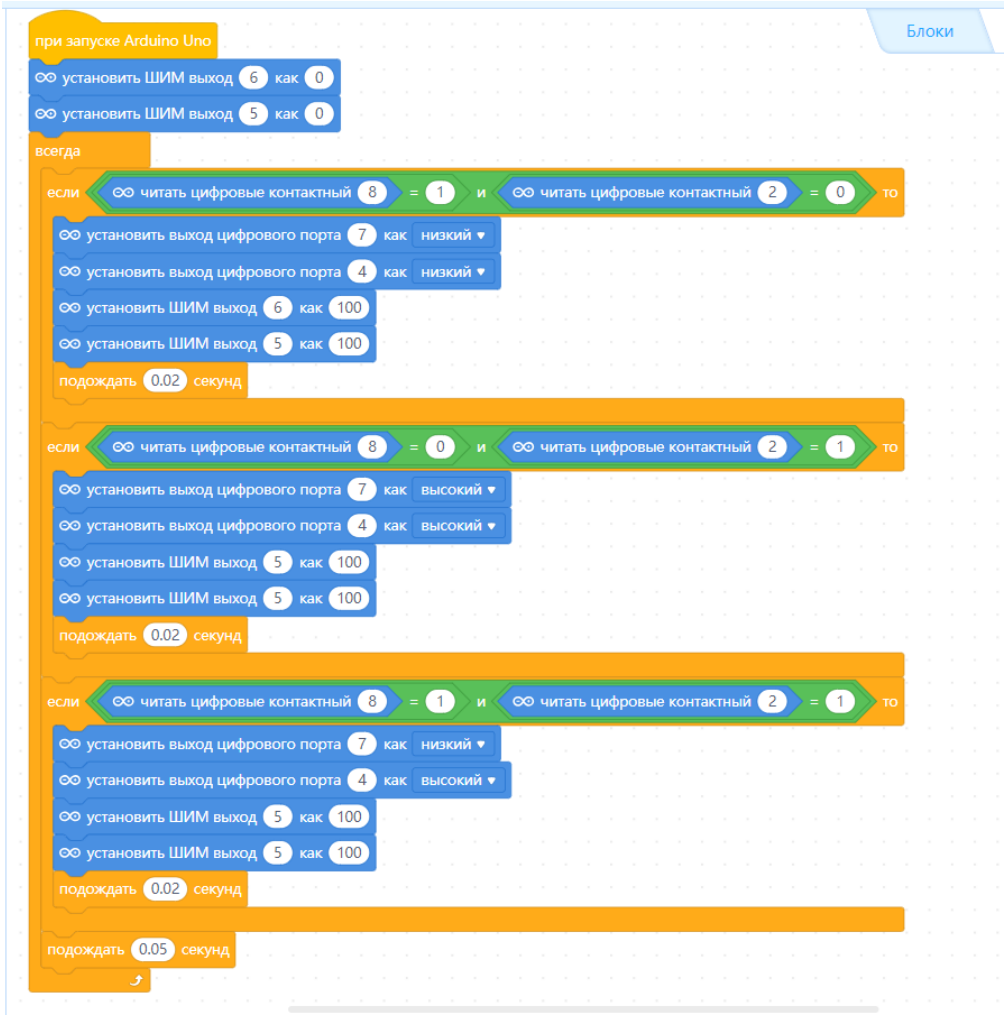


Рис.73 Программа для движения по линии в MBlock5

Задание 3 (Уровень С)

Создайте программу для более плавного и быстрого движения робота вдоль линии.

Пример решения.

- Изменим в предыдущей программе скорости моторов, но не их направление. Тем самым мы добьемся плавного движения.

```
int l1 =8;
int l2 =2;
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;

void setup() {
 pinMode(l1,INPUT);
pinMode(l2,INPUT);
pinMode(m1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  Serial.begin(9600);
}

void loop() {

    if ( digitalRead(l1)== 1 && digitalRead(l2)== 0 ){
      analogWrite(v1, 100);
      analogWrite(v2, 50);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, LOW);
       delay(20);
      }

       else if ( digitalRead(l1)== 0 && digitalRead(l2)== 1 ){
      analogWrite(v1, 50);
      analogWrite(v2, 100);
      digitalWrite(m1, LOW);
       digitalWrite(m2, HIGH);
       delay(20);
      }
      else if( digitalRead(l1)== 0 && digitalRead(l2)== 0 )
      {
       analogWrite(v1, 100);
      analogWrite(v2, 100);
     digitalWrite(m1, LOW);
digitalWrite(m2, HIGH);
delay(20);
      }
  Serial.println("left_line");
  Serial.println(digitalRead(l1));
  Serial.println("right_line");
  Serial.println(digitalRead(l2));
 delay(50);
}
```

Рис.74 Программа для плавного движения по линии

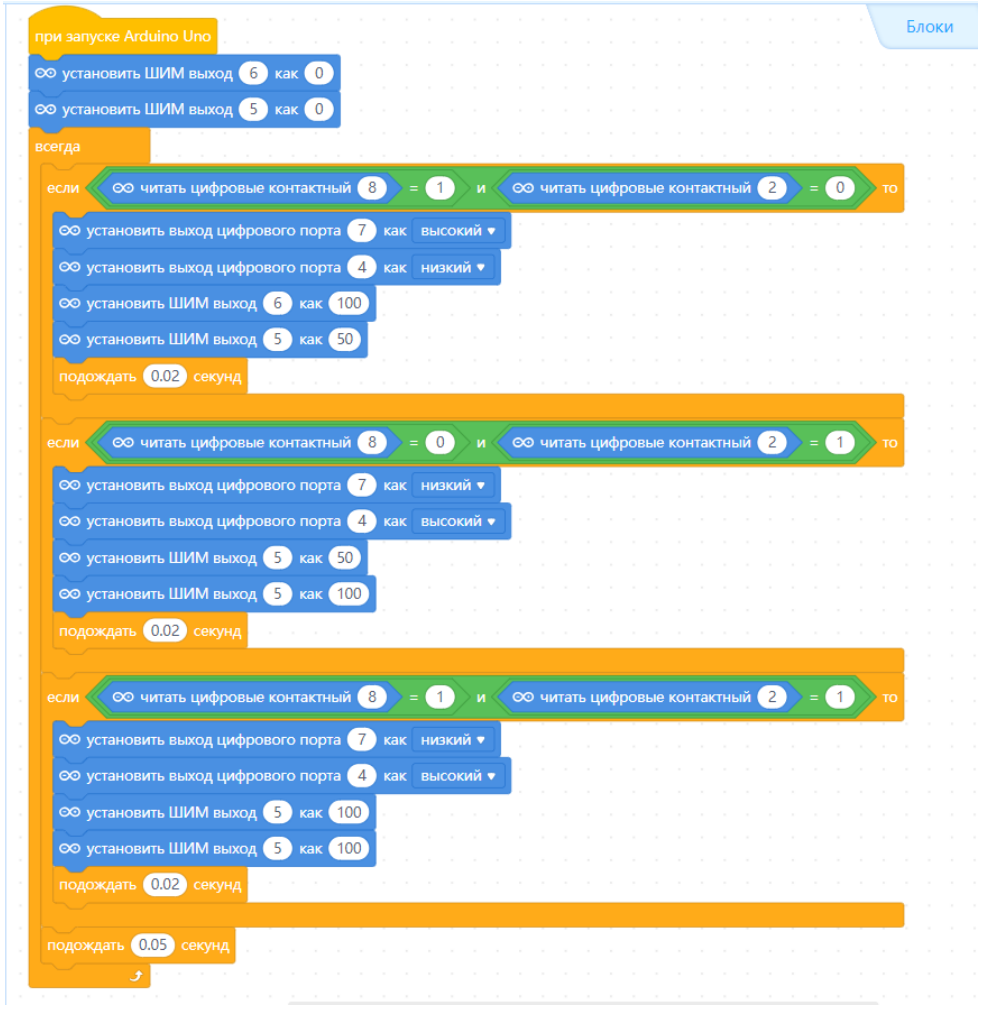


Рис.75 Программа для плавного движения по линии в MBlock5

8.1.5. Управление по IR

Методические рекомендации.

Тема: «Дистанционное управление по IR»

Цель: изучить процесс создания и программирования роботплатформ, работу которых можно контролировать по IR

Задачи:

- изучить и закрепить на практике процесс создания мобильных трёхколёсных роботов для управления по IR.
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Пример заданий.

Задание 1 (Уровень А)

Соберите мобильного робота согласно главе 5.5.

Задание 2 (Уровень В)

Создайте программу, с помощью которой можно управлять роботом через IR модуль, с помощью IR пульта.

Пример решения:

- Используя знания по созданию программ для управления моторами робота и работы с IR модулем, совместим в программу. В программе будет задействован алгоритм, который постоянно считывает данные с модуля и если пришёл сигнал, то он будет сравнивать данный сигнал с прописанными условиями.

```
#include <IRremote.h>

int RECV_PIN = 8;
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;
IRrecv irrecv(RECV_PIN);

decode_results results;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
  pinMode(m1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
}

void loop() {
  if (irrecv.decode(&results)) {
    //Serial.println(results.value, HEX);
    Serial.println(results.value);
    irrecv.resume();

    if (results.value == 16720605){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
     digitalWrite(m1, LOW);
      digitalWrite(m2, HIGH);
     }
    else if (results.value == 16761405){
      analogWrite(v1, 0);
      analogWrite(v2, 0);
      }
  else if (results.value == 16712445){
       analogWrite(v1, 100);
       analogWrite(v2, 100);
       digitalWrite(m1, HIGH);
        digitalWrite(m2, LOW);
       }
 else if (results.value == 16753245){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, HIGH);
      }
     else if (results.value == 16769565){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, LOW  );
       digitalWrite(m2, LOW);

        }
   }
 }
```

Рис.76 Программа для управление по IR

8.1.6. Управление по Bluetooth

Методические рекомендации.

Тема: «Дистанционное управление по Bluetooth»

Цель: изучить процесс создания и программирования робоплатформ, работу которых можно контролировать по Bluetooth.

Задачи:

- изучить и закрепить на практике процесс создания мобильных трёхколёсных роботов для управления по Bluetooth.
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий.

Задание 1 (Уровень А)

Соберите мобильного робота согласно главе 5.5.

Задание 2 (Уровень В)

Создайте программу, с помощью которой можно управлять роботом через Bluetooth модуль, с помощью смартфона.

Пример решения.

- Используя знания по созданию программ для управления моторами робота и работы с Bluetooth модулем, совместим в программу. В программе будет задействован алгоритм, который постоянно считывает данные с модуля и если пришёл сигнал, то он будет сравнивать данный сигнал с прописанными условиями.

```
int ldr =8;
int ser_enable = A4;

int x;

int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;
void setup() {
pinMode(m1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  pinMode(ser_enable, OUTPUT);
  digitalWrite(ser_enable, LOW);
  delay(1);
  digitalWrite(ser_enable, HIGH);
Serial.begin(9600);
}

void loop() {

  if (Serial.available())
  {
     x=Serial.read();
    if (x == '1') // При получении символа "W" движемся вперед
    {
 analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, LOW);
       digitalWrite(m2, HIGH);
delay(1000);
digitalWrite(m1, LOW);
digitalWrite(m2, LOW);
analogWrite(v1, 0);
  analogWrite(v2, 0);
   }
else if (x=='2'){
 analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m2, LOW);
       digitalWrite(m1, HIGH);
delay(1000);
digitalWrite(m1, LOW);
digitalWrite(m2, LOW);
analogWrite(v1, 0);
  analogWrite(v2, 0);
  }
else if (x=='3'){
 analogWrite(v1, 100);
     analogWrite(v2, 100);
     digitalWrite(m1, LOW);
      digitalWrite(m2, LOW);
delay(1000);
digitalWrite(m1, LOW);
digitalWrite(m2, LOW);
analogWrite(v1, 0);
  analogWrite(v2, 0);
  }
 else if(x=='4'){
    analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, HIGH);
delay(1000);
digitalWrite(m1, LOW);
digitalWrite(m2, LOW);
analogWrite(v1, 0);
  analogWrite(v2, 0);
   }
else  {
//x=0;
digitalWrite(m1, LOW);
digitalWrite(m2, LOW);
analogWrite(v1, 0);
  analogWrite(v2, 0);

}
 Serial.println(x);
   }

 }
```

Рис.77 Программа для управление по Bluetooth

9. Инженерные проекты

9.1. Сортировщик цвета

Сортировщик цвета – устройство, относящаяся к инженерным проектам.

Методические рекомендации.

Тема: «Сортировщик цвета»

Цель: изучить процесс создания и программирования устройства, способного определять цвета и сортировать предметы по цвету.

Задачи:

- изучить и закрепить на практике процесс создания сортировщика цветов
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Пример заданий.

Задание 1 (Уровень А)

Соберите устройство согласно инструкции.

Рис1 Сортировщик цветов

Рис.2 Сортировщик цветов

Рис.3 Сортировщик цветов

Рис.4 Сортировщик цветов

Рис.5 Сортировщик цветов

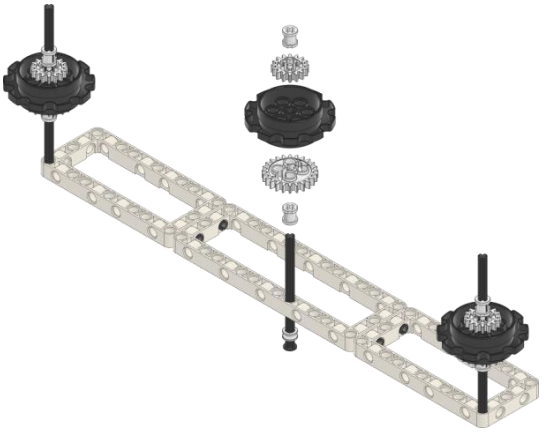


Рис.6 Сортировщик цветов

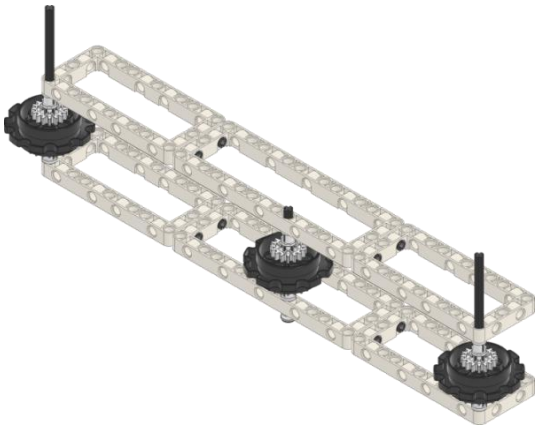


Рис.7 Сортировщик цветов

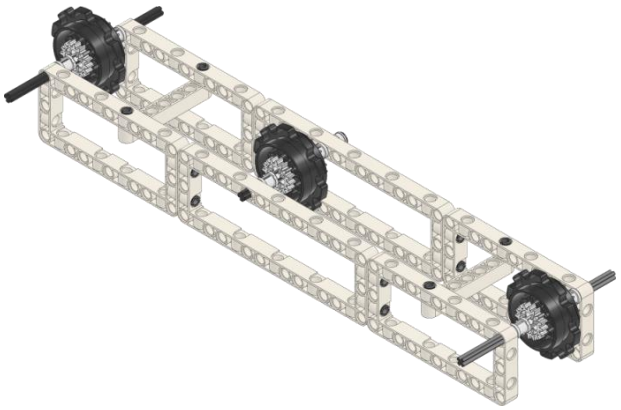


Рис.8 Сортировщик цветов

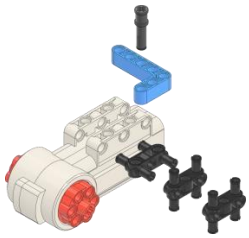


Рис.9 Сортировщик цветов

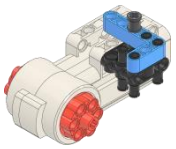


Рис.10 Сортировщик цветов

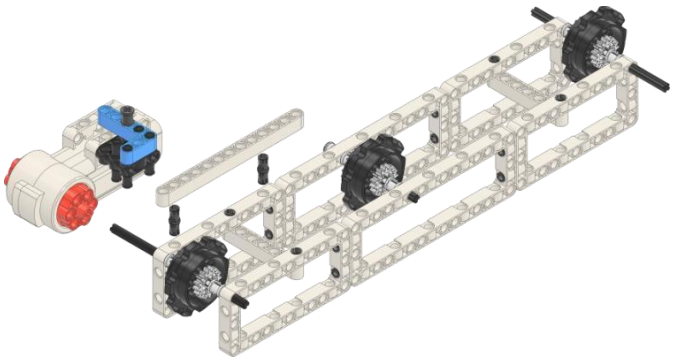


Рис.11 Сортировщик цветов

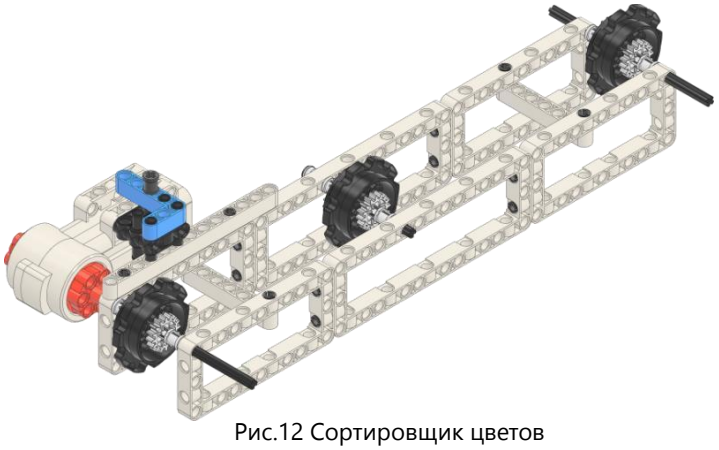


Рис.12 Сортировщик цветов

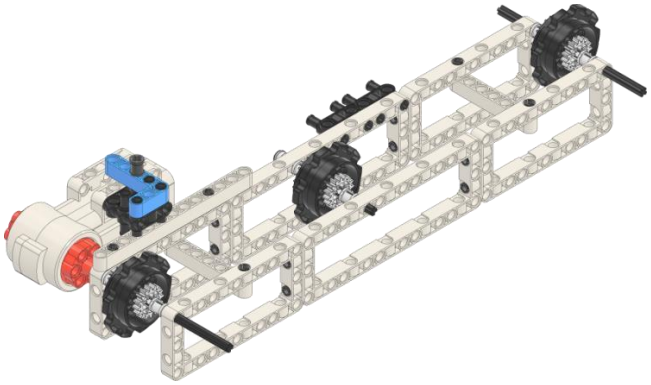


Рис.13 Сортировщик цветов

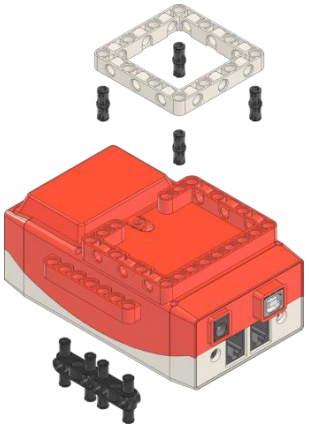


Рис.14 Сортировщик цветов

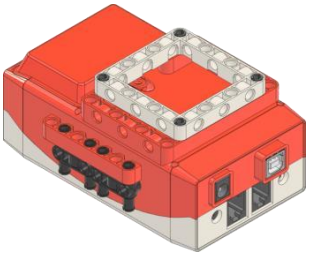


Рис.15 Сортировщик цветов

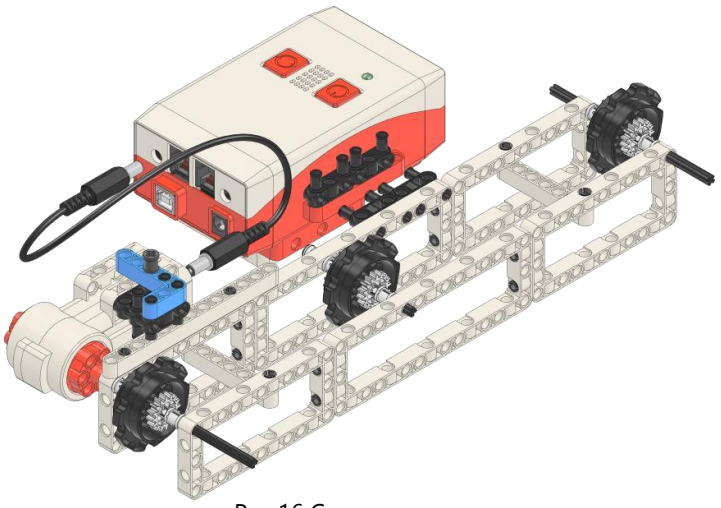


Рис.16 Сортировщик цветов

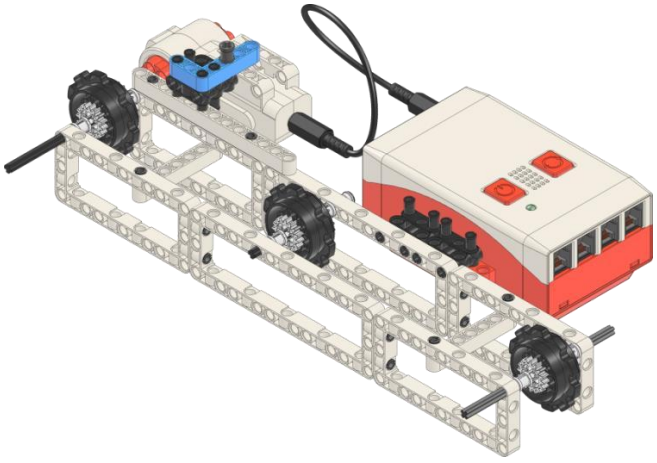


Рис.17 Сортировщик цветов

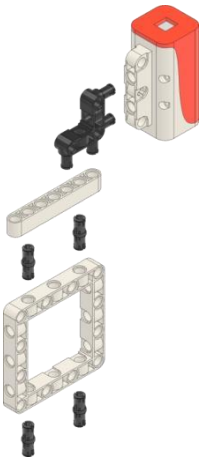


Рис.18 Сортировщик цветов

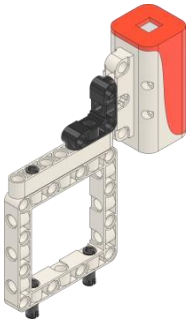


Рис.19 Сортировщик цветов

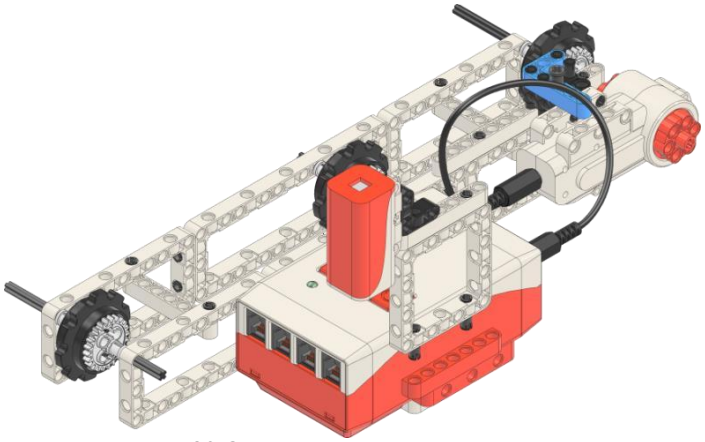


Рис.20 Сортировщик цветов – датчик цвета

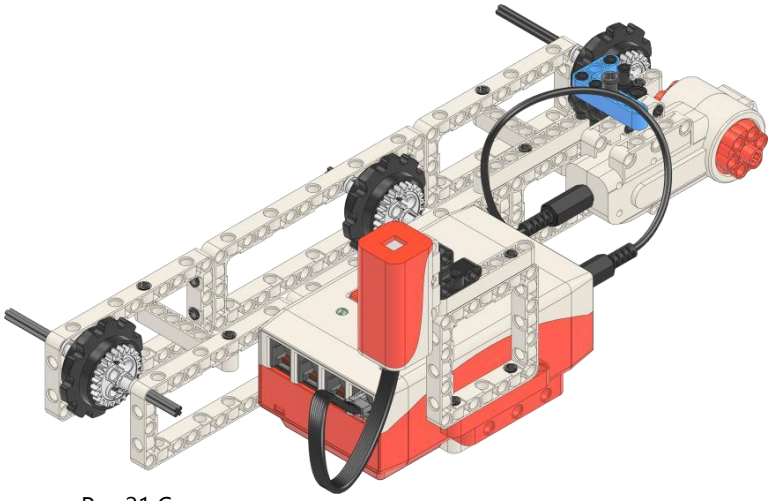


Рис.21 Сортировщик цветов

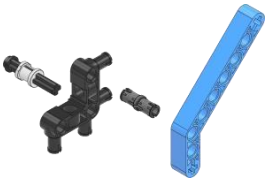


Рис.22 Сортировщик цветов

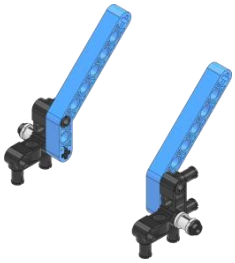


Рис.23 Сортировщик цветов

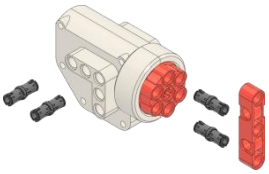


Рис.24 Сортировщик цветов

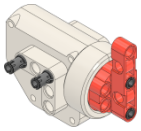


Рис.25 Сортировщик цветов

Рис.26 Сортировщик цветов

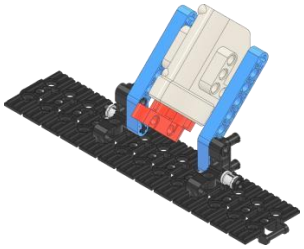


Рис.27 Сортировщик цветов

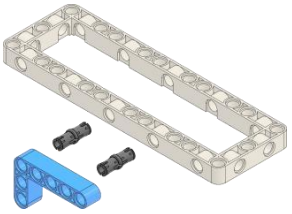


Рис.28 Сортировщик цветов

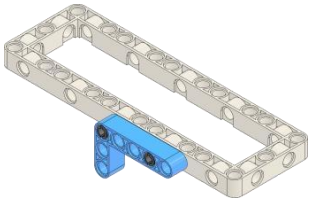


Рис.29 Сортировщик цветов

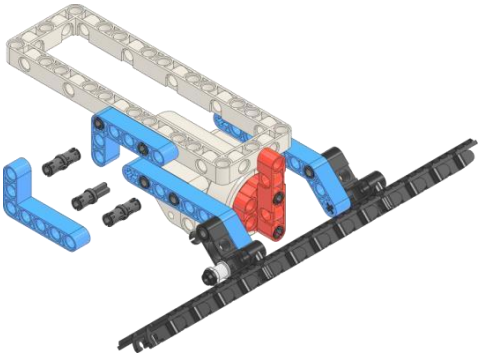


Рис.30 Сортировщик цветов

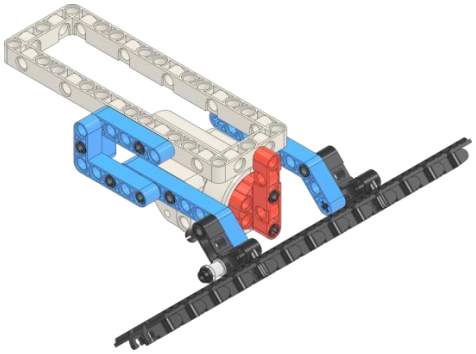


Рис.31.Сортировщик цветов

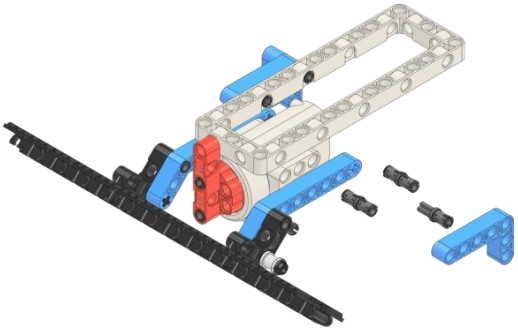


Рис.32.Сортировщик цветов

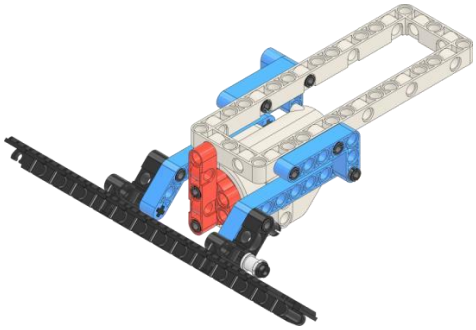


Рис.33 Сортировщик цветов

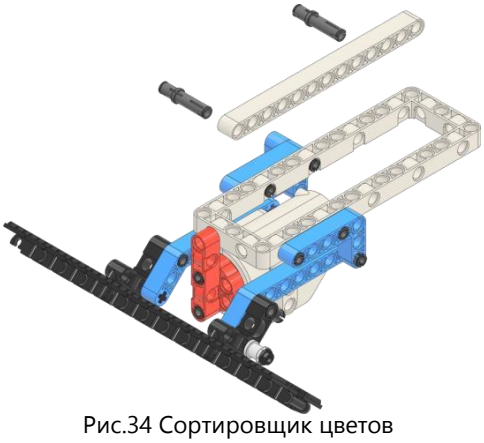


Рис.34 Сортировщик цветов

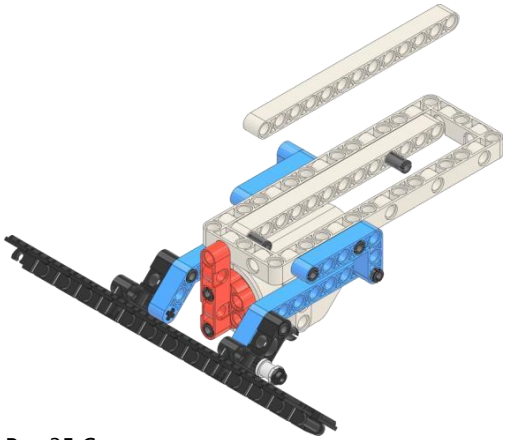


Рис.35 Сортировщик цветов

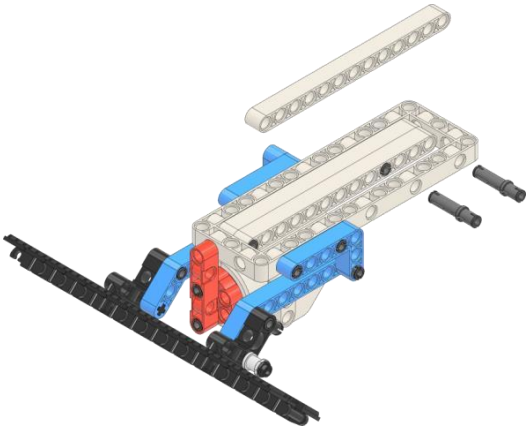


Рис.36 Сортировщик цветов

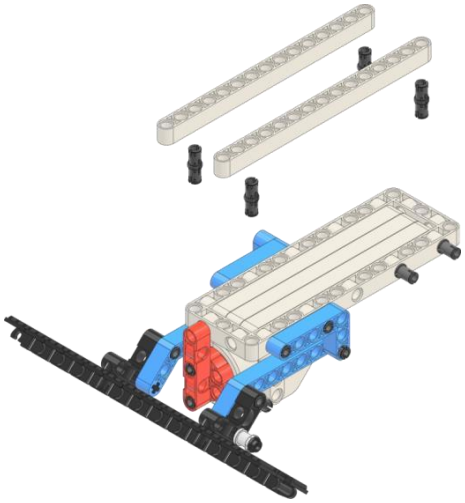


Рис.37 Сортировщик цветов

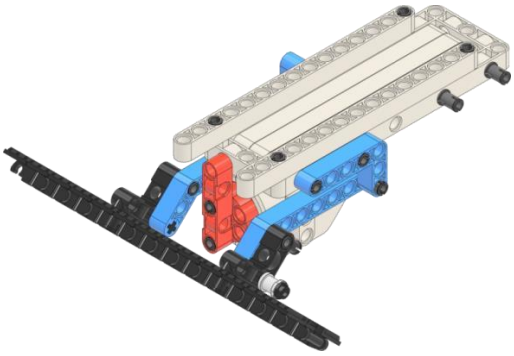


Рис.38 Сортировщик цветов

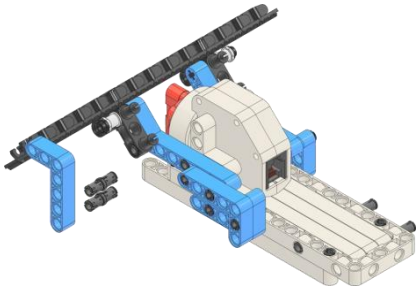


Рис.39 Сортировщик цветов

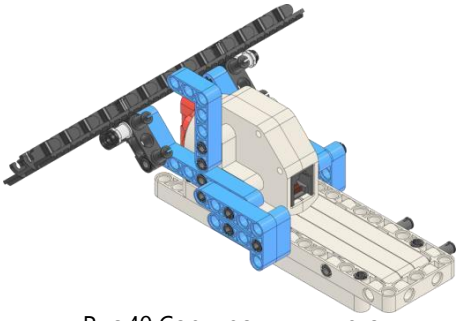


Рис.40 Сортировщик цветов

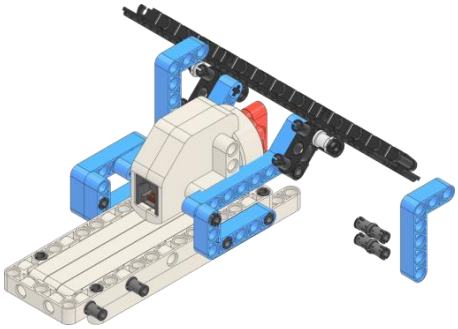


Рис.41 Сортировщик цветов

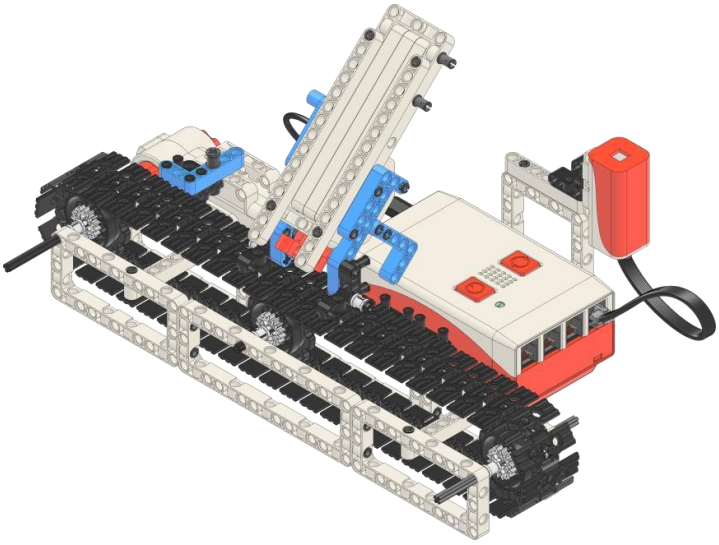


Рис.42 Сортировщик цветов

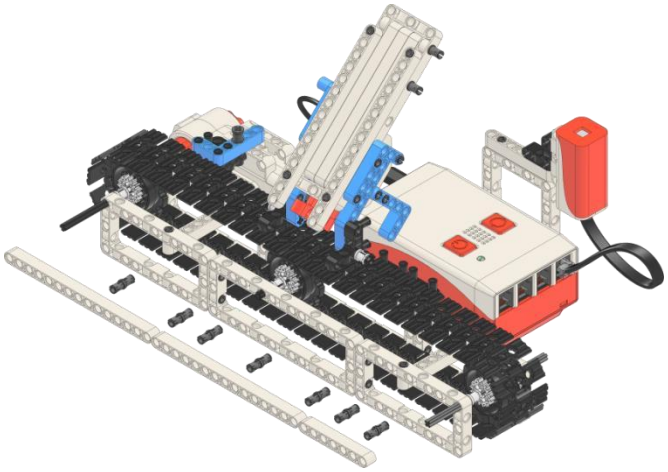


Рис.43 Сортировщик цветов

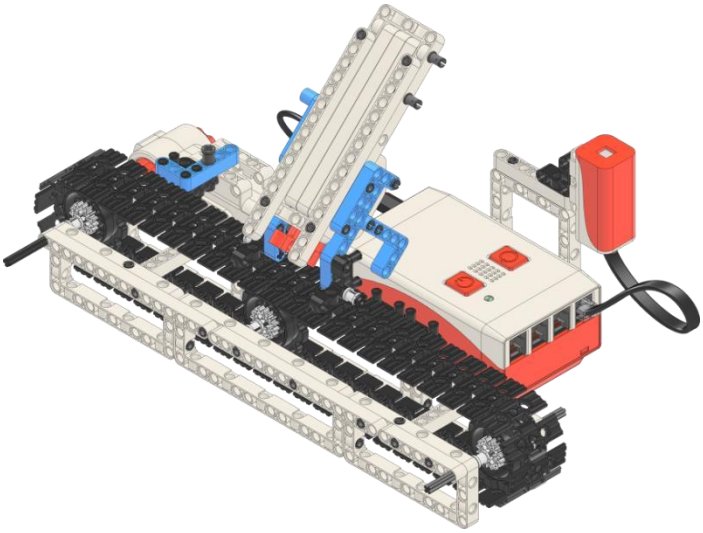


Рис.44 Сортировщик цветов

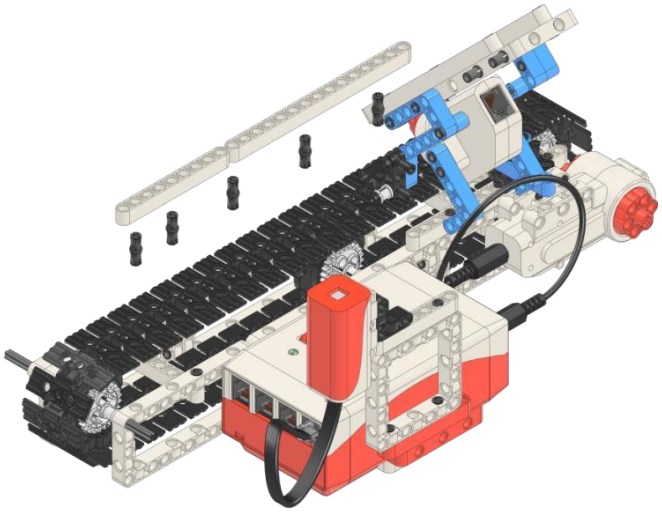


Рис.45 Сортировщик цветов

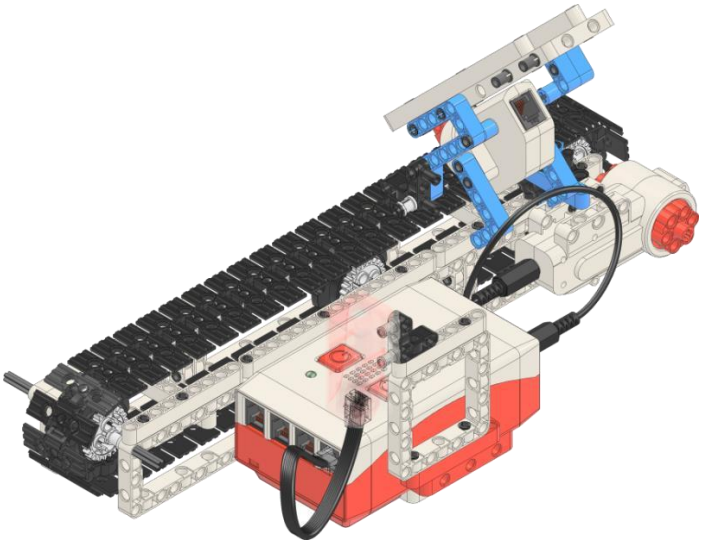


Рис.46 Сортировщик цветов

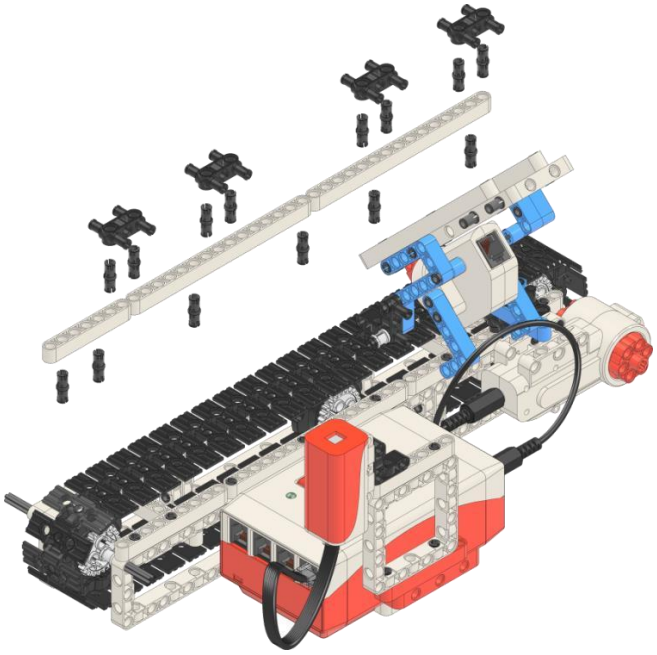


Рис.47 Сортировщик цветов

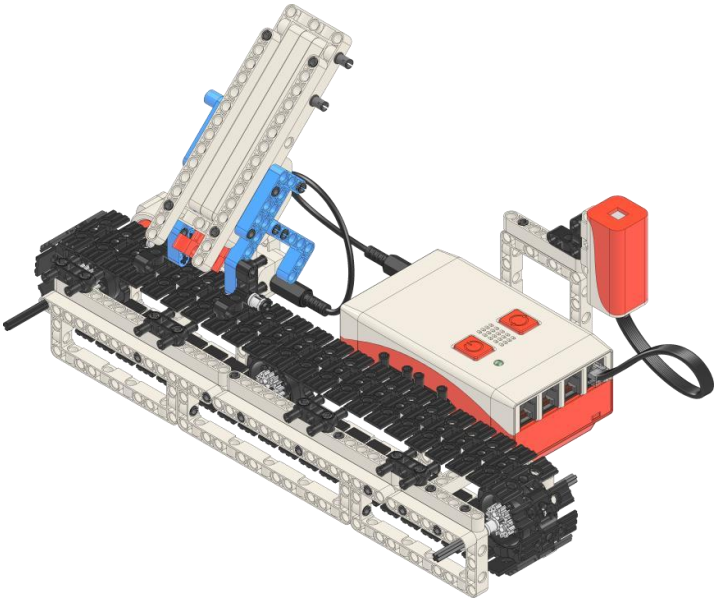


Рис.48 Сортировщик цветов

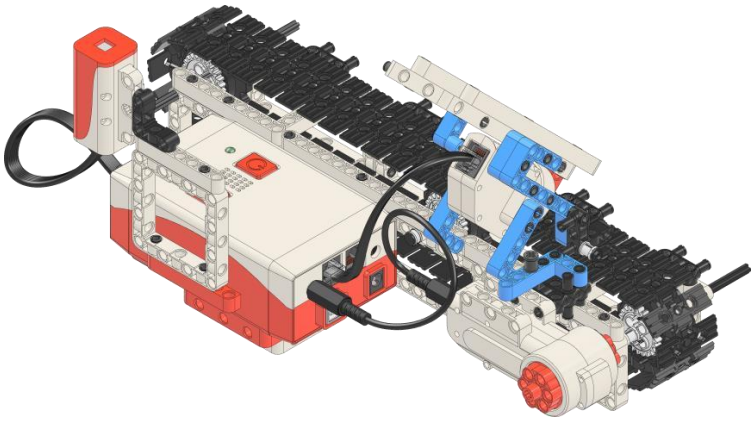


Рис.49 Сортировщик цветов

Сервомотор подключаем к первому порту, а датчик цвета к пятому.

Задание 2 (Уровень В)

Составьте программу, с помощью которой устройство начнёт сортировать детали.

Пример решения.

- Для начала нам необходимо откалибровать датчик цвета, опираясь на главу 4.5. Показываем датчику цвета детали для сортировки и выписываем значения RGB – red blue green. Значения будут варьироваться от 0 до 255. Необходимо для каждого цвета провести до трёх измерений, чтобы вычленить интервал.

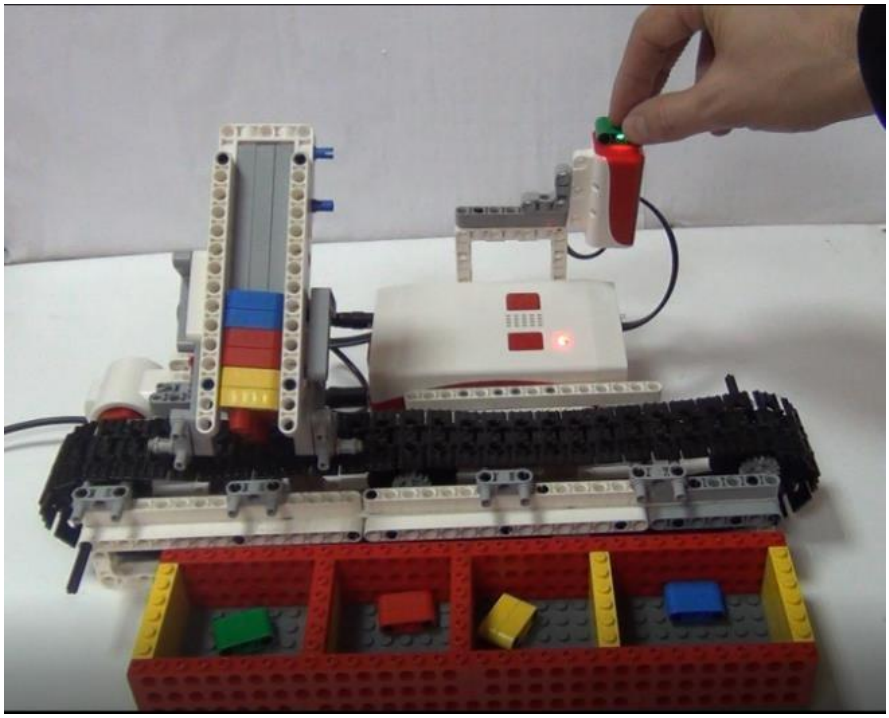


Рис.50 Калибровка датчика цвета

Далее необходимо откалибровать работу сервопривода. Нужно определить при каком угле запирающий язычок держит детали, а при каком отпускает. Затем нужно вычислить опытным путём время отпускания. Если это не рассчитать, то все детали в боксе выпадут одновременно.

Промежуток времени, примерно, - 100 мс. Всё будет зависеть от поверхности деталей и наклона бокса.

После данных процедур пришло время откалибровать работу dc мотора. Нужно прописать направление и время работы мотора, так чтобы он перемещал бокс на нужное расстояние. Для того чтобы не сбиться в сортировке, необходимо возвращать бокс в исходную точку после каждого выброса. Так мы создадим более понятный алгоритм работы. Время движения бокса к ячейке будет соответствовать времени его движения в исходную точку.

Добавим для визуализации звуковой сигнал, как только мы показываем деталь датчику цвета и звуковой сигнал, когда нужно его убрать – это необходимо сделать, чтобы он не записал количество больше, чем мы ему показали.

Для запоминания порядка расположения деталей в боксе используется массив. Для каждого цвета присваивается значение и записывается под определённым индексом массива. Для нашего варианта рассмотрим запись в массив четырёх цветных тел.

```
#include <Wire.h>
#include "Adafruit_TCS34725.h"

// Pick analog outputs, for the UNO these three work well
// use ~560  ohm resistor between Red & Blue, ~1K for green (its brighter)
#define redpin 8
#define greenpin 3
#define bluepin 2
// for a common anode LED, connect the common pin to +5V
// for common cathode, connect the common to ground

// set to false if using a common cathode LED
#define commonAnode true
#include <Servo.h>

Servo myservo;

int p=A0;
int m1 = 7;
int v1 = 6;
int kk=0;
int mas[]={0, 0, 0, 0};

// our RGB -> eye-recognized gamma color
byte gammatable[256];


Adafruit_TCS34725 tcs = Adafruit_TCS34725(TCS34725_INTEGRATIONTIME_50MS, TCS34725_GAIN_4X);
```

Рис.51 Первая часть программы

```
void setup() {
  Serial.begin(9600);
  //Serial.println("Color View Test!");
  pinMode(p, OUTPUT);
  pinMode(m1, OUTPUT);
  pinMode(v1, OUTPUT);
  analogWrite(6, 0);
  myservo.attach(11);
  myservo.write(80);
    delay(80);

    myservo.write(150);
    delay(1000);
  if (tcs.begin()) {
    //Serial.println("Found sensor");
  } else {
    Serial.println("No TCS34725 found ... check your connections");
    while (1); // halt!
  }
```

Рис.52 Вторая часть программы

```
  // use these three pins to drive an LED
#if defined(ARDUINO_ARCH_ESP32)
  ledcAttachPin(redpin, 1);
  ledcSetup(1, 12000, 8);
  ledcAttachPin(greenpin, 2);
  ledcSetup(2, 12000, 8);
  ledcAttachPin(bluepin, 3);
  ledcSetup(3, 12000, 8);
#else
  pinMode(redpin, OUTPUT);
  pinMode(greenpin, OUTPUT);
  pinMode(bluepin, OUTPUT);
#endif

  // thanks PhilB for this gamma table!
  // it helps convert RGB colors to what humans see
  for (int i=0; i<256; i++) {
    float x = i;
    x /= 255;
    x = pow(x, 2.5);
    x *= 255;

    if (commonAnode) {
      gammatable[i] = 255 - x;
    } else {
      gammatable[i] = x;
    }
    //Serial.println(gammatable[i]);
  }
}
```

Рис.53 Третья часть программы

```
// The commented out code in loop is example of getRawData with clear value.
// Processing example colorview.pde can work with this kind of data too, but It requires manual conversion to
// [0-255] RGB value. You can still uncomments parts of colorview.pde and play with clear value.
void loop() {
  float red, green, blue;

  tcs.setInterrupt(false);  // turn on LED

  delay(60);  // takes 50ms to read

  tcs.getRGB(&red, &green, &blue);

  tcs.setInterrupt(true);  // turn off LED

  Serial.print("R:\t"); Serial.print(int(red));
  Serial.print("\tG:\t"); Serial.print(int(green));
  Serial.print("\tB:\t"); Serial.print(int(blue));
Serial.print("mas:\t"); Serial.print(mas[0]);
Serial.print("mas:\t"); Serial.print(mas[1]);
Serial.print("mas:\t"); Serial.print(mas[2]);
Serial.print("mas:\t"); Serial.print(mas[3]);
```

Рис.54 Четвёртая часть программы

```
// GREEN
if(red>61 && red<64 && green>123 && green<127 && blue>70 && blue<74){
  mas[kk]=1;
  tone(p, 100);
  delay(500);
  kk=kk+1;
   tone(p, 100);
  delay(1000);
   tone(p, 100);
  delay(1000);
  }
// RED
else if (red>183 && red<192 && green>43 && green<47 && blue>40 && blue<45){
  mas[kk]=2;
  tone(p, 300);
  delay(500);

  kk=kk+1;
   tone(p, 100);
  delay(1000);
   tone(p, 100);
  delay(1000);

  }
```

Рис.55 Пятая часть программы

```
// YELLOW
else if (red>180 && red<190 && green>133 && green<141 && blue>59 && blue<64){
  mas[kk]=3;
  tone(p, 500);
  delay(500);
  kk=kk+1;
   tone(p, 100);
  delay(1000);
   tone(p, 100);
  delay(1000);

  }
// BLUE
else if (red>39 && red<45 && green>85 && green<89 && blue>130 && blue<139){
  mas[kk]=4;
  tone(p, 1000);
  delay(500);
  kk=kk+1;
  tone(p, 100);
  delay(1000);
   tone(p, 100);
  delay(1000);

  }
```

Рис.56 Шестая часть программы

```
//WHITE
else if (red>170 && red<256 && green>200 && green<256 && blue>150 && blue<256){

  tone(p, 2000);
  delay(500);
for(int bb=0; bb<5; bb++)
{
  if (mas[bb]==2){
  analogWrite(6, 110);
  digitalWrite(7, HIGH);
  delay(400);
  analogWrite(6, 0);
  myservo.write(80);
    delay(100);

  myservo.write(150);
    delay(1000);
  delay(1000);
  analogWrite(6, 110);
  digitalWrite(7, LOW);
  delay(400);
  analogWrite(6, 0);
  }
```

Рис.57 Седьмая часть программы

```
 else if (mas[bb]==3){
  analogWrite(6, 110);
  digitalWrite(7, HIGH);
  delay(700);
  analogWrite(6, 0);
  myservo.write(80);
    delay(100);
  myservo.write(150);
    delay(1000);
  delay(1000);
  analogWrite(6, 110);
  digitalWrite(7, LOW);
  delay(700);
  analogWrite(6, 0);
  }

  else if (mas[bb]==4){
  analogWrite(6, 120);
  digitalWrite(7, HIGH);
  delay(1200);
  analogWrite(6, 0);
  myservo.write(80);
    delay(100);
  myservo.write(150);
    delay(1000);
  delay(1000);
  analogWrite(6, 120);
  digitalWrite(7, LOW);
  delay(1200);
  analogWrite(6, 0);
  }
```

Рис.58 Восьмая часть программы

```
    else if (mas[bb]==1){
   tone(p, 770);
   delay(300);
   tone(p, -1);
   myservo.write(80);
     delay(100);
   myservo.write(150);
     delay(1000);
   }
   }
mas[0]=0;
mas[1]=0;
mas[2]=0;
mas[3]=0;
   }


else
{
   tone(p, -1);

   }

   Serial.print("\n");
```

Рис.59 Девятая часть программы

```
#if defined(ARDUINO_ARCH_ESP32)
   ledcWrite(1, gammatable[(int)red]);
   ledcWrite(2, gammatable[(int)green]);
   ledcWrite(3, gammatable[(int)blue]);
#else
   analogWrite(redpin, gammatable[(int)red]);
   analogWrite(greenpin, gammatable[(int)green]);
   analogWrite(bluepin, gammatable[(int)blue]);
#endif

}
```

Рис.60 Десятая часть программы

9.2. Манипулятор

Манипулятор – робот, способный захватывать и перемещать предметы в пространстве. Данное устройство имеет две степени свободы.

Методические рекомендации.

Тема: «Манипулятор»

Цель: изучить процесс создания и программирования устройства способного захватывать и перемещать предметы.

Задачи:

- изучить и закрепить на практике процесс создания манипулятора
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Пример заданий.

Задание 1 (уровень А)

Соберите конструкцию согласно инструкции, которая будет способна захватывать и перемещать объект.

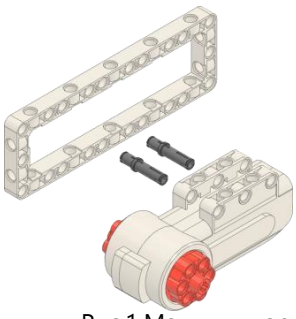


Рис.1 Манипулятор

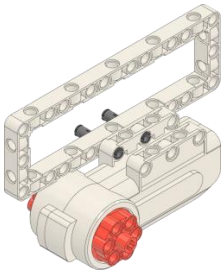


Рис.2 Манипулятор

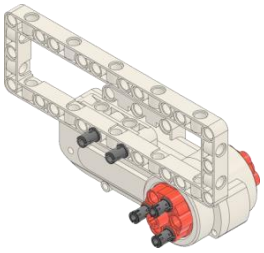


Рис.3 Манипулятор

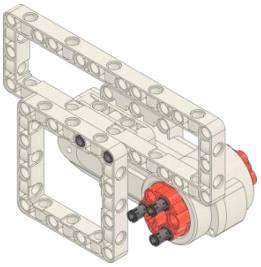


Рис.4 Манипулятор

Рис.5 Манипулятор

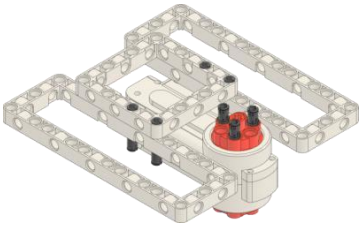


Рис.6 Манипулятор

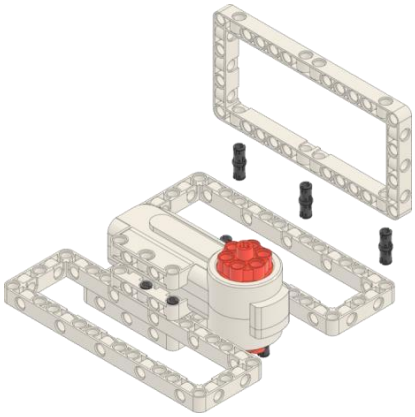


Рис.7 Манипулятор

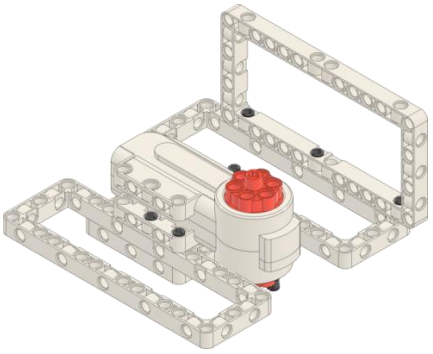


Рис.8 Манипулятор

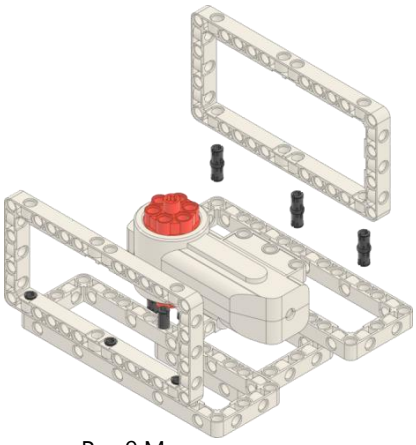


Рис.9 Манипулятор

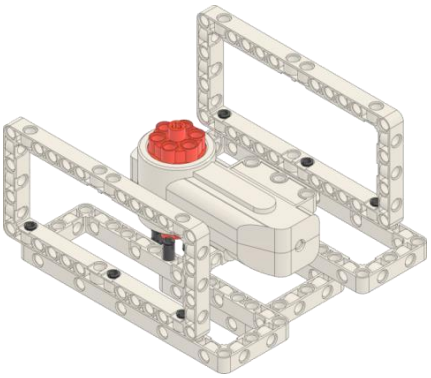


Рис.10 Манипулятор

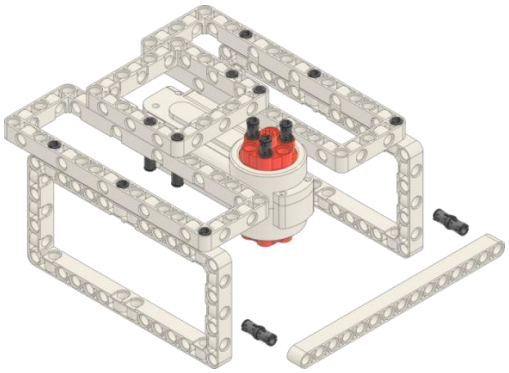


Рис.11 Манипулятор

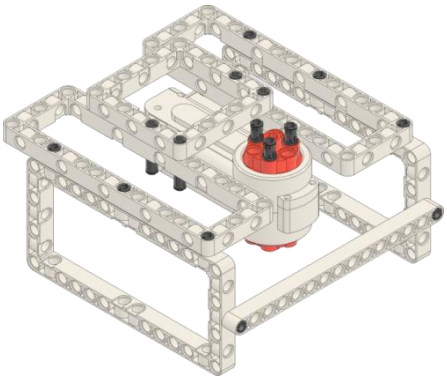


Рис.12 Манипулятор

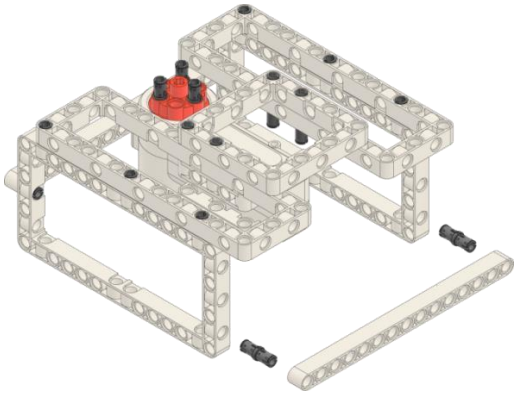


Рис.13 Манипулятор

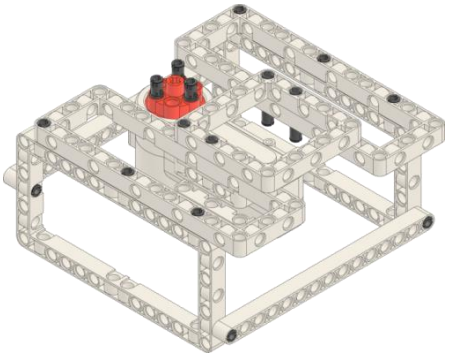


Рис.14 Манипулятор

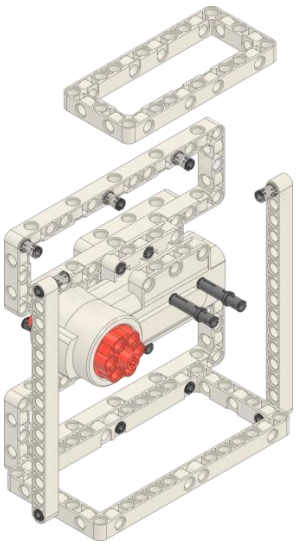


Рис.15 Манипулятор

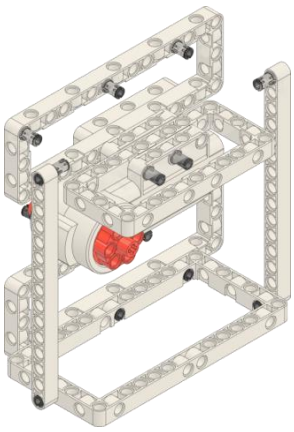


Рис.16 Манипулятор

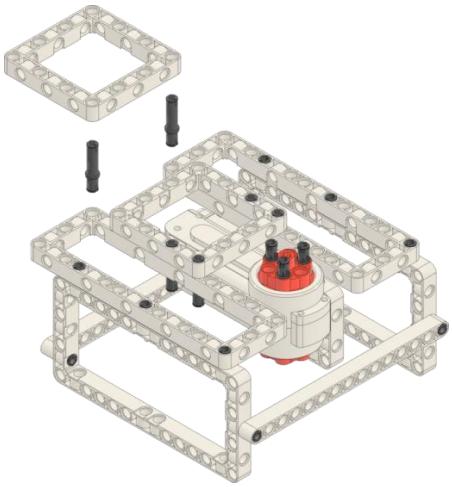


Рис.17 Манипулятор

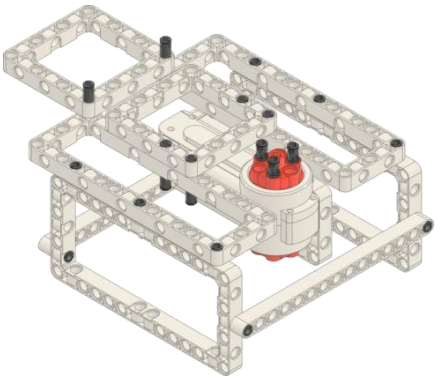


Рис.18 Манипулятор

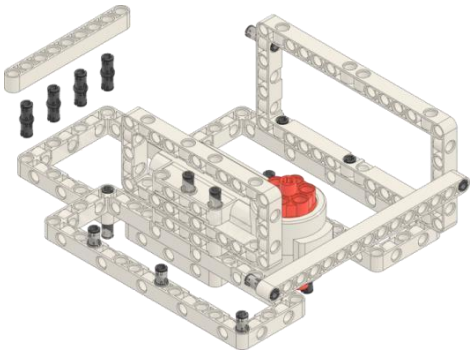


Рис.19 Манипулятор

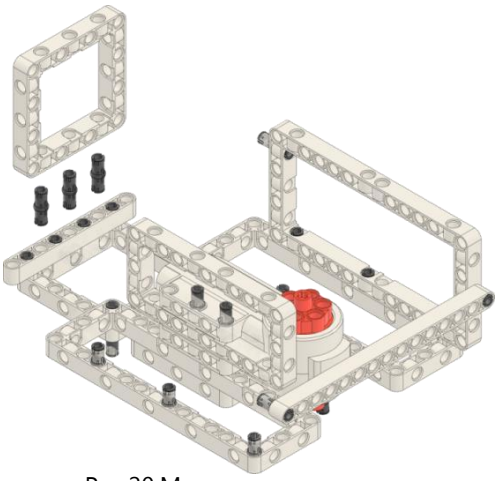


Рис.20 Манипулятор

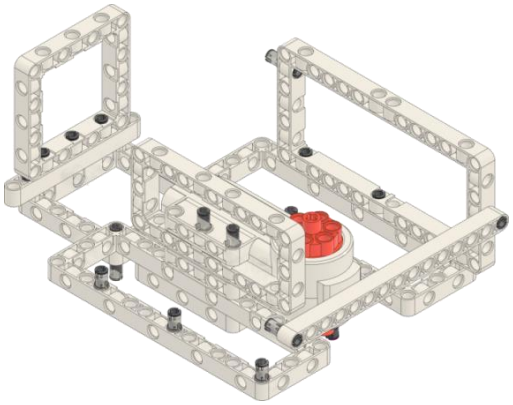


Рис.21 Манипулятор

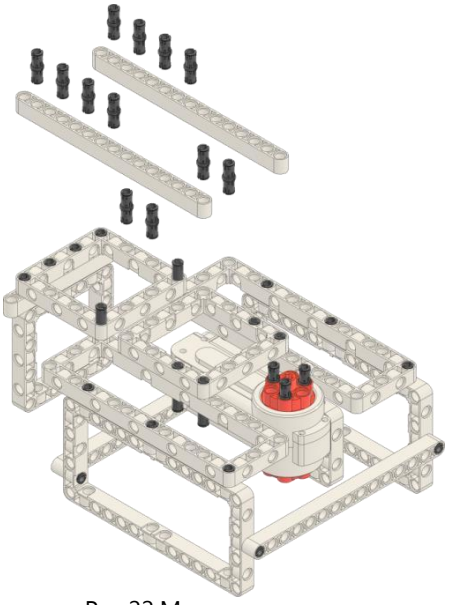


Рис.22 Манипулятор

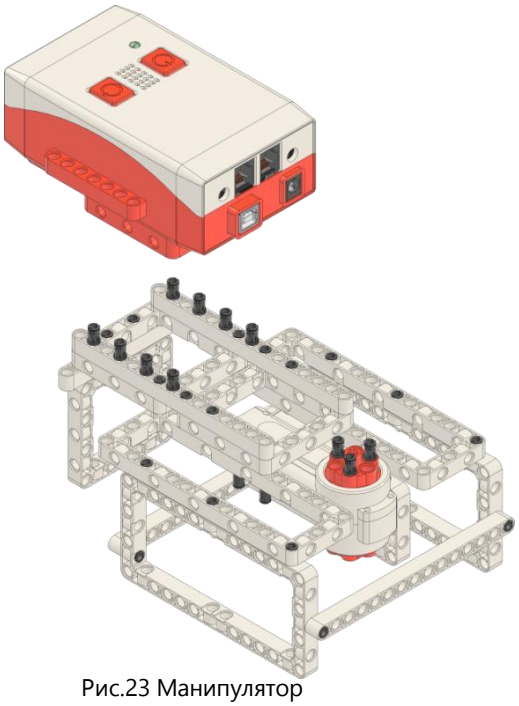


Рис.23 Манипулятор

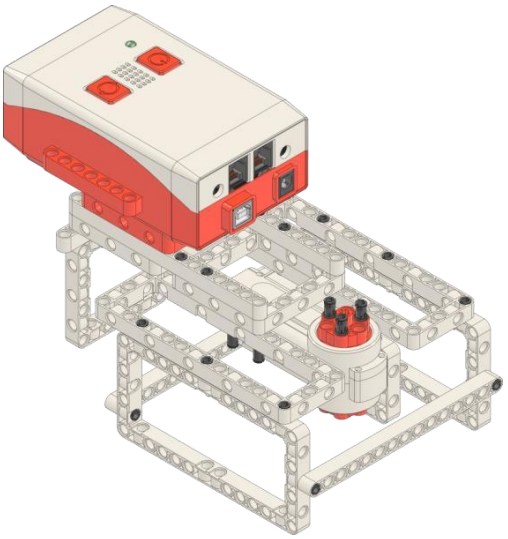


Рис.24 Манипулятор

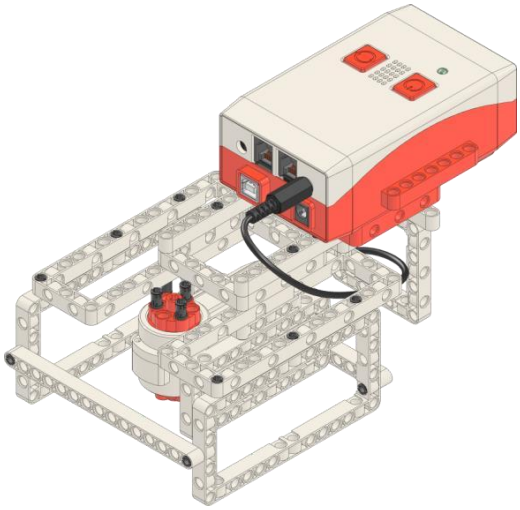


Рис.25 Манипулятор

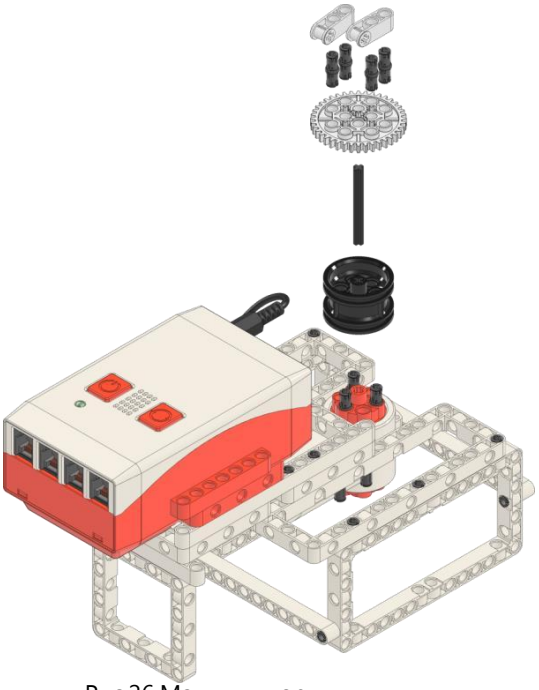


Рис.26 Манипулятор

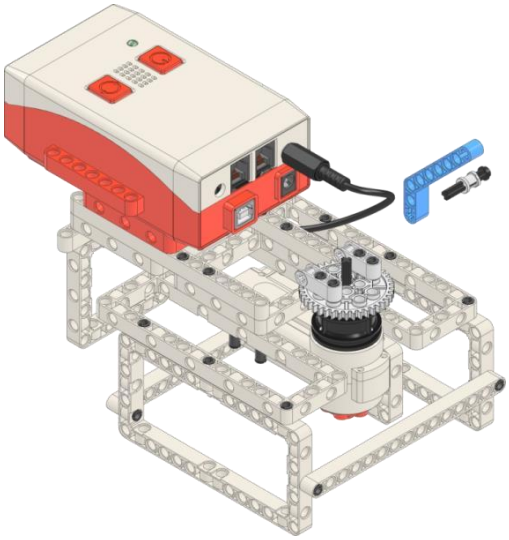


Рис.27 Манипулятор

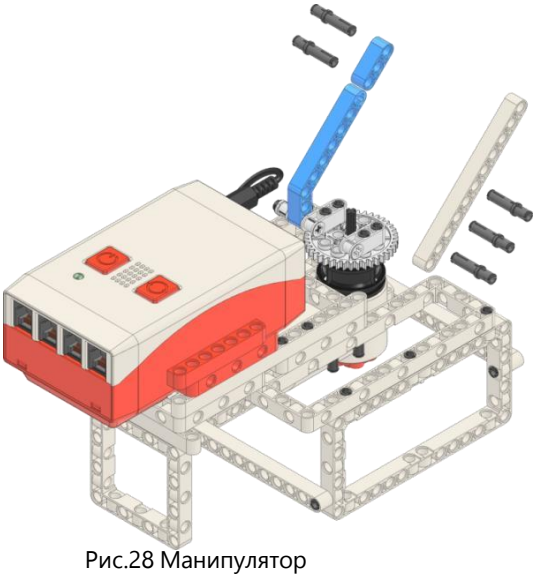


Рис.28 Манипулятор

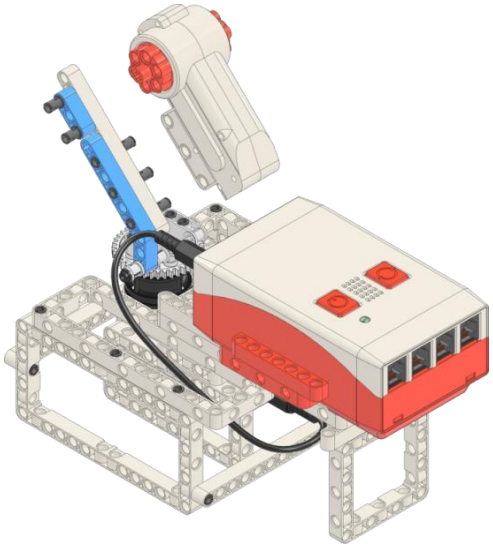


Рис.29 Манипулятор

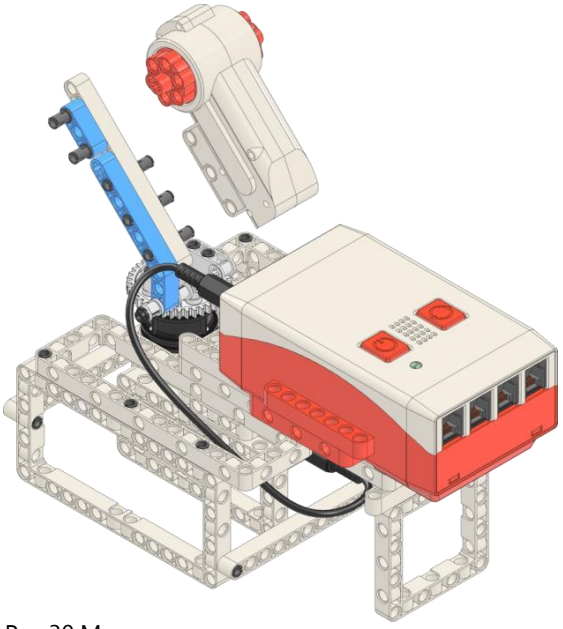


Рис.30 Манипулятор

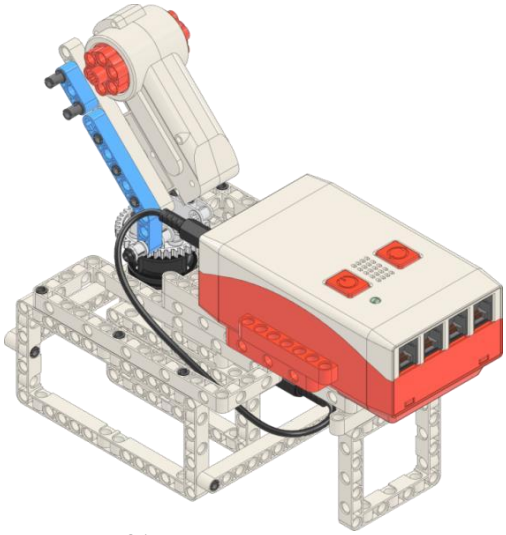


Рис.31 Манипулятор

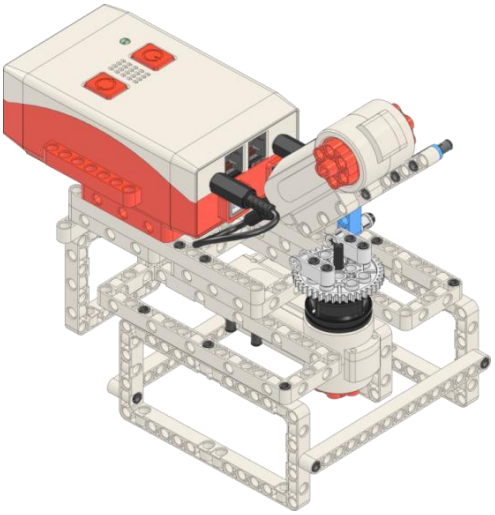


Рис.32 Манипулятор

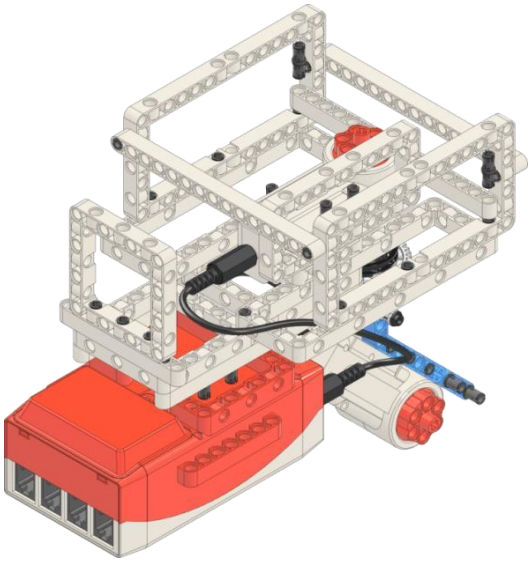


Рис.33 Манипулятор

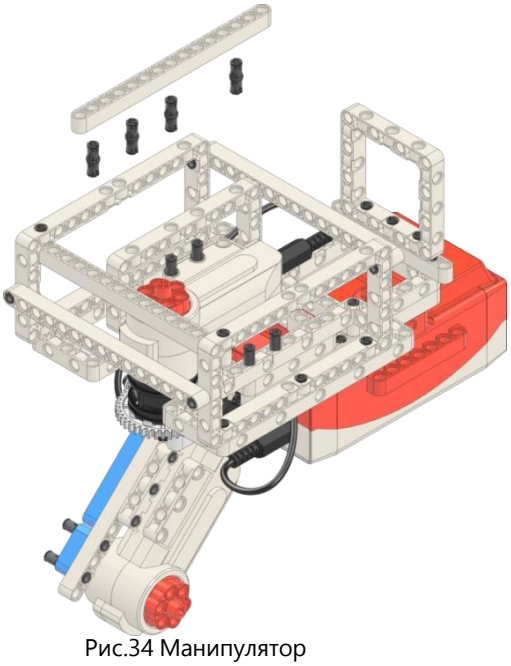


Рис.34 Манипулятор

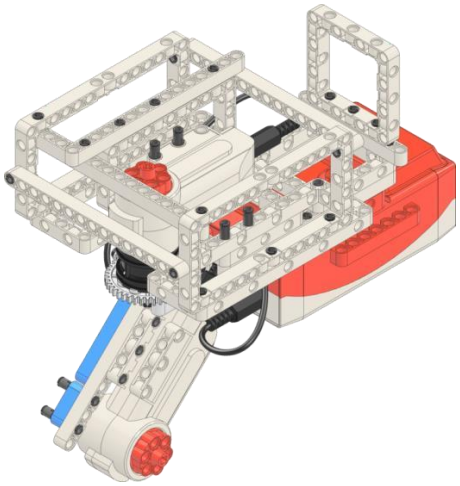


Рис.35 Манипулятор

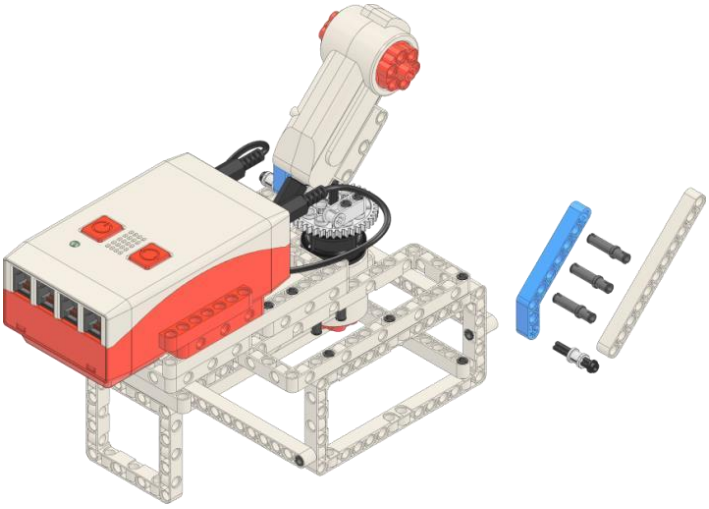


Рис.36 Манипулятор

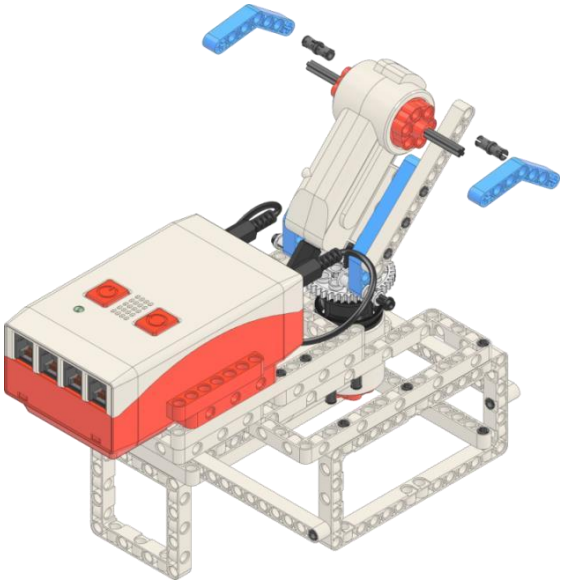


Рис.37 Манипулятор

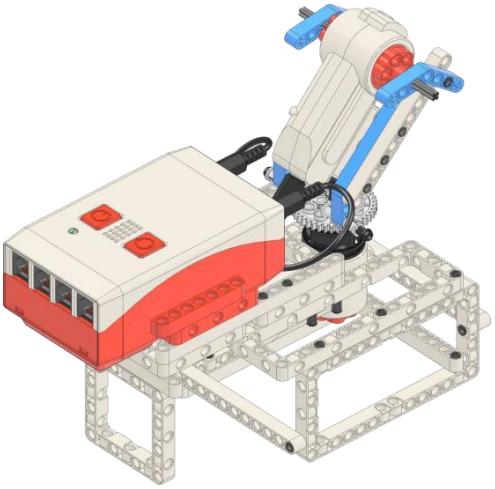


Рис.38 Манипулятор
Применим две оси размером 4.

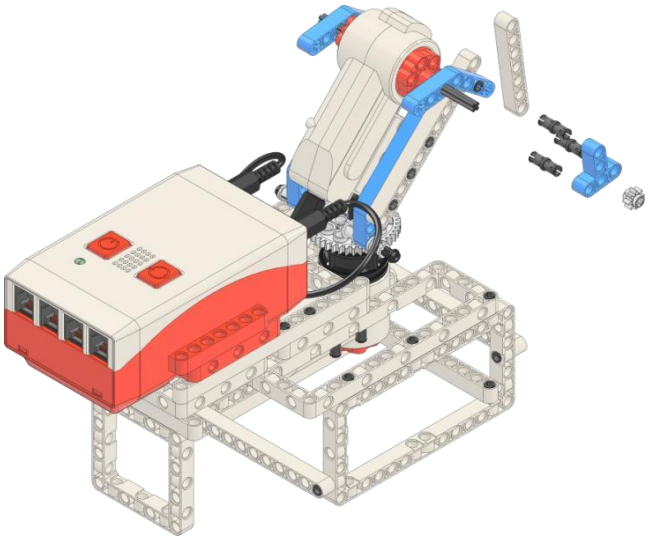


Рис.39 Манипулятор

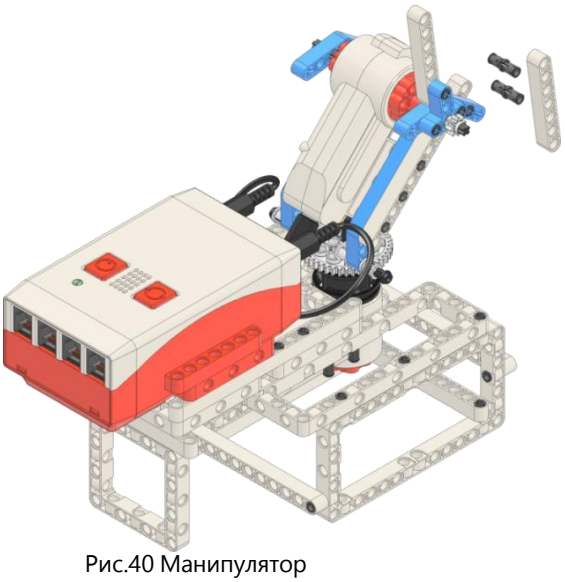


Рис.40 Манипулятор

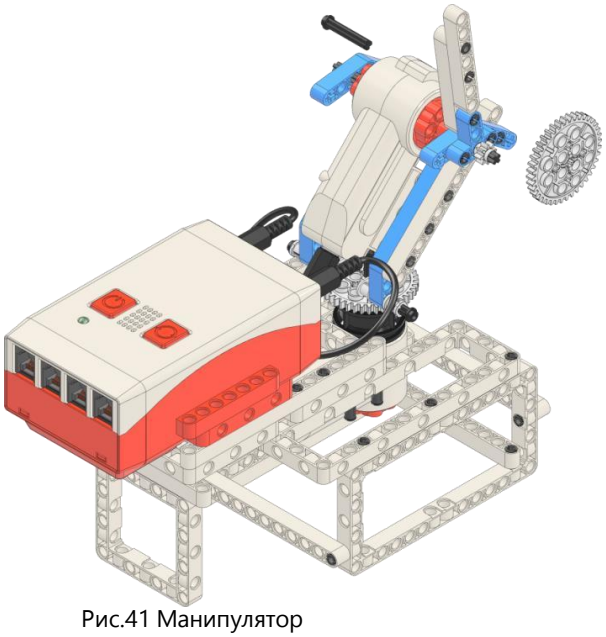


Рис.41 Манипулятор

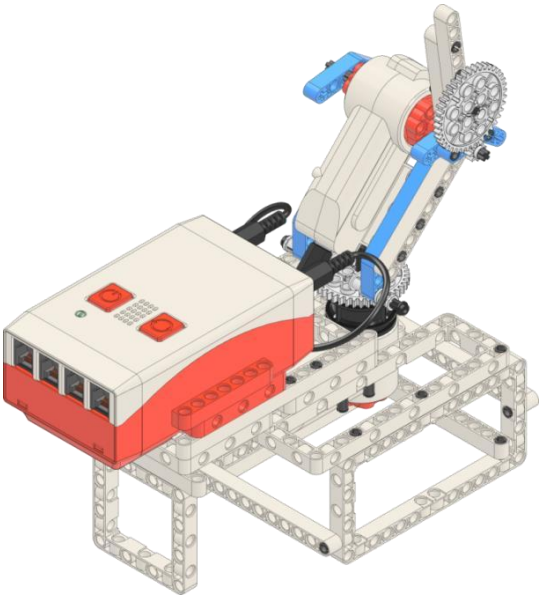


Рис.42 Манипулятор

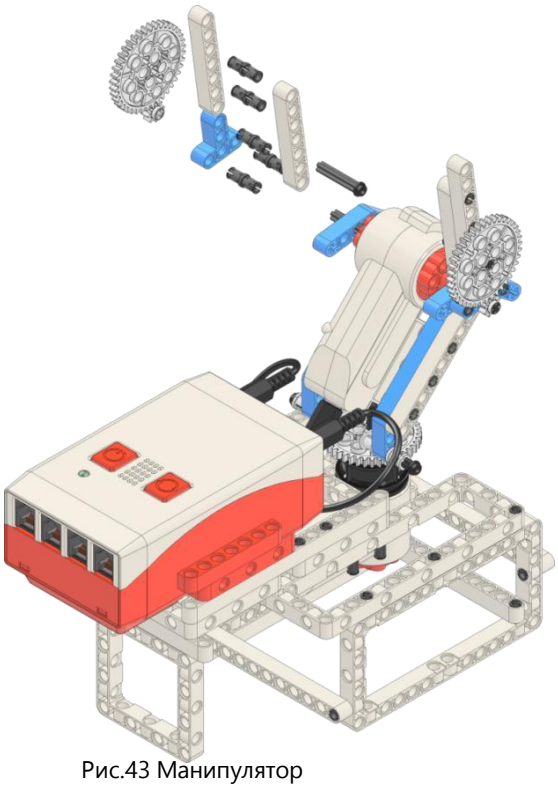


Рис.43 Манипулятор

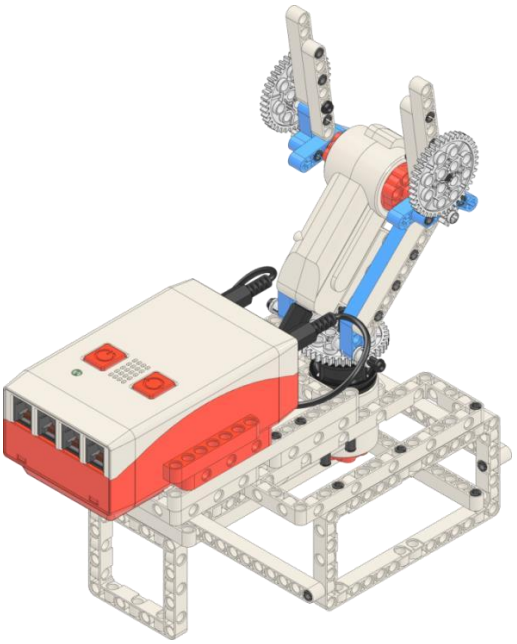


Рис.44 Манипулятор

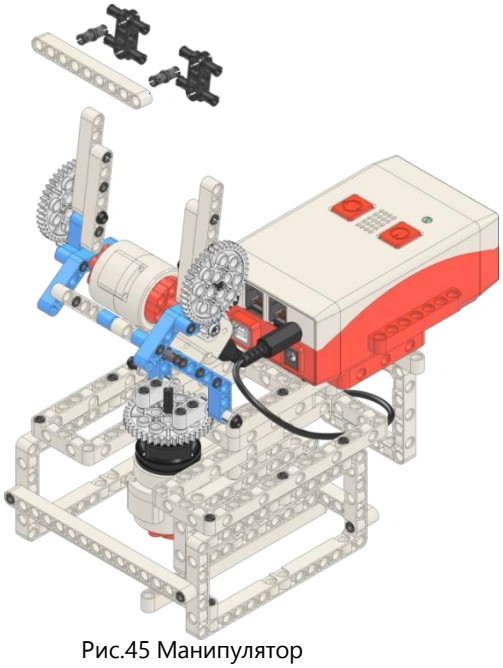


Рис.45 Манипулятор

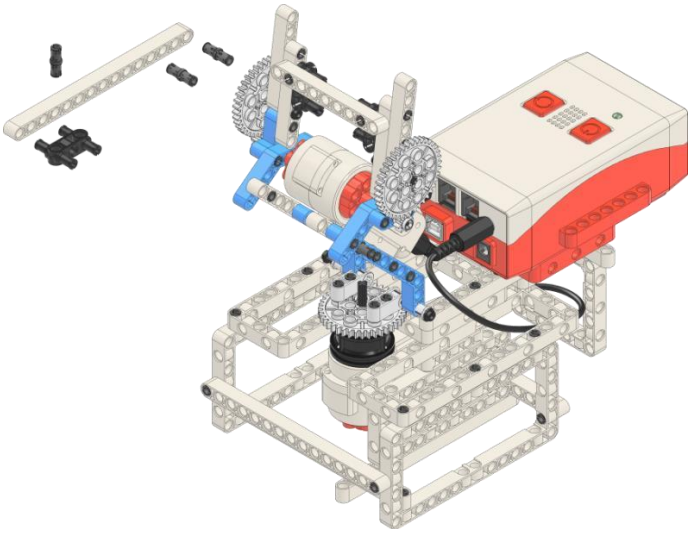


Рис.46 Манипулятор

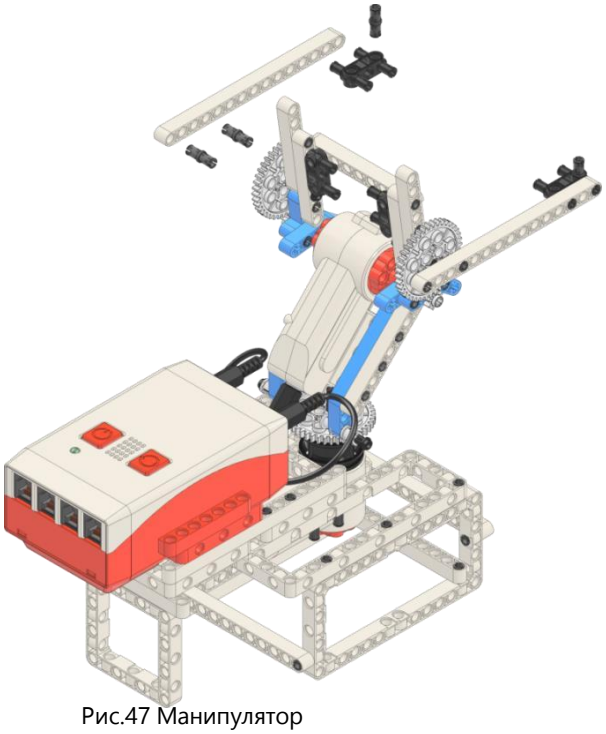


Рис.47 Манипулятор

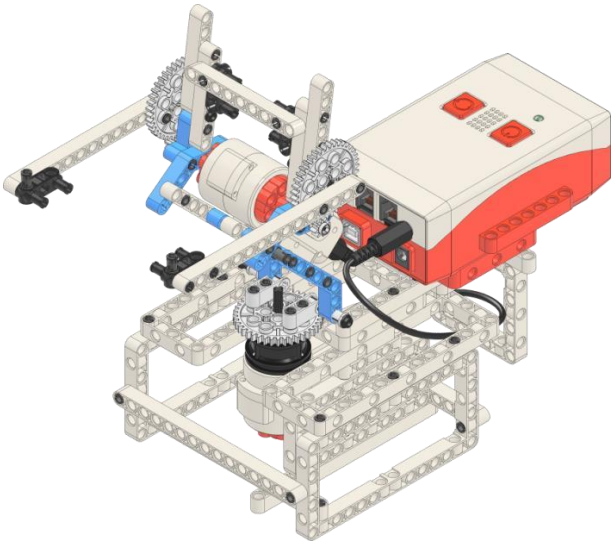


Рис.48 Манипулятор

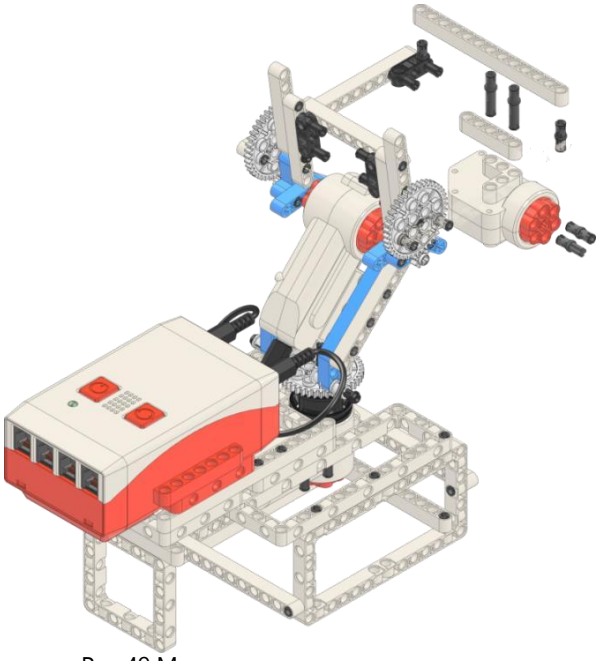


Рис.49 Манипулятор

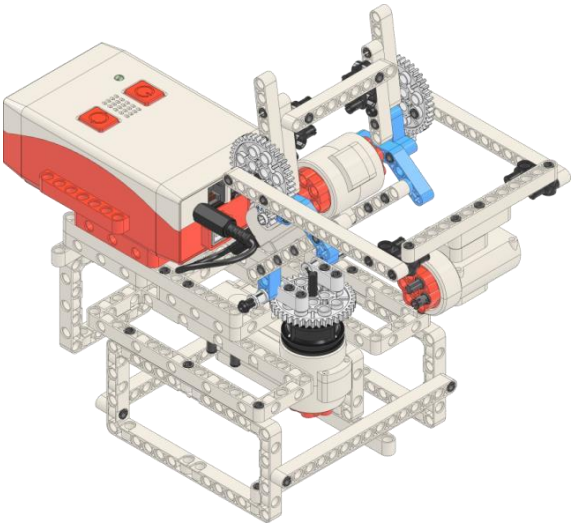


Рис.50 Манипулятор

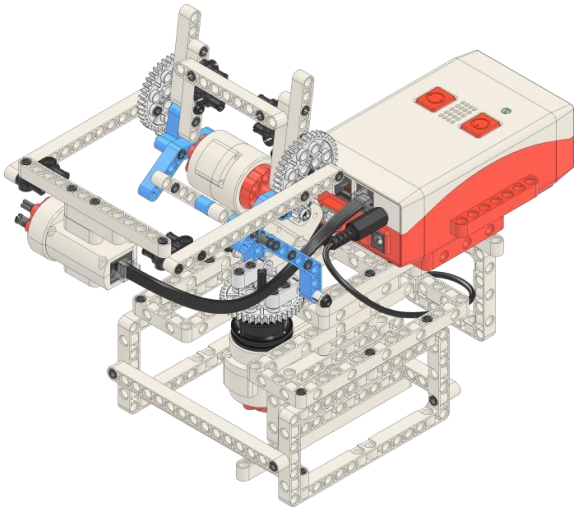


Рис.51 Манипулятор

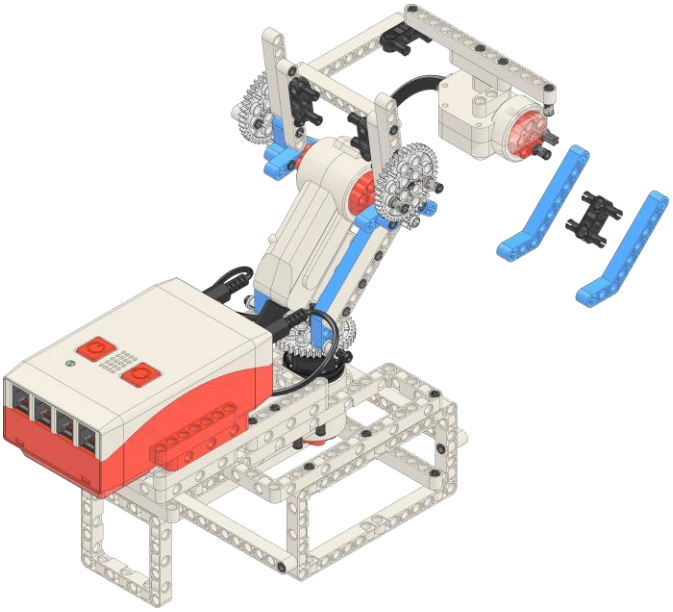


Рис.52 Манипулятор

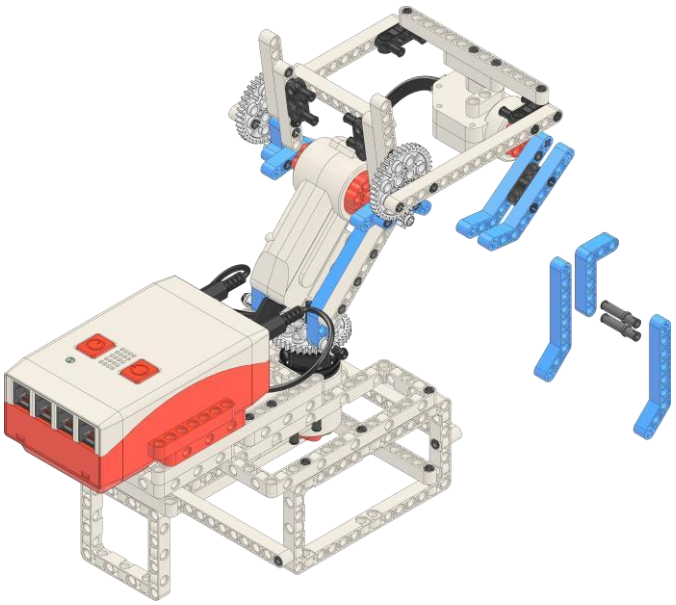


Рис.53 Манипулятор

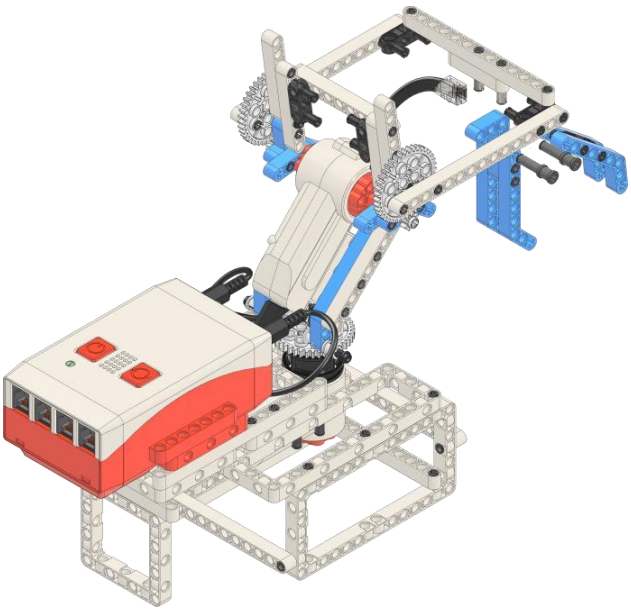


Рис.54 Манипулятор

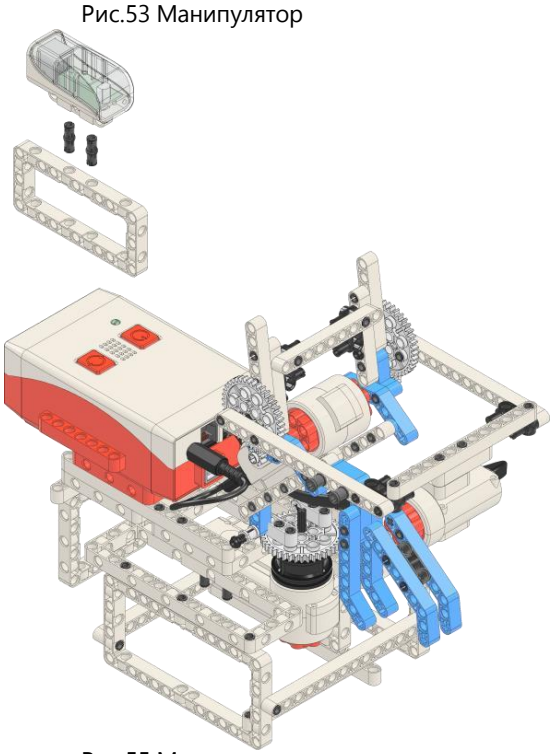


Рис.55 Манипулятор

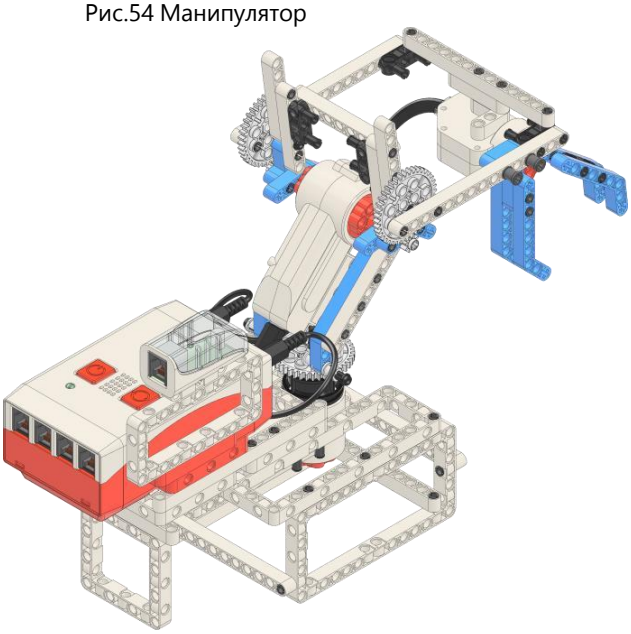


Рис.56 Манипулятор

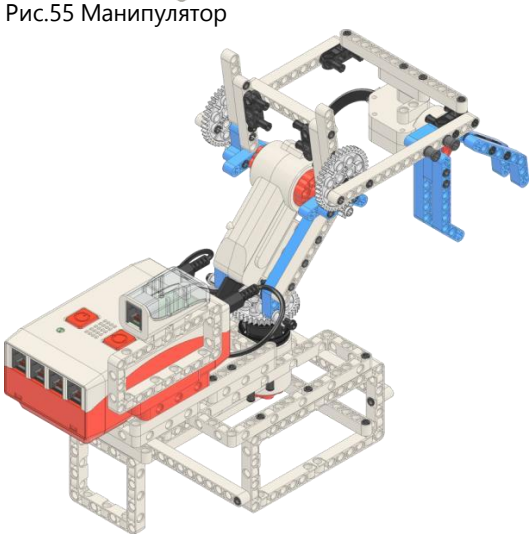


Рис.57 Манипулятор

Задание 2 (Уровень В)

Создать программу, с помощью которой можно управлять манипулятором посредством IR модуля.

Пример решения.

- Используя полученные знания и наработки по главе 4.2, 4.6 и 5.6 создадим программу, под названием **manipulator.ino**.

```
#include <Servo.h>
#include <IRremote.h>

int RECV_PIN = 8;
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;
IRrecv irrecv(RECV_PIN);

decode_results results;
Servo myservo;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
  pinMode(m1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  myservo.attach(3);
  myservo.write(170);
}
```

Рис.58 Начало программы «Манипулятор»

```
void loop() {
  if (irrecv.decode(&results)) {
    //Serial.println(results.value, HEX);
    Serial.println(results.value);
    irrecv.resume();

    if (results.value == 16720605){
      analogWrite(v1, 100);

      digitalWrite(m1, LOW);
      delay(300);
       analogWrite(v1, 0);


      }
    else if (results.value == 16761405){
      analogWrite(v1, 0);
      analogWrite(v2, 0);
      }

 else if (results.value == 16712445){
      analogWrite(v1, 100);

      digitalWrite(m1, HIGH);
        delay(300);
       analogWrite(v1, 0);

      }
```

Рис.59 Вторая часть программы «Манипулятор»

```
 else if (results.value == 16753245){

      analogWrite(v2, 127);

       digitalWrite(m2, HIGH);

       delay(200);
analogWrite(v2, 0);

      }
     else if (results.value == 16769565){
      analogWrite(v2, 127);

       digitalWrite(m2, LOW);

       delay(200);
analogWrite(v2, 0);
      }

      // else if (results.value == 16769055){
      //myservo.write(10);
     // }
      else if (results.value == 16754775){
      myservo.write(170);
      }
        else if (results.value == 16748655){
      myservo.write(90);
      }

  }
}
```

Рис.60 Заключительная часть программы «Манипулятор»

9.3. Копировальщик (Ресурсный набор)

Копировальщик – робот, способный копировать изображение. Данное устройство имеет две степени свободы.

Методические рекомендации.

Тема: «Копировальщик»

Цель: изучить процесс создания и программирования устройства способного копировать рисунки.

Задачи:

- изучить и закрепить на практике процесс создания копировальщика
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Пример заданий.

Задание 1 (Уровень А)

Соберите установку согласно инструкции.

Рис1 Копировальщик

Рис.2 Копировальщик

Рис.3 Копировальщик

Рис.4 Копировальщик

Рис.5 Копировальщик

Рис.6 Копировальщик

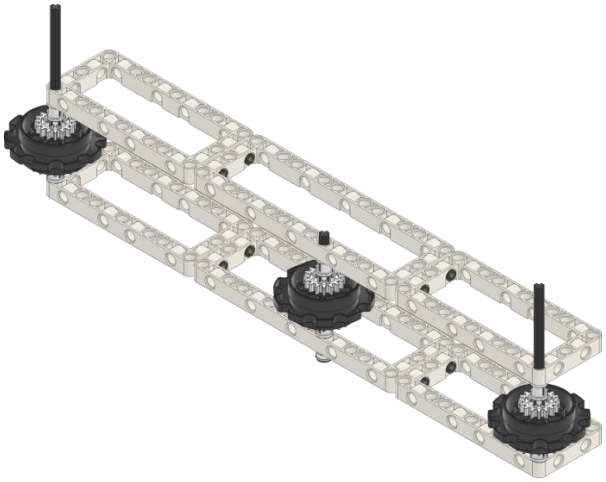


Рис.7 Копировальщик

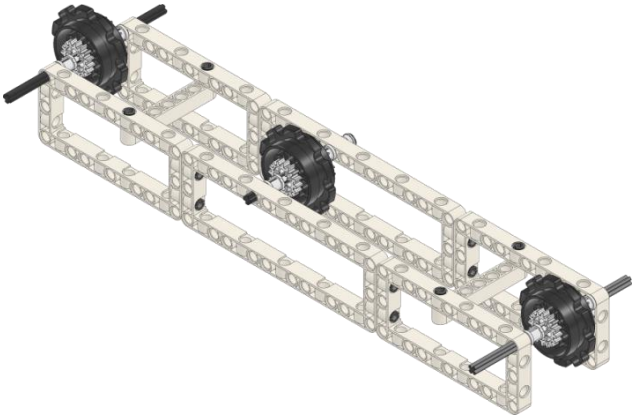


Рис.8 Копировальщик

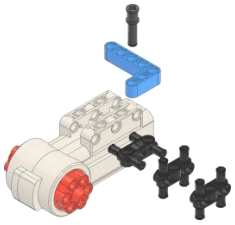


Рис.9 Копировальщик

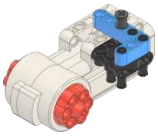


Рис.10 Копировальщик

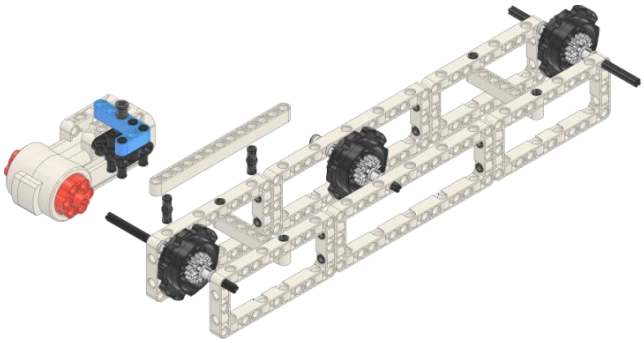


Рис.11 Копировальщик

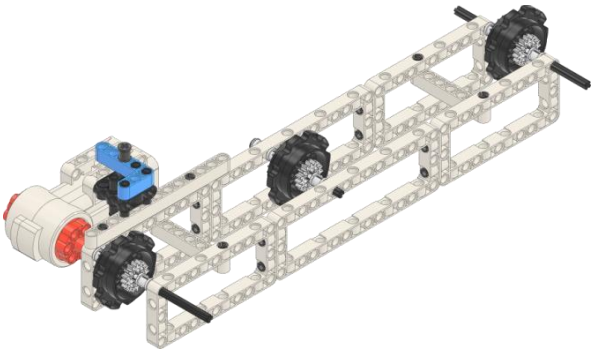


Рис.12 Копировальщик

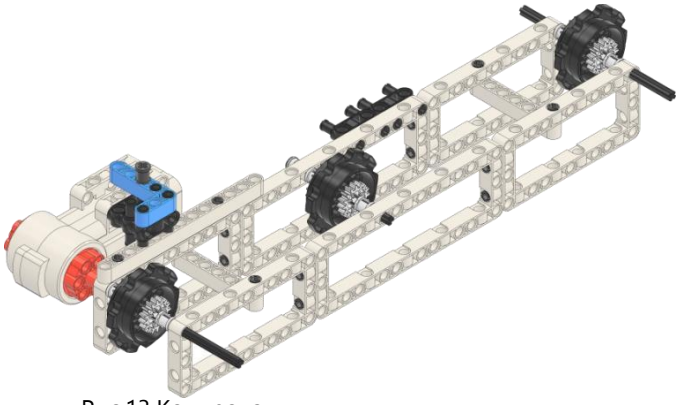


Рис.13 Копировальщик

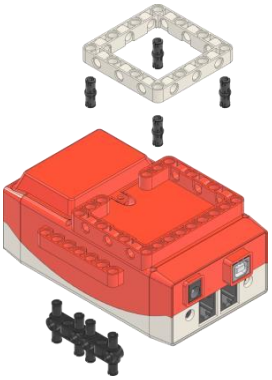


Рис.14 Копировальщик

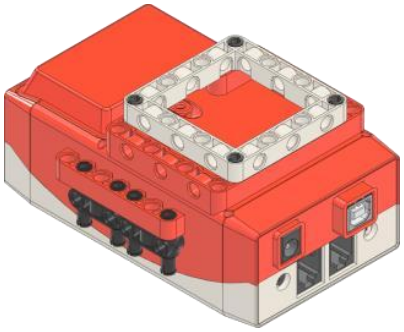


Рис.15 Копировальщик

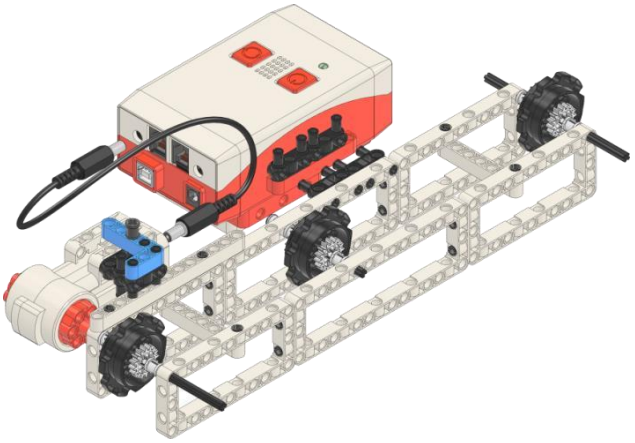


Рис.16 Копировальщик

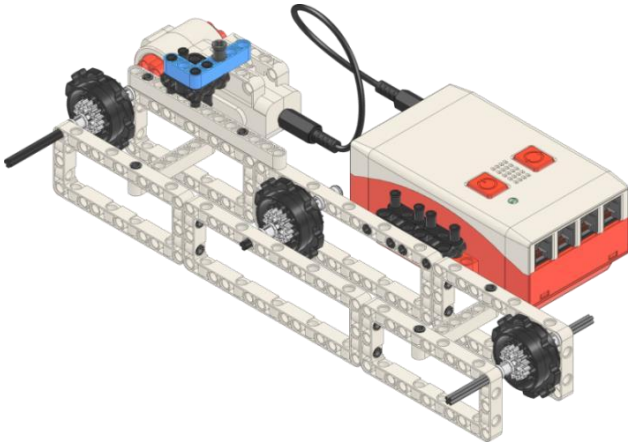


Рис.17 Копировальщик

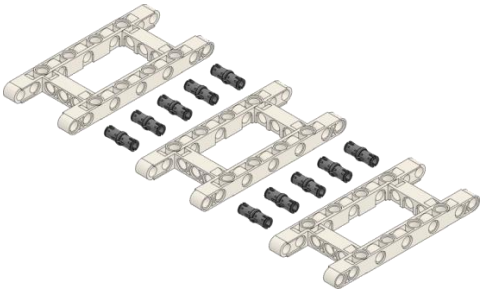


Рис.18 Копировальщик

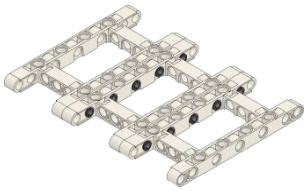


Рис.19 Копировальщик

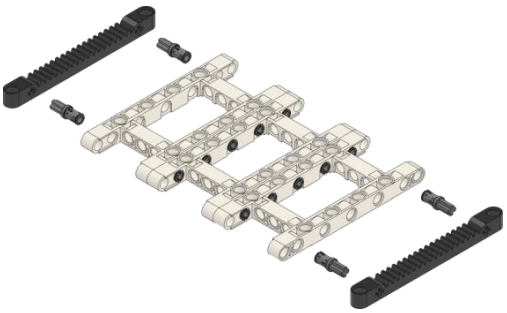


Рис.20 Копировальщик

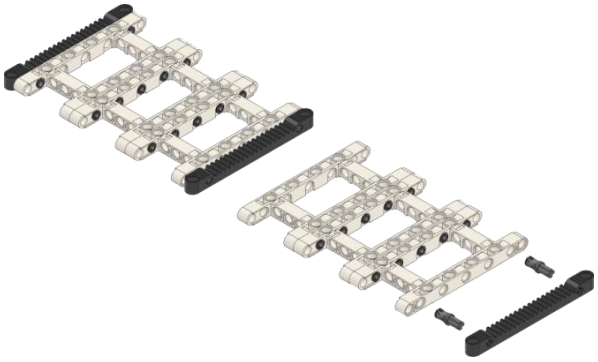


Рис.21 Копировальщик

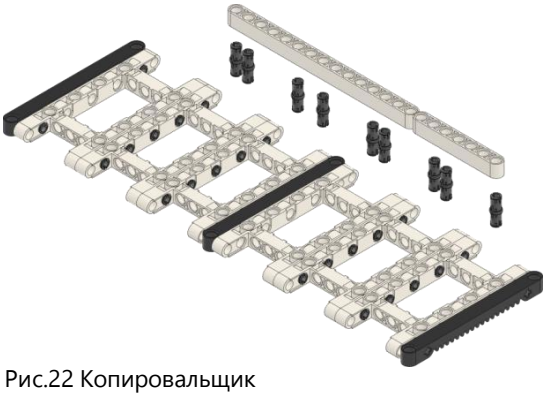


Рис.22 Копировальщик

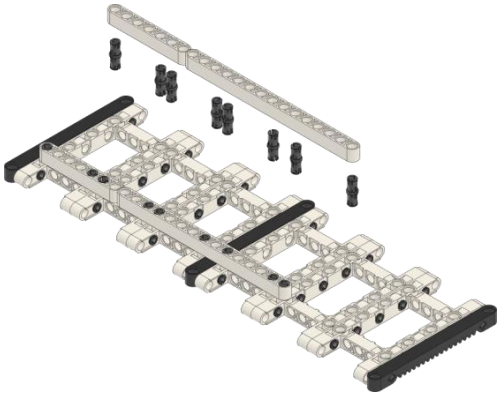


Рис.23 Копировальщик

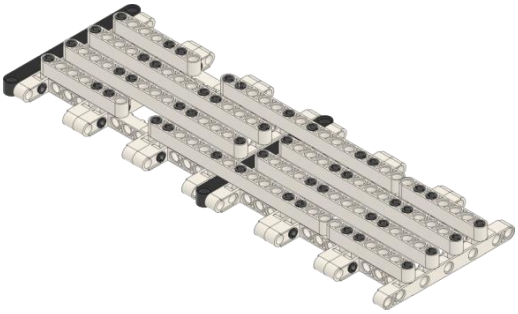


Рис.24 Копировальщик

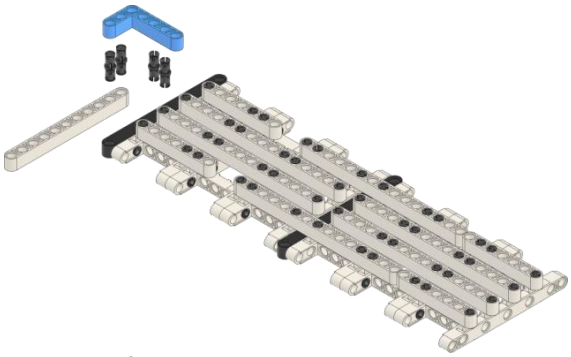


Рис.25 Копировальщик

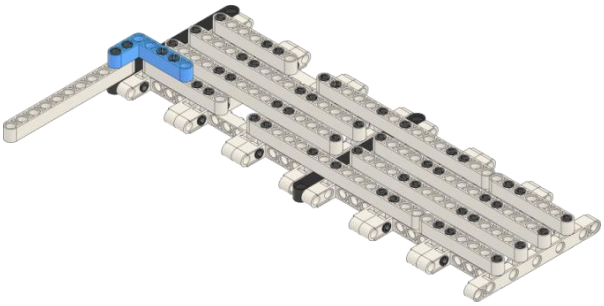


Рис.26 Копировальщик

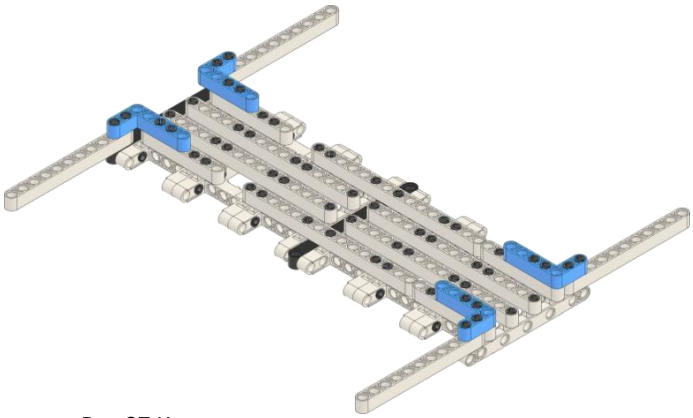


Рис.27 Копировальщик

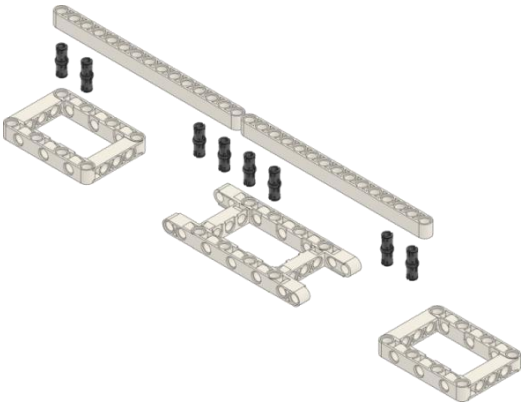


Рис.28 Копировальщик

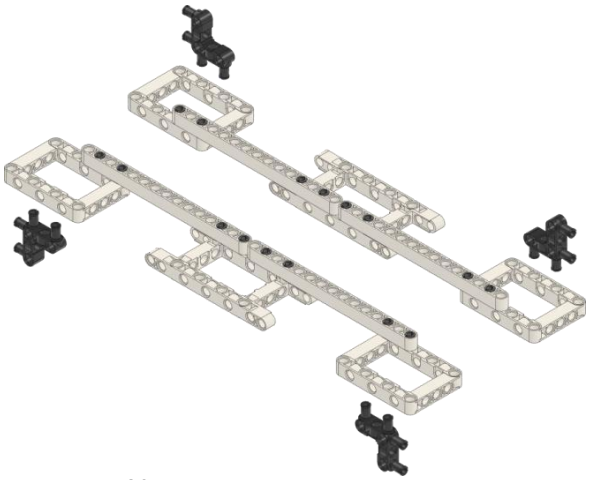


Рис.29 Копировальщик

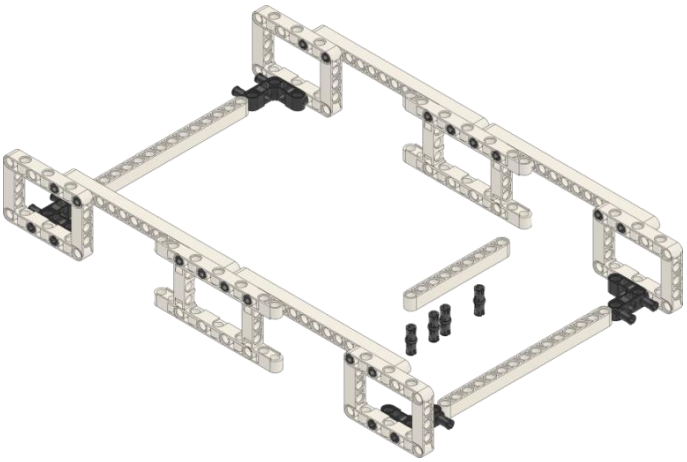


Рис.30 Копировальщик

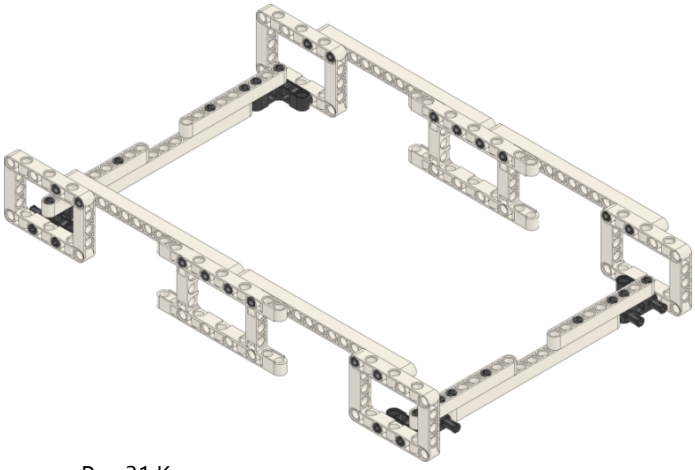


Рис.31 Копировальщик

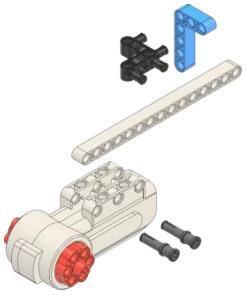


Рис.32 Копировальщик

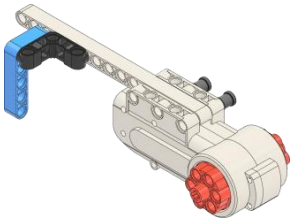


Рис.33 Копировальщик

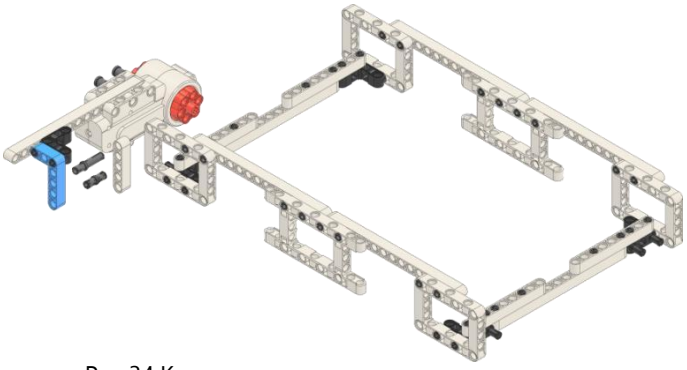


Рис.34 Копировальщик

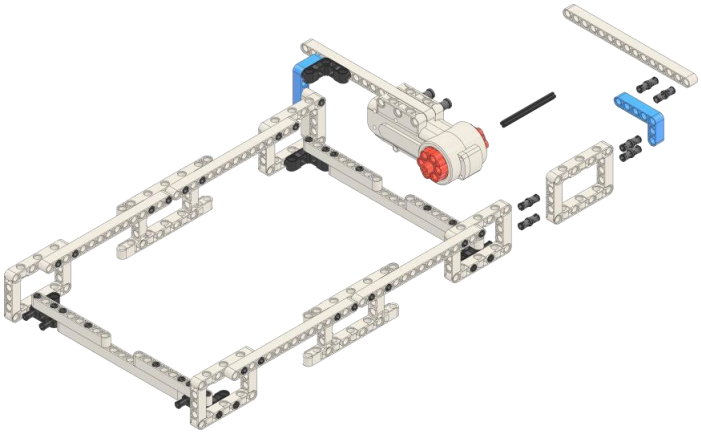


Рис.35 Копировальщик

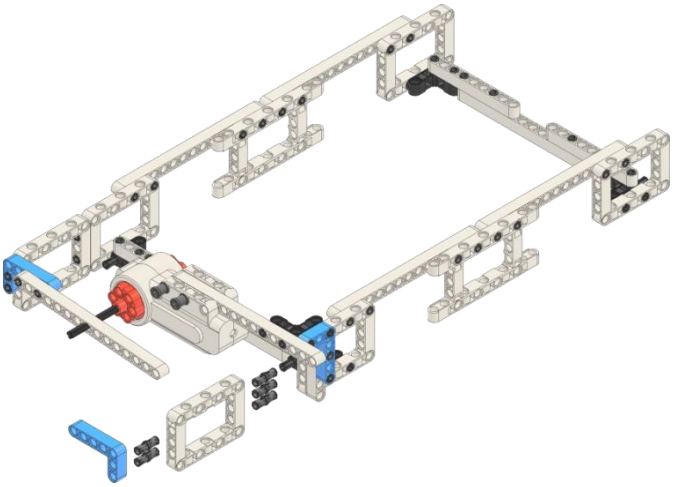


Рис.36 Копировальщик

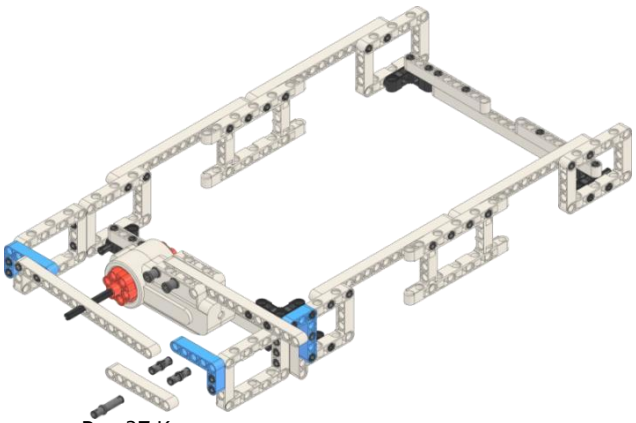


Рис.37 Копировальщик

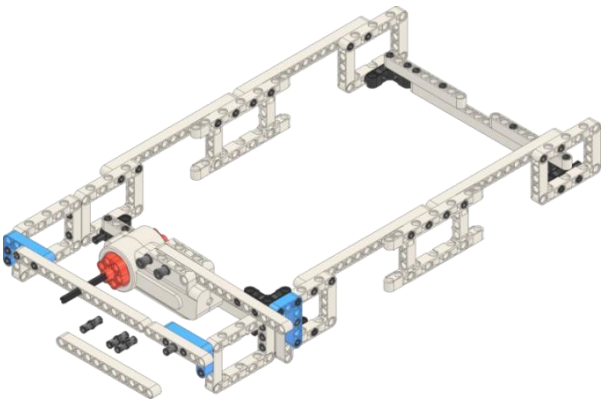


Рис.38 Копировальщик

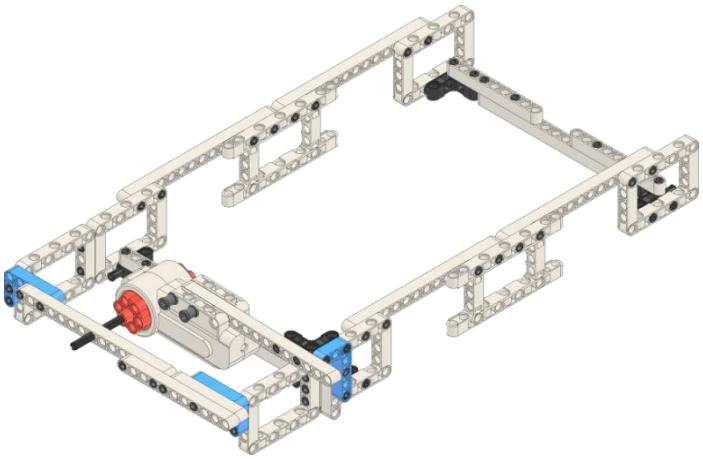


Рис.39 Копировальщик

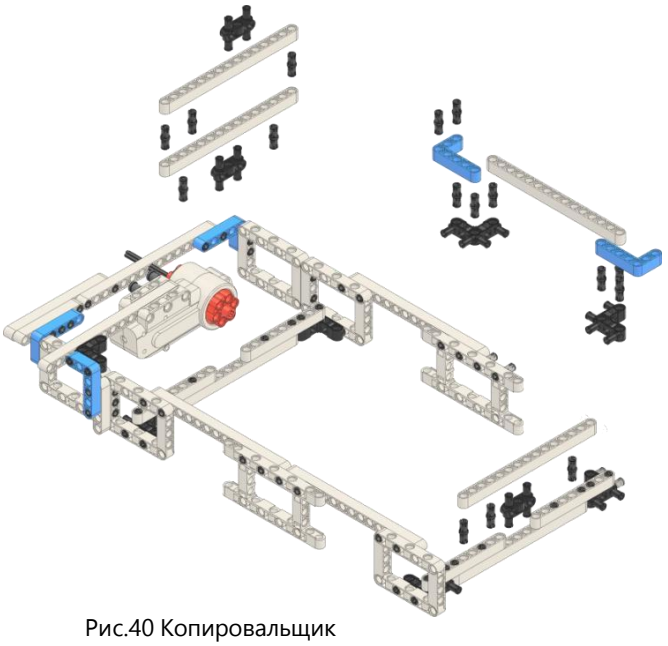


Рис.40 Копировальщик

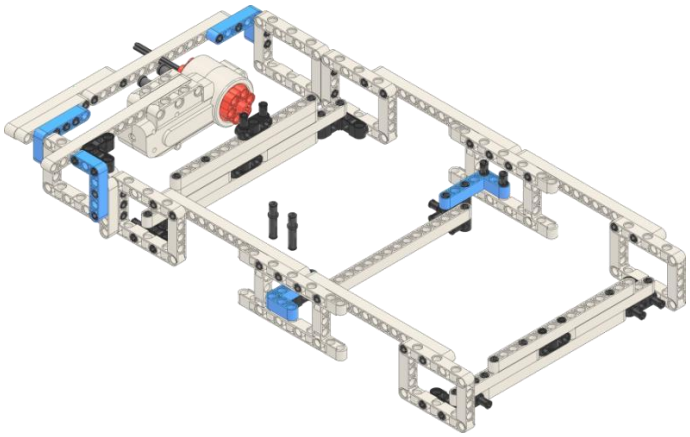


Рис.41 Копировальщик

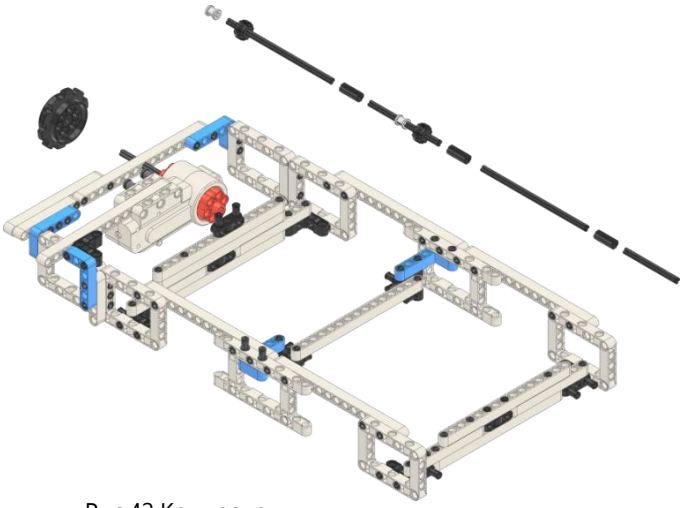


Рис.42 Копировальщик

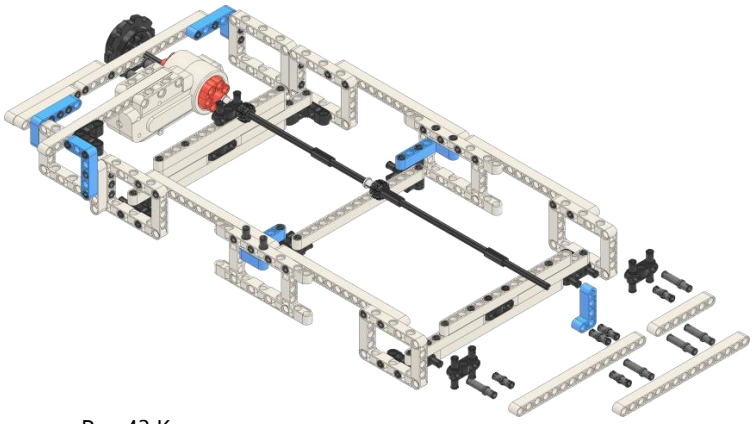


Рис.43 Копировальщик

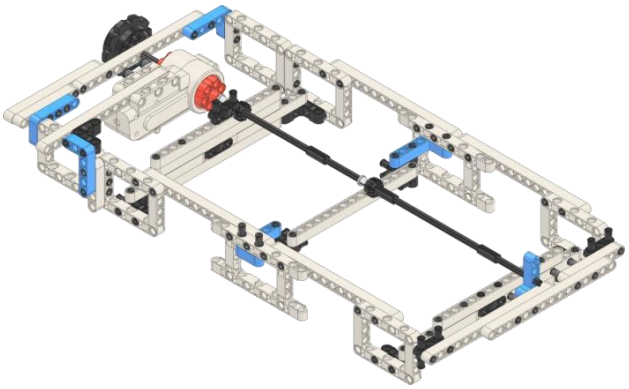


Рис.44 Копировальщик

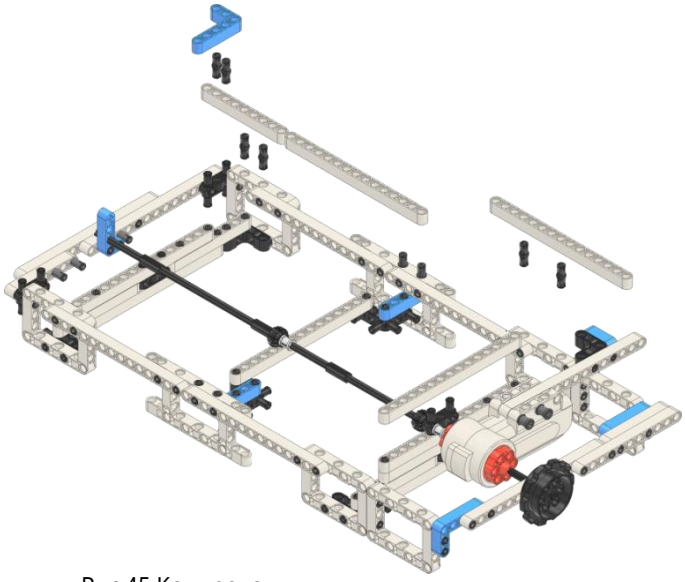


Рис.45 Копировальщик

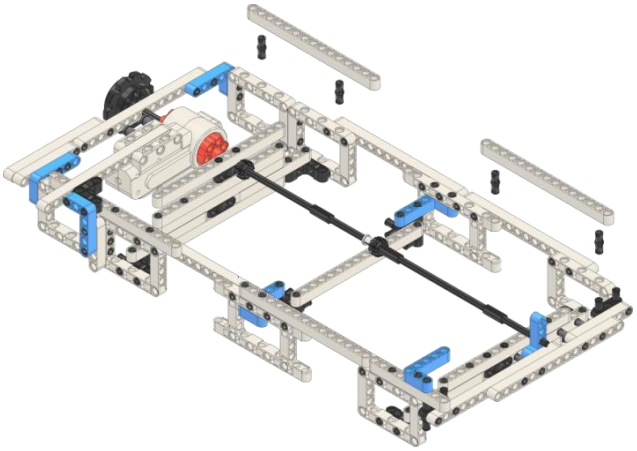


Рис.46 Копировальщик

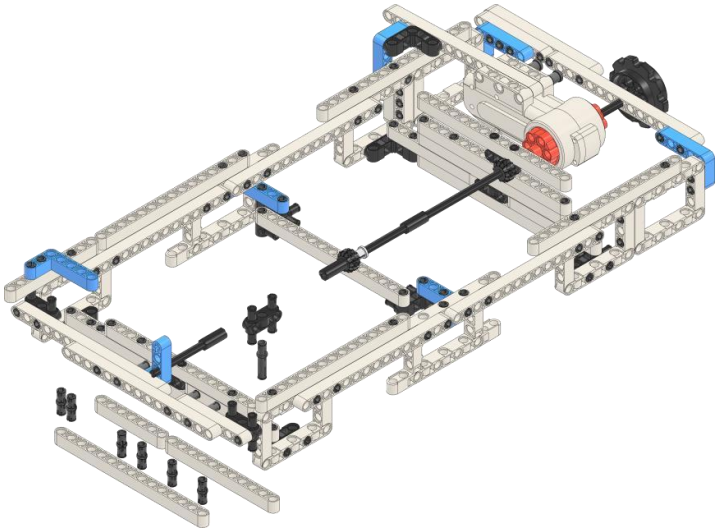


Рис.47 Копировальщик

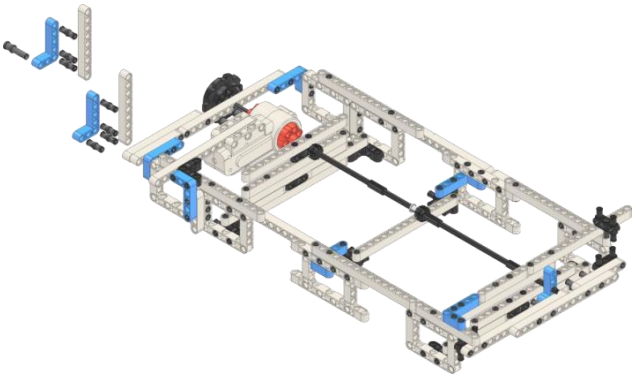


Рис.48 Копировальщик

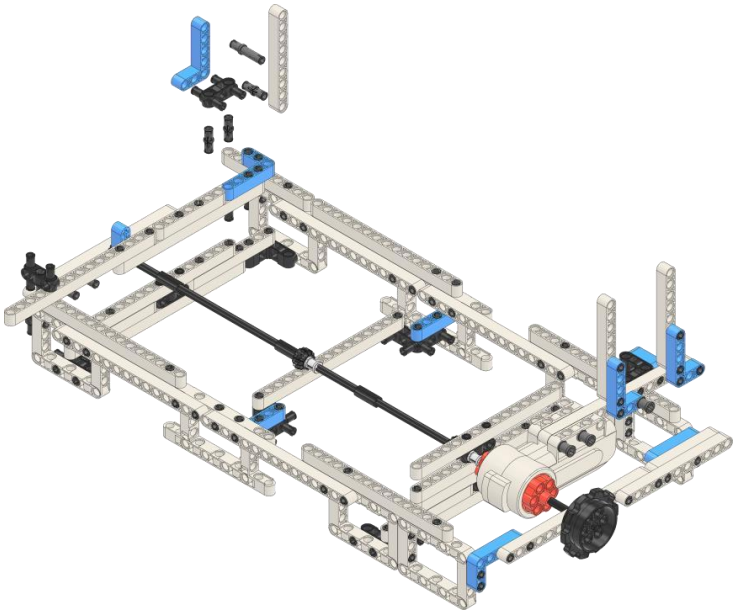


Рис.49 Копировальщик

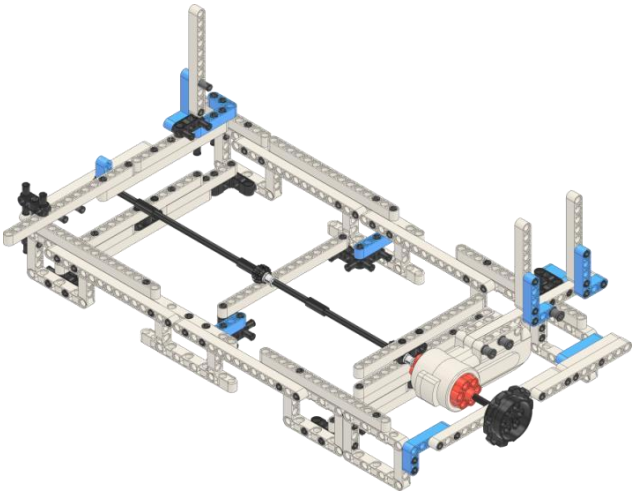


Рис.50 Копировальщик

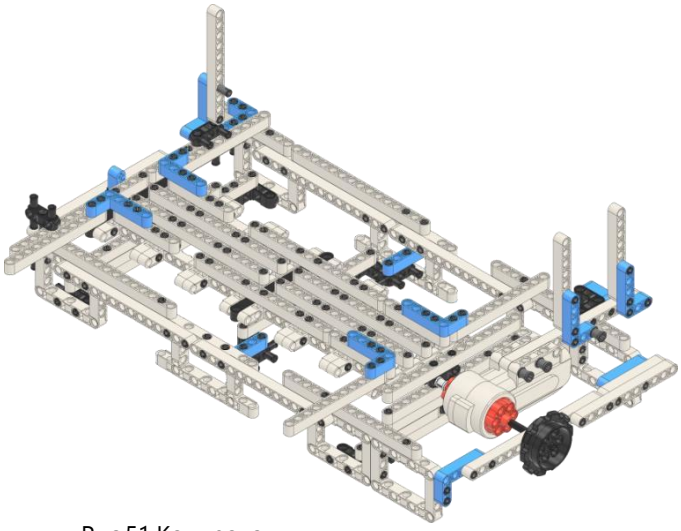


Рис.51 Копировальщик

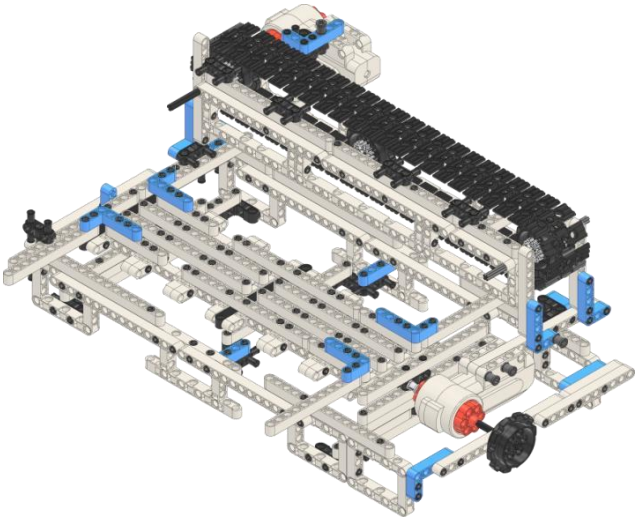


Рис.52 Копировальщик

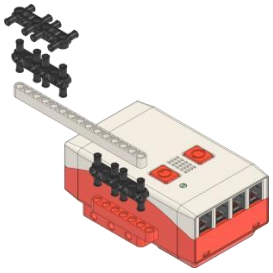


Рис.53 Копировальщик

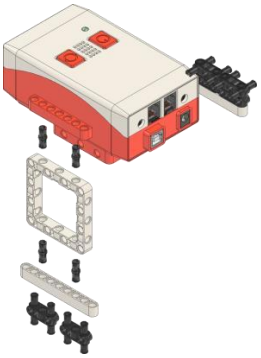


Рис.54 Копировальщик

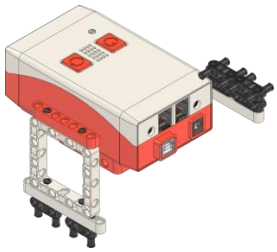


Рис.55 Копировальщик

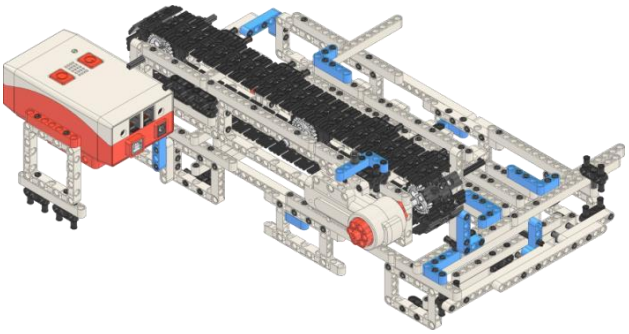


Рис.56 Копировальщик

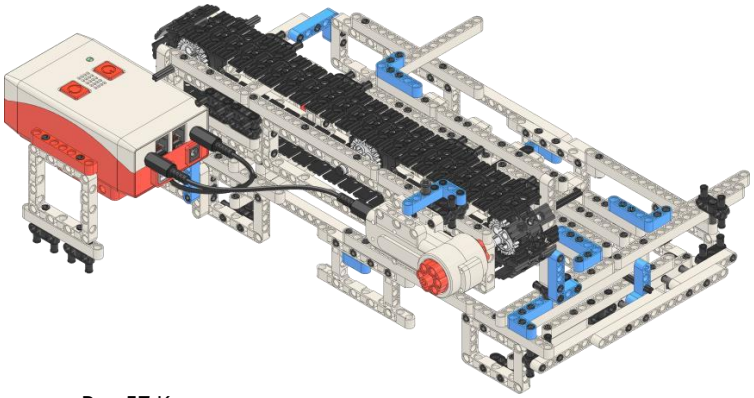


Рис.57 Копировальщик

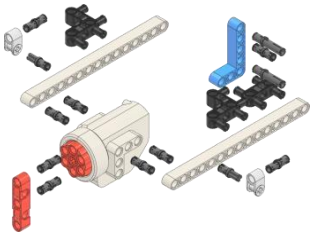


Рис.58 Копировальщик

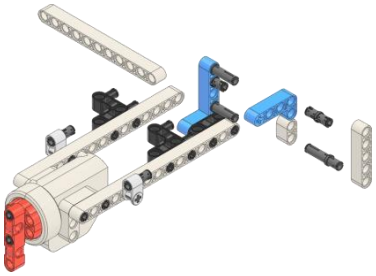


Рис.59 Копировальщик

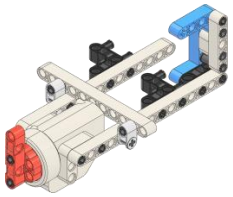


Рис.60 Копировальщик

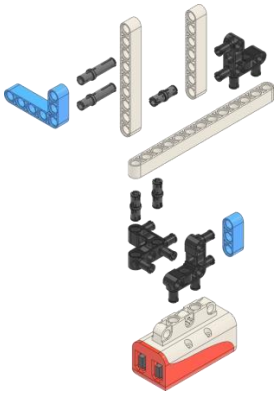


Рис.61 Копировальщик

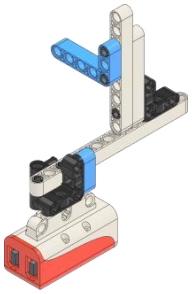


Рис.62 Копировальщик

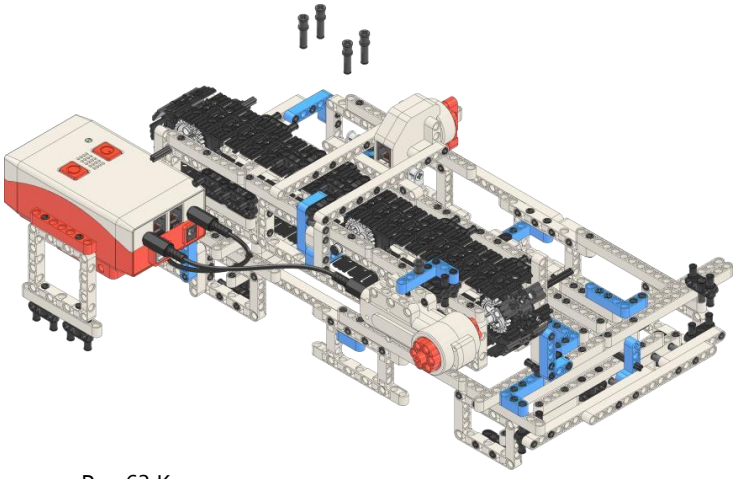


Рис.63 Копировальщик

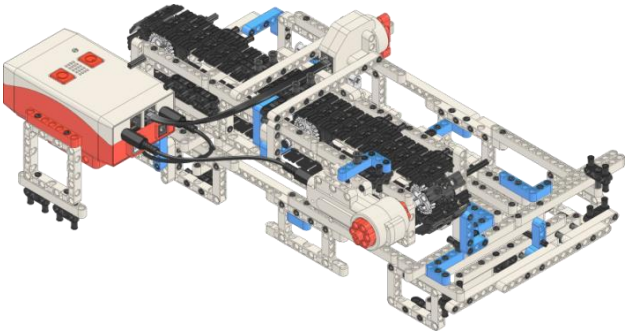


Рис.64 Копировальщик

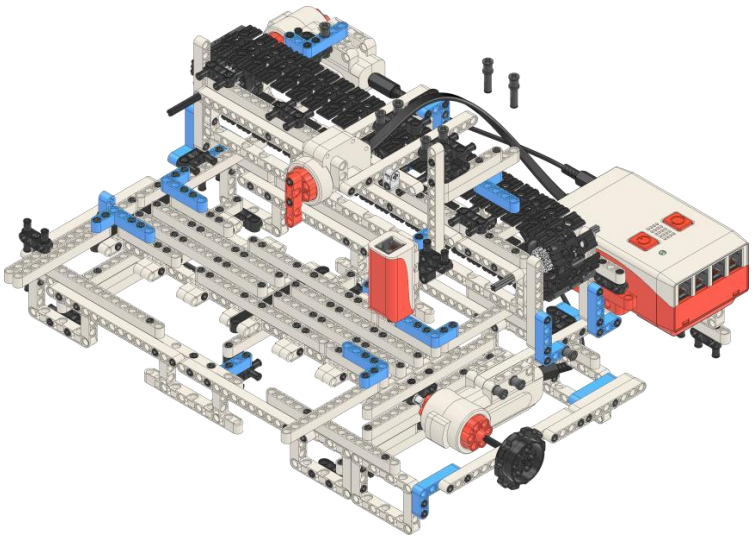


Рис.65 Копировальщик

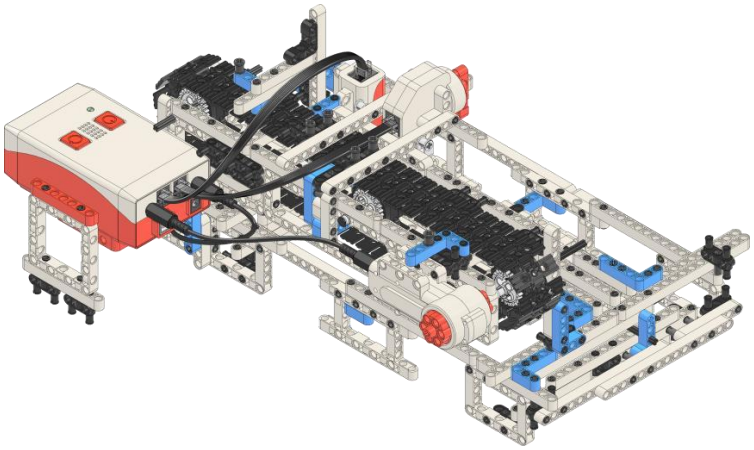


Рис.66 Копировальщик

Соединительные провода для датчика линии и сервопривода рекомендуется взять от 35 см.

Задание 2 (Уровень В)

Создайте программу, с помощью которой устройство будет способно копировать изображение.

Пример решения.

Рассмотрим самую примитивную реализацию работы устройства. К сервоприводу прикрепляется ручка. На движущемся столике располагают простой рисунок с чётко выраженными границами на белом фоне. Гусеничная платформа с датчиком линии и сервоприводом начинает постепенное движение в одну сторону, после определённого количества шагов, движение начинается в обратную сторону. После этого подвижный столик сдвигается на шаг и процесс повторяется. Таких повторений должно быть пока столик не дойдёт до конечной точки перемещения.

Как только датчик линии видит тёмный цвет, то блок управления подаёт сигнал на сервопривод опустить ручку. В результате этого действия ручка начнёт рисовать линию. Как только датчик линии увидит белый цвет, то сервопривод поднимет ручку.

Результат скопированного изображения будет содержать много погрешностей. В первую очередь размер рисунка будет зависеть от размера шага ленты и столика. Качество рисунка будет зависеть от точности расположения ручки и минимального отклонения датчика линии от поверхности. Расстояние между датчиком линии и рисунком должно быть в пределах 5 - 12 мм.

Пример рисунка.

```
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 =  5;

int l1 =11;
int l2 =10;

void setup() {
  pinMode(m1, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(v1, OUTPUT);
  pinMode(v2, OUTPUT);
  myservo.attach(3);
  myservo.write(60);

  analogWrite(6, 0);
  analogWrite(5, 0);
  pinMode(l1,INPUT);
  pinMode(l2,INPUT);
  Serial.begin(9600);

 for (int i=0; i<30; i++)
 {


 for (int i=0; i<10; i++)
 {
  Serial.println("left_line");
  Serial.println(digitalRead(l2));
  if (digitalRead(l2)==0){
   myservo.write(20);
     }
     else{
       myservo.write(60);
       }
   digitalWrite(m1,LOW);

 analogWrite(6, 100);
  //analogWrite(5, 100);
  delay(50);

 analogWrite(6, 0);
 analogWrite(5, 0);
 delay(500);


 }
```

Рис.67 Начало программы Копировальщик

```
 digitalWrite(m2,HIGH);

analogWrite(5, 100);
 //analogWrite(5, 100);
 delay(30);
 analogWrite(5, 0);
delay(500);

if (digitalRead(l2)==0){
  myservo.write(20);
    }

    else{
      myservo.write(60);
      }

for (int i=0; i<10; i++)
{
  Serial.println("left_line");
  Serial.println(digitalRead(l2));
if (digitalRead(l1)==0){
  myservo.write(20);
    }

    else{
      myservo.write(60);
      }
digitalWrite(m1,HIGH);
analogWrite(6, 100);
 //analogWrite(5, 100);
 delay(50);

  analogWrite(6, 0);
  analogWrite(5, 0);
delay(500);
    }
}
}

void loop() {
  }
```

Рис.68 Конец программы Копировальщик

9.4. Роботанк

Роботанк – робот на гусеничной платформе.

Методические рекомендации.

Тема: «Роботанк»

Цель: изучить процесс создания и программирования устройства на гусеничной платформе.

Задачи:

- изучить и закрепить на практике процесс создания роботанка
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Пример заданий.

Задание 1 (Уровень А)

Соберите мобильную гусеничную платформу согласно инструкции.

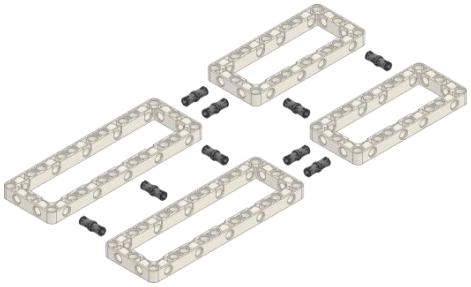


Рис.1 Роботанк

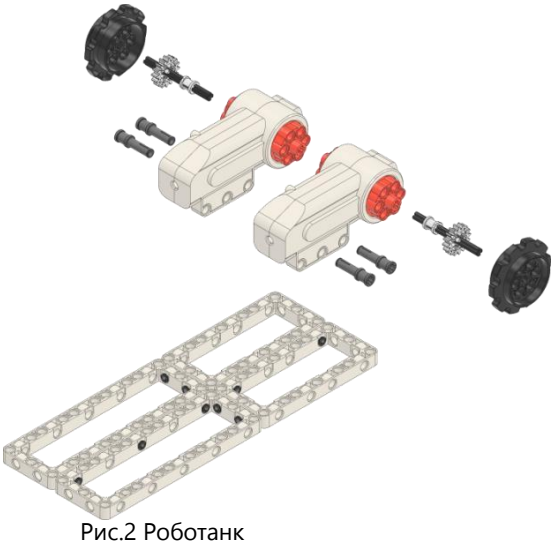


Рис.2 Роботанк

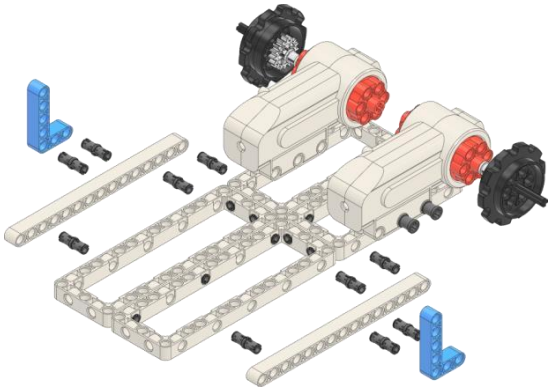


Рис.3 Роботанк

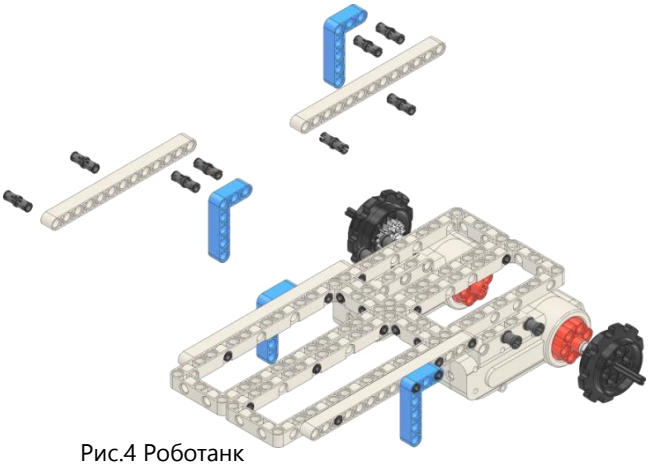


Рис.4 Роботанк
Используем оси размером 6.

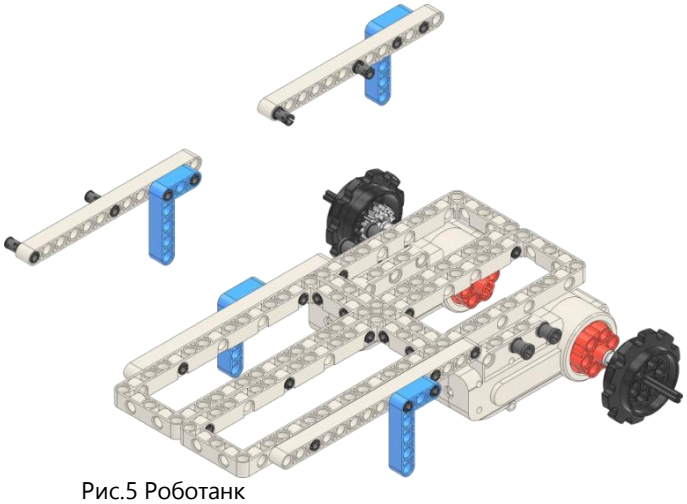


Рис.5 Роботанк

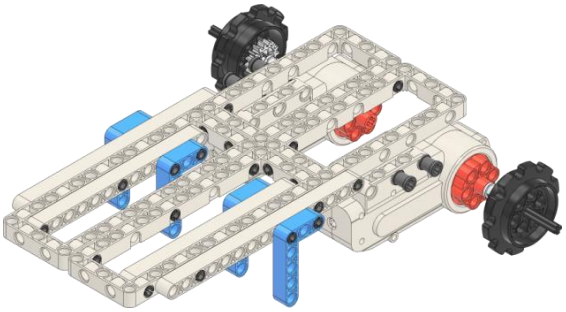


Рис.6 Роботанк

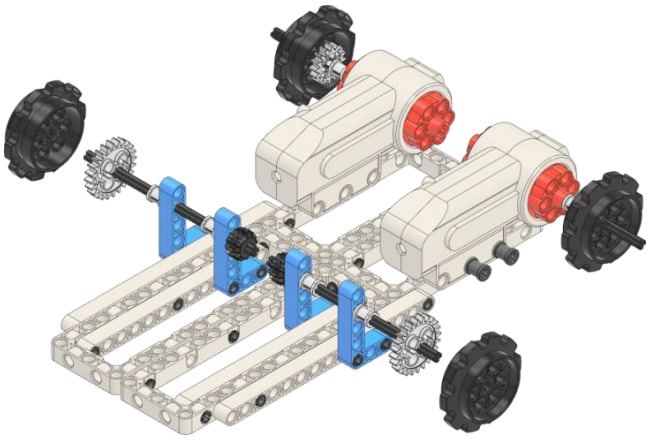


Рис.7 Роботанк

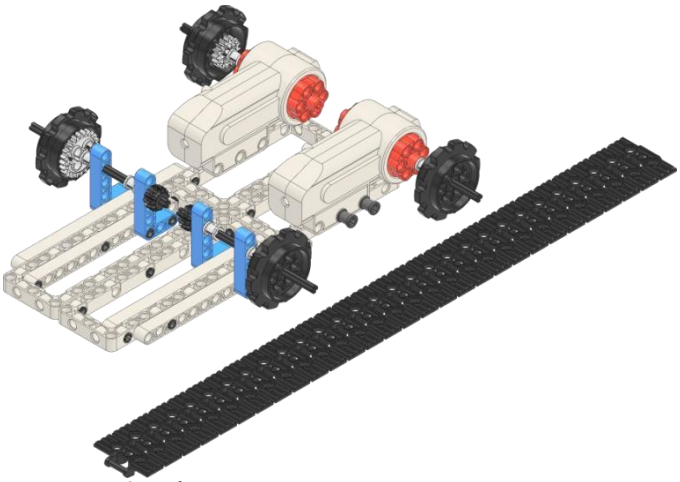


Рис.8 Роботанк

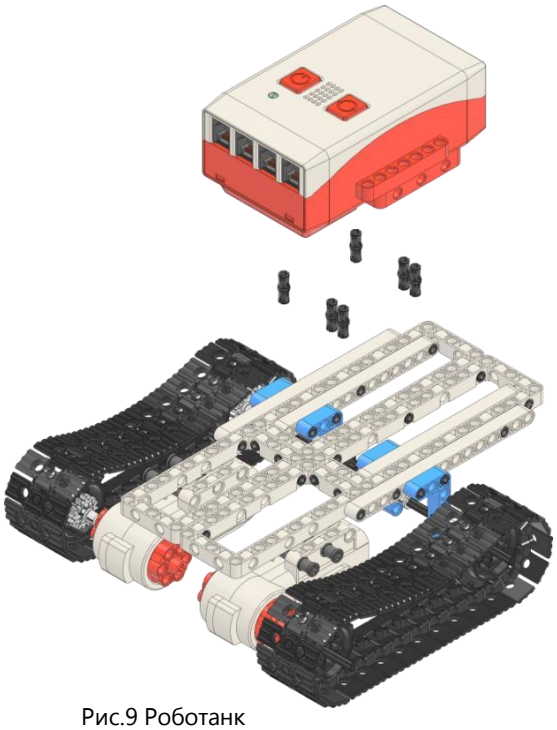


Рис.9 Роботанк

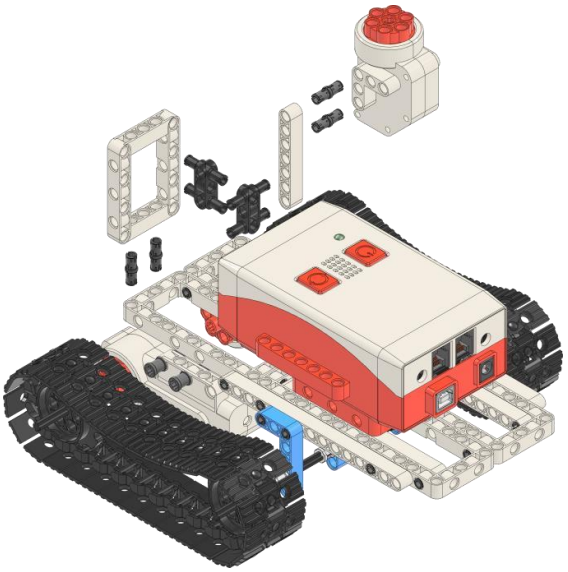


Рис.10 Роботанк

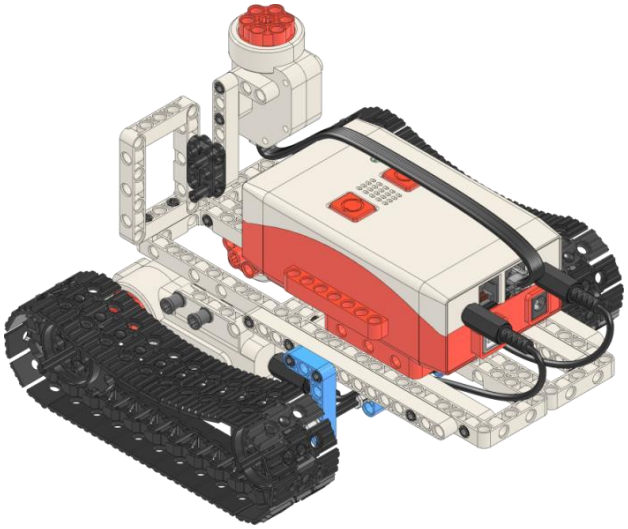


Рис.11 Роботанк

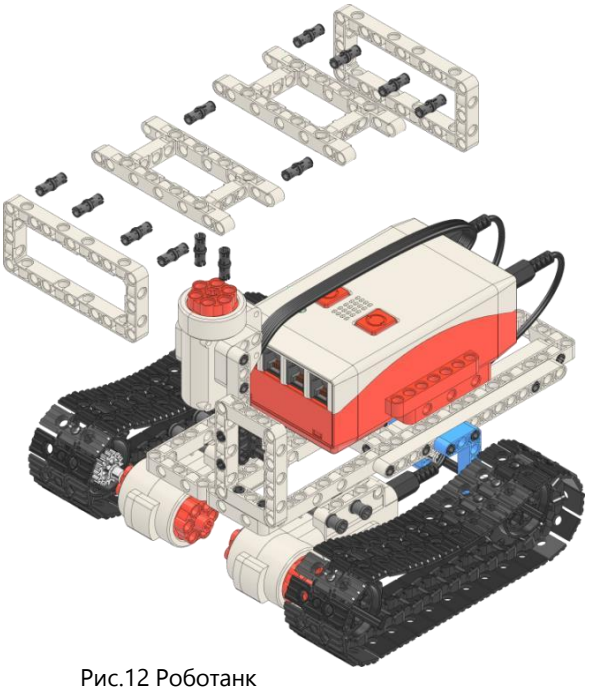


Рис.12 Роботанк
Используются оси размером 12.

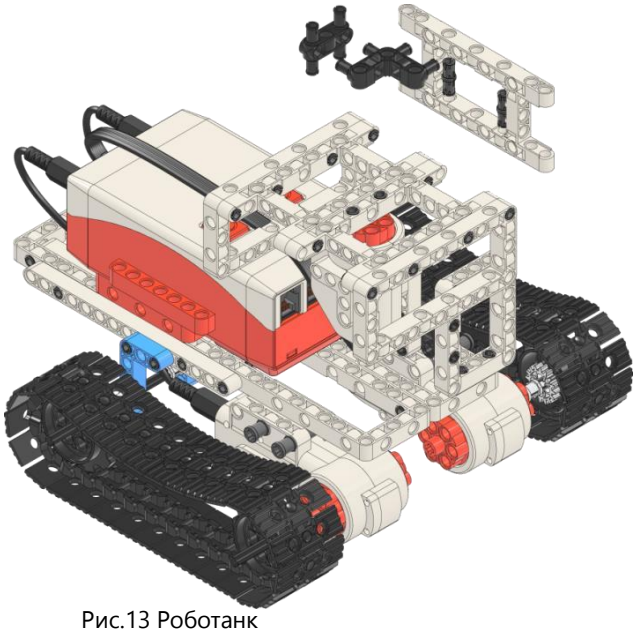


Рис.13 Роботанк

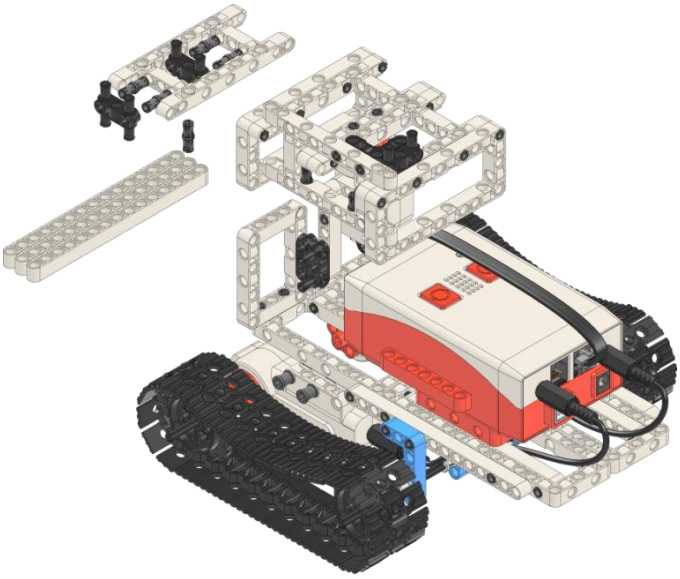


Рис.14 Роботанк

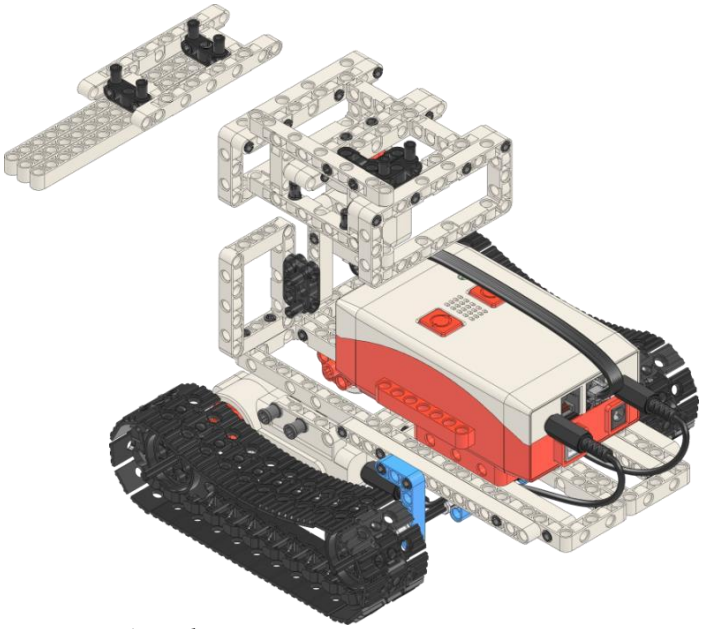


Рис.15 Роботанк

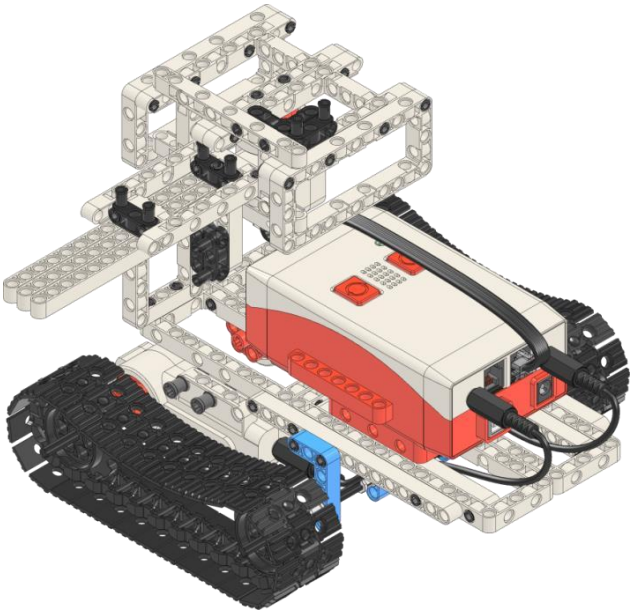


Рис.16 Роботанк

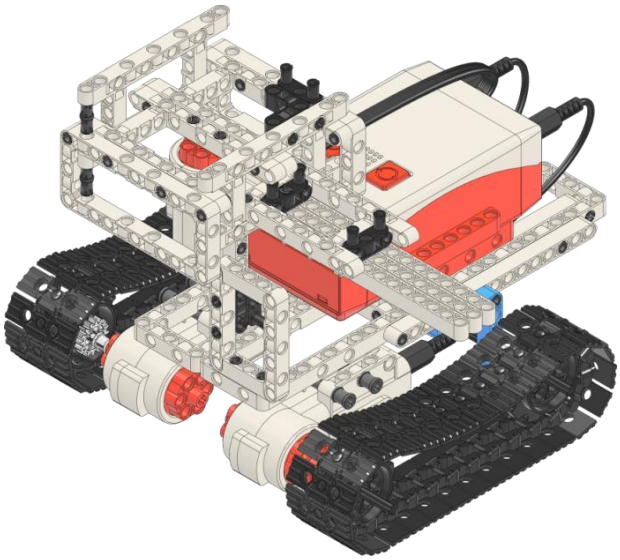


Рис.17 Роботанк

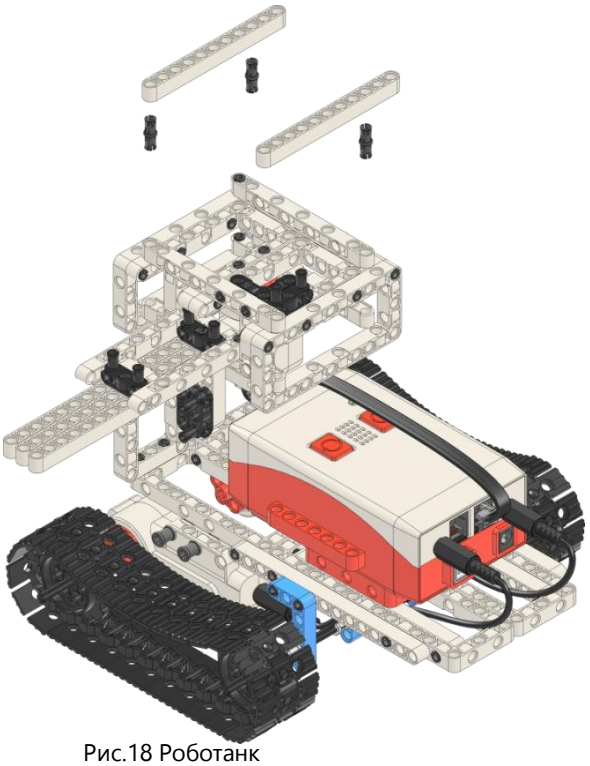


Рис.18 Роботанк

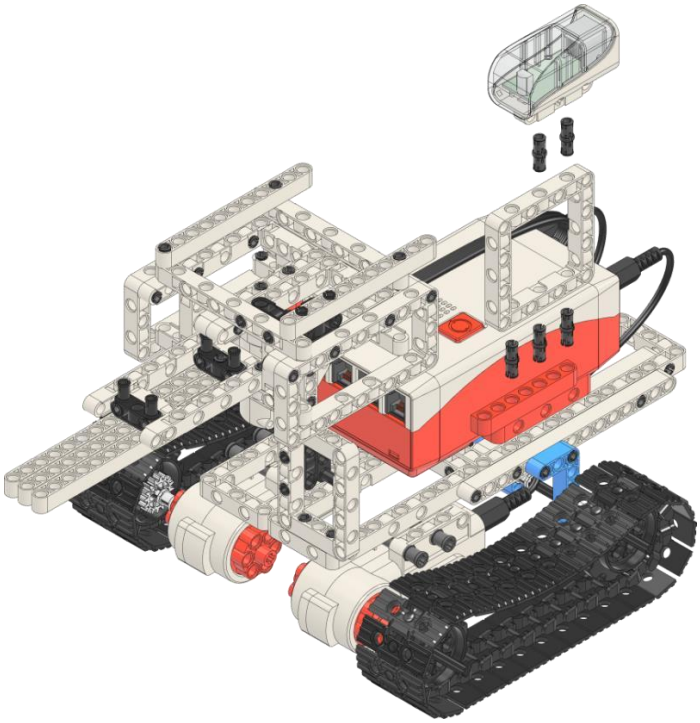


Рис.19 Роботанк

Задание 2 (Уровень В)

Создать программу, с помощью которой можно будет управлять танком посредством IR пульта.

Пример решения.

Для реализации данной программы, воспользуемся информацией, которую почерпнули из глав 4.6 и 5.6. Опираясь на эту информацию создадим программу, где с помощью сигналов по IR можно будет управлять движением как самого танка, так и башни.

```
#include <Servo.h>
#include <IRremote.h>

int RECV_PIN = 8;
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;
IRrecv irrecv(RECV_PIN);

decode_results results;
Servo myservo;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
  pinMode(m1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(v2,OUTPUT);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  myservo.attach(12);
  myservo.write(10);
}
```

Рис.20 Первая часть программы Роботанк

```
void loop() {
  if (irrecv.decode(&results)) {
    //Serial.println(results.value, HEX);
    Serial.println(results.value);
    irrecv.resume();

    if (results.value == 16720605){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, LOW);
       digitalWrite(m2, HIGH);
      }
    else if (results.value == 16761405){
      analogWrite(v1, 0);
      analogWrite(v2, 0);
      }
 else if (results.value == 16712445){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, LOW);
      }
 else if (results.value == 16753245){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, HIGH);
      }
```

Рис.21 Вторая часть программы Роботанк

```
      else if (results.value == 16769565){
       analogWrite(v1, 100);
       analogWrite(v2, 100);
       digitalWrite(m1, LOW  );
        digitalWrite(m2, LOW);
       }

        else if (results.value == 16769055){
       myservo.write(10);
       }
       else if (results.value == 16754775){
       myservo.write(170);
       }
         else if (results.value == 16748655){
       myservo.write(90);
       }

  }
}
```

Рис. 22Заключительная часть программы Роботанк

9.5. Робот Муравей

Методические рекомендации.

Тема: «Робот Муравей»

Цель: изучить процесс создания и программирования устройства со стопоходящим механизмом.

Задачи:

- изучить и закрепить на практике процесс создания стопоходящего робота
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Пример заданий.

Задание 1 (Уровень А)

Собрать робота муравья согласно инструкции.

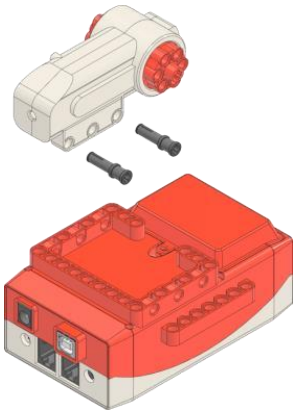


Рис.1 Робот муравей

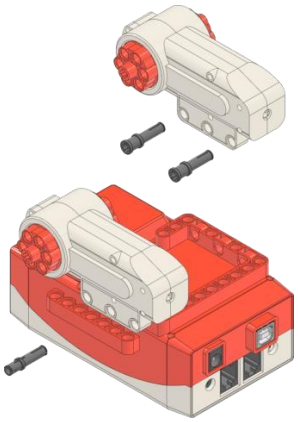


Рис.2 Робот муравей

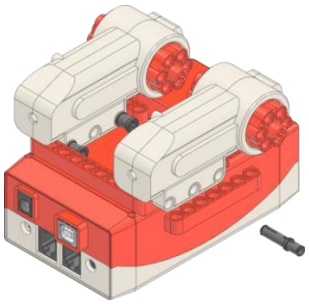


Рис.3 Робот муравей

Рис.4 Робот муравей

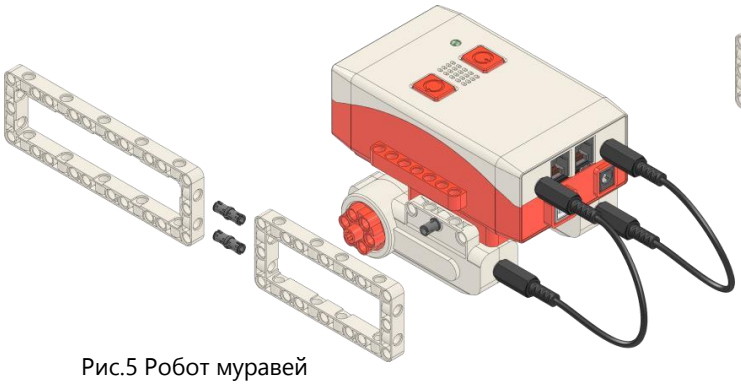


Рис.5 Робот муравей

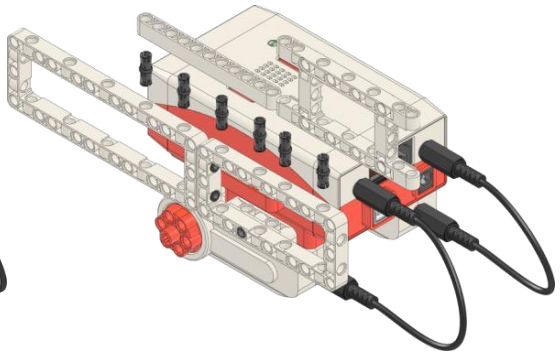


Рис.6 Робот муравей

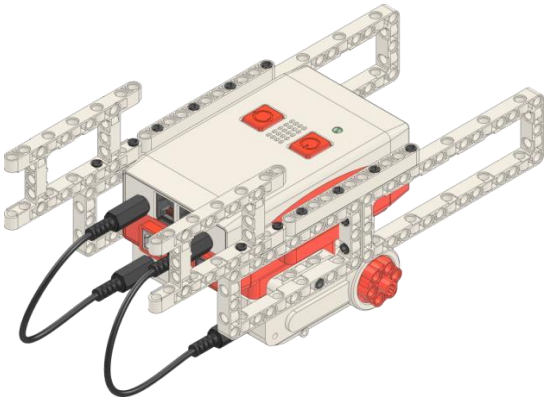


Рис.7 Робот муравей

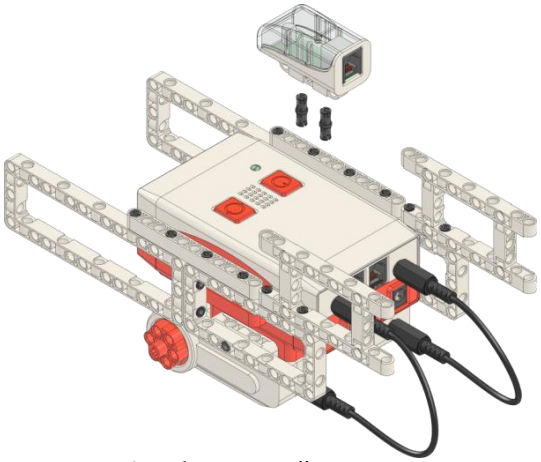


Рис.8 Робот муравей

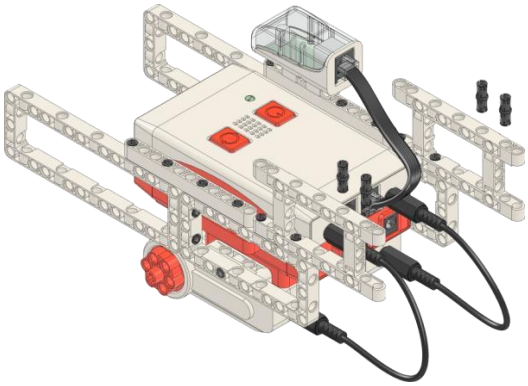


Рис.9 Робот муравей

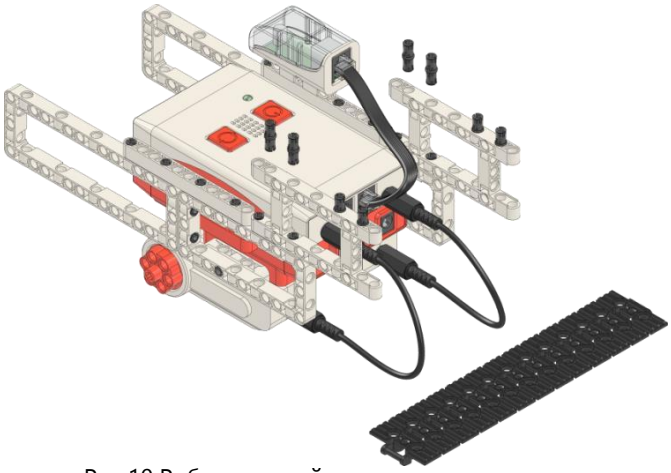


Рис.10 Робот муравей

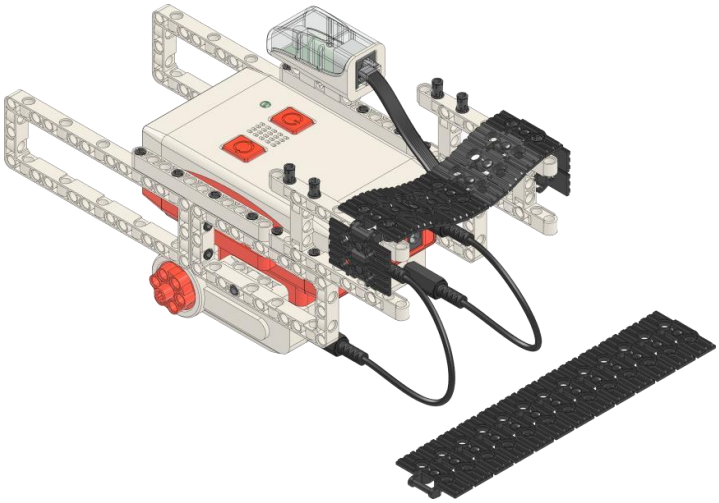


Рис.11 Робот муравей

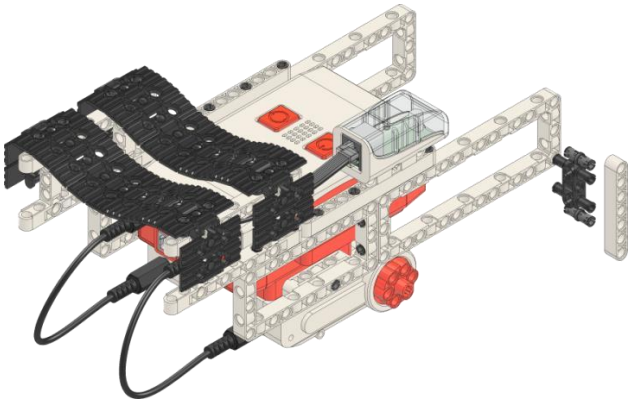


Рис.12 Робот муравей

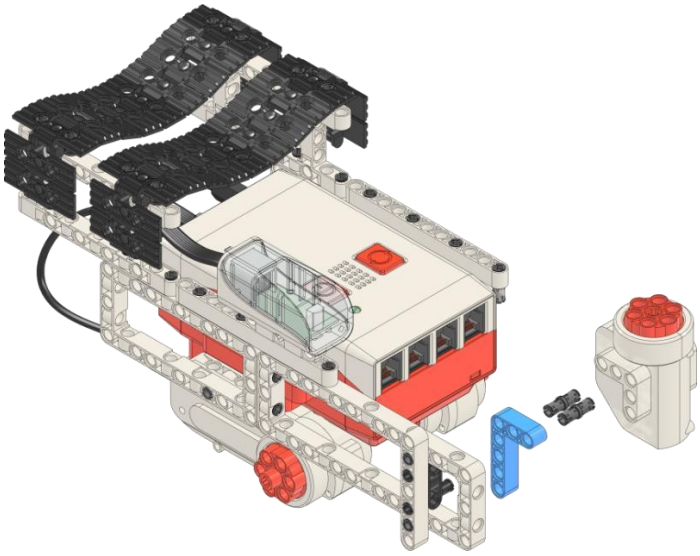


Рис.13 Робот муравей

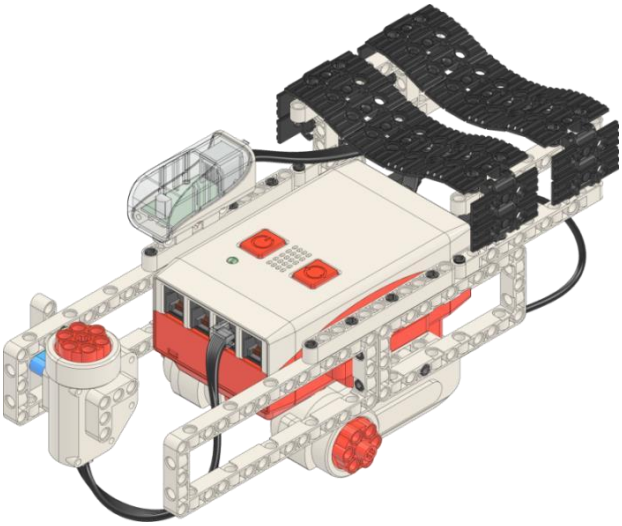


Рис.14 Робот муравей

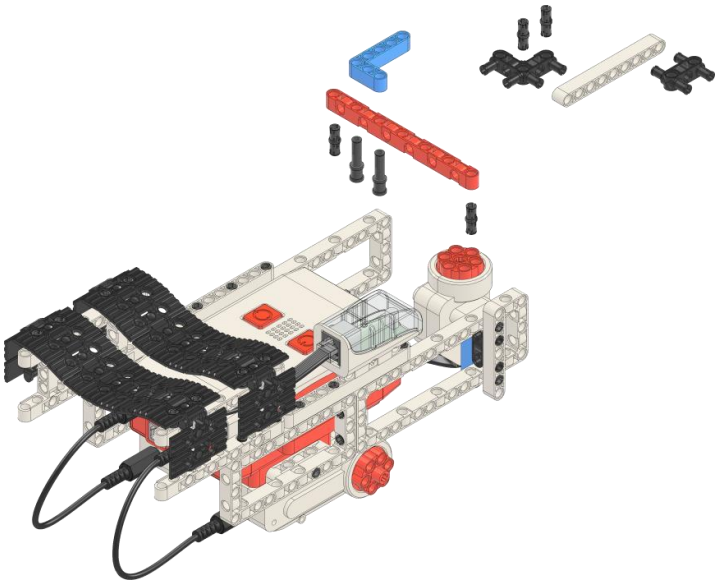


Рис.15 Робот муравей

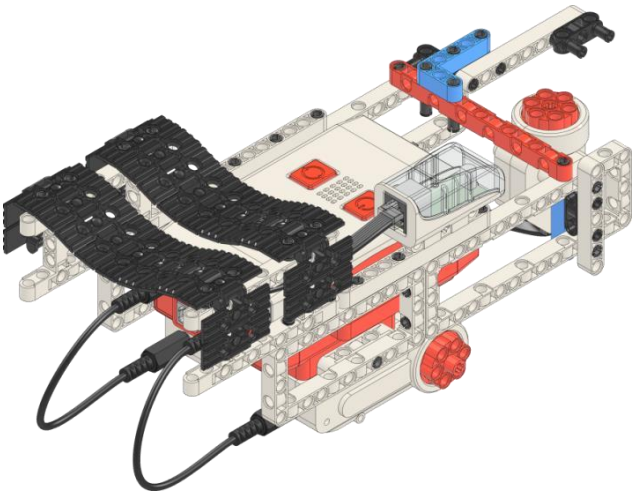


Рис.16 Робот муравей

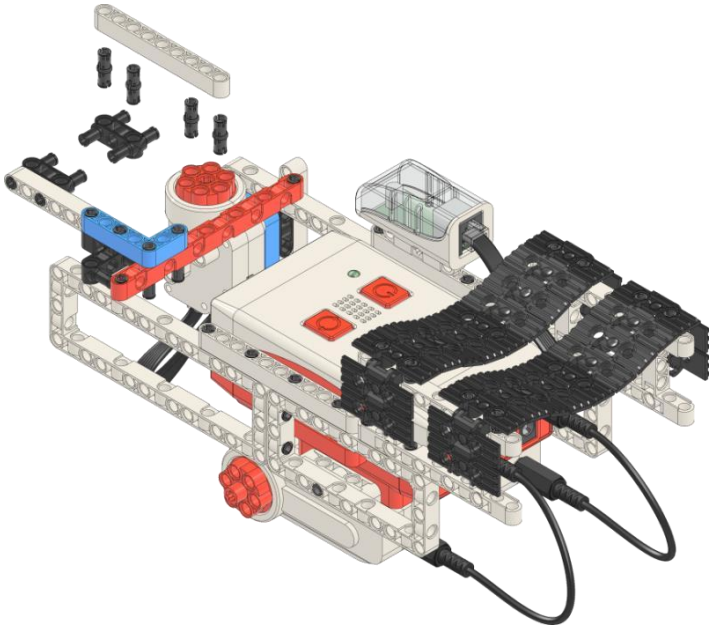


Рис.17 Робот муравей

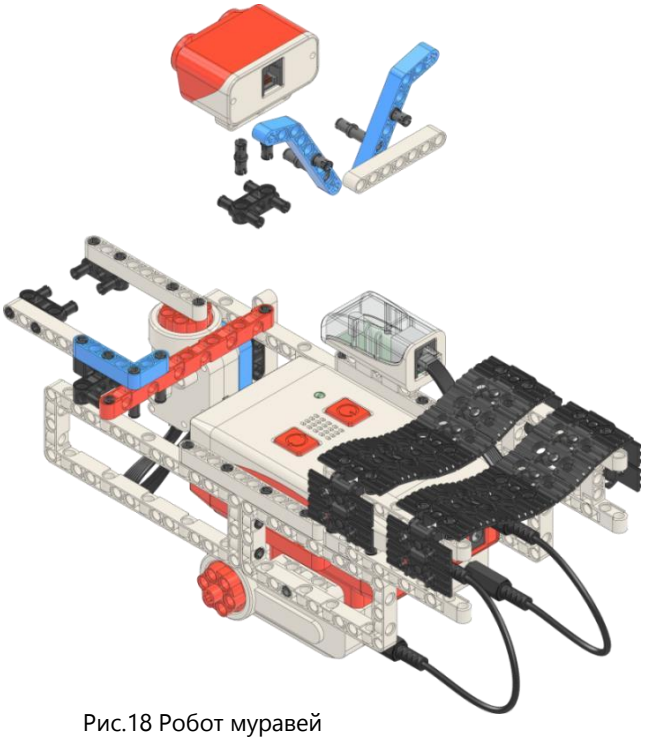


Рис.18 Робот муравей

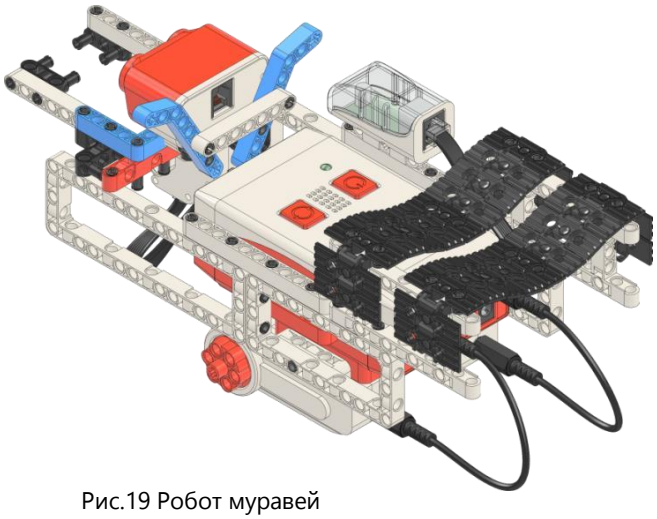


Рис.19 Робот муравей

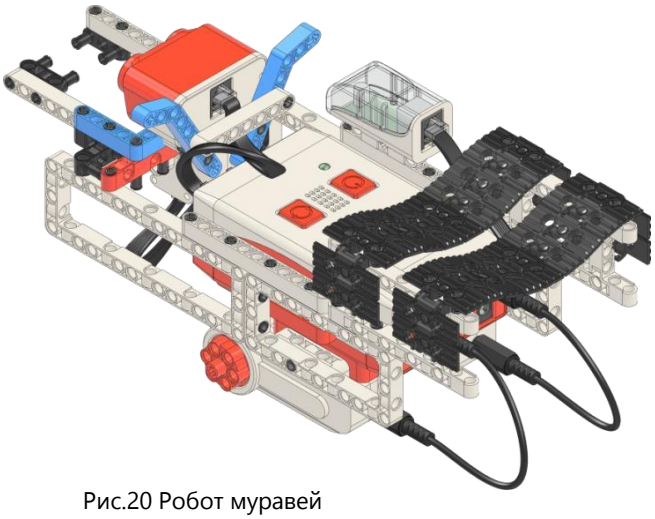


Рис.20 Робот муравей

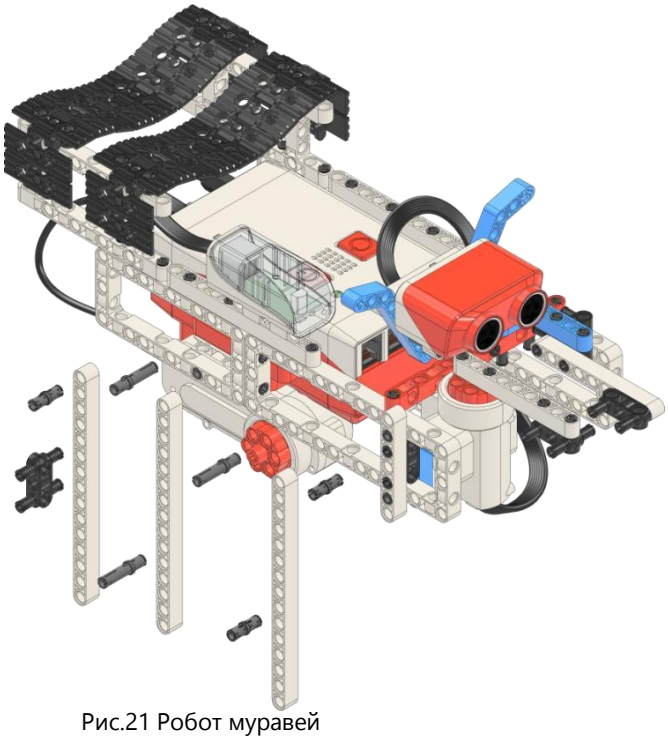


Рис.21 Робот муравей

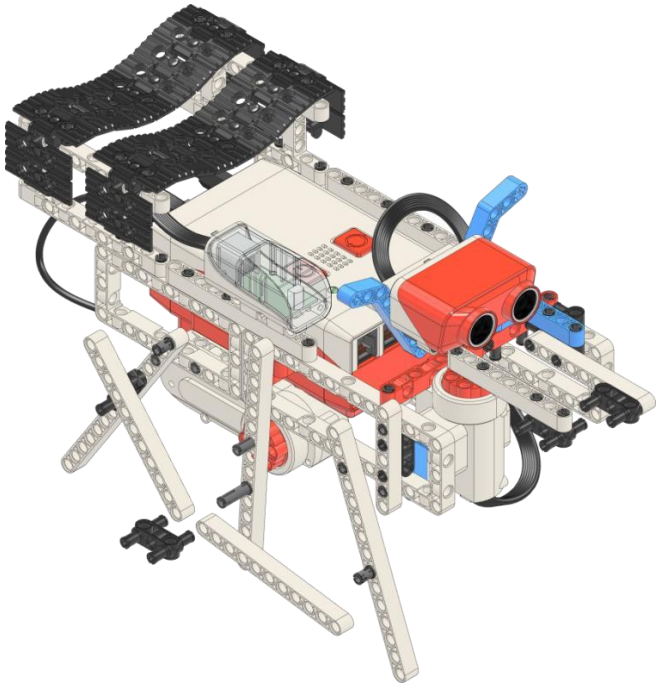


Рис.22 Робот муравей

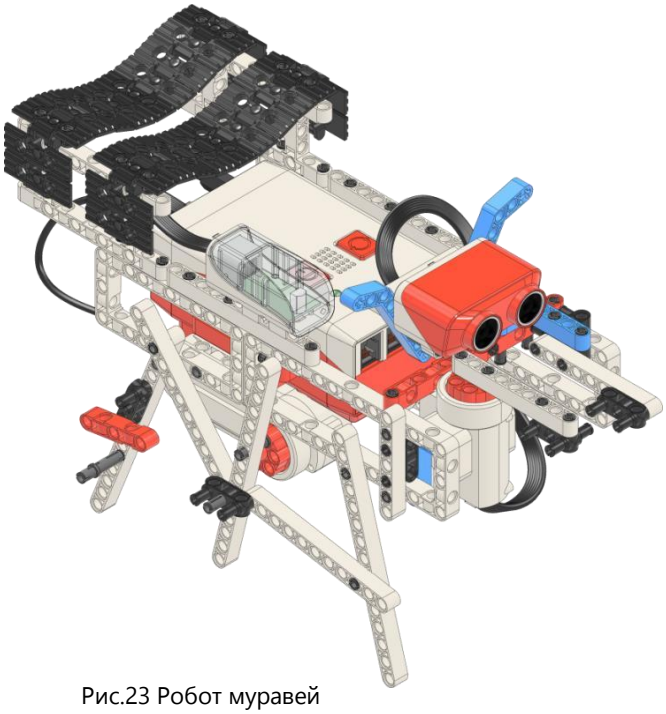


Рис.23 Робот муравей

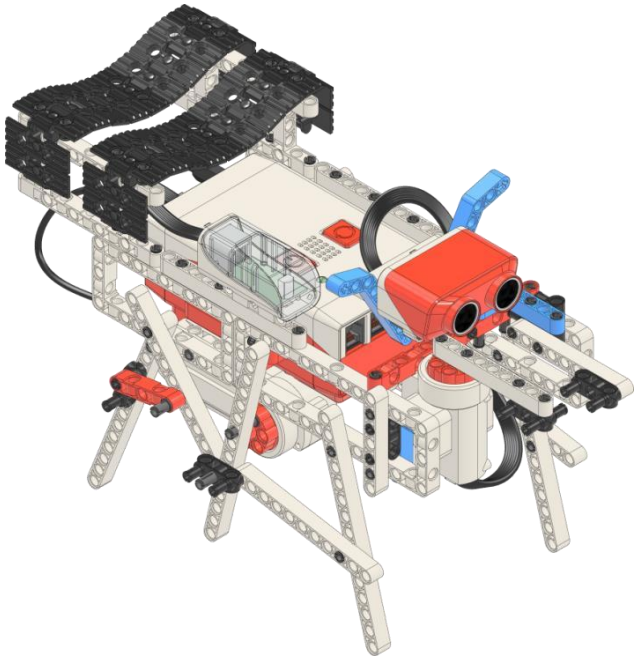


Рис.24 Робот муравей

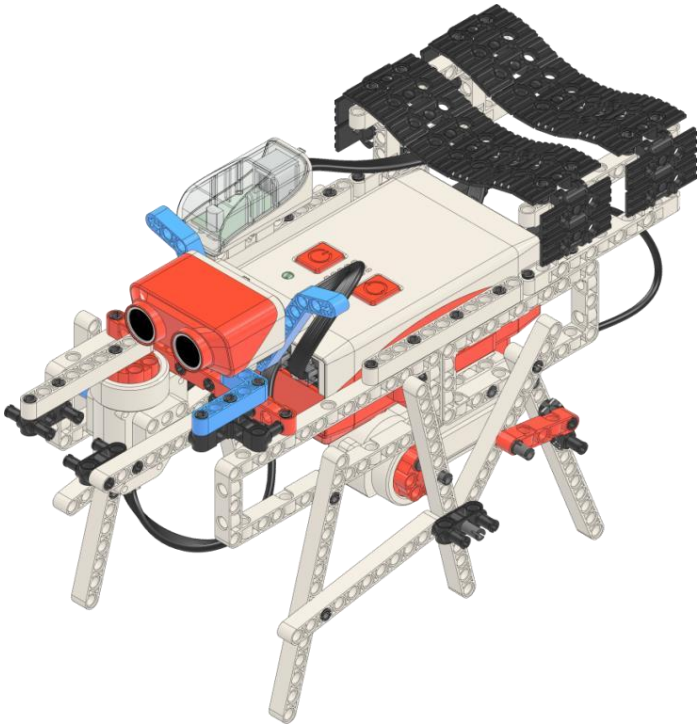


Рис.25 Робот муравей

Задание 2 (Уровень В)

Создать программу управления муравьём с помощью IR модуля: движение вперёд, движение назад, поворот налево поворот направо, остановка.

Для реализации подобной программы необходимо установить библиотеку по работе с модулем IR и протестировать программы по запуску двух моторов так, чтобы ноги перемещали робота по направлениям в плоскости.

Пример программы:

```
#include <Servo.h> //используем библиотеку для работы с сервоприводом
#include <IRremote.h>

Servo servo;
int m1 = 7;
int m2 = 4;

int v1 = 6;
int v2 = 5;

int trig = 8;
int echo = 2;
long dur, cm;

int RECV_PIN = 3;

IRrecv irrecv(RECV_PIN);

decode_results results;
```

Рис.26 Программа - Робот муравей (1 часть)

```
void setup()
{
  servo.write(30);
  Serial.begin(9600);
  irrecv.enableIRIn(); // Start the receiver
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);
  servo.attach(12);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  pinMode(trig, OUTPUT);

pinMode(echo, INPUT);

}
```

Рис.27 Программа - Робот муравей (2 часть)

```
void loop() {

  digitalWrite(trig, LOW);

delayMicroseconds(5);

digitalWrite(trig, HIGH);

delayMicroseconds(10);

digitalWrite(trig, LOW);
pinMode(echo, INPUT);

dur = pulseIn(echo, HIGH);
cm = (dur/2) / 29.1;

Serial.print(cm);
Serial.print("cm");
Serial.println();
```

Рис.28 Программа - Робот муравей (3 часть)

```
if ( cm<=15 ){
      servo.write(100);
      }
else{
    servo.write(10);
  }
 if (irrecv.decode(&results)) {
    Serial.println(results.value);
    irrecv.resume();
 if (results.value == 16720605){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, LOW);
       digitalWrite(m2, HIGH);
      }
    else if (results.value == 16761405){
      analogWrite(v1, 0);
      analogWrite(v2, 0);
      }
```

Рис.29 Программа - Робот муравей (4 часть)

```
 else if (results.value == 16712445){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, LOW);
      }
 else if (results.value == 16753245){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, HIGH);
       digitalWrite(m2, HIGH);
      }
     else if (results.value == 16769565){
      analogWrite(v1, 100);
      analogWrite(v2, 100);
      digitalWrite(m1, LOW  );
       digitalWrite(m2, LOW);
      }
```

Рис.30 Программа - Робот муравей (5 часть)

```
 else if (results.value == 16724175){
       servo.write(30);
       }

 else if (results.value == 16718055){
       servo.write(100);
       }

   }
     }
```

Рис.31 Программа - Робот муравей (6 часть)

Согласно данной программы робот муравей реагирует на приближение объекта к нему и если расстояние меньше 15 см, то запускается сервопривод на атаку.

Кроме этого муравей может управляться с помощью пульта IR (смотри главу, посвящённую модулю IR). Мощность моторов подобраны оптимально стабильному перемещению.

Задание 3 (Уровень С)

Подключите Bluetooth модуль и составьте программу по управлению роботом по Bluetooth.

9.6. Ультразвуковой терменвокс

Ещё одно интересное устройство это **терменвокс**. Терменвокс – это музыкальный инструмент, реагирующий на изменение электромагнитного поля. Здесь представлен ультразвуковой терменвокс – музыкальный инструмент реагирующий на изменение расстояния до объекта.

Методические рекомендации.

Тема: «Ультразвуковой терменвокс»

Цель: изучить процесс создания и программирования музыкального устройства.

Задачи:

- изучить и закрепить на практике процесс создания музыкального устройства
- изучить особенности управления данной конструкцией
- получить и закрепить на практике знания, умения и навыки в области работы со звуком и создания программ для управления звучанием.

Задание1 (Уровень А)

Соберите устройство.

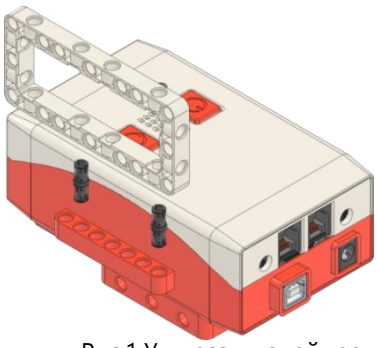


Рис.1 Ультразвуковой терменвокс

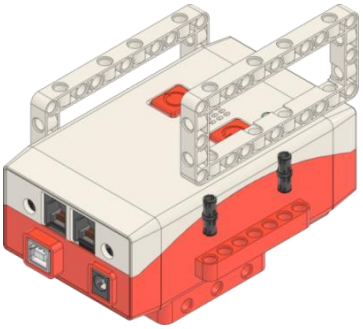


Рис.2 Ультразвуковой терменвокс

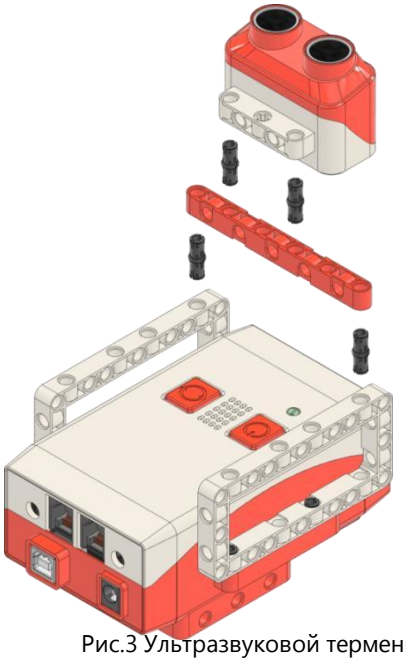


Рис.3 Ультразвуковой терменвокс

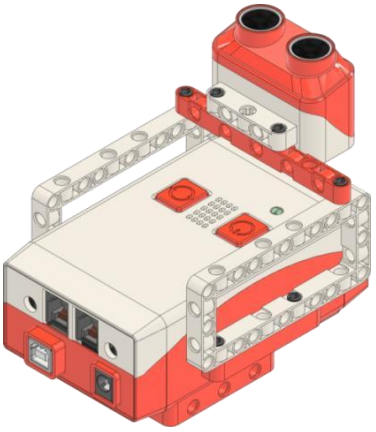


Рис.4 Ультразвуковой терменвокс

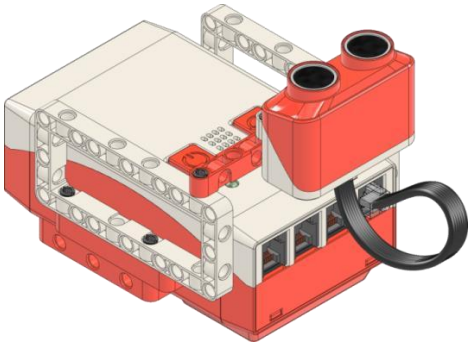


Рис.5 Ультразвуковой терменвокс

Задание 2 (Уровень В)
Составьте программу.

Программу необходимо создать так, чтобы в ней работал алгоритм на проверку расстояния от ультразвукового датчика и в зависимости от значения издавался бы звук определённой частоты. Для звука возьмём частоты нот для первой октавы.

Ноты	Суббконтр-октава	Контр-октава	Большая	Малая	Первая	Вторая	Третья	Четвертая	Пятая
ДО	16,35	32,70	65,41	130,82	261,63	523,26	1046,52	2093,04	4186,08
ДО диез	17,32	34,65	69,30	138,59	277,18	554,36	1108,72	2217,44	4434,88
РЕ	18,35	36,71	73,42	146,83	293,66	587,32	1174,64	2349,28	4698,56
РЕ диез	19,45	38,89	77,78	155,57	311,13	622,26	1244,52	2489,04	4978,08
МИ	20,60	41,20	82,41	164,82	329,63	659,26	1318,52	2637,04	5274,08
ФА	21,83	43,65	87,31	174,62	349,23	698,46	1396,92	2793,84	5587,68
ФА диез	23,12	46,25	92,50	185,00	369,99	739,98	1479,96	2959,92	5919,84
СОЛЬ	24,50	49,00	98,00	196,00	392,00	784,00	1568,00	3136,00	6272,00
СОЛЬ диез	25,96	51,91	103,83	207,65	415,30	830,60	1661,20	3322,40	6644,80
ЛЯ	27,50	55,00	110,00	220,00	440,00	880,00	1760,00	3520,00	7040,00
ЛЯ диез	29,14	58,27	116,54	233,08	466,16	932,32	1864,64	3729,28	7458,56
СИ	30,87	61,74	123,47	246,94	493,88	987,76	1975,52	3951,04	7902,08

Создадим программы в среде Arduino ide.

```
int trig = 8;
int echo = 2;
long dur, cm;
int p=A0;

void setup() {
Serial.begin (9600);

pinMode(trig, OUTPUT);

pinMode(echo, INPUT);
void loop() {

digitalWrite(trig, LOW);

delayMicroseconds(5);

digitalWrite(trig, HIGH);

delayMicroseconds(10);

digitalWrite(trig, LOW);

pinMode(echo, INPUT);
dur = pulseIn(echo, HIGH);

cm = (dur/2) / 29.1;

Serial.print(cm);
Serial.print("cm");
Serial.println();
delay(50);
if (cm>=0 & cm<5){

  tone(p, 261);
    delay(300);
  }
else if(cm>=5 & cm<10){

  tone(p, 293);
      delay(500);
  }
  else if(cm>=10 & cm<14){

  tone(p, 329);
      delay(300);
  }

  else if(cm>=15 & cm<20){

  tone(p, 349);

  }
  else if(cm>=20 & cm<25){

  tone(p, 440);
      delay(300);
  }
  else if(cm>=25 & cm<30){

  tone(p, 493);
      delay(300);
  }
   else if(cm==14 ){

   tone(p, -1);
       delay(1);
   }
  else {

   tone(p, -1);
       //delay(300);
   }
}
```

Рис.6 Программа Ультразвуковой терменвокс

8.1.Автоматизированные часы

Методические рекомендации.

Тема: «Автоматизированные часы»

Цель: изучить процесс создания и программирования устройства с часовым механизмом.

Задачи:

- изучить и закрепить на практике процесс создания устройств с часовым механизмом;
- изучить особенности управления роботом данной конструкции;
- изучить особенности понижающей и повышающей зубчатой передачи;
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Пример заданий

Задание 1 (Уровень А)

Соберите автоматизированные часы.

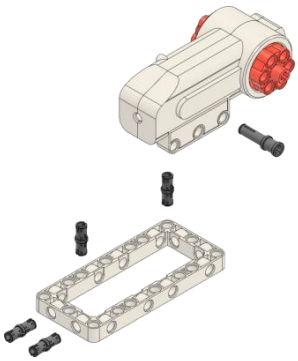


Рис.1 Часы

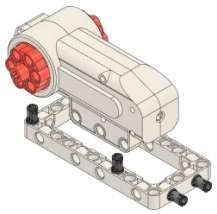


Рис.2 Часы

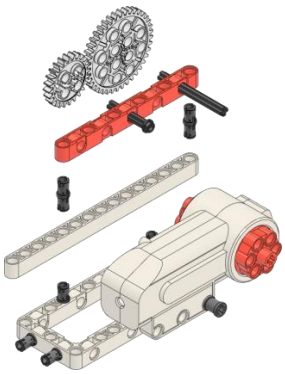


Рис.3 Часы

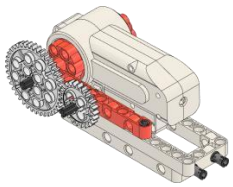


Рис.4 Часы

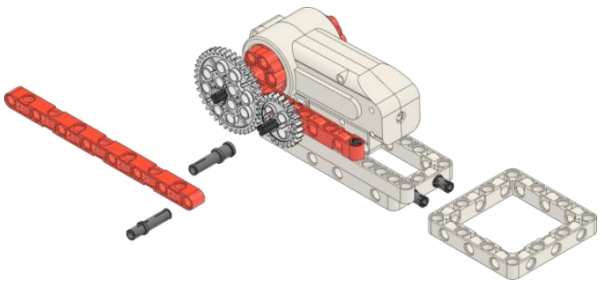


Рис.5 Часы

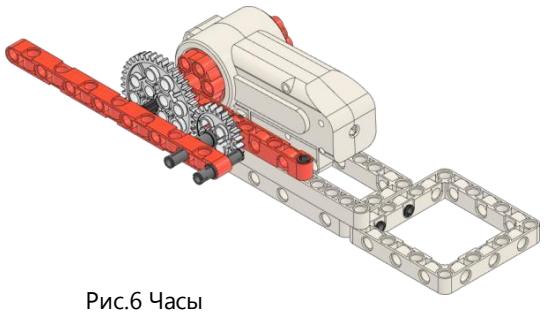


Рис.6 Часы

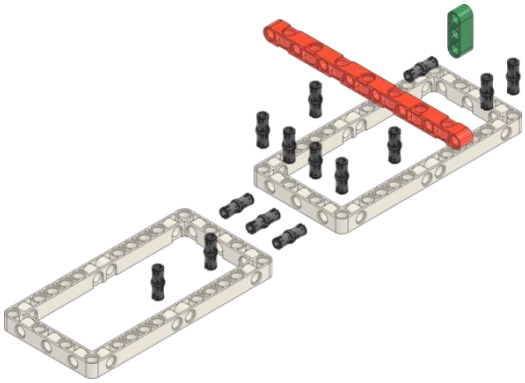


Рис.7 Часы

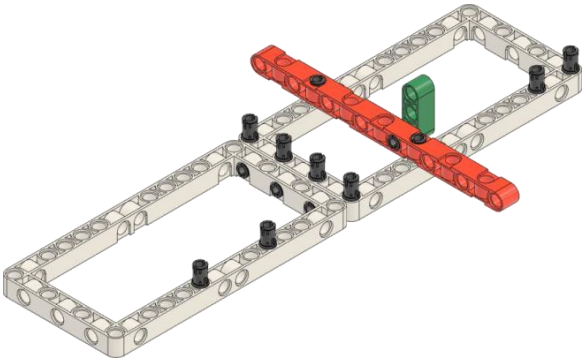


Рис.8 Часы

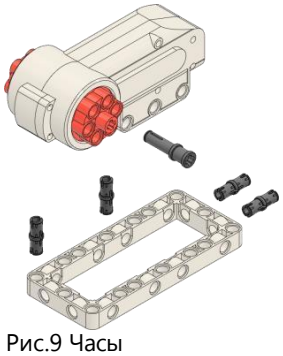


Рис.9 Часы

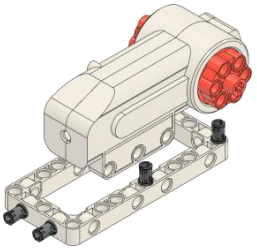


Рис.10 Часы

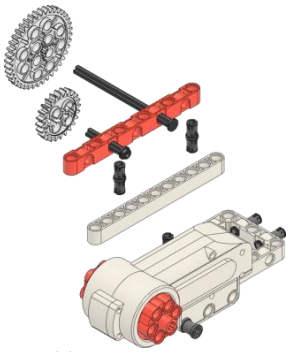


Рис.11 Часы

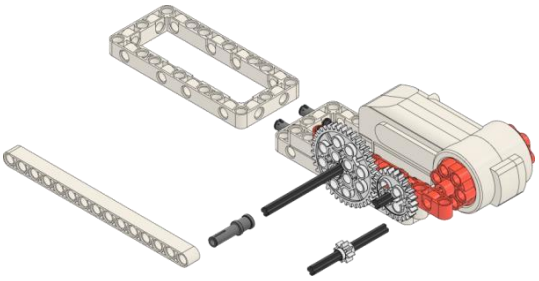


Рис.12 Часы

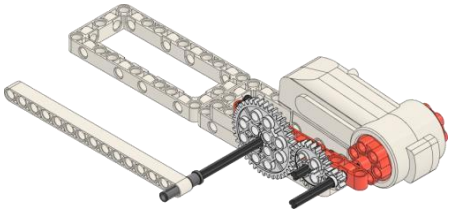


Рис.13 Часы

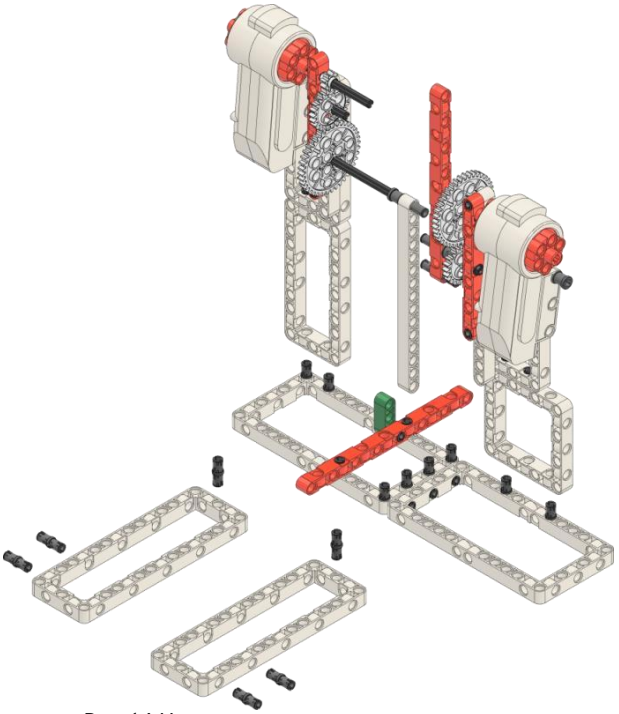


Рис.14 Часы

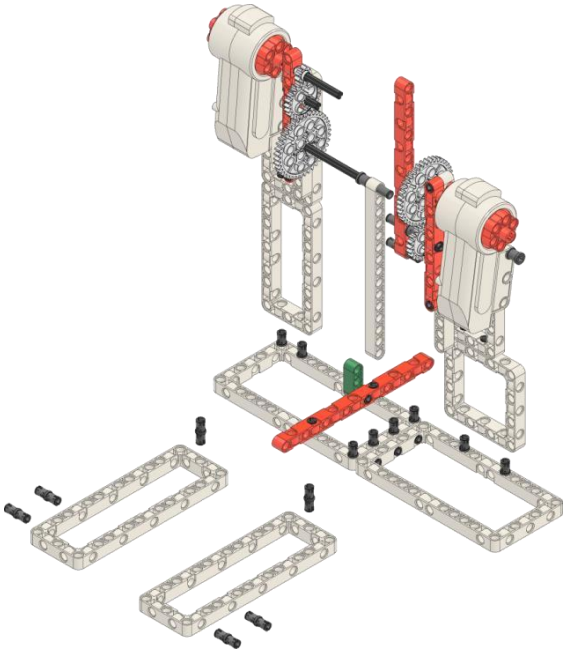


Рис.15 Часы

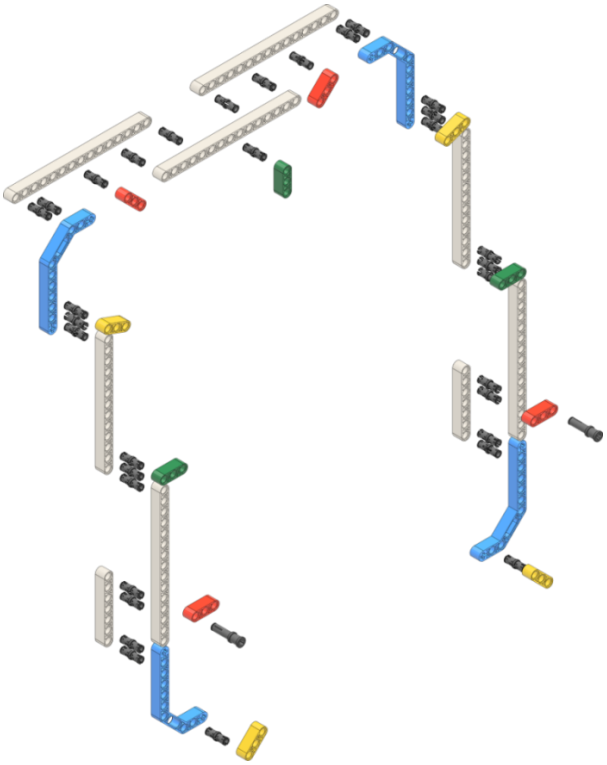


Рис.16 Часы

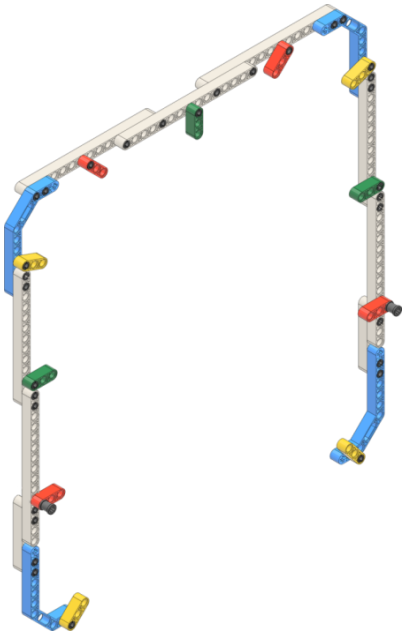


Рис.17 Часы

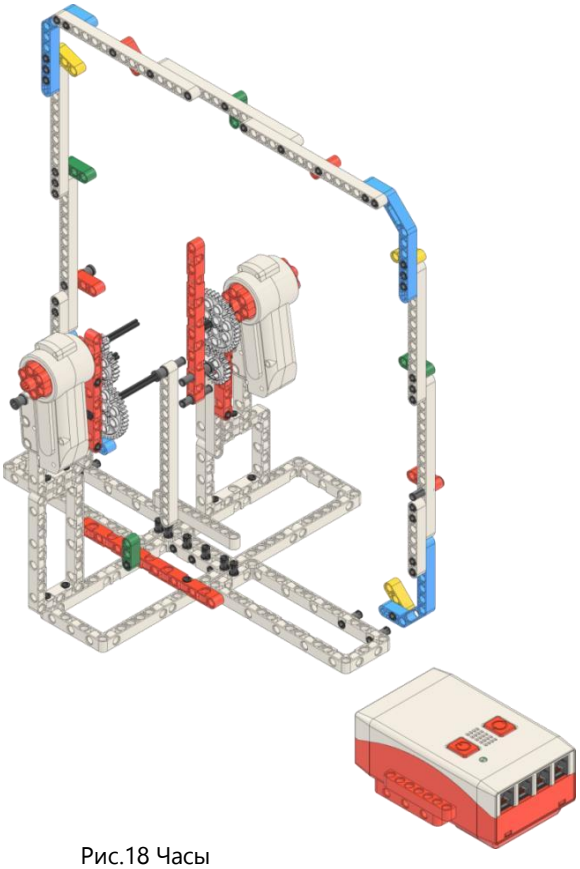


Рис.18 Часы

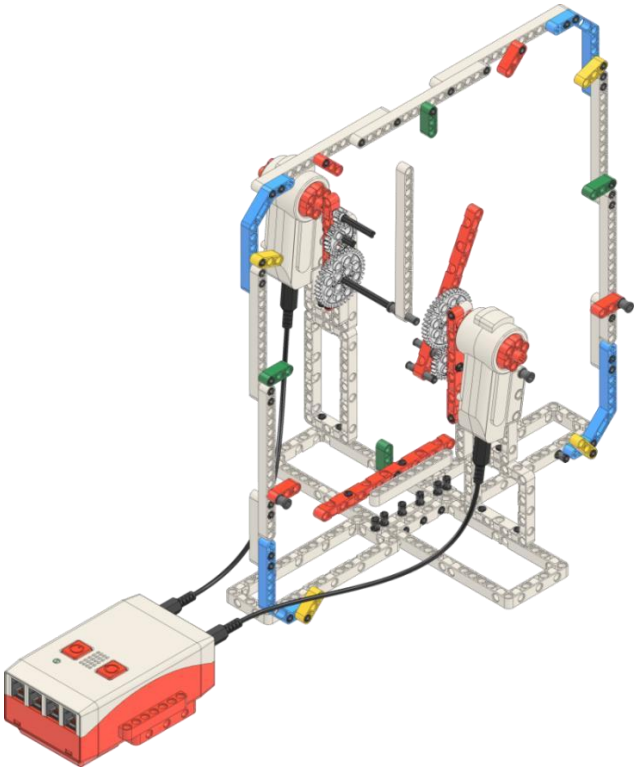


Рис.19 Часы

Задание 2 (Уровень В)

Создать программу, с помощью которой часы отсчитывали бы время сравнимое с реальными часами (секунды - минуты).

Для решения данной задачи нужно изучить передаточное отношение и рассчитать время движения и задержки моторов, а также их мощность.

time_clock §

```
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 = 5;

void setup()
{
  pinMode(m1,OUTPUT);
  pinMode(v1,OUTPUT);
  pinMode(m2,OUTPUT);
  pinMode(v2,OUTPUT);

  digitalWrite(m1, HIGH);
  digitalWrite(m2, LOW);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
}
```

Рис.20 Программа для работы часов (1 часть)

```
void loop() {

  analogWrite(v1, 100);
  analogWrite(v2, 100);
  delay(10);
  analogWrite(v1, 0);
  analogWrite(v2, 0);
  delay(500);

    }
```

Рис.21 Программа для работы часов (2 часть)

С помощью данной программы прибор отсчитывает время быстрее обычного, но это сделано для наглядности. Добейтесь регулированием параметров исходного отсчёта времени.

10. Физические эксперименты

Физика очень тесно связана с робототехникой, особенно механика. В данной главе рассмотрим в качестве примера возможность применения набора КЛИК на уроках физики для изучения определённых тем.

10.1. Равномерное прямолинейное движение

Ещё в древние времена люди дали два понятие состояниям тел. Тело может находиться в покое, а может двигаться.

Что же такое движение? Согласно общепринятому определению: движение – это изменение положения тела в пространстве, относительно неподвижного объекта, с течением времени (в более подробной форме - изменение координаты положения тела в пространстве относительно начала отсчёта с течением времени)

Если не сталкиваться с движением на практике и сопоставлять каждый элемент определения с явлением движения, то вряд ли возможно понять «сухую» теорию. Но мы с рождения сталкиваемся с данным явлением и можем различать простые формы движения и покоя исходя из жизненного опыта.

Так делали и люди жившие до н.э., но видов движений появлялось, с каждым годом, всё больше и больше. Поэтому решили сформулировать закон и подкрепить его математическими формулами, чтобы строго разграничить виды движений и дать точное определение, что такое движение.

Мы с вами познакомимся с самыми распространёнными видами, которые изучает физика, и которые применяются в механике и робототехнике. И первый вид – прямолинейное равномерное движение

Формула определяющее равномерное прямолинейное движение

$$S = \vartheta * t$$

Данная формула идеальна, если на всём участке пути скорость не менялась, т.е.

$$\vartheta = const$$

На практике скорость меняется, но если она сохраняется на определённых участках пути, то можно вычислить среднее её значение (среднюю скорость)

$$\vartheta_{ср} = S/t$$

S – весь путь, пройденный телом.

t – время, за которое тело прошло весь путь.

Методические рекомендации.

Тема «равномерное прямолинейное движение»

Цель: изучить физический смысл равномерного прямолинейного движения

Задачи:

- изучить и закрепить на практике процесс создания мобильной платформы
- изучить особенности равномерного движения
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Задание 1 (Уровень А)

Соберите мобильного робота согласно главе 5.5.

Задание 2 (Уровень В)

Создайте программу, при которой робот будет двигаться равномерно и прямолинейно, попутно вычисляя пройденный путь с течением времени.

Пример решения.

Из раздела одометрии робота (глава 4.1), можно почерпнуть, какой путь проходит робот за один оборот и сколько оборотов при данной мощности он совершает за одну секунду.

При мощности в 127 колесо совершает 1,83 оборота в секунду, а при мощности в 255 – 3,65 оборота в секунду.

Один оборот равен 155,43 мм

Составим программу на основе изложенных умозаключений. В программе робот будет двигаться пять секунд, попутно вычисляя каждую секунду пройденный путь.

```
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 =  5;

int t = 0;
int s = 0;
float l = 155.43;
float n = 0;


void setup() {
  Serial.begin(9600);
  pinMode(m1, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(v1, OUTPUT);
  pinMode(v2, OUTPUT);

  analogWrite(6, 0);
  analogWrite(5, 0);
```

Рис.1 Первая часть программы

Здесь мы укажем пины подключения моторов и введём переменные отвечающие за путь , длину колеса и количества оборотов. Чтобы видеть результат работы воспользуемся функцией **Serial.begin(9600).** Если подключиться по Bluetooth, то можно увидеть результат в реальном времени.

```
for (int i=0; i<5; i++)
{
  digitalWrite(m1,LOW);
digitalWrite(m2,HIGH);
analogWrite(6, 127);
 analogWrite(5, 127);
 delay(1000);
 n=n+1.83;
 s=l*n;
 Serial.print("s = ");
 Serial.print(s);
 Serial.print(" mm ");
 Serial.println();
  }
  analogWrite(6, 0);
analogWrite(5, 0);


}

void loop() {


}
```

Рис.2 Заключительная часть программы

Здесь запускается конечный цикл на пять шагов. Каждую секунду вычисляется путь и выводится на COM порт. Скорость моторов взяли равной 127.

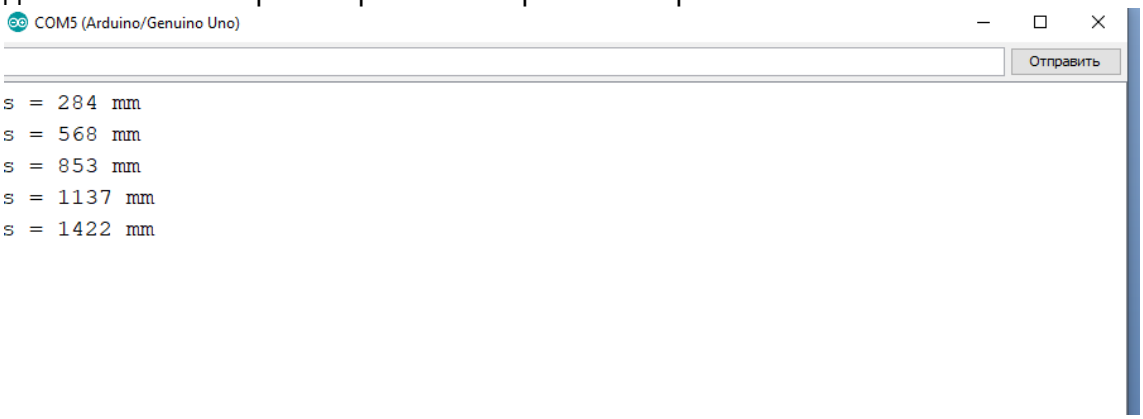


Рис.3 Результат работы программы

Рис.4 Код программы в MBlock5

Постройте график с осями скорости и времени.

10.1. Равноускоренное прямолинейное движение

В этой книге уже упоминалось, что есть множество видов движения и равномерное прямолинейное движение – не единственное движение. В этом параграфе мы рассмотрим прямолинейное равноускоренное движение.

Мы уже знаем, что движение характеризуется несколькими физическими величинами: время, скорость, перемещение. Но оказывается, прямолинейное равноускоренное движение характеризуется ещё одной величиной – ускорение тела.

$$\vec{a} = \frac{\vec{V_1}^2 - \vec{V_0}^2}{t}$$

Ускорение - векторная физическая величина, численно равная изменению скорости за единицу времени:

a = [м/с^2] - единица измерения.

Математические выражения равноускоренного прямолинейного движения:

$$s = s_0 + \vartheta_0 \cdot t + \frac{a \cdot t^2}{2}$$

$$2 \cdot a \cdot s = \vartheta_1^2 - \vartheta_0^2$$

$$a = \frac{\vartheta_1 - \vartheta_2}{t}$$

s – перемещение (м), ϑ - скорость (м/с), a – ускорение(м/с), t – время (с)

Методические рекомендации.

Тема: «равноускоренное прямолинейное движение»

Цель: изучить физический смысл равноускоренного прямолинейного движения

Задачи:

- изучить и закрепить на практике процесс создания мобильной платформы
- изучить особенности равноускоренное движение
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Задание 1(Уровень А)

Соберите мобильного робота согласно главе 5.5.

Задание 2 (Уровень В)

Создайте программу, при которой робот будет двигаться равномерно и прямолинейно, попутно вычисляя пройденный путь с течением времени.

Пример решения.

Из раздела одометрии робота (глава 4.1), можно почерпнуть, какой путь проходит робот за один оборот и сколько оборотов при данной мощности он совершает за одну секунду.

При мощности в 127 колесо совершает 1,83 оборота в секунду, а при мощности в 255 – 3,65 оборота в секунду.

Один оборот равен 155,43 мм

Составим программу на основе изложенных умозаключений. В программе робот будет двигаться десять секунд, попутно вычисляя каждую секунду пройденный путь за всё время. Параллельно скорость моторов будет возрастать на 10 от 126 до 255. Значение на которое увеличивается количество оборотов с каждой секундой не трудно рассчитать. Разница между начальным количеством оборотов и конечным при максимальной скорости равна

$$\Delta n = 3.65 - 1.83 = 1.82$$

А если принять во внимание, что скорость изменится за десять секунд со 126 до 255, то следовательно, каждую секунду скорость изменяется примерно на 13.

Поделим разницу оборотов на шаг изменения скоростей $h = \frac{\Delta n}{\Delta \vartheta} = \frac{1.82}{12.7} = 0.144$

Шаг скоростей мы выразили точнее.

Воспользуемся нашими вычислениями в программе.

```
int m1 = 7;
int m2 = 4;
int v1 = 6;
int v2 =  5;

float h = 0.144;
int s = 0;
int s1 =0;
float l = 155.43;
float n = 1.83;


void setup() {
  Serial.begin(9600);
  pinMode(m1, OUTPUT);
  pinMode(m2, OUTPUT);
  pinMode(v1, OUTPUT);
  pinMode(v2, OUTPUT);

  analogWrite(6, 0);
  analogWrite(5, 0);
```

Рис.5 Первая часть программы

```
for (int i=126; i<255; i+=13)
{
  digitalWrite(m1,LOW);
digitalWrite(m2,HIGH);
analogWrite(v1, i);
 analogWrite(v2, i);
 delay(1000);
  s=l*n;
 s1=s1+s;
 n=n+h;
 Serial.print("s1 = ");
 Serial.print(s1);
 Serial.print(" mm ");
 Serial.println();
  }
  analogWrite(6, 0);
  analogWrite(5, 0);


}

void loop() {


}
```

Рис.6 Заключительная часть программы

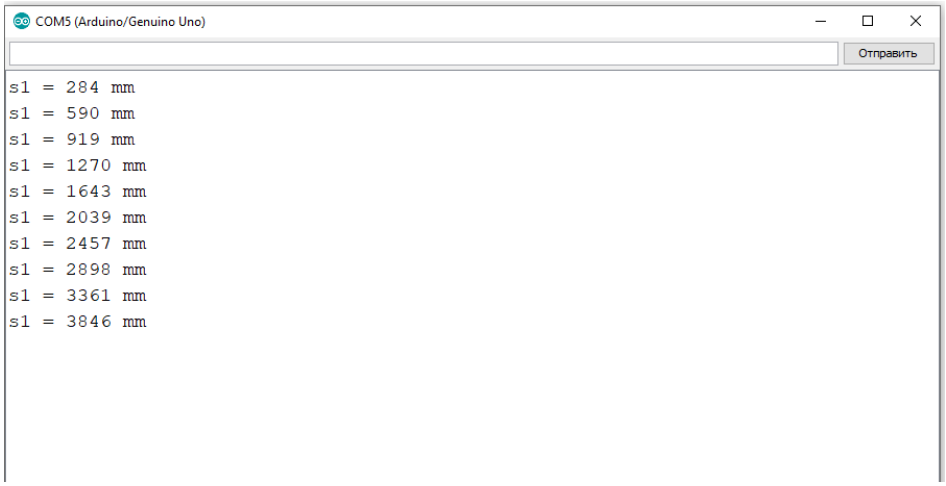


Рис.7 Результат работы программы

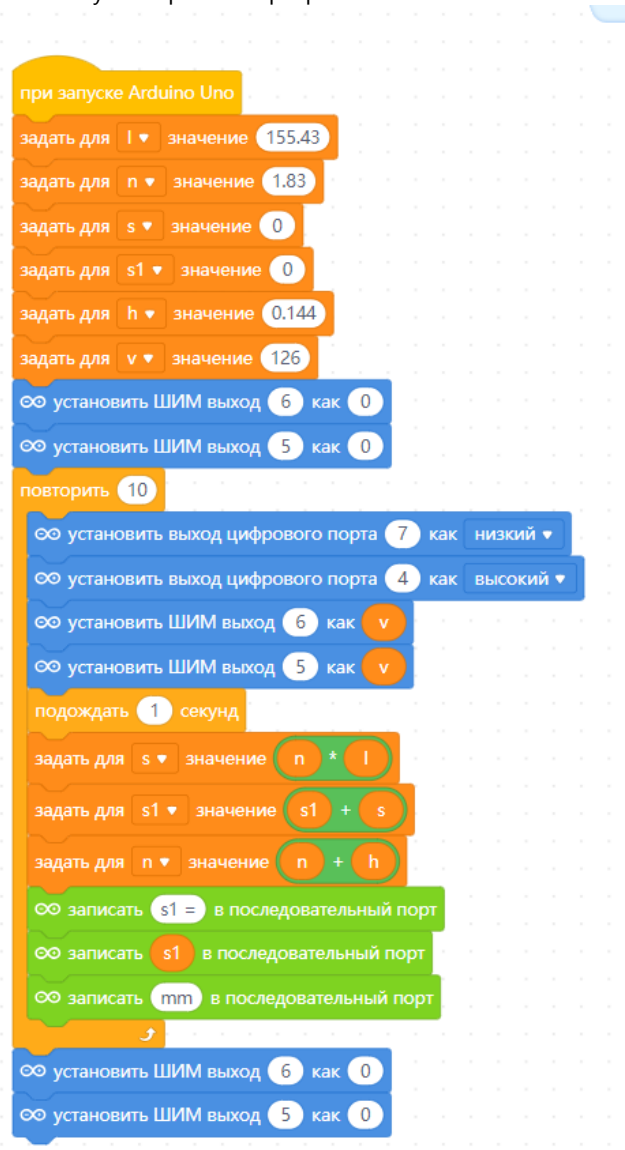


Рис.8 Код программы в MBlock5

Как видно для испытаний вам понадобиться пространство в 4 м. Проверьте работу робота, постройте график зависимости скорости от времени.

Во время испытания может возникнуть такой случай, что ваш робот идёт не по прямолинейной, а по криволинейной траектории. Это связано с тем, что один мотор имеет

чуть более малую скорость вращения, чем второй. Исправляется это путём нахождения значения для быстрого мотора и вычитание из него данного значения.

Пример представлен на рис. 4. Причина работы так двигателей может быть различна – механическое повреждение, неровное крепление колёс (плохая сцепка с поверхностью) , запаздывание по времени передача мощности на моторы, различное потребление моторами тока (разница в десятой части Вольта).

```
for (int i=126; i<255; i+=13)
{
  digitalWrite(m1,LOW);
digitalWrite(m2,HIGH);
analogWrite(v1, i);
 analogWrite(v2, i-6);
 delay(1000);
  s=l*n;
 s1=s1+s;
 n=n+h;
 Serial.print("s1 = ");
 Serial.print(s1);
 Serial.print(" mm ");
 Serial.println();
  }
  analogWrite(6, 0);
  analogWrite(5, 0);


}

void loop() {


}
```

Рис.9 Корректировка работы программы

10.2. Колебания

Колебания – повторяющейся во времени действия.

Виды колебаний.

1. Свободные.
2. Гармонические.
3. Вынужденные.
4. Затухающие.

Примеры: маятники, струна гитары, игла швейной машинки, поршень в цилиндре двигателя и т.д.

Данный вид движения характеризуется такими физическими величинами как:

R - радиус окружности (м)

$\Delta\varphi$ - угол поворота (рад)

ω - угловая скорость ($\frac{рад}{с}$)

υ - скорость вращения ($\frac{м}{с}$)

T - период вращения (с)

ν - частота вращения ($\frac{1}{с}$ или $Гц$)

a_n - нормальное ускорение (центростремительное ускорение) ($\frac{м}{с^2}$)

a_τ - тангенциальное ускорение (касательное ускорение) ($\frac{м}{с^2}$)

Единица измерения рад – радиан, видоизменённая градусная мера угла.

Пример:

$30^\circ = \frac{\pi}{6}\ рад$

$\nu = \frac{1}{T}$, $\omega = 2\pi\nu$,

$T = \frac{t}{N}$, где t – время колебания, N – количество колебаний.

Методические рекомендации.

Тема: «Колебательные движения»

Цель: изучить физический смысл колебаний

Задачи:

- изучить и закрепить на практике процесс создания экспериментальной установке для изучений колебаний
- изучить особенности колебаний
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Задание 1 (уровень А)

Соберите конструкцию согласно инструкции.

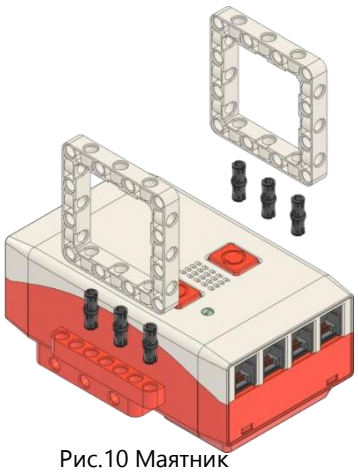


Рис.10 Маятник

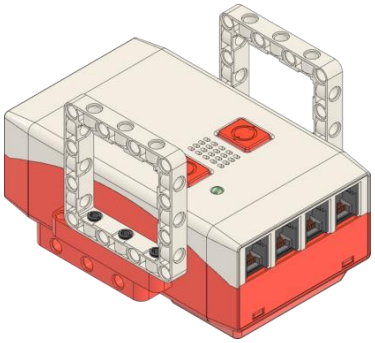


Рис.11 Маятник

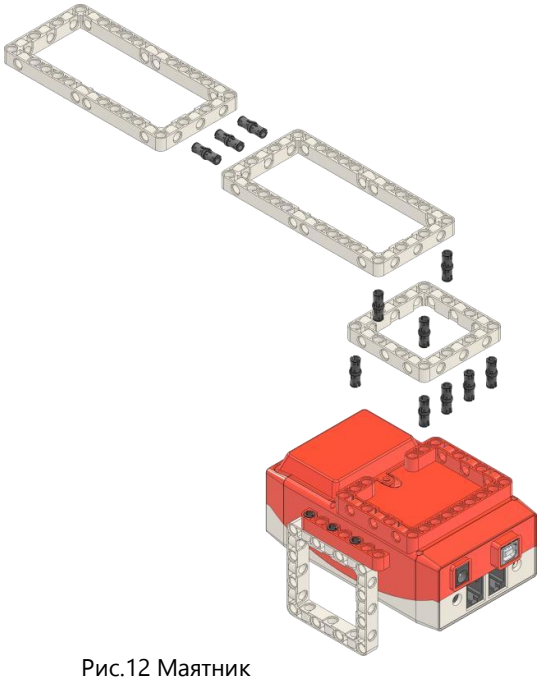


Рис.12 Маятник

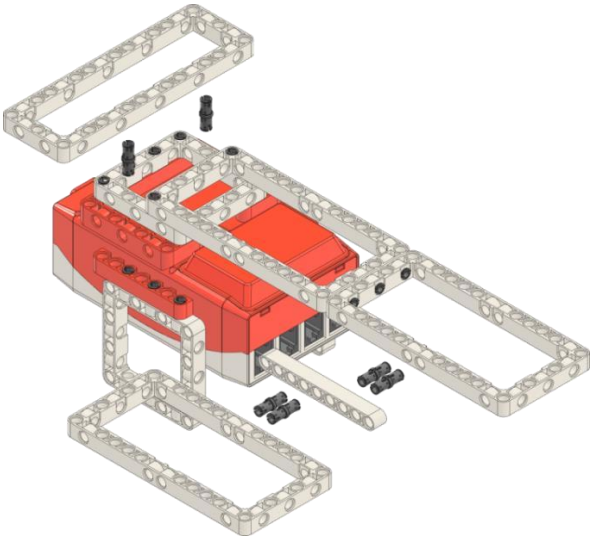


Рис.13 Маятник

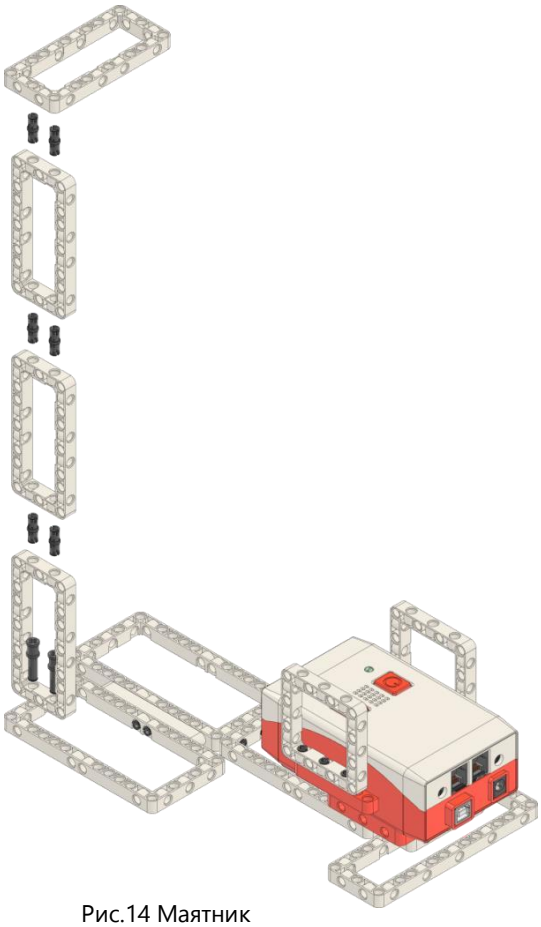


Рис.14 Маятник

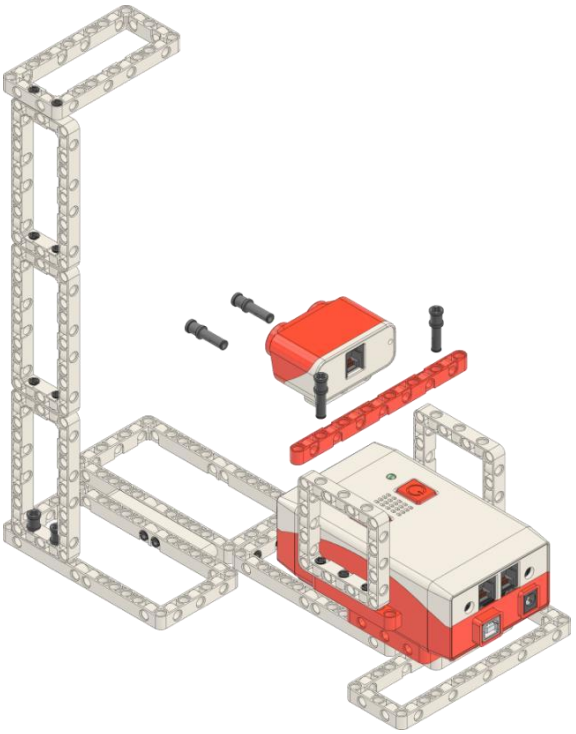


Рис.15 Маятник

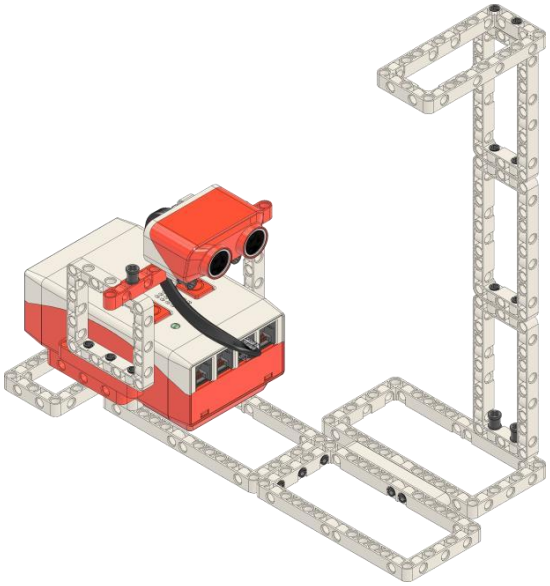


Рис.16 Маятник

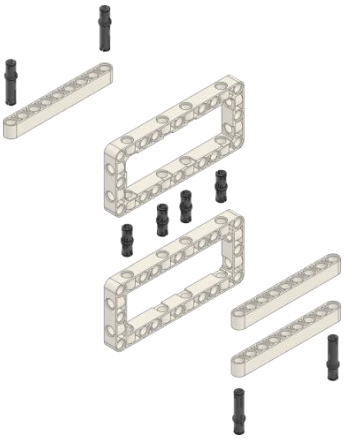


Рис.17 Маятник

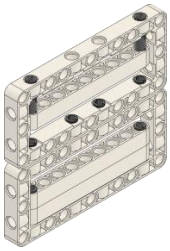


Рис.18 Маятник

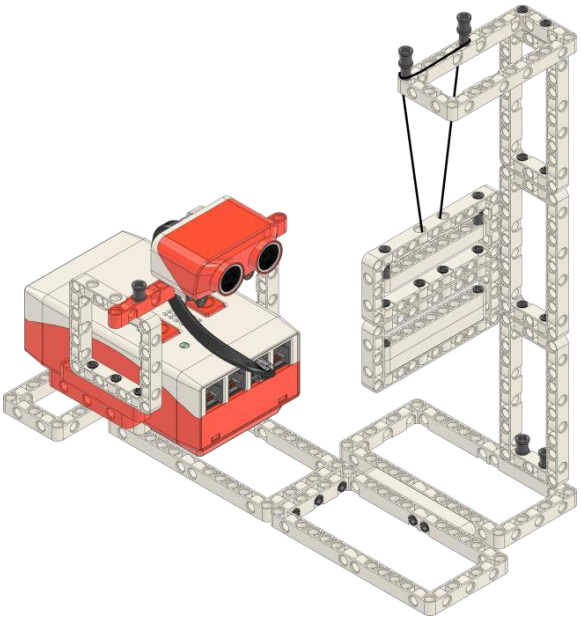


Рис.19 Маятник

Задание 2 (уровень В)

1. Создайте программы для ультразвукового датчика, чтобы он определял расстояние до маятника. Выведите значения через монитор порта;
2. Определите амплитуду колебания в первый промежуток времени;
3. Дождитесь примерно равномерного колебания и за определённый промежуток времени посчитайте количество колебаний;
4. На основании этого вычислите период и частоту колебания, сравните с теоретическими расчётами вычисления частоты и периода колебания нитяного маятника.

11. Контроллер Makeblock CyberPi

11.1. Программируемый контроллер Makeblock CyberPi.

CyberPi - это мощный микроконтроллер, разработанный Makeblock Education для компьютерных наук иобучения STEAM.

Благодаря сетевым возможностям преподаватели могут проводить очень интерактивные и интеллектуальные уроки. Это может быть, например, подключение нескольких CyberPi через беспроводную локальную сеть (Wi-Fi-LAN) в классе, проведение интерактивной викторины или мониторинг среды в классе и обмен информацией.
CyberPi также может напрямую подключаться к Интернету для распознавания речи, сбора данных обокружающей среде или передачи данных через Google Spreadsheets.

В качестве среды программирования используется mBlock5. Он представляет собой платформу программирования для различных устройств, включая CyberPi, разработанную для обеспечения расширенного образовательного опыта и непрерывного роста для учащихся. Благодаря расширениям в mBlock5 преподавателимогут легко интегрировать передовые технологии, такие как Data Science, IoT и AI, в свои уроки. Благодаря интеграции блочного кодирования и Python, mBlock5 предлагает учащимся путь обучения для развития от базовых до профессиональных навыков программирования.

CyberPi имеет несколько встроенных датчиков и исполнительных механизмов, которые можно программировать. Учащиеся могут визуализировать текст, данные и изображения на цветном дисплее, записывать звуковые данные или воспроизводить аудиофайлы в зависимости от входа датчика. Это позволяет создавать увлекательные учебные проекты, актуальные для учащихся в реальном мире, например, звуковую машину, объединяющую светодиоды и динамик в CyberPi, или создание собственного игрового контроллера благодаря встроенным в CyberPi кнопкам, джойстику и гироскопу. Это некоторые из проектов, которые являются частью этих уроков.

Сфера деятельности может быть дополнительно расширена за счет объединения CyberPi с платами расширения и множеством различных компонентов. Например, Pocket Shield имеет встроенную батарею, которая делает CyberPiпортативным и автономным для выполнения действий при отключении от компьютера. Pocket Shield также помогает подключаться к другим внешним компонентам, таким как сервоприводы или двигатели постоянного тока, чтобы обеспечить движение в проекте. Он также может подключаться к светодиодным лентам и широкому спектру датчиков и исполнительных механизмов сторонних производителей, которые обычно используются для проектов DIY.

1. **Внешний вид контроллера**

Контроллер не продаётся отдельно, а является частью наборов CyberPi Go Kit, CyberPi Innovation Add-On-pack или mBot2. Питание контроллера обеспечивается через аккумулятор, установленный в специальном шилде или через разъём USB Type-C.

Внешний вид и функционал Pocket Shild:

Вот так выглядит контроллер вместе с шилдом в наборе CyberPi Go Kit:

11.2. Описание функционала.

Контроллер предназначен для использования в образовательных робототехнических Проектах, а также для изучения Интернета вещей (IoT) и Искуственного интеллекта (Ai).

Чип ESP32, на базе которого сделана эта плата, содержит в себе достаточно производительный ARM-процессор, а также модули беспроводной связи Wi-Fi и Bluetooth.

Даже не подключая внешних датчиков, уже можно начать работу с этим контроллером, потому что «на борту» у него уже есть как сенсоры, так и исполнительные устройства.

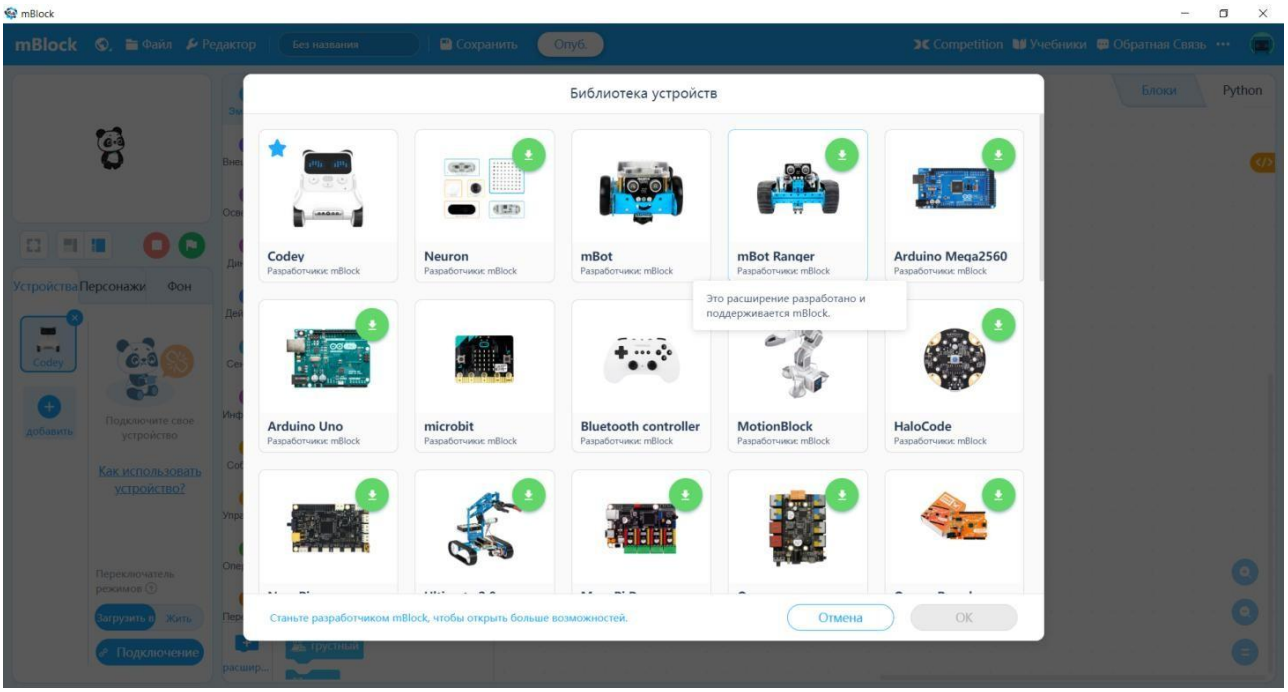
Гироскоп с акселерометром поможет определить положение в пространстве, микрофон – среагировать на звук, программируемая кнопка – среагировать на ее нажатие, а микрофон и динамик записать или воспроизвести звук.

Из исполнительных устройств у нас есть модуль из 5-ти полноцветных RGB-светодиодов, динамик и полноцветный экран.

Теперь пришло время рассмотреть, что именно мы можем делать с таким набором сенсоров, но сначала поговорим о среде программирования для этого контроллера.

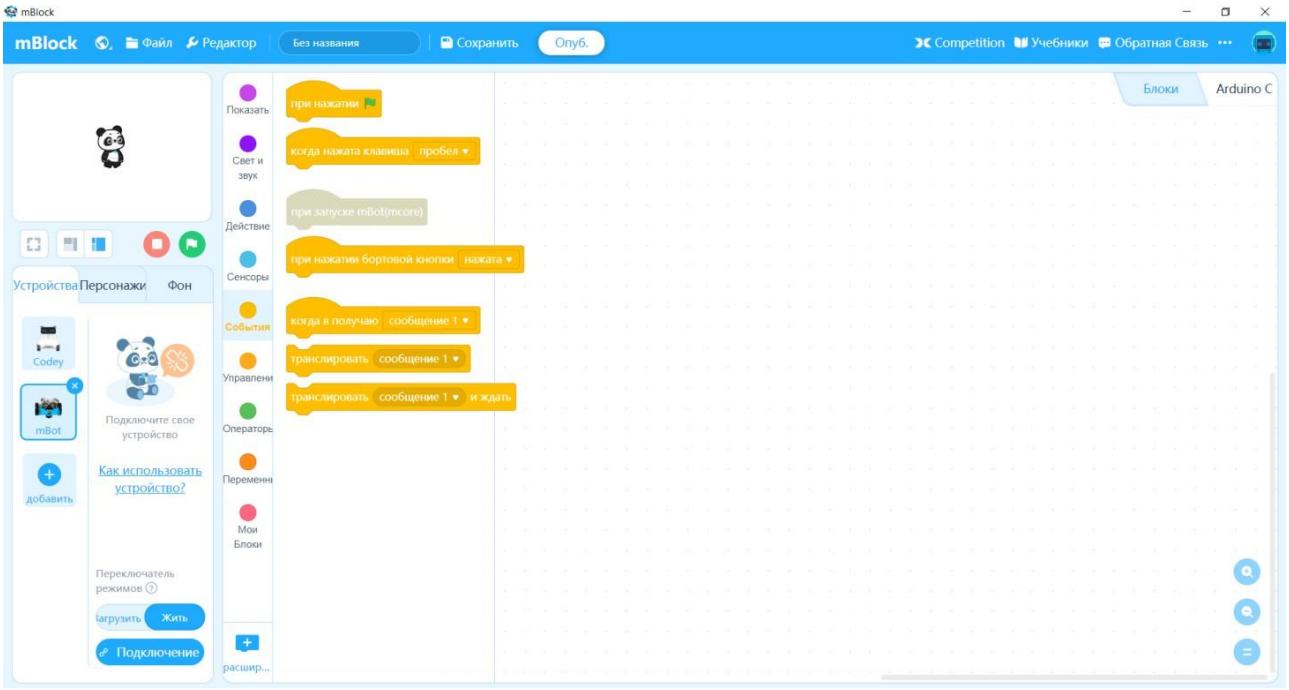
Подключение вашего устройства

Что бы подключить вашего робота (или просто контроллер) к ПО Mblock, слева в соответствующей зоне нажмите на «+ добавить» и увидите список поддерживаемых устройств с их изображениями, как на скришшоте:



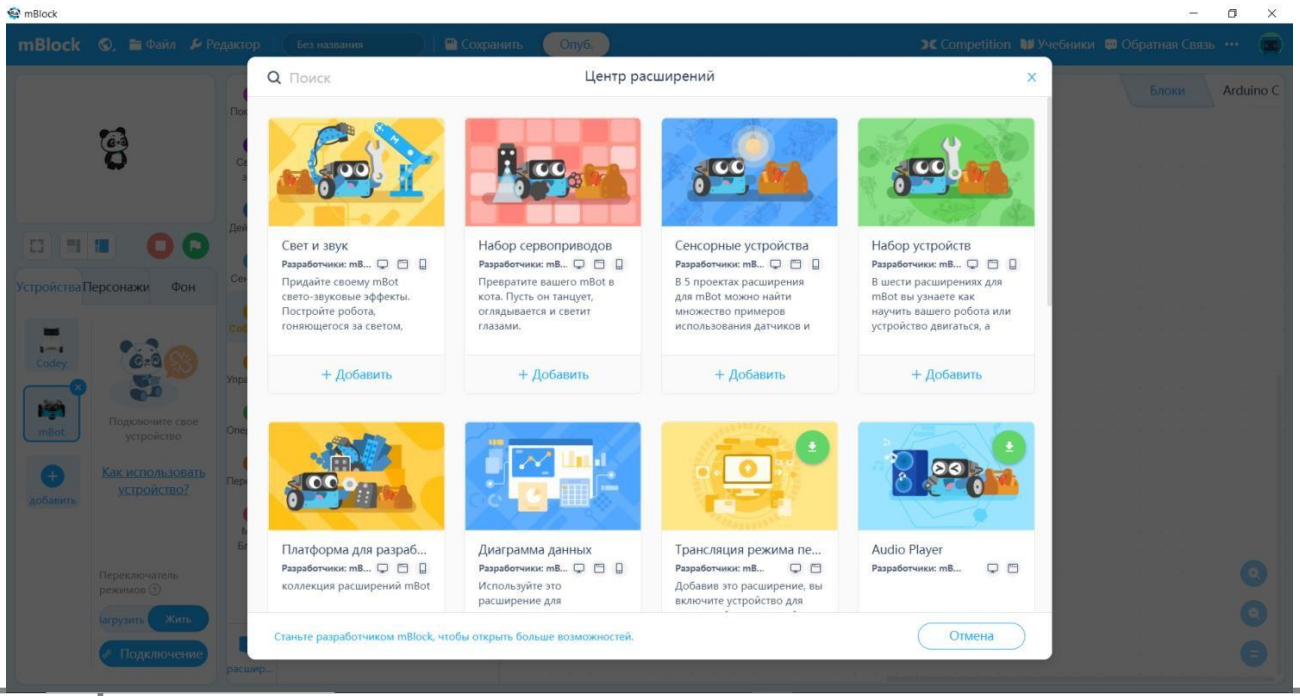
Если в вашей версии программы есть какие-либо обновления для конкретного устройства, вы увидите на его иконке зеленый кружок со стрелкой. Нажмите на него для получения
обновлений.

Нажмите на значок нужного вам устройства, чтобы добавить его в ваш проект. Выбранное устройство появится в окне программы слева. Нажав на значёк вашего устройства и затем на ОК справа снизу, вы подключите все возможности ПО Mblock, которые доступны именно для него.



Например, выбрав робота mBot, мы получим доступ ко всем возможностям программирования контроллера mCore, входящего в данный набор.

В окне типов программных блоков внизу можно увидеть значёк «+». Нажав на него мы можем добавить дополнительные функции программирования для нашего устройства через открывшийся «Центр расширений».



Например, здесь можно подключить программные блоки для RGB массива линии, датчика цвета, камеры и другие.

Что бы устройство было подключено к ПО Mblock, необходимо подсоединить его кабелем к USB порту вашего компьютера или использовать для подключения специальный USB-Bluetooth Dongle от Makeblock.

После этого нажимаем кнопку «Подключение» в нижней части экрана и выбираем порт к которому подключено устройство. Изображение панды в окне подключения покажет, что произошло соединение. Наш робот готов для заливки программы и управления.

Есть 2 режима программирования. За них отвечает кнопка-переключатель

«Загрузить/жить» Если мы хотим создать автономного робота, то переключатель должен

быть в положении
«Загрузить». Если предполагается выполнение программы по нажатию на флажок, то выбираем режим «Жить».

11.3. Использование примеров программ.

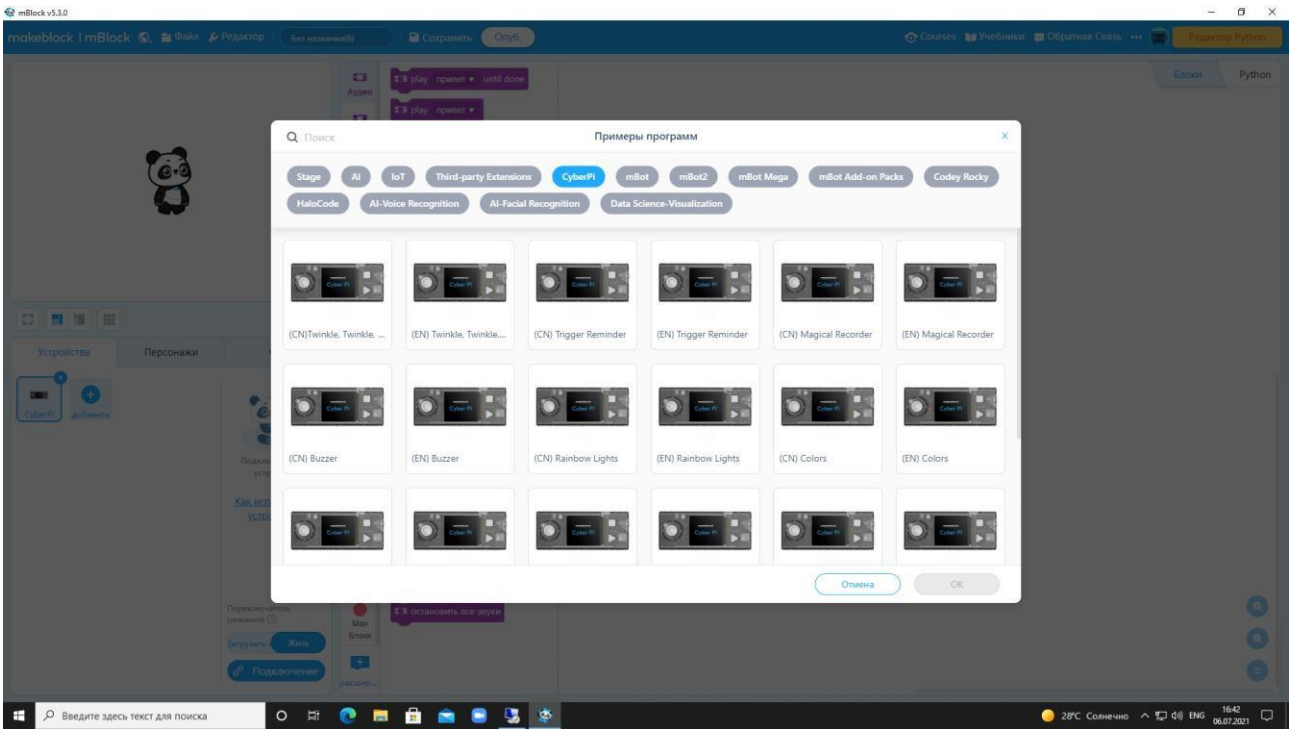
Для подготовки к работе выполните следующие действия:

1. В окне подключенных устройств добавьте нашу плату HaloCode
2. Подсоедините контроллер к компьютеру с помощью кабеля USB (USB – Micro USB) или при помощи Bluetooth Dongle. Для соединения с Bluetooth Dongle, воткните его в свободный
   порт USB на вашем компьютере и нажмите на нём кнопку с изображением значка Bluetooth.
   Светодиод на Bluetooth Dongle начнёт быстро мигать. Подайте питание на контроллер (можно с USB-зарядки от телефона или с Power Bank, если у вас нет «родного»
   аккумулятора) и поднесите контроллер к Bluetooth Dongle. Подключение будет установлено автоматически и светодиоды и на Bluetooth Dongle и на контроллере перестанут мигать. При следующем подключении этого же контроллера к Bluetooth Dongle нажимать на кнопку уже не надо (если вы не нажимали ее больше и не

сбросили подключение) – поключение будет установлено автоматически.

3. Нажмите кнопку «Подключить» в нижней части окна ПО Mblock и выберите COM-порт, соответствующий вашему устройству. При этом, если вы используете Bluetooth Dongle, то не нужно выбирать Bluetooth, так как устройство эмулирует полноценное подключение по COM-порту.
4. Наше устройство подключено и готово для работы.

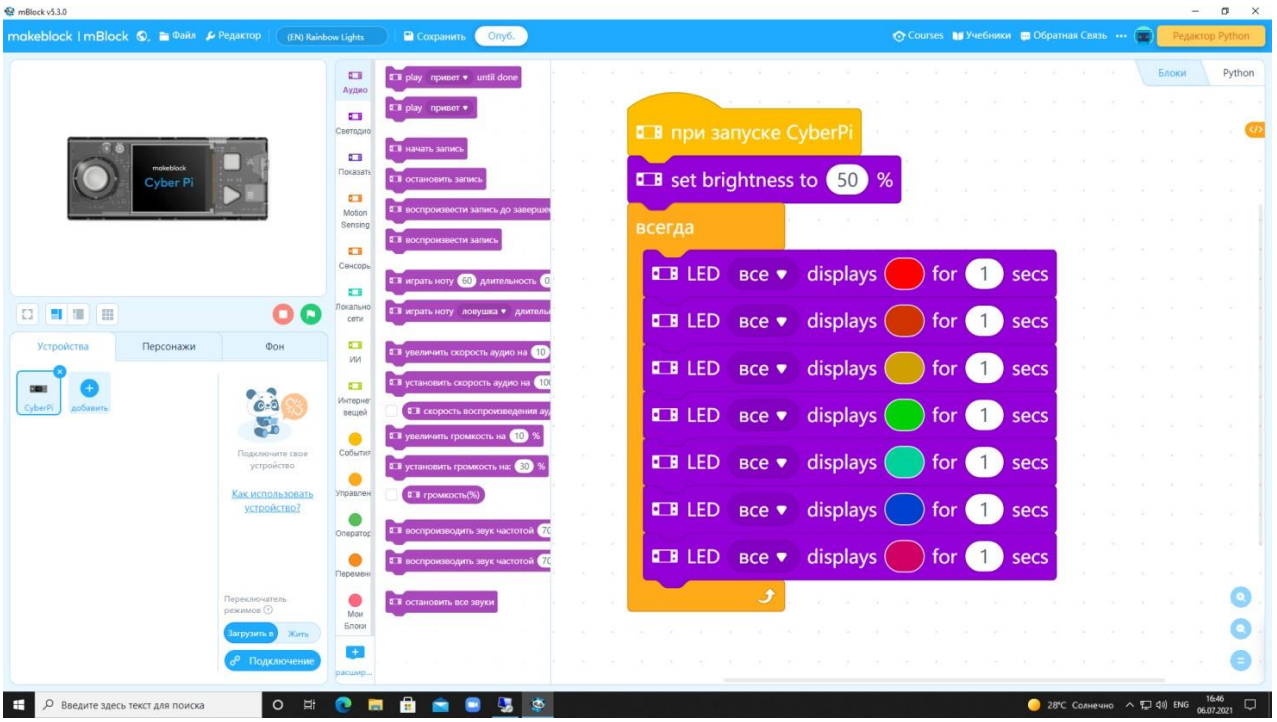
Как было описано выше, примеры программ находятся в меню «Учебники» в верхнем меню справа. Нажимаем на это меню и заходим в примеры программ и выбираем в открывшемся окне CyberPi



На экране мы увидим весь список программ-примеров, которые относятся именно к CyberPi.

Не все из них мы можем использовать с «голым» контроллером, но есть примеры, которые вполне работают. Выберите самый первый – «Rainbow Button».

После выбора примера, произойдёт загрузка соответствующей программы:



Как мы видим, программа очень простая и состоит из События «При запуске CyberPi», установки яркости свечения на 50% и результата действия в виде горящих определённым цветом светодиодов. Светодиоды в бесконечном цикле меняют цвет каждую секунду.

Нажмите мышкой на цвет в блоке программы. Раскрасьте светодиоды так, как вам нравится.

Подключите контроллер CyberPi компьютеру в порт USB

Нажмите «Подключить» в Mblock, убедитесь, что устройство

подключено. Замените шапку программы на «При нажатии на

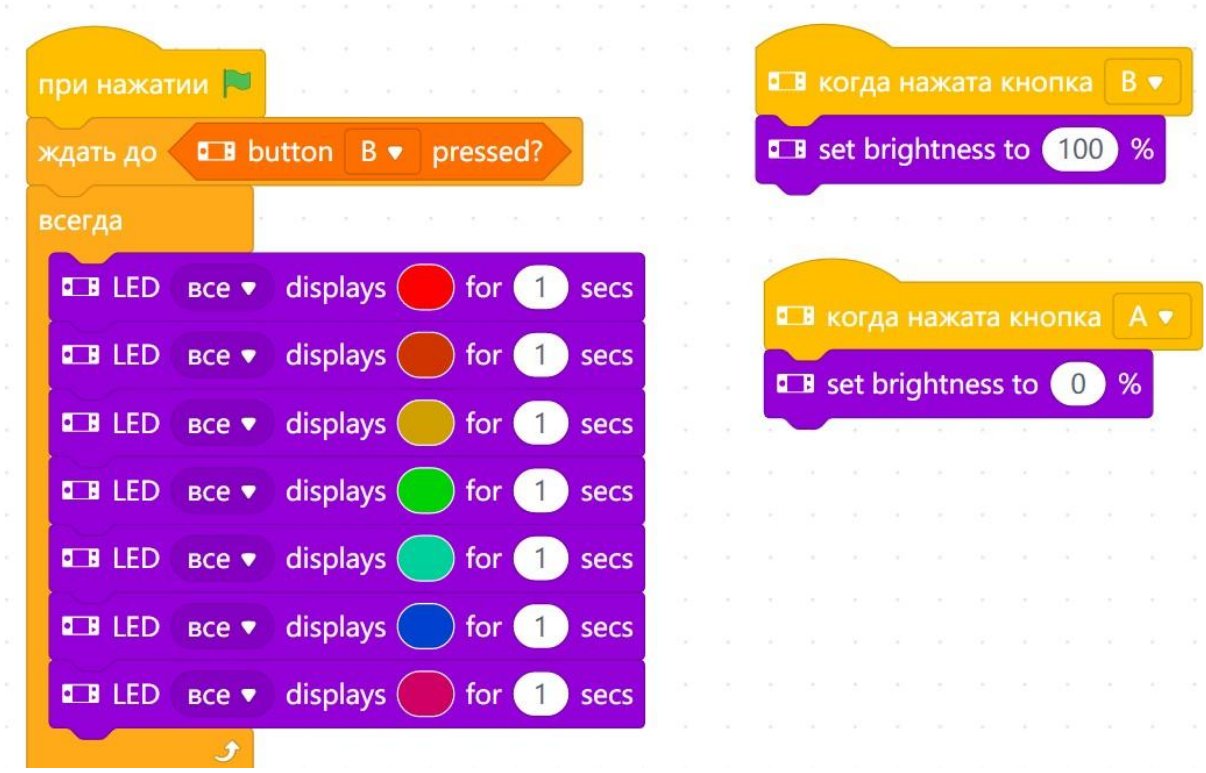
флажок»

Проверьте, что переключатель режимов загрузки находится в положении «Жить» и нажмите на зеленый флажок над областью подключения.
Светодиоды загорятся тем цветом и в том порядке, которые вы задали выше в программе.

Обратите внимание, что светодиоды зажглись и не гаснут, потому, что мы не предусмотрели это в программе.

Давайте немного модифицируем нашу программу. Пусть при нажатии на прямоугольную кнопку (это кнопка A при составлении программ), светодиоды гаснут, а при нажатии на кнопку B (это треугольная кнопка-стрелка) работа светодиодов возобновляется.

Составьте программу, как показано на рисунке:



Запустите программу с помощью флажка и нажмите на кнопку B на контроллере. Светодиоды будут мигать до тех пор, пока мы не нажмём на кнопку A. При повторном нажатии на кнопку B, мигание светодиодов возобновится.

Изменяйте программу, чтобы получить другие световые эффекты.

Попробуйте зажигать не все светодиоды, а только определённые, добейтесь эффекта меняющего цвет «бегущего огня».

Попробуйте работу других примеров. Посмотрите, как выглядит программа для тех или иных действий.

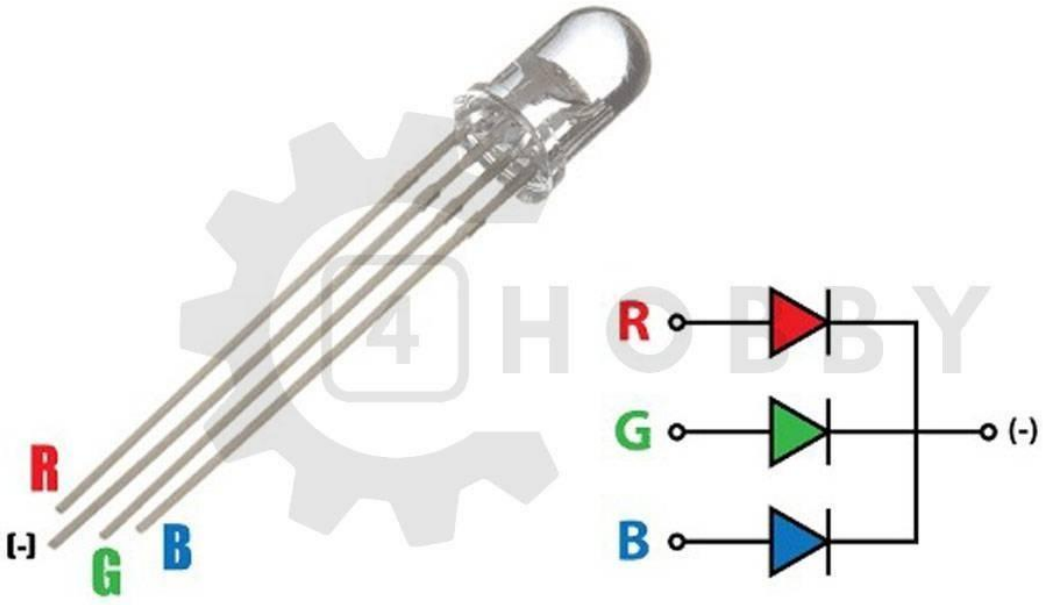
11.4. Работа с встроенными датчиками контроллера CyberPi

На плате контроллера, как уже говорилось выше, распаяны следующие устройства:

- Микрофон
- Датчик освещенности
- Гироскоп с акселерометром
- Модуль из 5-ти RGB-светодиодов
- Программируемые кнопки
- Джойстик
- Динамик

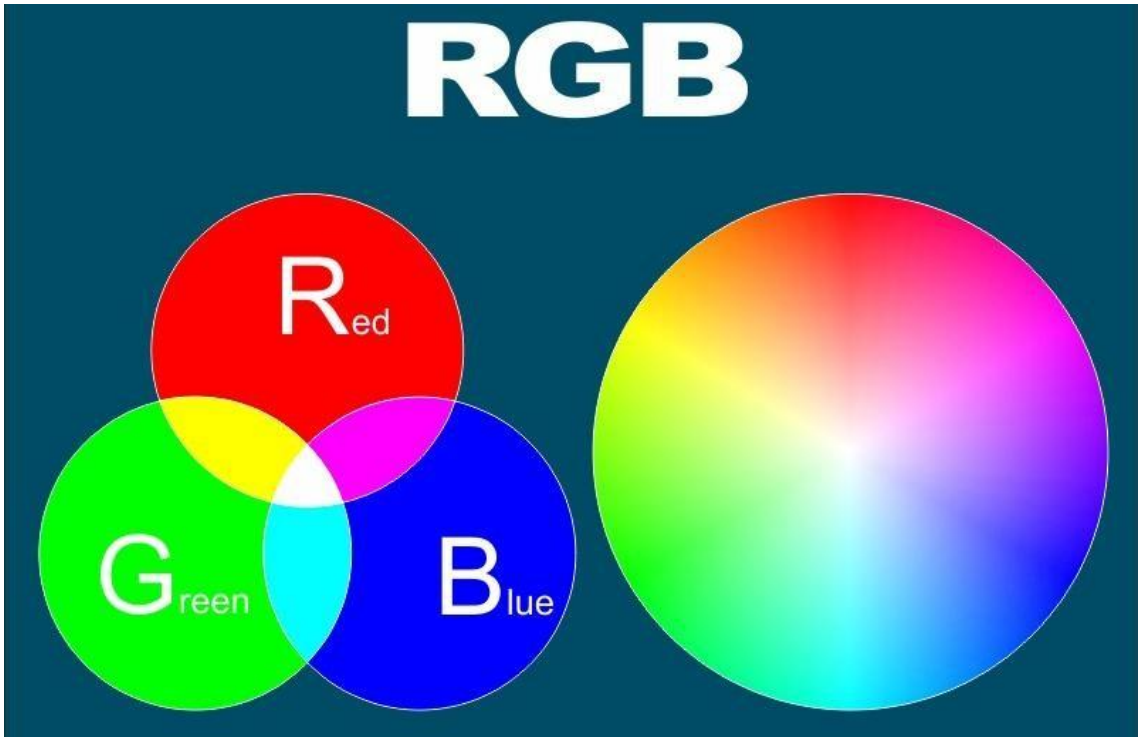
RGB-светодиоды

Что такое RGB-светодиод? По сути, это три светодиода красного (R, red), зелёного (G, green) и синего (B, blue) цветов, смонтированные в одном корпусе очень близко друг к другу.



Когда эти светодиоды зажигаются по-отдельности, то мы видим чистый цвет (красный, зеленый или синий). Сочетание этих цветов в разной интенсивности даст нам остальные цвета спектра. Если все три светодиода горят с максимальной яркостью, то мы увидим белый цвет.

Именно так цвета смешиваются на экранах ваших телевизоров, на больших рекламных панелях и при цветной печати на принтере.

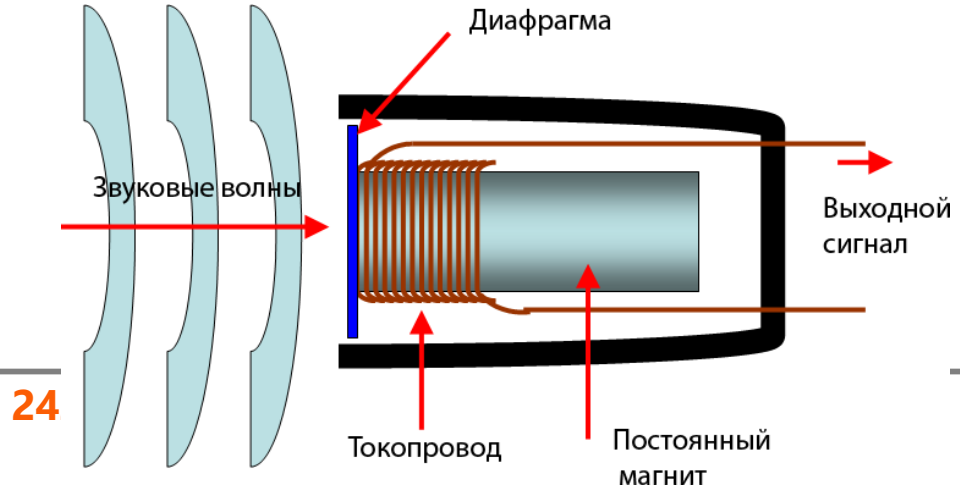


Как могут использоваться кнопка и светодиоды, мы рассмотрели уже в предыдущем разделе. Рассмотрим теперь работу остальных устройств.

Микрофон

Микрофон это электроакустический прибор, способный преобразовывать акустические колебания в электрический сигнал. Таким образом, мы, получив некий электрический сигнал, можем на него тем или иным образом среагировать. В проектах робототехники, не связанных с распознаванием голоса, чаще всего важен сам факт наличия такого сигнала, например в
проектах, где происходит включение чего-то по хлопку в ладоши.

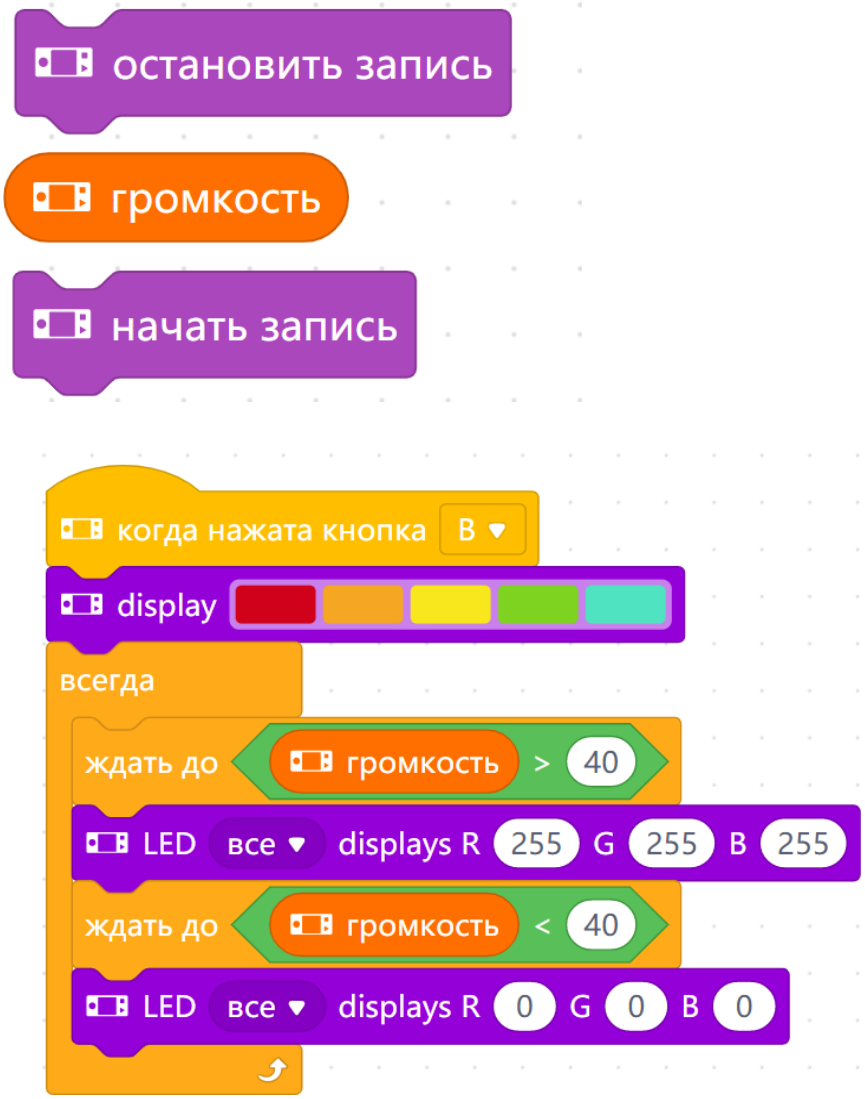
Принцип действия микрофона



Принцип работы микрофона заключается в том, что давление звуковых колебаний воздуха, воды или твердого вещества действует на тонкую мембрану микрофона. В свою очередь, колебания мембраны возбуждают электрические колебания; в зависимости от типа микрофона для этого используются явление электромагнитной индукции, изменение ёмкости конденсаторов или пьезоэлектрический эффект.

За работу с микрофоном в CyberPi отвечают несколько команд в меню «Сенсоры» и «Аудио». С помощью встроенного микрофона мы можем не только определить уровень громкости, но и записать звук в встроенную память контроллера.

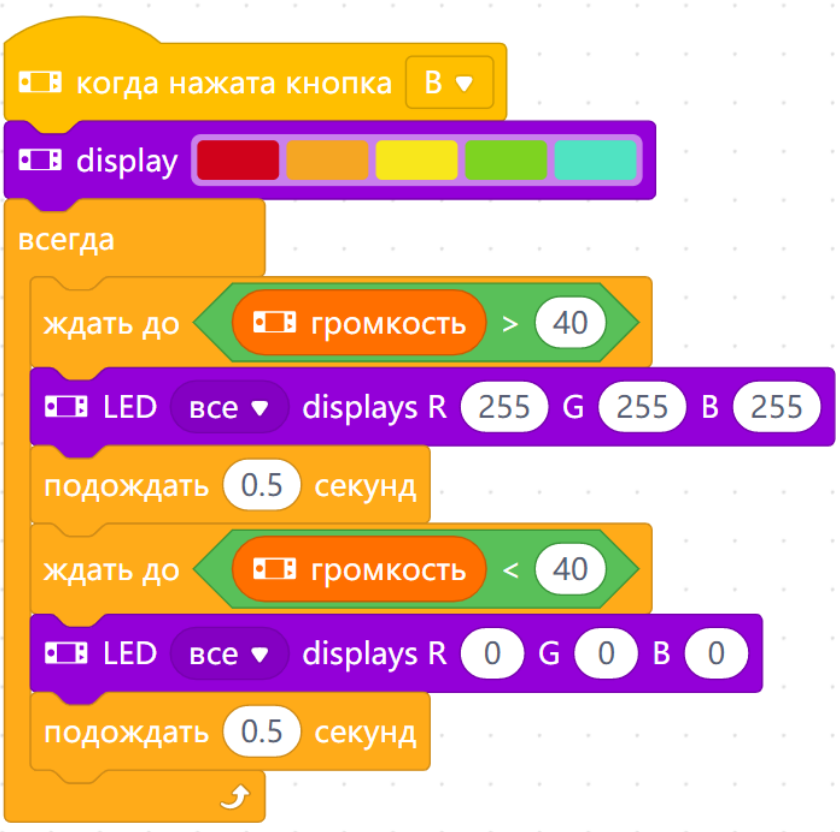
Давайте напишем программу включения и выключения светодиодов по хлопку (или по другому звуку громкостью выше, чем задана в программе).



Запускать программу будем по нажатию на треугольную кнопку «B». Работа программы начинается после нажатия кнопки на контроллере. В начале все

светодиоды у нас будут погашены. Далее введём повторяющийся цикл «Всегда» и внутри него будет собственно рабочая часть нашей программы. Сначала мы ждём сигнала микрофона свыше 40 Децибел и при получении подобного сигнала включаем все светодиоды на максимальную яркость белым светом. При повторном получении сигнала все светодиоды выключатся.

При тестировании работы программы, вы можете заметить, что результат не совсем такой, как предполагалось. Контроллер, например, зажигается и тут же гаснет или наоборот гаснет и тут же снова зажигается. Это происходит от того, что длительность звукового сигнала больше, чем необходимо для срабатывания контроллера, а уровень сигнала, на который он срабатывает в программе установлен слишком низким. Для исправления этой ситуации немного изменим нашу программу следующим образом:



Мы ввели в программный код задержки, которые позволяют не реагировать повторно на звуковой сигнал длительностью до 0,5 секунды и уменьшили чувствительность, подняв уровень сигнала, на который среагирует система, до 50 Дб. Приём с задержкой используется довольно часто и для других датчиков.

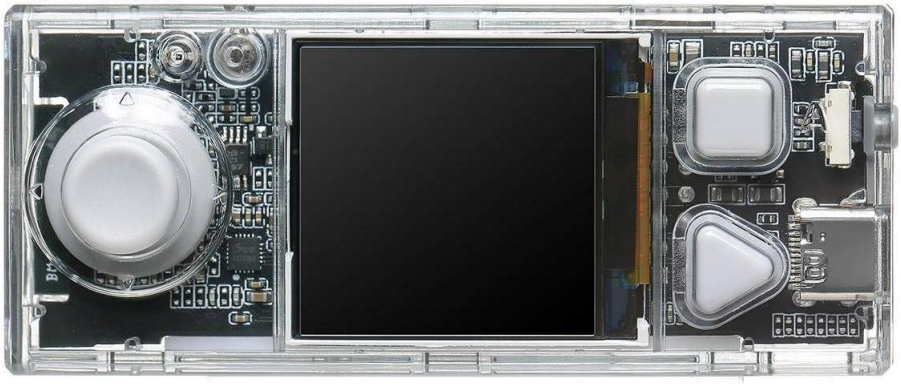
Поэкспериментируйте с параметрами, чтобы понять, как будет от них зависеть работа запрограммированной системы.

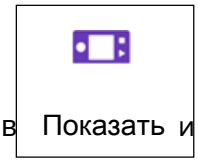
Сделайте систему автономной, переключив Mblock в режим «Загрузка» и загрузив программу непосредственно в контроллер с помощью появившейся кнопки «Загрузить». Убедитесь в работоспособности автономной системы, выключив компьютер и подав на контроллер питание с аккумулятора в Pocket Shild или с Power

Bank через разъём USB Type-C.

Встроенный ЖК-экран

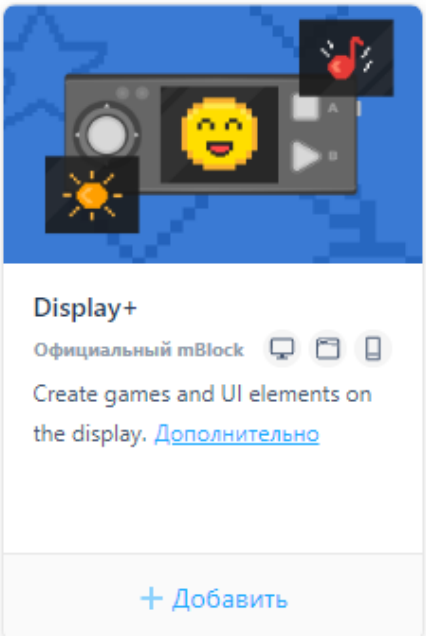
В контроллер CyberPi встроен полноцветный жидкокристаллический дисплей (диагональ 1,44 дюйма, разрешение 128x128 пикселей), способный отображать как текстовую, так и графическую информацию. Это очень удобно, когда надо просмотреть данные, получаемые из облака или с другого устройства по сети или вывести состояние устройства в текущий момент времени или данные с встроенных датчиков.



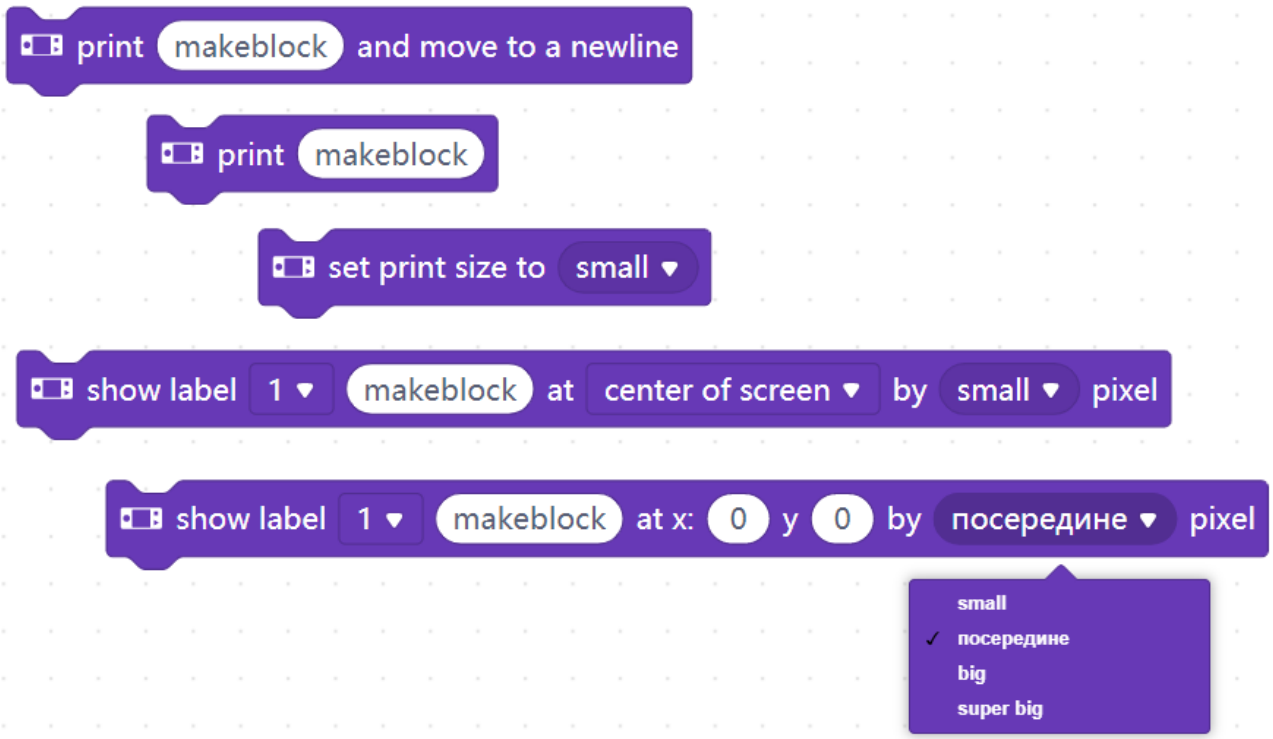
За отображение информации на экране в mBlock 5 отвечает группа блоков 

и

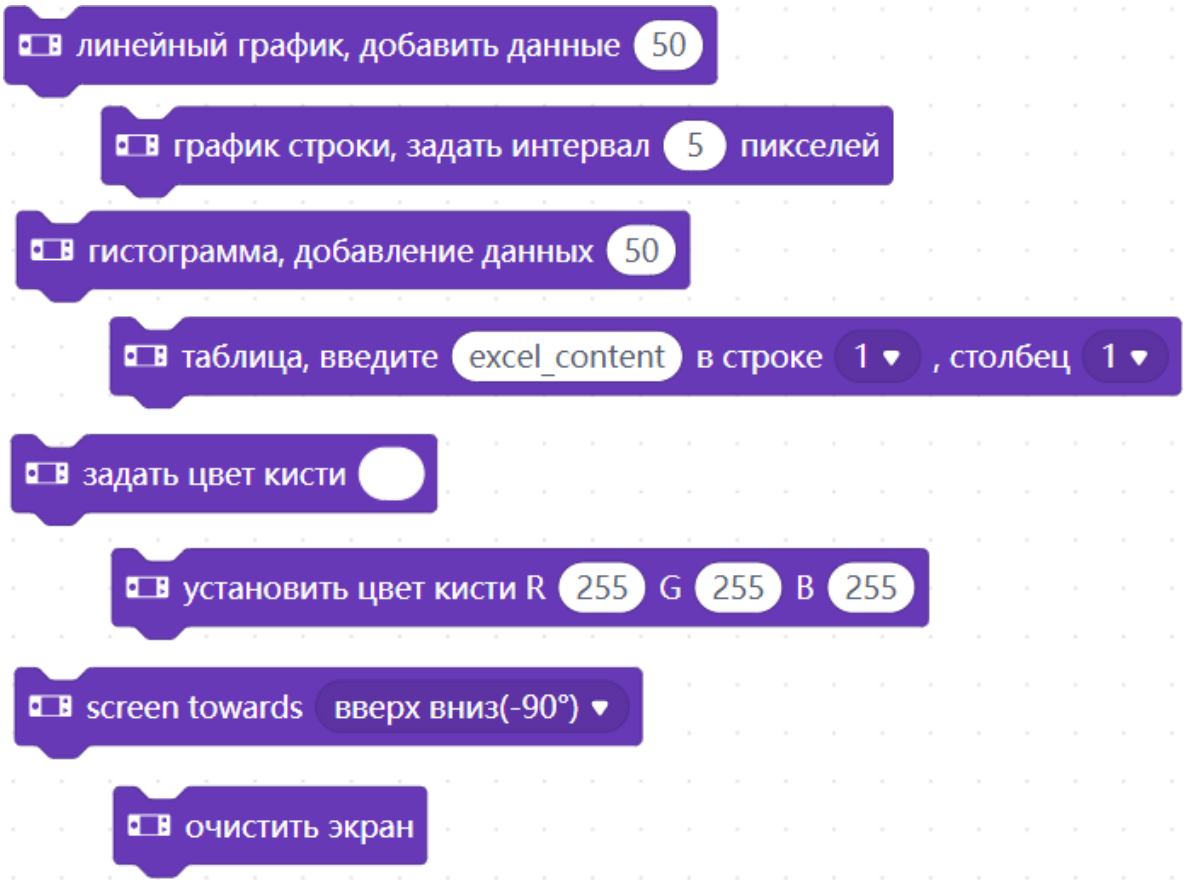
Дополнительное расширение Display+ позволяющее более широко использовать возможности экрана.



Рассмотрим блоки, расположенные в группе «Показать»:



Эти блоки выводят на экран тектовую информацию с нежным размере шрифта и в нужном месте экрана. Это могут быть как надписи, задаваемые пользователем, так и числовые данные, получаемые из облака или со встроенных датчиков CyberPi.



Кроме этого, рассматривая эти блоки, мы видим, что можно строить графики, гистограммы, таблицы, задать цвет кисти и положение объекта на экране. Также, есть функция очистки
экрана, позволяющая удалить с него все изображения, подготовив к выводу новых.

Гироскоп и акселерометр.

Этот датчик используется для измерения ускорения (акселерометр) и угловой скорости (гироскоп). Вместе эти значения дают представление о положении объекта в пространстве.

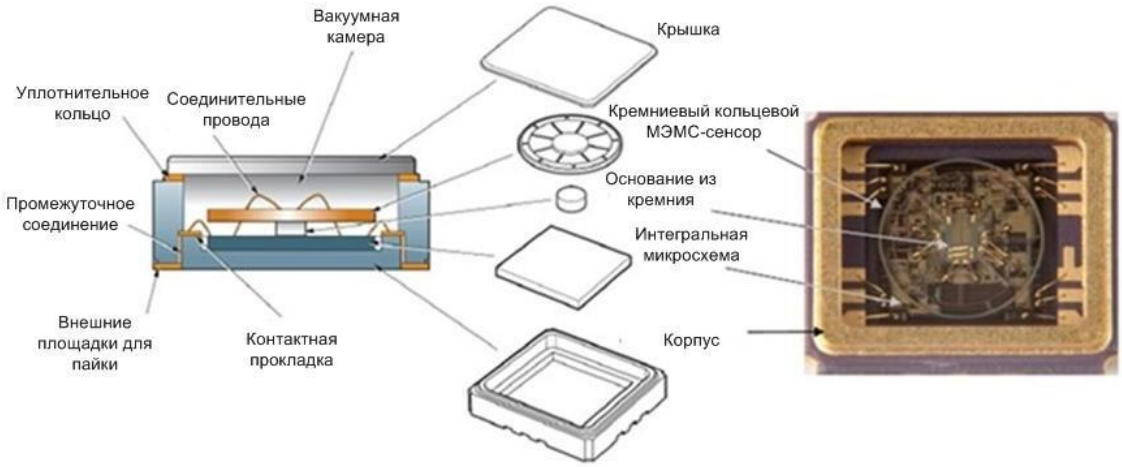
В последнее время, во всём мире стали широко использовать датчики, основанные на микроэлектромеханических системах, так называемых МЭМС. Популярность данных устройств обусловлена рядом причин, таких как, простота их использования, относительно низкая цена и малые габариты. Подобные датчики, как правило, оснащаются интегрированной электроникой обработки сигнала и не имеют движущихся частей. Этим обусловлена их высокая надежность и способность обеспечивать стабильные показания в достаточно жестких условиях окружающей среды (перепады температур, удары, влажность, вибрация, электромагнитные и высокочастотные помехи). Совокупность данных преимуществ побуждает производителей систем для различных сфер применения (от авиа и автомобилестроения до бытовой техники) использовать в своих разработках те или иные МЭМС-сенсоры.

Именно к такому типу относится и датчик Гироскопа с акселерометром, применяемый на контроллере CyberPi

Рассмотрим работу МЭМС-сенсора более подробно:

Как правило, подобные гироскопы выпускаются в герметичных керамических LCC корпусах которые можно устанавливать на системные платы. Датчик состоит из пяти основных
компонентов:

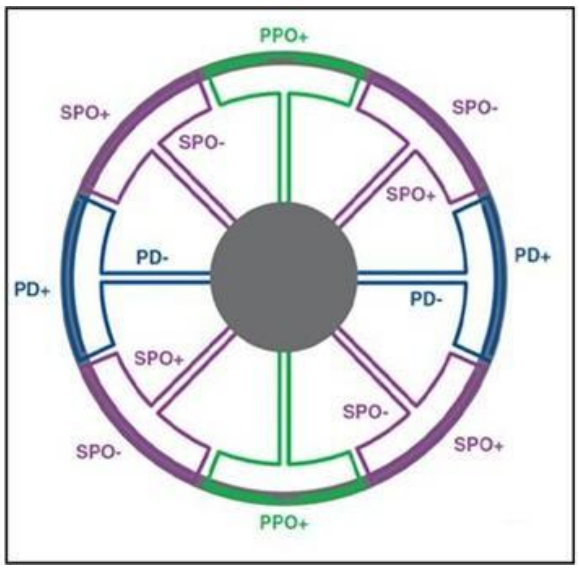
- кремниевый кольцевой МЭМС-сенсор
- основание из кремния
- интегральная микросхема гироскопа (ASIC),
- корпус
- крышка



Кремниевый кольцевой МЭМС-сенсор, микросхема и кремниевое основание размещены в герметичной части корпуса с вакуумом, частично заполненным азотом.

Принцип действия системы гироскопа

Описываемые гироскопы обычно являются твердотельными устройствами и не имеют движущихся частей за исключением сенсорного кольца, которое имеет возможность отклоняться. Оно показывает величину и направление угловой скорости за счет
использования эффекта «силы Кориолиса». Во время вращения гироскопа силы Кориолиса действуют на кремниевое кольцо, являясь причиной радиального движения по периметру кольца.



На рисунке показана структура кремниевого кольца сенсора, показывающая приводы первичного движения «PD» (одна пара), первичные снимающие преобразователи «PPO» (одна пара) и вторичные снимающие преобразователи «SPO» (две пары).

Если гироскоп подвергается воздействию угловой скорости, то на кольцо действуют силы Кориолиса: по касательной к периметру кольца относительно главных осей. Эти силы деформируют кольцо, что вызывает радиальное движение вторичных снимающих
преобразователей. Данное движение, определяемое на вторичных снимающих преобразователях, пропорционально прилагаемой угловой скорости.

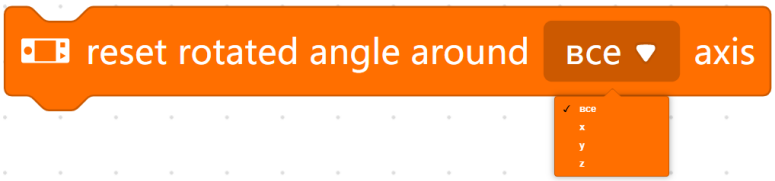
Работа с гироскопом контроллера CyberPi

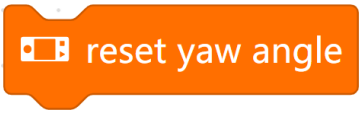
Программные блоки для работы с гироскопом:

Вывод на экран CyberPi Значений перемещений

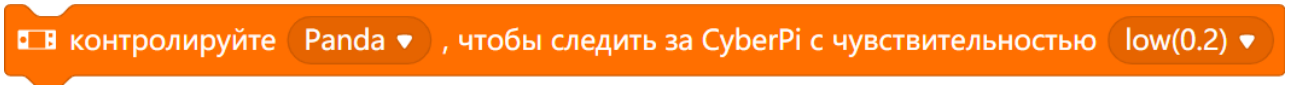
Значения перемещений:

Установка параметров





Контроль персонажа с настройкой чувствительности

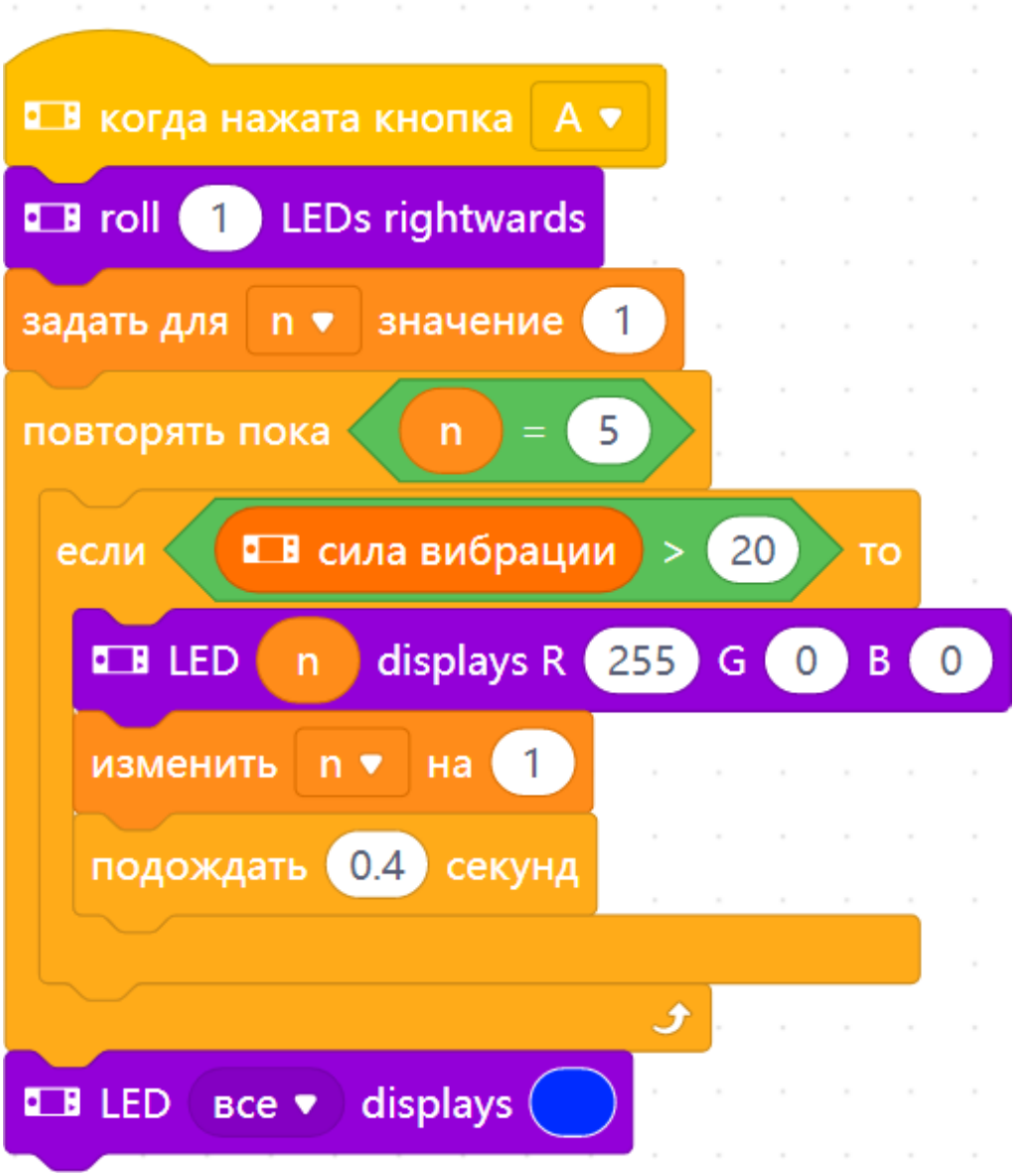


То есть, наш контроллер может реагировать на вибрацию, наклон, вращение и ускорение, что позволяет нам использовать наш датчик в большом количестве различных проектов.

Вибрация

Рассмотрим пример:

В данной программе описан алгоритм поочередного зажигания светодиодов на плате красным цветом. Как только все светодиоды загорятся красным, всё кольцо изменит цвет на синий. Выполнение программы начинается при нажатии на кнопку на контроллере.

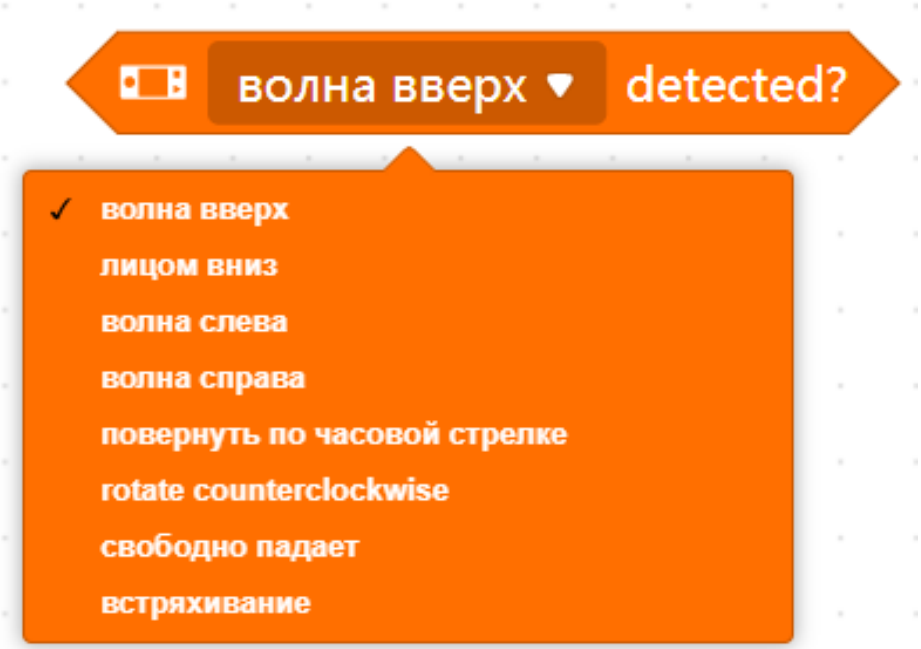


Попробуйте изменить количество и цвет загорающихся светодиодов.

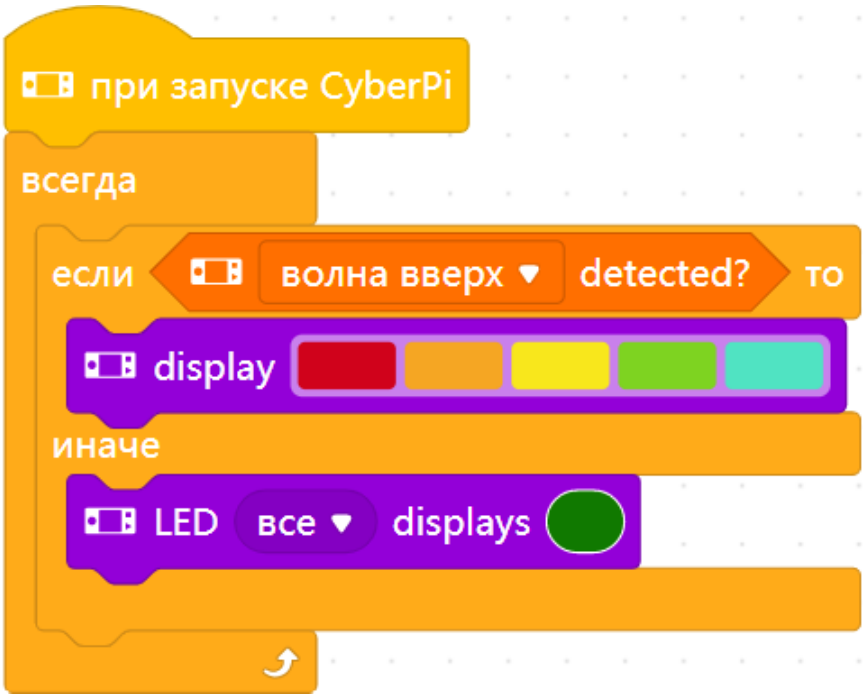
Положение контроллера в пространстве

В программных блоках есть блок, запрограммированный на несколько действий с контроллером контроллера в пространстве. Это сильно облегчает программирование

для некоторых случаев. Вот эти стандартные действия: положение экраном вниз, движение вверх, влево или вправо, вращение по и против часовой стрелки, свободное падение и встряхивание.



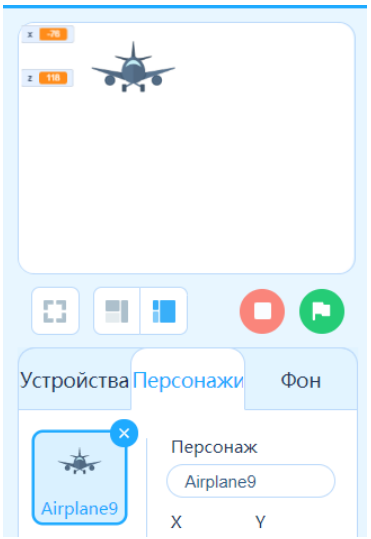
Рассмотрим пример, в котором используется этот блок:

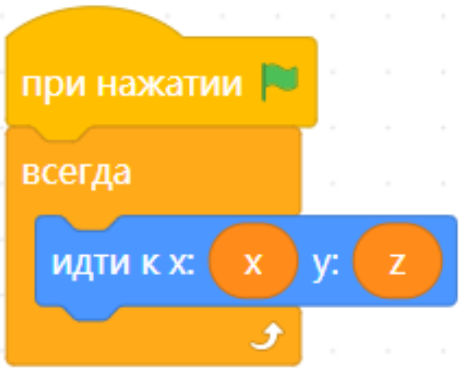


В нём при движении вверх светодиоды загораются разными цветами, а в любом другом случае зелёным цветом. Попробуйте в работе и другие варианты, изменяя значения с помощью раскрывающегося списка в этом блоке.

Кроме этих программных предустановок, мы можем использовать и точные значения поворота, смещения, наклона и силы вибрации, что очень важно при создании серьёзных DIY-проектов. А сейчас, давайте попробуем использовать функционал нашего гироскопа для управления изображением панды в окне анимации на экране ПО Mblock. Для этого нам требуется запрограммировать совместно наш контроллер и спрайт на экране компьютера.

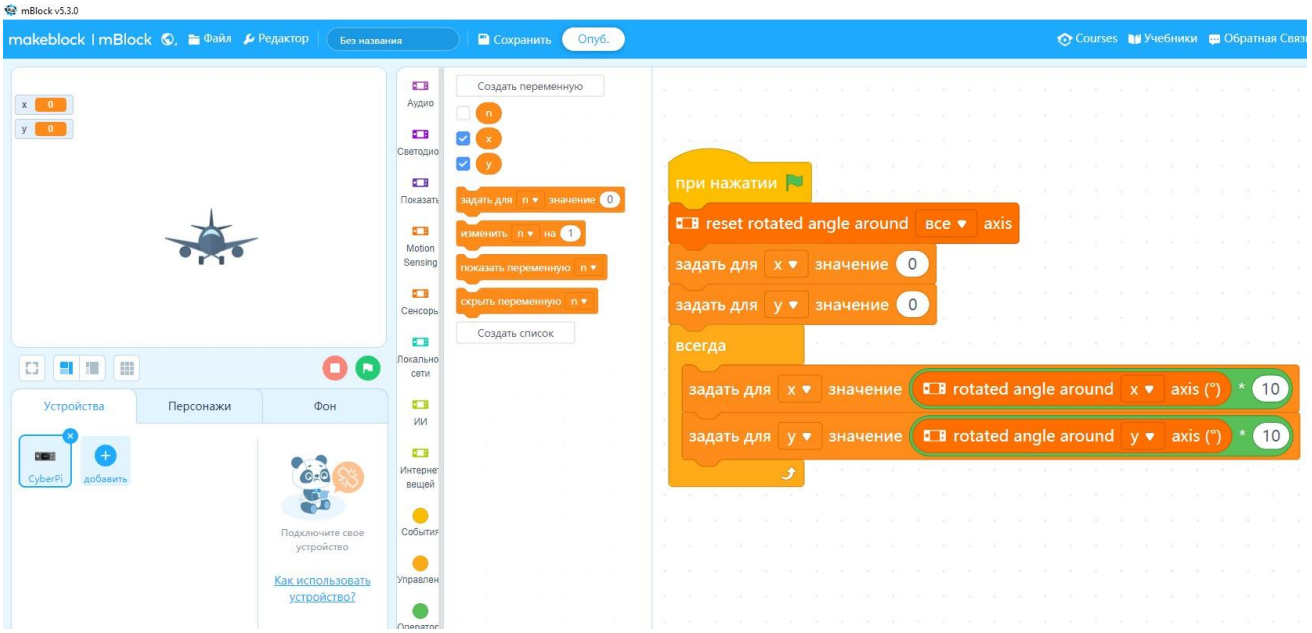
Сначала перейдите в меню «Персонажи», нажмите на кнопку «Костюмы» и выберите изображение самолёта. Здесь же для нашего самолёта составим вот такую простую
программу:





Переменных **X** и **У** у нас ещё не создано, по этому пока что в эти окошки ничего не добавляем.

Теперь переходим в наши «Устройства» и выбираем плату CyberPi. В окне типов программных блоков выбираем «Переменные» и создаём переменные X и Z. Это будут координаты положения нашего самолётика на экране. Используем созданные переменные в создаваемой программе. Сначала присвоим им нулевые значения и в дальнейшем будем их изменять в зависимости от положения нашего контроллера в пространстве.



Раскройте спрайт на весь экран для лучшего эффекта и запустите программу при помощи флажка. Нажмите флажок для запуска процесса управления (контроллер CyberPi должен быть подключен к компьютеру кабелем или по Bluetooth). Теперь, наклоняя контроллер по осям X и Y, мы можем управлять изображением самолёта на экране компьютера.

Точно так же можно управлять и реальным объектом, а не только изображенным на экране.Попробуйте изменить программу таким образом, что бы движение влево-вправо осуществлялось движением контроллера влево-вправо.

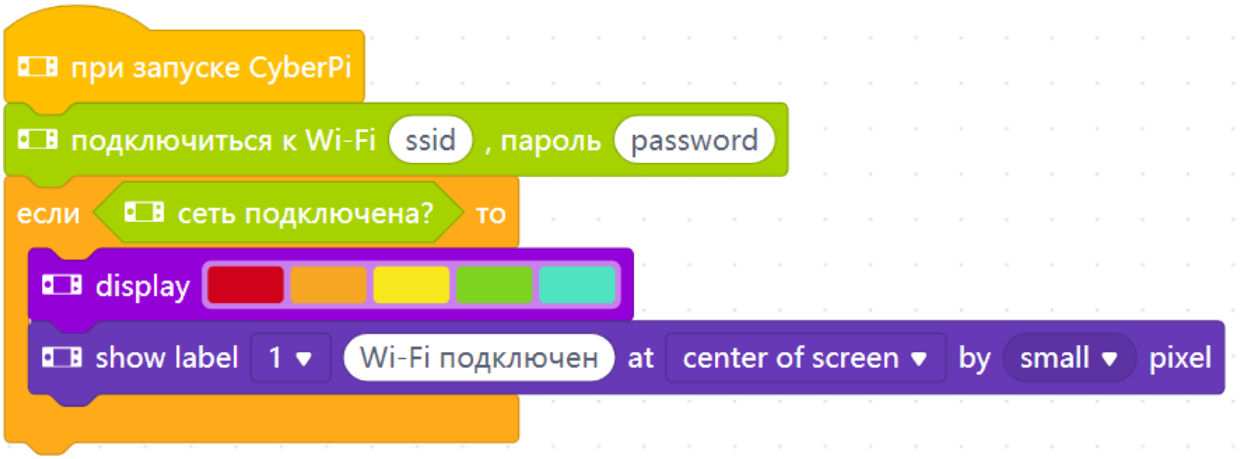
Добавьте к изображению самолёта фон в виде неба и получите настоящий мультфильм с управляемым полётом самолёта.

Подключение контроллера к Wi-Fi сети

Наш контроллер CyberPi содержит «на борту» модуль Wi-Fi, что даёт возможность прямого подключения этой платы к сети. Это простой и эффективный способ обмена информацией между удалёнными системами. Необходимые блоки для программирования подключения вы найдёте в группе команд Wi-Fi.

В этом же разделе вы найдёте команды для отправки и приёма широковещательных сообщений через облако Makeblock, к которому подключаетесь автоматически в момент входа в аккаунт.

Программный код для подключения к Wi-Fi очень простой и не вызывает сложностей. Вот как он может выглядеть:



При включении платы контроллера, происходит подключение к сети с именем, которое необходимо ввести с паролем, который вы тоже должны ввести. После этого контроллер переходит в режим ожидания подтверждения о подключении. Когда подключение установлено, мы увидим индикацию загорающимися светодиодами и надписью на экране. Всё, контроллер подключен к сети.

Важно отметить, что контроллер не просто подключается к сети Wi-Fi, а выполняет подключение в ваш аккаунт в облаке Makeblock. Что это даёт, рассмотрим в следующем разделе. А пока попробуйте подключить ваш контроллер к вашей сети.

Поэтому, для программирование IoT или отправления данных в облако вам понадобиться учётная запись для облака. Для этого вам достаточно пройти регистрацию в самом приложении mBlock5 или на сайте.

Управление контроллером через интернет

Итак, мы научились подключать наш контроллер к сети интернет и к своему аккаунту в облаке Makeblock. Рассмотрим как можно использовать это подключение на практике.

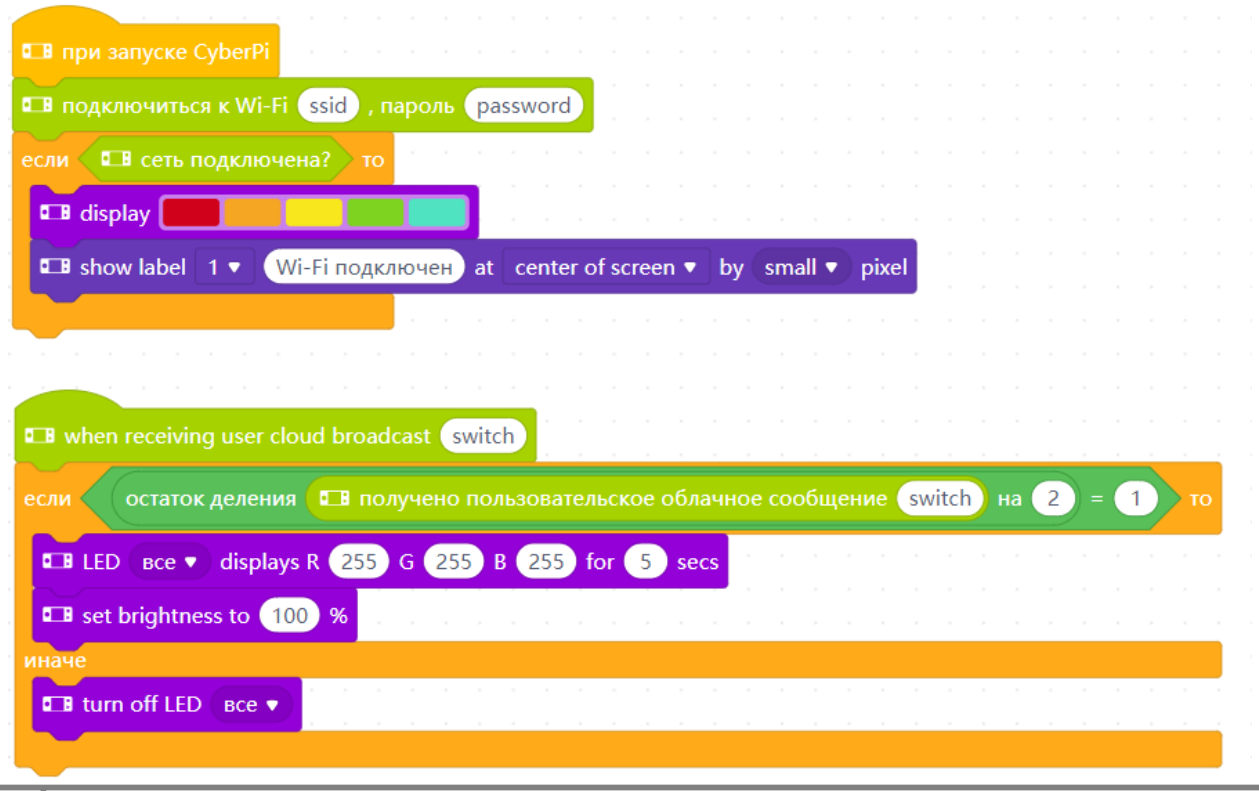
Как уже говорилось, при подключении к Wi-Fi наш контроллер автоматически осуществляет вход в аккаунт Makeblock и может как передать туда данные, так и получить данные из аккаунта. Разберем подробней пример программы удалённого управления.

Мы можем увидеть, что здесь есть две части программы. Одна для устройства CyberPi, а вторая для спрайта.

В данном случае наш компьютер выступает в роли облачного сервера к которому контроллер выполнит подключение после входа в аккаунт. Та часть программы,

Которая пишется для спрайта это программа управления нашими устройствами, подключенными в аккаунт (их может быть несколько). Таким образом можно организовать облачное взаимодействие нескольких устройств. Также, программирование спрайта позволяет нам создать удобный графический интерфейс взаимодействия для нашей системы.

Программа для контроллера:



Как мы видим, программа разбита на 2 части. В первой части осуществляется подключение к сети Wi-Fi и происходит индикация подключения.

Во второй части описаны действия, которые будут выполнены при получении сигнала из облака. Здесь возможно 2 варианта полученного сообщения.

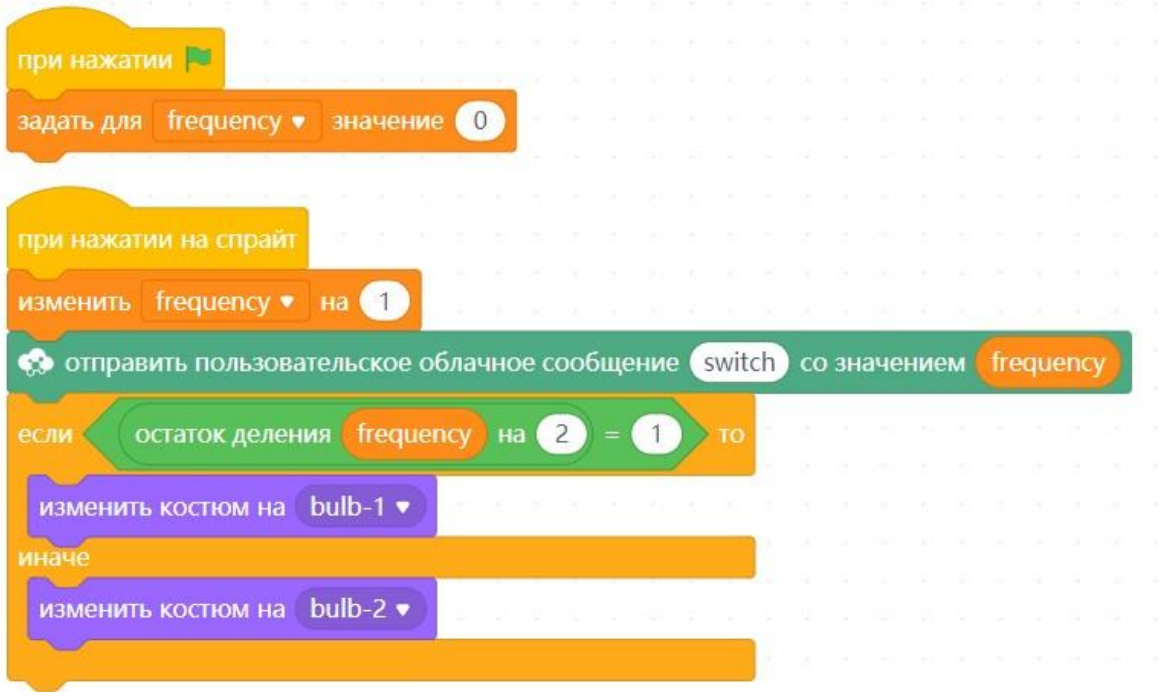
1. Чётное число
2. Нечётное число

В первом случае остаток от деления на 2 будет равен 0 и наши светодиоды будут выключены.

Во втором случае остаток от деления на 2 будет равен 1 и наше светодиодное кольцо на контроллере загорится белым цветом с максимальной яркостью.

Теперь рассмотрим, откуда именно берётся эта команда, приходящая на наш контроллер из интернета.

Программа для спрайта:



Мы видим, что для работы счётчика создана переменная «frequency» и в начале ей присваивается нулевое значение.

Удалённое управление зажиганием и отключением светодиодов на нашем контроллере происходит по нажатию на спрайт в виде лампочки.

Когда мы нажимаем на спрайт, наша переменная изменяется на 1 и это значение передаётся в аккаунт Makeblock в виде широковещательного сигнала. То есть, оно

будет получено всеми устройствами, которые сейчас подключены к вашему аккаунту. Если в программе устройства есть соответствующая этому сообщению команда, то оно выполнит запрограммированное действие.

Далее в программе прописано изменение внешнего вида спрайта для наглядности. Если число чётное, то мы увидим спрайт с погасшей лампочкой, а если нечётное, то с горящей.
Когда мы делаем большой проект, например для управления умным домом, то логично использовать спрайты в виде кнопок и создать более практичный интерфейс.

Опробуйте данный пример на практике.

Создайте программу с обратной связью, когда изображение спрайта будет переключаться «по хлопку», улавливаемому микрофоном контроллера. Работа в беспроводной

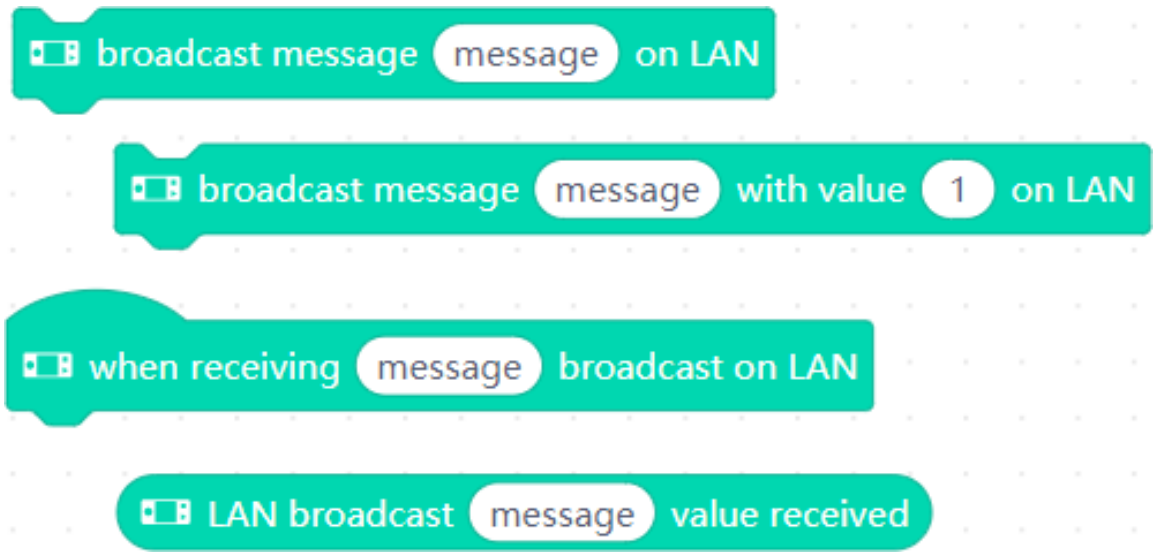
локальной сети. Ещё одной собенностью контроллера CyberPi является возможность

обмена

данными внутри локальной беспроводной сети. Для обмена данными, все устройства CyberPi должны быть подключены к одной точке доступа.

Подключение осуществляется так же, как при подключении к сети интернет, но обмен данными происходит не через облачные сервисы, а напрямую.

Для этого существует отдельный тип блоков, объединенный в группу LAN:



Сразу видно, что возможности передачи и приёма сообщений внутри локальной сети практически аналогичны работе через облачные сервисы в сети интернет. Есть возможность передачи значений и выполнения действий при получении определенного значения сообщения.

Это даёт возможность практически мгновенно обмениваться информацией устройствам, находящимся в рамках одной локальной сети. При этом, расстояния, благодаря современным технологиям, могут быть довольно значительными.

На этом мы закончим рассмотрение основных функций контроллера CuberPi в режиме «без дополнительных устройств».

12. Занятия с микроконтроллером Cyber Pi

Серия уроков с CyberPi была разработана для учащихся от 11 до 14 лет и их преподавателей. Это коллекция из 9занятий начального уровня, обзор содержания которых представлен ниже:

Название деятельности	**Описание**	**Ключевые идеи**
1. Познакомьтесь с CyberPi	Встречайте CyberPi, многофункциональный микроконтроллер с множеством датчиков, кнопок и полноцветным экраном. Откройте для себя множество ключевых функций CyberPi, изучив примеры программ в mBlock.	• Компоненты и особенности CyberPi. • Навигация по программному обеспечению mBlock. • Установите связь между программным обеспечением и оборудованием.
2. Звуковая машина	На этом уроке студенты создают дискотеку, используя встроенные светодиоды и динамик. Эта программа будет использовать кнопки CyberPi для запуска событий и запуска скриптов. Студенты также запрограммируют кнопку для отключения всех звуков и индикаторов, а также кнопку для перезапуска CyberPi.	• Компоненты ввода и вывода на CyberPi. • Написание алгоритма. • Создание программы в mBlock.

3. Диктофон	Объединив динамик, микрофон и интегрированное хранилище, студенты превратят CyberPi в карманный аудиомагнитофон и устройство воспроизведения. Посредством итеративного процесса студенты будут оценивать свои проекты и улучшать свои диктофоны.	• Записывайте звук с помощью CyberPi. • Воспроизведение записей. • Используйте итеративный процесс проектирования.
4. Итерация диктофона	Продолжая работу над проектом «Звукозапись», студенты получат обратную связь от коллег и обдумают свое первоначальное решение. Затем студенты спланируют и создадут многофункциональный проект звукозаписи.	• Собирайте и оценивайте отзывы коллег. • Используйте итеративный процесс проектирования.
5. Игровой контроллер	На этом уроке учащиеся превратят CyberPi в игровой контроллер, объединив программирование устройства и сценическое программирование в mBlock. Студенты изучат примеры программ, чтобы узнать, как CyberPi может управлять движением спрайта. Затем с помощью парного программирования учащиеся изменят существующую игру, чтобы запрограммировать игровойконтроллер CyberPi.	• Совместите программирование персонажа и программирование устройства. • Парное программирование. • Декомпозиция и абстракция.
6. Данные с датчиков	Узнайте, как встроенные датчики CyberPi отображают громкость и интенсивность освещения в окружающей среде. Студенты узнают о представлении данных и построении графиков значений датчиков.	• Представление данных. • Отладка программ. • Что такое датчики.
7. Микшер цветов	Студенты познакомятся с переменными для создания цветового микшера CyberPi. Эта программа будет использовать джойстик и кнопки для управления значениями цвета R, G, B всех встроенных светодиодов. Затем учащиеся будут использовать условные выражения, чтобы гарантировать, что значения R, G, B не выходят за пределы допустимого диапазона.	• Хранение данных с переменными. • Использование условных операторов.

8. Измеритель силы встряски	На этом уроке студенты создадут забавную игру с CyberPi, в которой игрок встряхивает CyberPi в течение десяти секунд. Учащиеся запрограммируют игру, чтобы подсчитать, во сколько раз сила сотрясения превышает 50.	• Ведение счета. • Использование таймера CyberPi. • Отображение текста на дисплее CyberPi.
9. Подарок с сигнализацией.	Студенты будут использовать CyberPi для создания программы, которая определяет, встряхнул ли друг их подарок на день рождения. Используя беспроводную связь, студенты будут отправлять сообщения между вычислительными устройствами, позволяя одному устройству управлять другим.	• Использование беспроводных сетей. • Связь между устройствами.

12.1. Урок 1: Знакомство с Cyber Pi.

Продолжительность:

45 минут Уровень

сложности:

Начальный ★ **Цели**

К концу этого урока студенты смогут:

- Определить ключевые особенности CyberPi.
- Подключить CyberPi к компьютеру с помощью программного обеспечения mBlock.
- Изучат примеры программ для CyberPi с помощью программного обеспечения mBlock.

★ Обзор

Встречайте CyberPi - многофункциональный микроконтроллер с множеством датчиков, кнопок и полноцветным экраном. Откройте для себя множество ключевых функций CyberPi, изучив примеры программ в mBlock.

Ключевые моменты

- Компоненты и особенности CyberPi
- Навигация по программному обеспечению mBlock
- Установите связь между программным обеспечением и оборудованием.

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- CyberPi с кабелем USB-C

- Pocket Shild (опция)
- Примеры программ, включенных в программное обеспечение mBlock

План урока (45 минут)

Тайминг	Контент
5 минут	• Встречайте CyberPi
15 минут	• Экскурсия по мБлоку 5 • Подключите CyberPi • добавить расширение CyberPi • для тестирования живого режима • Изучите пример программы • Тестовый режим загрузки
20 минут	• Изучите примеры программ
5 минут	• Подумайте о функциях и возможностях CyberPi • Идеи мозгового штурма для программ CyberPi • Домашнее задание

Введение

1. Помогите ученикам распаковать CyberPi. Покажите учащимся следующие компоненты и предложитеучащимся найти их в своем наборе CyberPi Kit:

a. CyberPi

6. Кабель USB-C

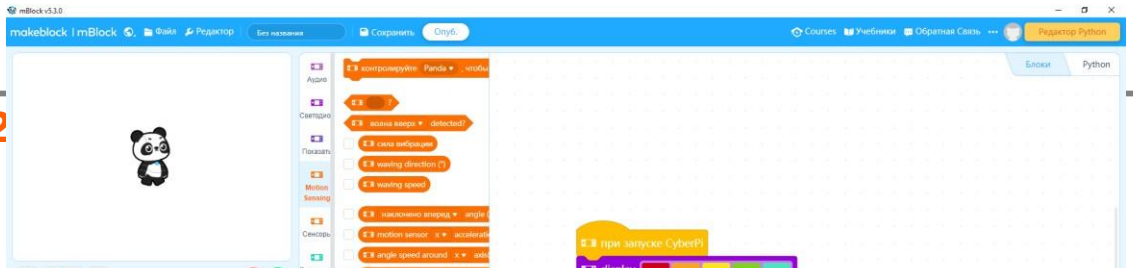
c. Pocket Shild (не входит в базовый комплект)

d. Датчики mBuild (не входят в базовый комплект)

2. Попросите учащихся прочитать надписи на коробке CyberPi и краткое руководство. Затем попросите учащихсянаписать краткое изложение того, что они узнали о функциях и возможностях CyberPi.

Теоретическая часть

Экскурс в mBlock 5

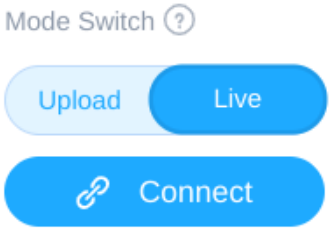
1. Откройте программное обеспечение mBlock 5 или веб-версию mBlock 5.
2. Познакомьте студентов со следующими ключевыми областями интерфейса программы:
3. 

Название области	Функции
Меню	• Выбор языка • Создание, открытие и сохранение файлов • Примеры программ, файл справки и т.д.
Сцена	• Просмотр стадии проекта • Выбор и редактировать спрайтов и фонов. • Подключение аппаратных устройств
Область блоков	• Найдите и выберите блоки сценария, собранные вкатегории с цветовой кодировкой • Найдите и добавьте расширения
Область программирования	• Комбинируйте блоки для создания программ илискриптов. • Перетаскивайте блоки и располагайте их в определенном порядке для управления спрайтами,фоном и / или устройствами.

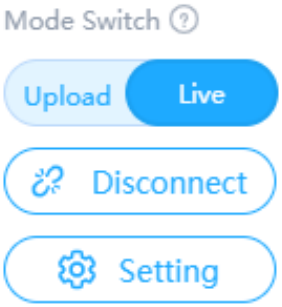
Подключите CyberPi

1. Подключите CyberPi к компьютеру с помощью прилагаемого кабеля.CyberPi должен загрузиться, и на экране отобразится либо последняя загруженная программа, либо главное меню.
2. На вкладке «Устройства» в mBlock нажмите кнопку «Добавить». ВыберитеCyberPi и нажмите OK.
3. Щелкните кнопку Подключить. Затем выберите порт USB и нажмите «Подключение».
4. В случае успешного подключения кнопка изменится на «Отключить».Если вы используете английский язык интерфейса, то это выглядит так:

Не подключено



Подключено



1. Обратите внимание, CyberPi подключен в режиме реального времени. Давайте проверим соединение.
2. В области блоков выберите категорию светодиодов.

Нажмите на блок: 

. Наблюдайте за CyberPi. Светодиодная лента должна отображатьуказанные цвета.

3. Используйте примеры программ.
    1. Щелкните Учебники справа сверху в Меню. Выберите "Примеры программ".
    2. Выберите ярлык CyberPi, чтобы увидеть примеры программ для CyberPi.

3. Найдите и выберите программу «Rainbow Lights».

4. Предложите учащимся прочитать код и предсказать, что произойдет.

5. Подключите CyberPi в режиме реального времени («Жить»).

6. Нажмите на блок 

. Наблюдайте за CyberPi, поскольку светодиодная лента отображаетуказанные цвета. (Обратите внимание: светящаяся желтая рамка окружает сценарий в области сценария, указывая на то, что сценарий запущен.) Щелкните блоки, чтобы остановить программу.

7. Объясните учащимся разницу между режимом реального времени и режимом загрузки.

Режим	Описание
Реального времени (Жить)	• Программа запускается на компьютере (не хранится на CyberPi) • CyberPi должен оставаться подключенным к компьютеру. • Проект mBlock должен оставаться открытым. • Должен использоваться для сценического программирования
Загрузки	• Программа загружена и хранится на CyberPi. • Программа запускается и выполняется внутри CyberPi • CyberPi можно отключить от компьютера • Программное обеспечение mBlock может быть закрыто • Программа останется на CyberPi до тех пор, пока на ее место не будетзагружена новая программа.

1. Переключите CyberPi в режим загрузки и нажмите кнопку «Загрузить».

2. Появится окно «Ход загрузки», которое исчезнет после завершения загрузки. CyberPi перезагрузится и будетвыполнять программу Rainbow Lights. Каждый раз при запуске CyberPi запускается программа Rainbow Lights.

12.2. Звуковая машина

Предмет: Информатика

Продолжительность: 45 минут

Уровень сложности: Начальный

Цели

К концу этого урока студенты смогут:

- Определите входы и выходы CyberPi.
- Напишите алгоритм для планирования и разработки программы в mBlock.
- Создайте программу в mBlock, используя кнопки CyberPi для запуска событий.
- Выберите и используйте программные блоки для управления динамиком и светодиодной лентой.

Обзор

На этом уроке студенты создают звуковую машину, используя встроенные светодиоды и динамик. Эта программа будет использовать кнопки CyberPi для запуска событий и запуска скриптов. Студенты такжезапрограммируют кнопку для отключения всех звуков и индикаторов, а также кнопку для перезапуска CyberPi.

Ключевые моменты

- Компоненты ввода и вывода на CyberPi
- Написание алгоритма
- Создание программы в mBlock

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- CyberPi с кабелем USB-C
- Pocket Shild (опция)
  - Пример программы, Sound Mashine, включенный в программное обеспечение mBlock

План урока (45 минут)

Тайминг	Контент
5 минут	• Входные и выходные данные
15 минут	• Планируйте программу с псевдокодом • Напишите алгоритм • Напишите и загрузите программу • Перезагрузите CyberPi.
20 минут	• Создание звуковой машины
5 минут	• Презентация проектов • Домашнее задание

Введение.

Входные и выходные данные

1. Обсудите со студентами следующие определения.

Понятие	Определение*
Input	Устройство или компонент, позволяющий передавать информацию компьютеру.
Output	Любое устройство или компонент, получающий информацию с компьютера.

2. На примере смартфона попросите учащихся работать в парах или небольших группах, чтобысоздать список входов и выходов для мобильного телефона.

   Некоторые примеры могут включать:

Смартфон	
Input	**Output**
Микрофон Сенсорный экран Кнопки GPS Датчик движения (поворот экрана) Датчик освещенности Камера Соединение с интернет Датчик температуры Порт зарядки Bluetooth	Динамик Экран Наушники Вибрация Соединение с интернет Светодиоды (фонарик / подсветка камеры) Порт зарядки Bluetooth

Предложите учащимся поразмышлять над Уроком 1. Познакомьтесь с CyberPi и создайте списоквходных и выходных данных для CyberPi. Поощряйте студентов вернуться к документации по продукту, прилагаемой к CyberPi, если они застряли.

CyberPi	
Input	**Output**
Микрофон Кнопки (A, B & Home) Джойстик Порт зарядки Bluetooth Датчик движения (гироскоп) Датчик освещённости Звук / микрофон	Динамик Экран Светодиодная лента Светодиодный индикатор (показывает зарядку и питание)

С использованием Pocket Shield и mBuild Kit	
Мультидатчик касания «Ползунок» Ультразвуковой датчик расстояния Сторонние датчики	Моторы (в том числе с энкодером и серво) Светодиодная лента Сторонние модули

Теоретическая часть

Создание алгоритма программы

1. Обсудите со студентами важность создания алгоритма программы перед ее разработкой в программном обеспечении. Познакомьте учащихся с алгоритмом, который может быть полезныминструментом для планирования проекта mBlock.

Понятие	Определение
Алгоритм	Письменная последовательность шагов для программы, написанная на родном языке программиста.

2. Используя следующее описание, подведите студентов к написанию алгоритма для проекта.

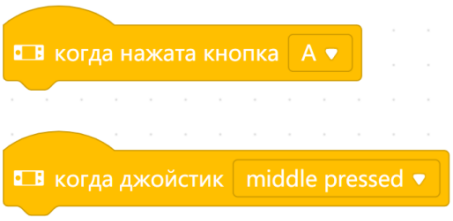
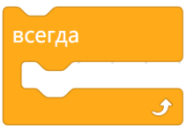
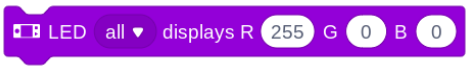
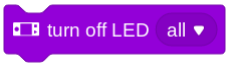
Звуковая машина	
Описание проекта	Создайте проект, в котором CyberPi непрерывно издает звук после нажатия кнопки A и включает все светодиоды после нажатия кнопки B. При нажатии средней кнопки джойстика выключаются и звуки, и свет.
Алгоритм	Когда нажата кнопка A: Включить все светодиоды одним цветом Когда нажата кнопка B: Включить воспроизведение динамиком одной ноты. Если нажата средняя кнопка Джойстика: Остановить воспроизведение звука и погасить

3. Попросите учащихся изучить приведенный выше алгоритм и определить входные и выходныеданные для программы.

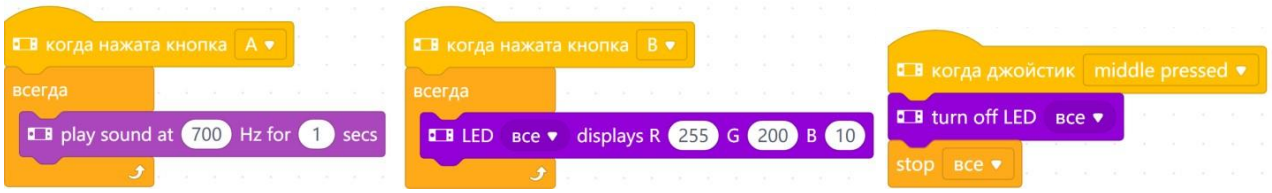
Проект «Звуковая машина»	
Input	**Output**
Кнопка A Кнопка B Средняя кнопка джойстика	Динамик Светодиоды

Написание программы

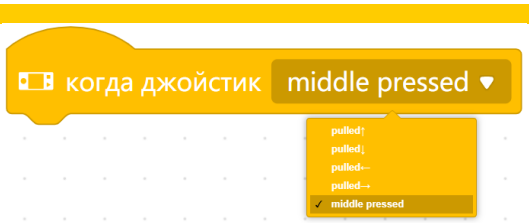
1. Теперь, когда написан алгоритм, пора узнать о новых блоках, необходимых для программы.

Категория	Блок	Функции
События		Определяет действие на CyberPi, которое запускает выполнение прикрепленных действий..
Управление		Блок бесконечного цикла Непрерывно выполнять действия, вложенные в блок.
		Останавливает все скрипты, включая все циклы.
АУДИО		Воспроизведите звук на зуммере CyberPi в течение определенного времени. Диапазон частот: от 0 до 1000 Герц
Светодиоды		Зажечь все или отдельный светодиод на борту контроллера заданным цветом. Диапазон значений цвета: от 0 до 255
		

2. Запустите программу mBlock 5 или mBlock 5 Web version. Добавьте CyberPi на вкладке «Устройства» и подключитесь в режиме реального времени.
3. 3. Напомните учащимся, как перетаскивать блоки из категорий с цветовой кодировкой в областиблоков. Попросите их собрать каждый из следующих сценариев:



Внимание: Опцию, нажатия средней кнопки джойстика, можно найти в раскрывающемся меню.



4. Протестируйте программу в режиме реального времени и / или в режиме загрузки.
5. Предложите учащимся поэкспериментировать с различными значениями для зуммера исветодиодных блоков, чтобы понаблюдать за работой CyberPi..
6. **Дополнительно:** если позволяет время, научите студентов значениям цвета RGB. Или разрешитеим использовать палитру цветов для определения значений RGB определенных цветов.

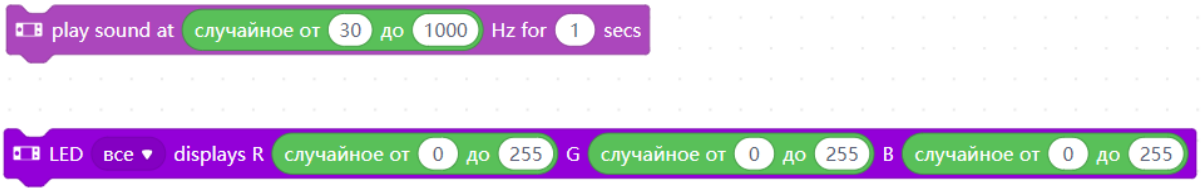
Рандомизация вывода

7. В приведенном выше коде, были запрограммированы конкретная частота зуммера и значениецвета RGB. CyberPi навсегда повторяет один и тот же звук и один и тот же цвет светодиода.

Следующий блок может использоваться, чтобы позволить программе выбирать случайное значение каждый раз, когда цикл навсегда повторяется.

Category	Block	Function
Операторы	случайное от 1 до 10	Компьютер или CyberPi выбирают случайное значение из указанного диапазона

1. Измените предыдущие программы, включив в них следующие случайные блоки:



Обратите внимание, эти диапазоны соответствуют диапазону значений, принятых для каждого блока.

2. 2. Предложите учащимся протестировать новую программу. Обратите внимание: учащиеся могутизменить значение секунд в блоке звукового сигнала воспроизведения на десятичную дробь, еслиим нужен более быстрый звуковой сигнал.

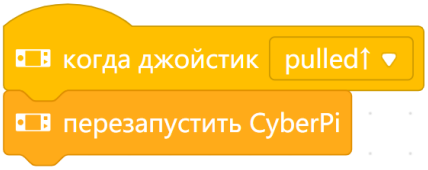
Перезагрузка CyberPi

Программирование кнопки для перезапуска CyberPi может быть полезным инструментом на

следующих уроках. Итак, научимся программировать кнопку для ручного перезапуска CyberPi.

Category	Block	Function
Управление	перезапустить CyberPi	Перезапускает или перезагружает устройство CyberPi. CyberPi воспроизведет последнюю загруженную на устройство программу.

3. Попросите учащихся создать следующий сценарий и протестировать программу:



Практическая часть.

Создать звуковую машину

1. Дайте студентам время поэкспериментировать с существующим кодом. Предложите им попробовать разные значения параметров в блоках.

2. Предложите учащимся изучить другие блоки в категориях «Светодиод» и «Аудио» в областиблоков.
3. После того, как они исследуют различные блоки. Попросите учащихся написать свой алгоритмдля расширенной версии проекта «Звуковая машина».

Предложите учащимся создать свой проект, используя алгоритм в качестве руководства.

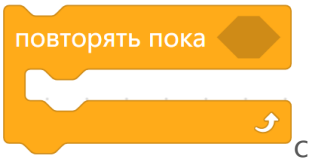
Вывод.

Демонстрация Проекта:

1. Сгруппируйте учащихся в пары и попросите учащихся представить свою звуковую машинуCyberPi своему партнеру.
2. Попросите учащихся задать друг другу следующие вопросы:

- Чем вы больше всего гордитесь в своем проекте?
- Что было самым сложным в этом проекте?

Дополнительное задание:

- • Запрограммируйте кнопку остановки, используя блок 

с блоком 

• Используйте кнопки джойстика, чтобы запрограммировать несколько различных звуков и световых шоу.

12.3. Диктофон

Предмет: Информатика

Продолжительность: 45 минут Уровень сложности: Начальный

Цели

К концу этого урока студенты:

• Создадут программу в mBlock, которая записывает и воспроизводит аудио.

• Исследуют итеративный процесс для разработки решения вычислительной проблемы.

Обзор

Объединив динамик, микрофон и интегрированное хранилище, студенты превратят CyberPi в карманныйдиктофон. Посредством итеративного процесса студенты будут оценивать свои проекты и улучшать свои диктофоны.

Ключевые моменты

• Запись звука с помощью CyberPi

• Воспроизведение записей

• Использование итеративного процесса проектирования.

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- CyberPi с кабелем USB-C
- Pocket Shild (опция)
- Пример программ, включенные в программное обеспечение mBlock:

  - CyberPi – Lesson 3 – Sound Recorder 1

  - CyberPi – Lesson 3 – Sound Recorder 2

План урока (45 минут)

Тайминг	Контент
10 минут	• Развивающиеся вычислительные решения
10 минут	• Составить план • Создать диктофон на основе Create Sound Recorder 1.0
20 минут	• Создать диктофон на основе Create Sound Recorder 2.0
5 минут	• Выводы • Дополнительные задания

Введение

Развивающиеся вычислительные решения

3. Обсудите со студентами, как вычисления решают повседневные проблемы реального мира. В классе составьте список вычислительных решений, с которыми студенты и их семьи сталкиваются ежедневно. Некоторые примеры могут включать:
   - o Смартфон
   - o Будильник
   - o Холодильная сигнализация
   - o Автомобили o Общественный транспорт
   - o GPS
   - o микроволновая печь
   - o Отчет о погоде или приложение
4. Обсудите со студентами, как вычислительные решения развивались с течением времени. Вот конкретный пример, которым вы можете поделиться:

Эволюция получения маршрута к пункту назначения

- Использование распечатанной дорожной карты или карты общественного транспорта для определения маршрута из одного места в другое до отправления.
  Использование веб-сайта на компьютере для создания инструкций для печати до отправления. (например, Mapquest или транзитный веб-сайт)
- Использование мобильного устройства для просмотра веб-сайта с указаниями по маршруту.
- Использование мобильного устройства для просмотра карты во время маршрута (маршруты не указаны). (Обратите внимание, это была первая итерация приложения Google Maps; у него не было направлений.)
- Использование мобильного устройства для просмотра карты с маршрутами проезда или общественного транспорта.
- Использование мобильного устройства для предоставления пошаговых маршрутов движения по GPS (пока нет поддержки для транспорта или пешеходов).
- Использование мобильного устройства для предоставления пошаговых GPS-маршрутов движения с корректировкой маршрута на основе данных о дорожном движении в реальном времени.
- Использование мобильного устройства для предоставления маршрутов движения, ходьбы или общественного транспорта с помощью GPS в режимереального времени.

Использование GPS-направлений на смартфоне или мобильном устройстве со временем эволюционировало. Например, в первой итерации Google Maps отсутствовали многие ключевые функции (пешеходные маршруты, общественный транспорт, движение в реальном времени, закрытые дороги, информация о пунктеназначения и т. Д.), Которые со временем добавлялись посредством оценки решения, отзывов пользователей и Сбора данных.

Теоретическая часть

Составьте план.

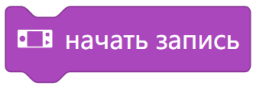
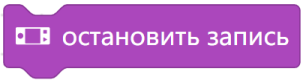
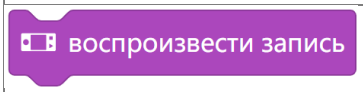
4. Используя следующую постановку задачи, помогите студентам спланировать решение проблемы.

Вам предоставлен образец решения. Обратите внимание, что исходное решение должно быть очень простым. Это позволит учащимся повторять и добавлять функции в процессе разработки многофункционального устройства записи звука.

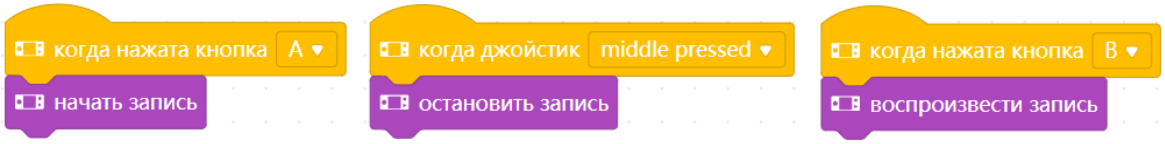
Sound Recorder 1.0	
Проблема	Один из ваших одноклассников посещает уроки иностранного языка. Они хотели бы попрактиковаться в произношении слов, которые выучили в классе, но к тому времени, как они вернутсядомой, они забывают, как их учитель сказал эти слова. Как можно использовать CyberPi, чтобы помочь однокласснику сэтой проблемой?
Предложенное решение	Создайте звукозапись с помощью CyberPi, чтобы записать, какучитель произносит слова, а затем воспроизведите запись во время учебы дома.
Алгоритм	Когда нажата кнопка A: Начать запись При нажатии средней кнопки джойстика:Остановить запись Когда нажата кнопка B:Воспроизвести запись

Создание Sound Recorder 1.0

5. Теперь, когда программа спланирована, пора узнать о новых блоках, необходимых для программирования:

Категории	Блоки	Функции
АУДИО		Старт записи Аудио. Лимит времени записи 10 секунд.
		Остановка записи Аудио
		Проиграть последнюю запись,сохранённую в CyberPi.

6. Запустите программу mBlock 5 или mBlock 5 Web version. Добавьте CyberPi на вкладке «Устройства» и подключитесь в режиме реального времени.
7. Напомните учащимся, как перетаскивать блоки из категорий с цветовой кодировкой в области блоков. Попросите студентов создать следующие сценарии и протестировать программу:



a. Спросите учащихся, готово ли устройство помогать однокласснику при использованиина уроке иностранного языка. Многие студенты начнут определять области улучшения этого функционала. Предложите учащимся провести мозговой штурм и составить список способов улучшить это базовое устройство звукозаписи. Если учащимся требуется дополнительное руководство, подумайте о том, чтобы задать следующие вопросы:

b. Как ваш одноклассник узнает, как использовать CyberPi Sound Recorder?

c. Как ваш одноклассник узнает, что CyberPi записывает?

d. Какие еще функции могут быть полезны вашему однокласснику?

e. Какие другие компоненты CyberPi можно использовать для создания более многофункционального решения? (т.е. дисплей, светодиодная лента, динамик, датчикдвижения)

Практическая часть

Планирование и создание Sound Recorder 2.0

1. 1. Объясните студентам, что программисты и разработчики программного обеспечения часто улучшают свои решения. (Обратите внимание, что именно поэтому программные приложенияобновляются и модифицируются.) Разработка программного обеспечения - это итеративный процесс.
2. 2. Используя список улучшений, которые они создали в предыдущем разделе, попросите учащихся определить три наиболее важные функции, которые они хотели бы добавить впрограмму звукозаписи.

3. Попросите учащихся описать и обосновать функции, которые они будут добавлять. Затем, попросите учащихся написать алгоритм для каждой функции. Примерный план представленниже:

Sound Recorder 2.0	
Функция # 1	Добавьте на дисплей инструкции, сообщающие пользователю, какие кнопки нажимать для управления CyberPi Sound Recorder.
	Когда CyberPi запускается: Отображение на экране: «Нажмите A, чтобы начать запись, нажмите джойстик, чтобы остановить, нажмите B, чтобы играть»
Функция # 2	Используйте светодиоды, чтобы сообщить пользователю, когда CyberPi выполняет запись.

	Когда нажата кнопка A: Установите все светодиоды на зеленый Начать запись При нажатии средней кнопки джойстика: Установите все светодиоды на красный Остановить запись Когда нажата кнопка B: Установите все светодиоды на синий Воспроизвести запись
Функция # 3	Добавьте возможность изменять громкость CyberPi.
	Когда CyberPi запускается: Установите громкость на 50% Когда джойстик нажат вверх: Увеличить громкость на 10% Когда джойстик нажат вниз: Уменьшить громкость на 10%

Обратите внимание, что в этот урок включены два примера программ. Это должно быть на усмотрение учителя. Студентов следует побуждать к мозговому штурму и созданию программына основе своих собственных идей.

4. Студенты создают **Sound Recorder 2.0,** используя алгоритмы как руководство.

Выводы

Итоги занятия:

1. 1. Предложите учащимся поделиться некоторыми идеями, которые они включили в свои диктофоны.
2. Напомните учащимся, что на следующем занятии они будут продолжать разработку проекта «Звукозапись» с помощью итеративного процесса.

Дополнительное задание

- Предложите учащимся изучить UX-дизайн (дизайн пользовательского опыта) и его роль в разработке программного обеспечения.

Предложите студентам исследовать доступность и удобство использования ПО приразработке программного обеспечения.

12.4 Итерация диктофона

Предмет: Информатика Продолжительность:

45 минут Уровень сложности: Начальный

Цели

К концу этого урока студенты смогут:

- Предоставить конструктивную обратную связь о вычислительном решении.
- Изменить существующую программу, чтобы улучшить взаимодействие с пользователем.
- Следовать итеративному процессу для разработки решения вычислительной проблемы.

Обзор

Продолжая работу над проектом «Звукозапись», студенты получат обратную связь от сверстников ипоразмышляют над своим первоначальным решением. Затем студенты спланируют и создадут многофункциональный проект звукозаписи.

Ключевые моменты

- Собирать и оценивать отзывы коллег
- Использовать итеративный процесс проектирования

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- CyberPi с кабелем USB-C
- Pocket Shild (опция)
- Пример программы, Sound Mashine, включенный в программное обеспечение mBlock

План урока (45 минут)

Тайминг	Контент
15 минут	• Собирайте отзывы коллег
20 минут	• Планирование и создание Sound Recorder 3.0
10 минут	• Проектная документация • Дополнительные задания

Введение

Собирайте отзывы коллег

1. Со всем классом проведите формальную экспертную оценку и обратную связь. Объясните, как разработчики программного обеспечения полагаются на отзывы, обзоры и данные пользователей при планировании дополнительных итераций вычислительного решения.

2. Попросите учащихся положить свой план для Sound Recorder 2.0 на свой стол, открыть программу mBlock и поместить CyberPi с загруженным диктофоном на свой стол.

3. Если позволяет время, предложите учащимся перемещаться по комнате и давать отзывы о проектахсвоих одноклассников. Некоторые наводящие вопросы для обратной связи могут включать:

   - Что вам больше всего нравится в их проекте?
   - Были ли ясны инструкции по использованию записывающего устройства? Вам приходилосьгадать

или делать какие-либо предположения о том, как его использовать?

- Могут ли они что-нибудь добавить в свой проект, чтобы сделать его более удобным для пользователя?

Есть ли функция, которая, по вашему мнению, могла бы улучшить их проект записывающегоустройства?

Практическая часть

Планирование и создание Sound Recorder 3.0

1. Попросите учащихся просмотреть полученные отзывы и придумать список идей для SoundRecorder 3.0. Некоторые идеи по улучшению:

- Включите заголовок, который появляется при запуске CyberPi.
- Измените цвета текста на экране.
- Установите определенную продолжительность записи вместо использования кнопки остановки.
- Используйте джойстик для управления громкостью и / или скоростью воспроизведения.
- Отобразите текущий уровень громкости на экране или используйте светодиодную ленту, чтобы указать громкость.
- Включите светодиодную ленту во время записи.
- Используйте джойстик для управления продолжительностью записи, сохраняя продолжительность записи в переменной.

2. Следуя шагам из предыдущего урока, попросите учащихся определить улучшения, написатьалгоритм и создать свой проект Sound Recorder 3.0.

Выводы.

Проектная документация

Теперь, когда учащиеся создали многофункциональный диктофон, попросите их написать описание для своего диктофона, объясняющее функции, которые они включили в свою конструкцию.

Дополнительные задания

- Предложите учащимся собрать отзывы от различных заинтересованных сторон (например, родителей, учителей, друзей и т. Д.).

Попросите учащихся внести улучшения в проект другого учащегося.

12.5. Игровой контроллер.

Предмет: Информатика

Продолжительность: 45 минут Уровень

сложности: Начальный

Цели

К концу этого урока студенты смогут:

- Различать сценическое программирование и программирование устройства.
- Изучить и описать, как функционирует существующий проект.
- Создать программу в mBlock, используя CyberPi для управления спрайтами.
- Изменить существующую программу.

Обзор

На этом уроке студенты превратят CyberPi в игровой контроллер, объединив программирование устройства и сценическое программирование в mBlock. Студенты изучат примеры программ, чтобы узнать, как CyberPi может контролировать движение спрайта. Затем с помощью парного программирования учащиеся изменят существующую игру, чтобы запрограммировать игровой контроллер CyberPi.

Ключевые моменты

- Объединение сценического программирования и программирования устройства
- Парное программирование
- Декомпозиция и абстракция

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- CyberPi с кабелем USB-C
- Pocket Shild (опция)
- Примеры программ, включенные в программное обеспечение mBlock:

CyberPi – Lesson 5 – Space Adventures

CyberPi – Lesson 5 – Chase Game

План урока (45 минут)

Тайминг	Контент
5 минут	• Дискуссия о видеоиграх

15 минут	• Изучите примеры игр • Абстракция и декомпозиция
20 минут	• Парное программирование • Изменение примера проекта
5 минут	• Уважение интеллектуальной собственности • Дополнительные задания

Введение.

Дискуссия о видеоиграх

Попросите учащихся обсудить с партнером следующее:

- Вы играете в видеоигры? Если да, то как часто?
- На каком устройстве или игровой системе вы предпочитаете играть в игры?
- Какой ваш любимый игровой контроллер?

Какие особенности делают его вашим любимым?

Теоретическая часть

Изучите примеры игр

1. Назначьте студентов-партнеров и определите, кто является партнером A, а кто - партнером B.
2. Попросите каждого ученика открыть соответствующий пример проекта:
   - Партнер A, Lesson 5 – Chase Game
   - Партнер Б, Lesson 5 – Space Adventure
3. Попросите учащихся подключить CyberPi в режиме реального времени и сыграть в игру.
4. Каждый проект сочетает этапное программирование и программирование устройства длясоздания игры mBlock, управляемой CyberPi. Объясните учащимся следующее:

Термин	Определение
Сценическое программирование	Последовательности программных блоков, которые взаимодействуют со спрайтами и фоном сцены в mBlock.
Программирование устройства	Последовательности программных блоков, которые взаимодействуют с физическими вычислительными устройствами, подключенными в mBlock.

5. Попросите учащихся изучить назначенный им пример программы и различить этапноепрограммирование и программирование устройства.

Обратите внимание, что программирование устройств будет происходить на устройствах, перечисленных на вкладке «Устройства», а программирование этапа - на спрайтах и фонах,перечисленных на вкладках «Спрайты» и «Фон».

Абстракция и декомпозиция

1. Студенты будут использовать стратегии вычислительного мышления, абстракцию и декомпозицию, чтобы изучить примеры проектов и определить, как изменить существующую игру, чтобы добавить игровой контроллер CyberPi. Они будут использовать абстракцию, чтобыигнорировать или отфильтровывать части программы,

Термин	Определение
абстракция	Упрощение проблемы, скрывая, отфильтровывая или игнорируя ненужные детали.
декомпозиция	Разбиение проблемы на более мелкие части.

которые не нужны для задачи, и использовать декомпозицию, чтобы разбить этапы программирования сцены и
программирования устройств, которые необходимы для создания контроллера.

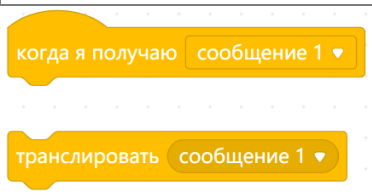
2. Попросите учащихся внимательно изучить назначенную им игру и выполнить следующее задание:

Игровой контроллер	
Проблема	Вам нужно взять существующую программу и добавить игровой контроллер CyberPi. Воспользуйтесь примером программы, чтобы узнать, как выполнить эту задачу.
Испытание	Напишите исчерпывающее объяснение взаимодействия CyberPi и сцены в примере игры.

Вычислительное мышление	
Абстракция	Поощряйте студентов использовать абстракцию для фильтрации ненужных частей программ, которые не взаимодействуют с CyberPi. Вот некоторые примеры: Space Game ○ Название, шар, астероиды и спрайты здоровья ○ Фон ○ Скрипты «когда я получаю gameStart» Chase Game ○ Спрайты Title и Bat ○ Фон ○ Скрипты «когда я получаю gameStart»

Декомпозиция	Предложите студентам разложить программу на части. Убедитесь, что учащиеся указали такие особенности, как: ○ Какие кнопки на CyberPi используются? ○ Какие спрайты находятся под контролем? ○ Как кнопки CyberPi управляют спрайтами?

3. С помощью приведенного выше упражнения учащиеся должны были обнаружить следующиеблоки:

Category	Block	Function
 События		Отправьте сообщение с одного устройства, спрайта или фона надругое. Используется для синхронизации действий.
 Управление		Условный цикл Выполняет вложенные внутрь действия, если выполняется условие.
 Сенсоры		Используется с условным оператором, чтобы определить, нажимается ли джойстик или перемещается пользователем..

Практическая часть

Парное программирование

5. Учащиеся будут работать с партнером, чтобы добавить игровой контроллер CyberPi в существующий проект. Познакомьте учащихся с ролями в парном программировании:

Парное программирование	
Навигатор	Отслеживает общую картину и решает, что делать дальше.
Ведомый	Человек, использующий компьютер, на самом деле пишет код.

6. Попросите учащихся менять роли каждые 3-5 минут во время части урока.

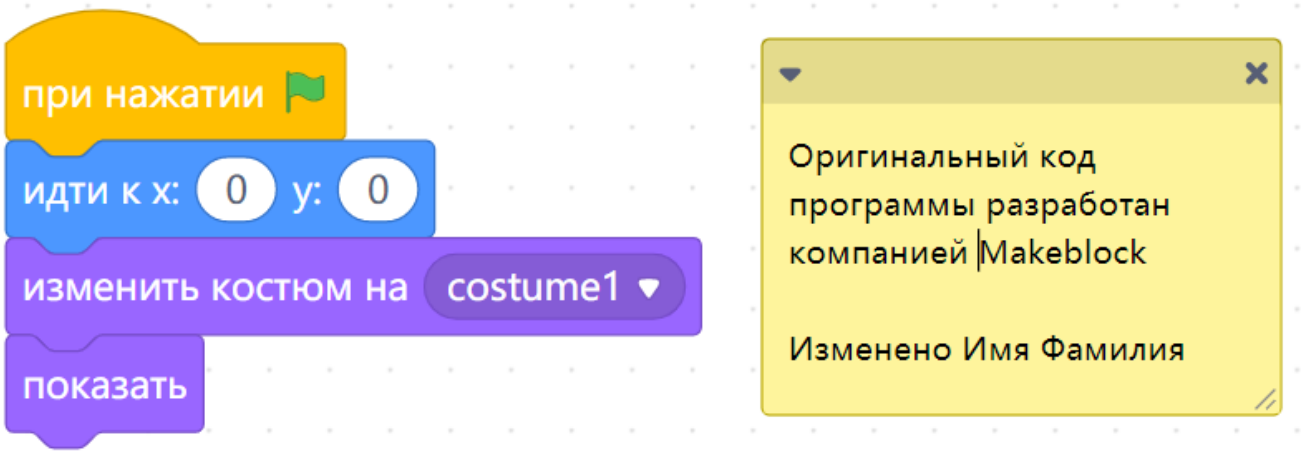
Изменение существующего проекта

7. Щелкните Учебники в правом верхнем углу. Выберите "Примеры программ".
8. Выберите метку Stage, чтобы увидеть примеры программ для этой сцены.
9. При парном программировании попросите учащихся выбрать пример программы и изменить сценарии программы, чтобы добавить игровой контроллер.

Выводы

Уважение интеллектуальной собственности

1. Обсудите важность уважения интеллектуальной собственности и поощрения авторов. Попросите учащихся добавить к своему проекту комментарий, в котором указывается разработчик. Чтобы добавить комментарий, щелкните правой кнопкой мыши на область сценария и выберите «Добавить комментарий».



Дополнительные задания

- Предложите студентам изучить интеллектуальную собственность, авторское право, творческие ресурсы и ссылки.
- Предложите учащимся создать новую игру, включающую сценическое программирование и программирование устройств.

• Попросите учащихся добавить в проект название и инструкции к игре.

12.6. Данные с датчиков

Предмет: Информатика Продолжительность:

45 минут Уровень сложности: Начальный

Цели

К концу этого урока студенты смогут:

- Описать, как датчики CyberPi обнаруживают окружающую среду.
- Отлаживать ошибки в программах в mBlock.
- Документировать программу, используя комментарии в mBlock.

Обзор

Узнайте, как встроенные датчики CyberPi отображают громкость и интенсивность света окружающей среды.Студенты узнают о представлении данных и построении графиков значений датчиков.

Ключевые моменты

- Представление данных
- Отладка программ
- Что такое датчики

Необходимо для урока

Для учителя:

- • Компьютеры с установленным mBlock 5 или веб-версией mBlock
- • CyberPi с кабелем USB-C
- • Pocket Shild (опция)
- • Пример программ, включенные в программное обеспечение mBlock:

  CyberPi – Lesson 6 – Sensor Meter CyberPi – Lesson 6 – Sensor Meter V2

План урока (45 минут)

Тайминг	Контент
5 минут	• Данные и общество
15 минут	• Изучение данных, получаемых от датчиков • Данные от датчика звука
20 минут	• Данные от датчика освещённости
5 минут	• Документация • Дополнительные задания

Введение.

Данные и общество

Найдите информацию об устройствах для умного дома, чтобы её его классу. Обсудите, как в этихрешениях используются датчики для обеспечения безопасности, удобства и автоматизации для потребителей.

Теоретическая часть.

Изучение данных с датчика.

1. Откройте программное обеспечение mBlock 5 или веб-версию mBlock 5. Добавьте CyberPi на вкладке «Устройства» и подключитесь в режиме реального времени.
2. Откройте пример проекта «Lesson 6 - Sensor Meter». Щелкните зеленый флаг, чтобы запустить программу. Обратите внимание на значения CyberPiVolume и CyberPiLightIntensity.
3. Попросите учащихся запустить программу и понаблюдать за показаниями датчиков и графиками в различных сценариях, например:
    - CyberPi на столе
    - Закрытие CyberPi (интенсивность света должна уменьшиться)
    - Включите фонарик на CyberPi (интенсивность света должна увеличиться)

Старайтесь вести себя как можно тише (громкость должна уменьшиться).

- Хлопайте в ладоши или разговаривайте рядом с CyberPi (громкость должна увеличиться)

4. Попросите учащихся определить минимальные и максимальные значения, которые датчики сообщают компьютеру. Обратите внимание, что датчик освещенности и датчик звука имеют диапазон от 0 до 100..
5. Познакомьте учащихся с блоками, которые используются для датчиков CyberPi.:

Категория	Блок	Функции
Сенсоры	яркость окружающего света	Сохраняет числовое значение, представляющее интенсивность света, обнаруженную датчиком освещенности на CyberPi.
	громкость	Сохраняет числовое значение, громкости, обнаруженное звуковым датчиком CyberPi.

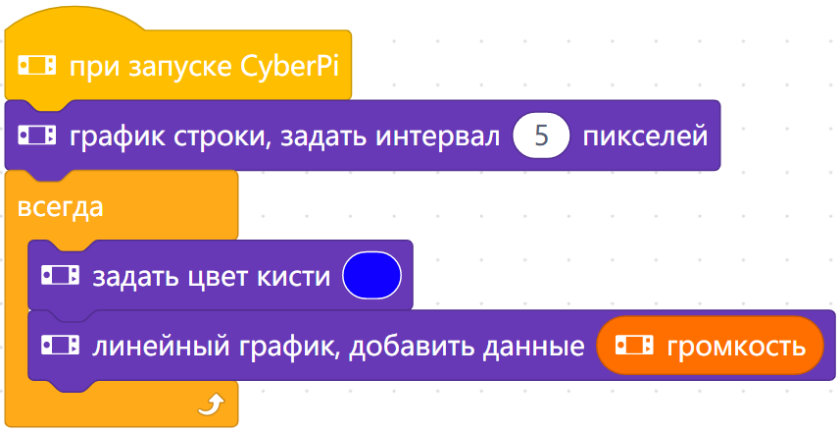
Данные от датчика звука

6. Познакомьте учащихся с блоками, предназначенными для создания линейной диаграммы на CyberPi:

Категория	Блок	Функция
Показать	линейный график, добавить данные 50	Наносит точку на линейной диаграмме на дисплее CyberPi.
	график строки, задать интервал 5 пикселей	Изменяет горизонтальный интервал линейной диаграммы.
	задать цвет кисти	Устанавливает цвет линейного графика.

7. Откройте программное обеспечение mBlock 5 или веб-версию mBlock 5. Добавьте CyberPi на вкладке «Устройства» и подключитесь в режиме реального времени.

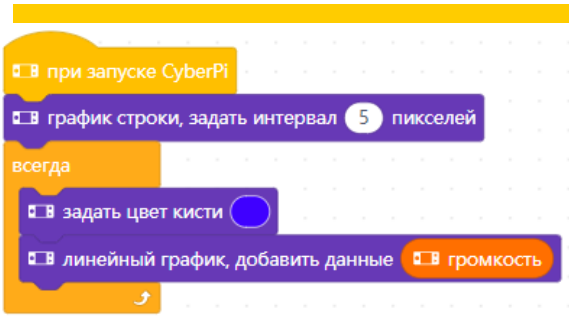
8. Попросите учащихся создать следующий сценарий и протестировать программу:

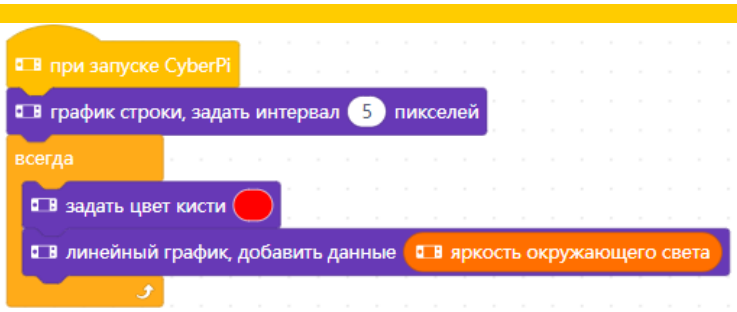


Практическая часть

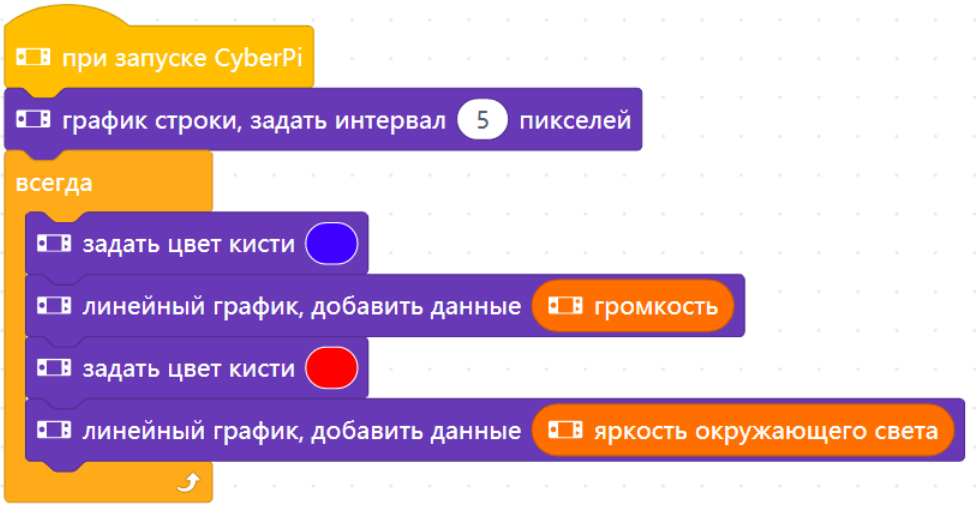
Данные от датчика освещённости

1. Попросите учащихся изменить код, включив в него красную линейную диаграмму для интенсивности света. CyberPi должен отображать обе диаграммы одновременно.
2. Некоторые решения сначала могут показаться правильными. Но поощряйте студентов критически мыслить и полностью проверять свое решение. Например, следующий код неправильно отображает данные. При тестировании этого кода вы должны заметить, что громкий звук рядом с CyberPi приводит к увеличению красной и синей линий.





3. Поощряйте студентов продолжать исправлять и отлаживать свой код, пока они не получат правильное решение. Следующее - правильное решение:



Выводы.

Документация

При программировании может быть полезно документировать программы, чтобы облегчить их отслеживание, тестирование и отладку. Часто над одним проектом работают несколько программистов. Попросите учащихся использовать комментарии, чтобы объяснить, как работает код.

Дополнительные задания

- Попросите учащихся запрограммировать светодиоды или динамик на реакцию на громкость или интенсивность света.
- Попросите учащихся добавить в проект название и инструкции к игре.
- Попросите учащихся запрограммировать спрайт или фон для смены костюмов в зависимости от громкости или интенсивности света. (Обратите внимание, что для хранения каждого значения датчика требуется переменная).

12.7. Цветовой микшер.

Предмет: Информатика

Продолжительность: 45 минут Уровень

сложности: Начальный

Цели

К концу этого урока студенты смогут:

- Использовать переменные для хранения значений.
- Изменять переменную на основе ввода данных пользователем.
- Написать программу в mBlock, которая выполняется, если выполняется условие.

Обзор

Студенты познакомятся с переменными для создания цветового микшера CyberPi. Эта программа будет использовать джойстик и кнопки для управления значениями цвета R, G, B всех встроенных светодиодов.Также, учащиеся будут использовать условные выражения, чтобы гарантировать, что значения R, G, B не выходят за пределы допустимого диапазона.

Ключевые моменты

- Хранение данных с переменными
- Использование условных операторов

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- CyberPi с кабелем USB-C
- Pocket Shild (опция)
- Пример программы, включенный в программное обеспечение mBlock:

CyberPi – Lesson 7 – Color Mixer

План урока (45 минут)

Тайминг	Контент
5 минут	• Игра с условными операторами
15 минут	• Хранение данных с переменной • Использование условного оператора
20 минут	• Завершение программы
5 минут	• Переменные • Дополнительные задания

Введение

Игра с условными операторами

8. Сыграйте со своим классом в игру, похожую на «Саймон говорит», чтобы продемонстрироватьусловные утверждения. Вместе со своими учениками прочтите каждое утверждение ниже и
попросите учеников подчиниться команде. (Не стесняйтесь писать свои собственные команды длясвоих учеников.)
    - ЕСЛИ в вашем имени есть буква «S», ЗАТЕМ поднимите руку.
    - ЕСЛИ у вас есть домашняя кошка, ТОГДА хлопните в ладоши.
    - ЕСЛИ вы занимаетесь спортом, ТОГДА топайте ногами.
    - ЕСЛИ на вас носки, ТОГДА прикоснитесь к ногам.
    - ЕСЛИ ваше любимое мороженое шоколадное, ТОГДА скажите «Ням».

9. Расскажите учащимся, что эти команды являются примерами условных выражений. В программировании условные операторы используются для выполнения определенных действий,если условие истинно.

Теоретическая часть.

7. Обсудите со студентами следующее определение:

Термин	**Определение**
Переменная	Величина, которая может изменяться и иметь множестворазличных значений.

8. Повторите со студентами следующую задачу и алгоритм:

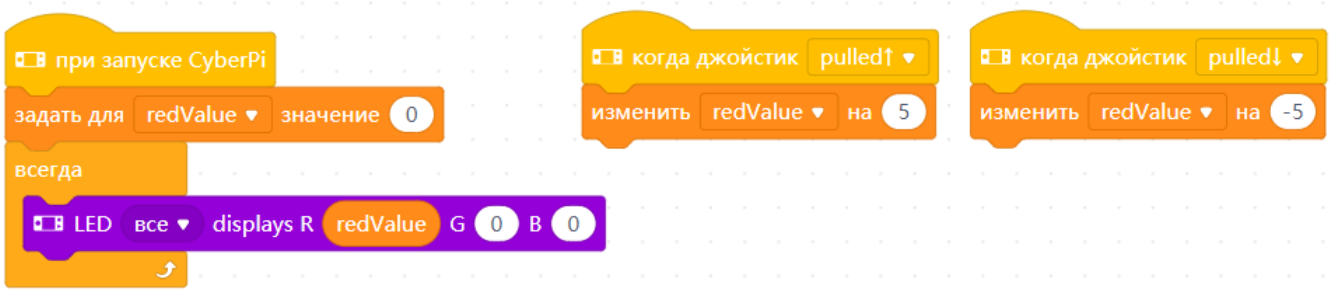
Цветовой микшер	
Описание Проекта	Создайте проект, в котором джойстик и кнопки CyberPi управляют значениями R, G, B всех светодиодов
Алгоритм	Когда джойстик поднят: Увеличьте значение красного (R) всех светодиодов на 5 Когда джойстик опущен: Уменьшите значение красного (R) всех светодиодов на 5 Когда джойстик вытянут вправо: Увеличьте значение зеленого (G) всех светодиодов на 5 Когда джойстик отведен влево:

	Уменьшите значение зеленого (G) всех светодиодов на 5 Когда нажата кнопка A: Увеличьте значение синего (B) всех светодиодов на 5 Когда нажата кнопка B: Уменьшите значение синего (B) всех светодиодов на 5

9. Чтобы запрограммировать этот проект, необходимо использовать переменную для хранения значения каждого из светодиодов. При запуске CyberPi переменные будут начинаться с нуля (0) ибудут регулироваться с помощью джойстика или кнопок.
10. Откройте программное обеспечение mBlock 5 или веб-версию mBlock 5. Добавьте CyberPi на вкладке «Устройства» и подключитесь в режиме реального времени.
11. Перейдите в раздел «Переменные» в области блокировки. Нажмите кнопку "Создать переменную".
12. Назовите новую переменную redValue и оставьте выделенной для всех спрайтов.
13. Познакомьте учащихся со следующими новыми блоками, которые теперь доступны в разделе «Переменные»:

Категория	Блок	Функции
Переменные	задать для redValue значение 0	Устанавливает для переменной определенное значение.
	изменить redValue на 1	Изменяет переменную на определенное значение (положительное или отрицательное).
	redValue	Возвращает текущее значение, хранящееся в переменной.
Операторы	< 0	Сравнивает два значения и определяет, меньше ли первое значение второго. Возвращает ИСТИНА или ЛОЖЬ.
	> 255	Сравнивает два значения и определяет, больше ли первое значение второго. Возвращает ИСТИНА или ЛОЖЬ.

14. Попросите учащихся создать следующие сценарии и протестировать программу, чтобы наблюдать, как элементы управления джойстиком вверх / вниз увеличивают красное значение всех светодиодов:



Использование условного оператора

15. Напомните учащимся, что значения R, G, B имеют диапазон от 0 до 255. В приведенной выше программе переменная redValue может быть изменена на значение, выходящее за пределы этого диапазона. Мы можем использовать условные операторы для управления минимальным и максимальным значениями. Попросите учащихся изменить код и добавить в программу следующиеусловные операторы:

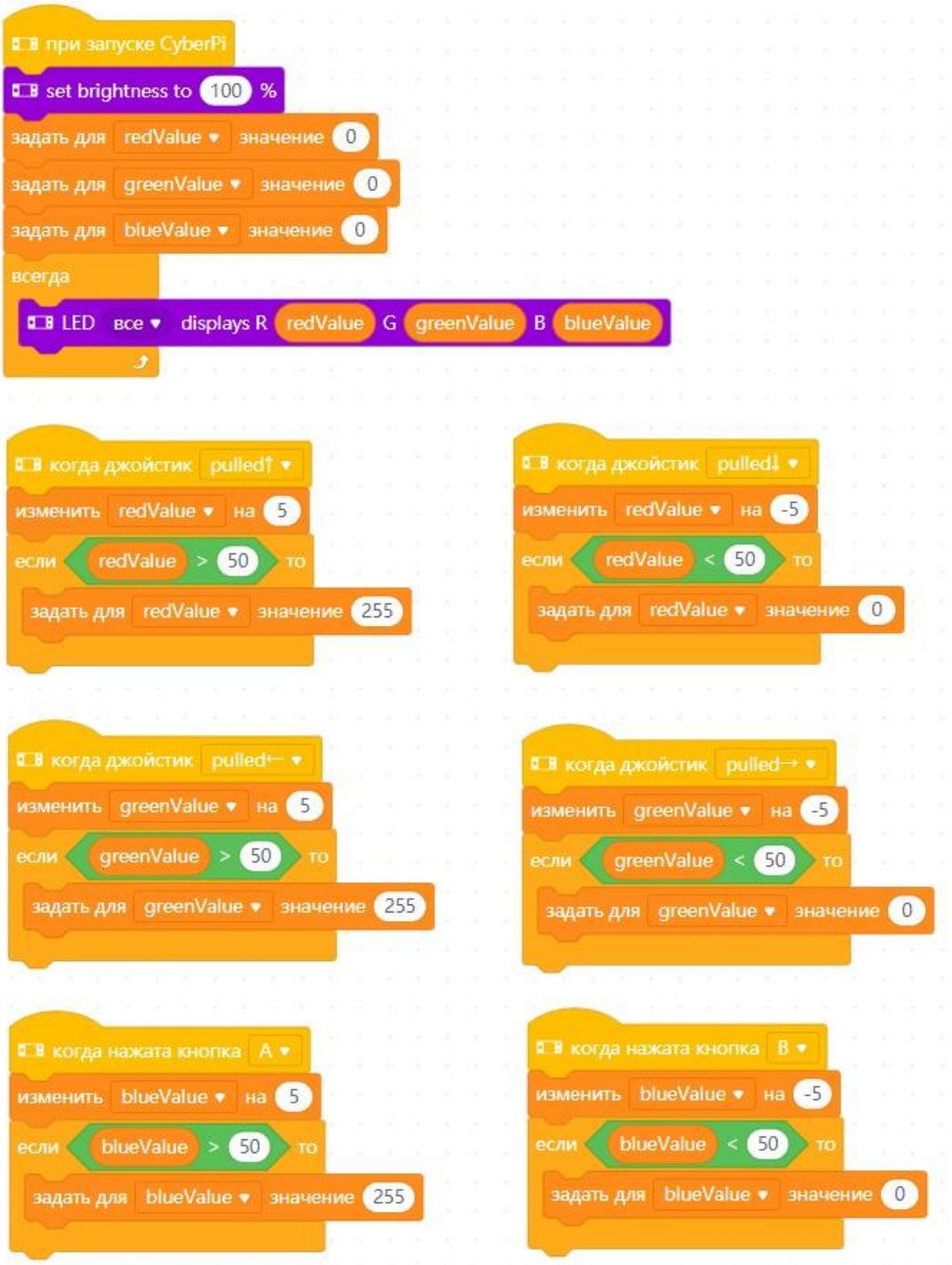
Обратите внимание: учащиеся также могут добавить блок, чтобы установить яркость на 100% при запуске программы.

16. Объясните учащимся, как условные выражения не позволяют redValue когда-либо быть больше 255или меньше 0.

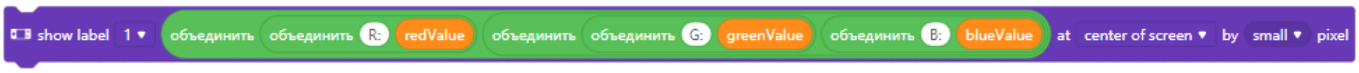
Практическая часть

Завершение программы

1. Попросите учащихся завершить программу на основе алгоритма, приведенного ранее в уроке. Нижеприведен пример завершенной программы:



2. Учащимся может быть полезно добавить следующий скрипт внутри цикла forever, чтобы увидетьзначения переменных на дисплее CyberPi:



Выводы.

Применение переменных в устройствах.

Обсудите со студентами, как переменные используются для хранения информации в различных вычислительных устройствах и приложениях. Обсудите со студентами следующие примеры:

- Фитнес-трекеры хранят количество шагов.
- Автомобили (с цифровым дисплеем) хранят количество пройденных миль.
- Мобильные устройства хранят уровень заряда батареи и сообщают его в процентах.
- На карточках магазинов хранится количество посещений до тех пор, пока вы не получите вознаграждение.
- Видеоигры хранят здоровье персонажа, уровень жизни и результаты.

Дополнительные задания.

- Попросите учащихся добавить название и инструкции к проекту.

Попросите учащихся создать переменную для LEDNumber и использовать среднюю кнопку джойстика для управления изменяемым светодиодом.

12.8. Измерение силы встряски.

Предмет: Информатика

Продолжительность: 45 минут

Уровень сложности: Начальный

Цели

К концу этого урока студенты смогут:

- Написать программу в mBlock, которая ведет счет.
- Написать программу, которая выполняет действия в течение определенного промежутка времени.
- Отображать текст на дисплее Cyberpi.

Обзор

На этом уроке ученики создадут забавную игру с CyberPi, в которой игрок встряхивает CyberPi в течение десяти секунд. Учащиеся запрограммируют игру, чтобы подсчитать, во сколько раз сила сотрясения превышает 50.

Ключевые моменты

- Ведение счета
- Использование таймера CyberPi
- Отображение текста на дисплее CyberPi

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- CyberPi с кабелем USB-C
- Pocket Shild (опция)
- Пример программы, включенный в программное обеспечение mBlock:

  Lesson 8 – Strength Meter

План урока (45 минут)

Тайминг	Контент
5 минут	• Обзор переменных
15 минут	• Определение силы встряски • Сохранение очков
20 минут	• Создание измерителя силы встряски 2.0
5 минут	• Датчик движения • Дополнительные задания

Введение.

Обзор переменных

Просмотрите переменные с классом и объясните, как можно использовать переменные для отслеживания результатов в проекте mBlock. Обсудите со студентами, как создать переменную, установить переменную и изменить переменную в mBlock.

Теоретическая часть.

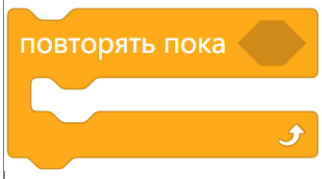
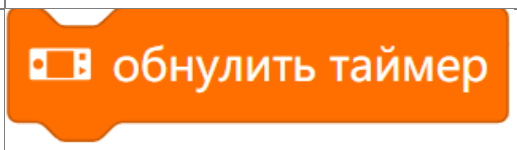
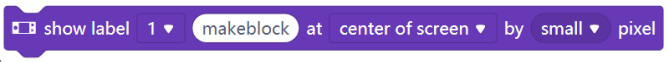
Обнаружение встряски

1. Повторите со студентами следующую задачу и алгоритм:

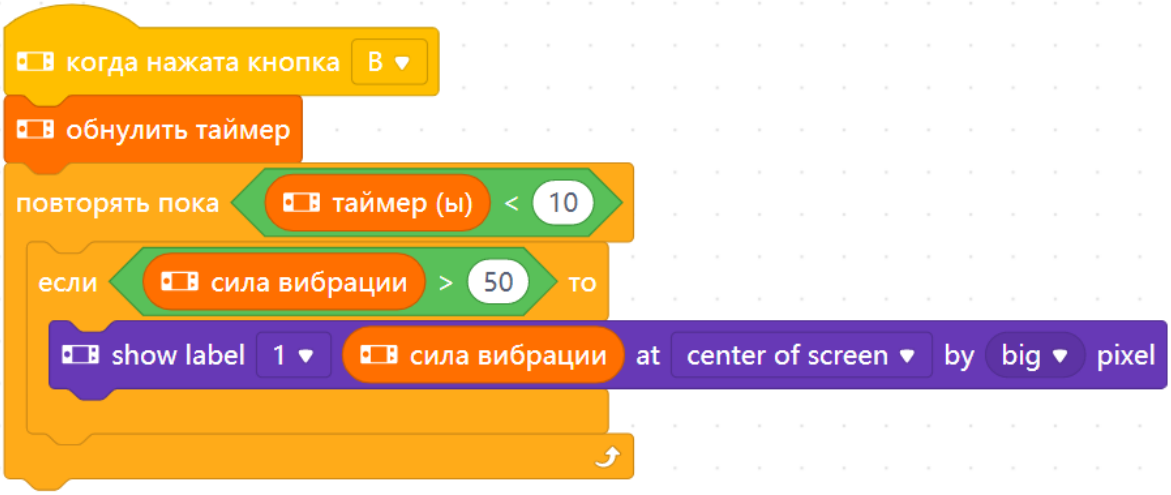
Измеритель силы встряски 1.0	
Описание проекта	Создайте игру, в которой пользователь встряхивает CyberPi в течение 10 секунд и зарабатывает балл за каждый раз, когда сила встряхивания превышает 50.

Алгоритм	Когда нажата кнопка B: Установить счет на ноль (0)Сбросить таймер Повторяйте встряхивание в течение 10 секунд. Если сила встряхивания больше 50, тогда изменить счет на один (1) Когда нажата кнопка A: Показать результат на дисплее CyberPi

2. Откройте программное обеспечение mBlock 5 или веб-версию mBlock 5. Добавьте CyberPi во вкладку «Устройства» и подключитесь в режиме реального времени.
3. Познакомьте учащихся со следующими новыми блоками:

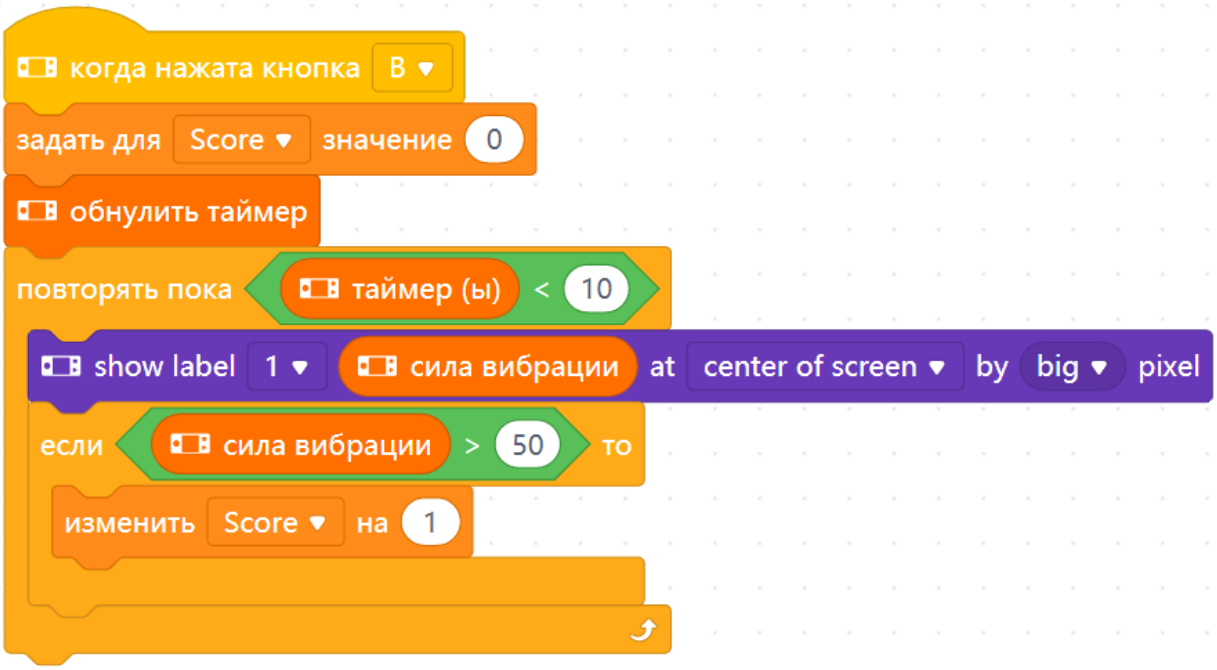
Category	Block	Function
Управление		Условный цикл Выполнять действия, вложенные в блок, покаусловие не станет равно ИСТИНА.
Сенсоры		Установить таймер CyberPi вноль (0).
Настройка движения		Возвращает значение, показывающее, насколько сильно CyberPi встряхивается.
Показать		Отображает текст на дисплее CyberPi в указанном положениии размере.

4. Попросите учащихся составить следующий сценарий и протестировать программу, чтобы увидеть значения силы тряски.:

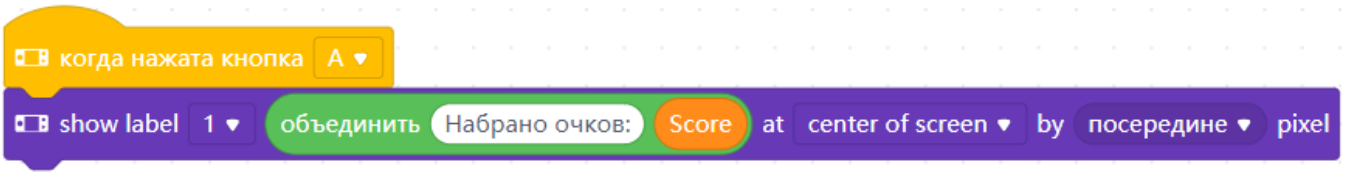


Ведение счёта

5. Создайте переменную с именем score и оставьте выбранной для всех спрайтов.
6. Попросите учащихся изменить программу следующим образом:



7. Теперь, когда CyberPi отслеживает счет, добавьте следующие скрипты для отображения счета на экране при нажатии кнопки A:



Предложите учащимся поиграть в игру и посмотреть, сколько очков они могут заработать.

Практическая часть

Создание Измерителя силы встряски 2.0

3. Создав простую версию игры Strength Meter, в программу можно добавить множество улучшений.Попросите учащихся определить области, в которых можно улучшить этот дизайн.
   Некоторые идеи по улучшению:
   - Используйте светодиодную ленту, чтобы сообщить пользователю, когда игра запущена.
   - Добавлять звуковой эффект каждый раз, когда зарабатывается очко.
   - Добавить звуковой эффект, когда время истекает.
   - Добавьте заголовок и инструкции.
   - Используйте джойстик и переменную, чтобы изменить сложность игры (например, сила сотрясения для легкого составляет 30, среднего - 50, а жесткого - 70).
4. Попросите учащихся разработать план и алгоритм улучшений, которые они хотели бы добавить всвой проект.

Попросите учащихся создать Измеритель силы встряски 2.0, используя свой псевдокод в качестве руководства.

Выводы.

Датчик движения

CyberPi имеет 3-осевой гироскоп и 3-осевой акселерометр, который определяет движение, ускорение и вибрацию. Блок силы тряски использует этот компонент для определения силы встряхивания CyberPi. Предложите учащимся составить список устройств, которые они используют, в которых используется гироскоп.
Некоторые примеры:

- Мобильные телефоны меняют ориентацию экрана в зависимости от поворота устройства.
- Экраны мобильных телефонов загораются, когда устройство берется в руки.
- Контроллеры видеоигр обнаруживают движение.
- Роботизированные пылесосы обнаруживают, если они упали или опрокинулись.

Дополнительные задания.

- Попросите учащихся запрограммировать светодиоды так, чтобы они загорались постепенно взависимости от силы света.

Попросите учащихся запрограммировать игру для двух игроков, в которой отслеживается счетобоих игроков, сравниваются результаты и объявляется победитель.

12.9. Подарок с сигнализацией

Предмет: Информатика

Продолжительность: 45 минут

Уровень сложности: Начальный

★ Цели

К концу этого урока студенты смогут:

- Использовать технологию беспроводной сети для связи между вычислительными устройствами.
- Решить проблему с помощью вычислительного решения.
- Определить способы использования беспроводной связи.

Обзор

Студенты будут использовать CyberPi для создания программы, которая определяет, встряхнул ли друг их подарок на день рождения. Используя беспроводную связь, студенты будут отправлять сообщения междувычислительными устройствами, позволяя одному устройству управлять другим.

Ключевые моменты

- Использование беспроводных сетей
- Связь между устройствами

Необходимо для урока

- Компьютеры с установленным mBlock 5 или веб-версией mBlock
- Два CyberPi с кабелем USB-C или один CyberPi и один Halocode
- Pocket Shild (опция)
- Пример программы, включенный в программное обеспечение mBlock:
  CyberPi – Lesson 9 – Gift Alarm

План урока (45 минут)

Тайминг	Контент
5 минут	• Интернет вещей
15 минут	• Беспроводные коммуникации в mBlock
20 минут	• Создание Подарка с сигнализацией
5 минут	• Идеи для мозгового штурма • Дополнительные задания

Введение

Интернет вещей

Проведите небольшое исследование в Интернете вещей. Предложите студентам исследовать и представить свои выводы. Их исследование должно привести их к тому, чтобы узнать больше о том, как подключенные к Интернету физические вычислительные устройства используются в следующих областях:

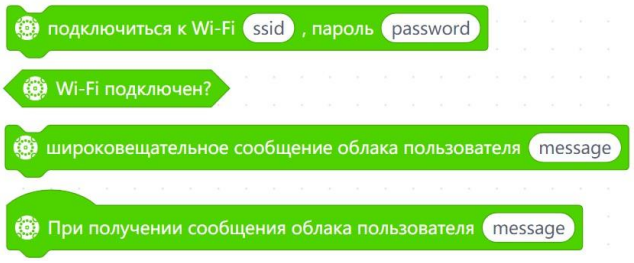
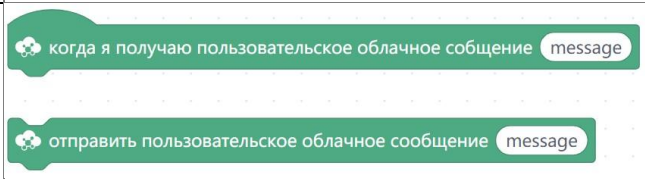
a. Умный дом
b. Мониторинг здравоохранения
c. Транспорт
d. Сельское хозяйство
e. Погода
f. Производство
g. Экологический исследования
h. Армия

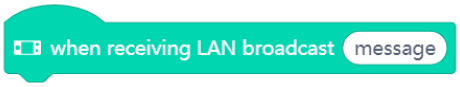
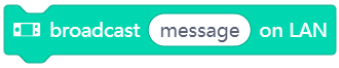
Теоретическая часть

Беспроводная связь в mBlock

В mBlock есть два типа беспроводной связи: Wi-Fi (с выходов в интернет) и LAN (локальное беспроводное соединение). Просмотрите следующую информацию и блоки для каждого типа.

Wi-Fi (с подключением к интернету)		
Используя соединение Wi-Fi, данные передаются с помощью функции облачных сообщений. Вы можете обмениваться данными между устройствами и проектами с одной и той же учетной записью вmBlock 5. Физическая близость или расстояние больше не являются ограничением, поскольку эти устройства не обязательно должны находиться в одном месте. Для использования каждое устройство должно быть подключено к Интернету. См. Блоки ниже дляподключения CyberPi, Halocode и проекта mBlock к облачным сообщениям.		
Категория	**Блоки**	**Функции**
Интернет вещей	подключиться к Wi-Fi ssid , пароль password	Подключает CyberPi к беспроводной сети.
	сеть подключена?	Возвращает TRUE, если CyberPiподключен к интернету.
	when receiving user cloud broadcast message	Блок событий, запускающий выполнение прикрепленных действий при получении определенного облачногосообщения.
	send user cloud broadcast message	Отправляет облачное сообщения.

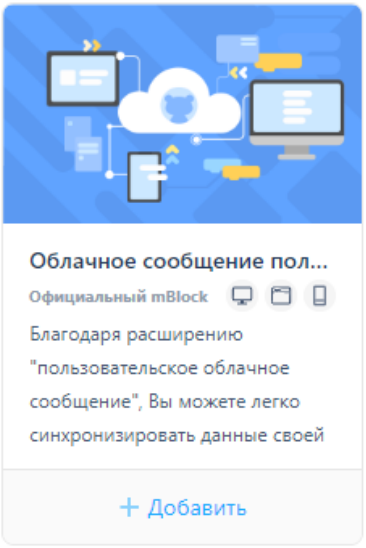
CyberPi может взаимодействовать со спрайтом Halocode или mBlock через облачные сообщения,передаваемые по Wi-Fi.		
Halocode  Wi-Fi		См. Описание блоков CyberPiвыше.
Спрайт  Облачное Сообщение		См. Описание блоков CyberPiвыше.

LAN (локальная беспроводная сеть)		
ЛВС (локальная сеть) - это сеть, которая связывает группу компьютеров или устройств в определенном месте. Группа компьютеров обменивается данными для отправки сообщений друг другу. Между CyberPi может быть сформирована локальная сеть, чтобы один CyberPi могуправлять другим.		
Category	**Block**	**Function**
Локальные сети		Блок событий, который запускает выполнение действий, связанных с получением конкретной широковещательной передачи по локальной сети.
		Отправляет сообщение по локальной сети.

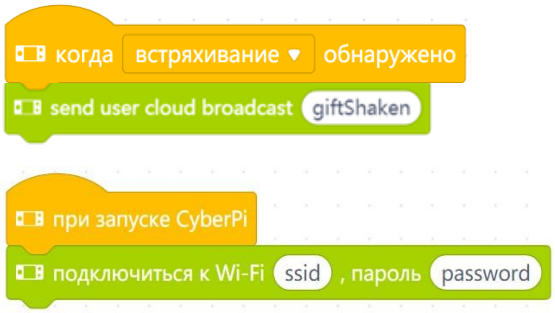
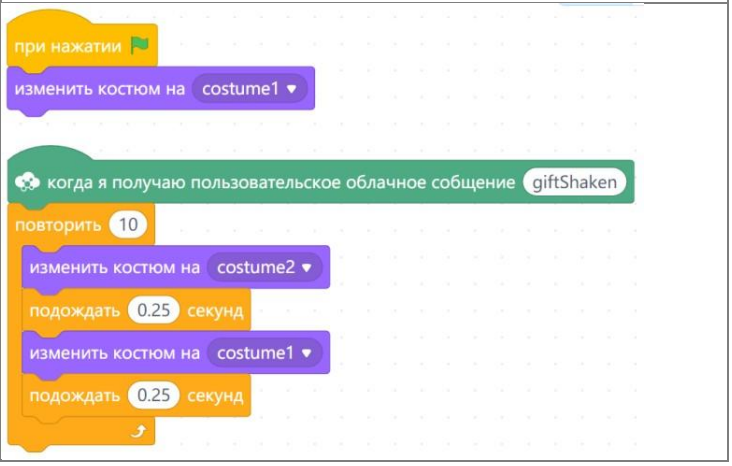
2. Выберите тип общения, который будет работать с вашим классом и учениками. Используйте следующие шаги в качестве примера того, как работает беспроводная связь в mBlock..

3. Откройте программное обеспечение mBlock 5 или веб-версию mBlock 5. Добавьте CyberPi на вкладке «Устройства» и подключитесь в режиме загрузки.

4. На спрайте панды в **области блоков**, снизу нажмите[расширения] . Найдите расширение **Облачное сообщения пользователя** и нажмите **+Добавить**.



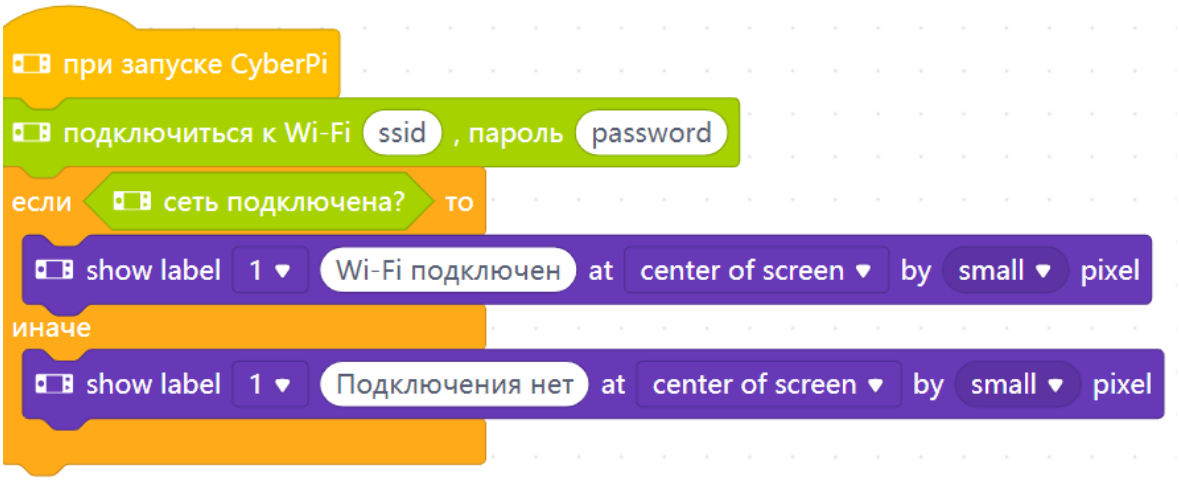
5. Попросите учащихся создать следующие сценарии:

CyberPi	Спрайт
	

6. Обновите SSID и пароль Wi-Fi, указав информацию о беспроводном маршрутизаторе, чтобы CyberPi мог подключиться.

7. Загрузите программу в CyberPi и протестируйте ее. Учащиеся должны наблюдать за шаганием панды при встряхивании CyberPi.

Если программа не работает, добавьте следующий код в скрипт CyberPi для устранения неполадок соединения Wi-Fi.



Практическая часть.

Создание Подарка с сигнализацией

1. Используя следующую постановку задачи, попросите учащихся спланировать решение проблемы. Есть много способов решить эту проблему. Вот один из вариантов:

Подарок с сигнализацией	
Проблема	Скоро день рождения твоего друга. Этот друг любит трясти подарками, чтобы понять, что внутри. Вы хотите придумать, как сделать бесшумный сигнал тревоги, который уведомит вас по беспроводной сети, если подарок встряхнут. Как mBlock, CyberPi и / или Halocode могут помочь вам определить,трясет ли ваш друг подарок, который вы дарите ему в этом году?
Предполагаемое решение	Запрограммируйте Halocode на отправку облачного сообщения, еслион встряхивается. Перед упаковкой закрепите Halocode и батарейный блок внутри подарка. Запрограммируйте CyberPi так, чтобы он воспроизводил звук имигал светодиодами, если получено облачное сообщение, указывающее, что подарок был потрясен. Бонус: спрайт в mBlock тоже трясётся, когда подарок встряхивают.

2. Попросите учащихся написать алгоритм для Подарка с сигнализацией, а затем создать проект, используя свой алгоритм в качестве руководства.

Выводы.

Идеи для мозгового штурма

Беспроводная связь и облачные сообщения устраняют такие препятствия, как длина кабеля и местоположение устройства. Вместе с классом придумайте список идей для программ, которым может

помочь беспроводная связь.

Некоторые идеи:

- Метеостанция с отчетом на отдельное устройство.
- Устройство сбора данных опросов в главном здании школы, которое сообщает результаты в класс.
- Рация или обмен текстовыми сообщениями между устройствами.

Дополнительные задания

- Предложите учащимся разработать собственный проект с использованием беспроводной связи.

Предложите учащимся изучить сети и их роль в обществе.

13. Робототехника со множеством контроллеров.

13.1. Свободное падение тел

Методические рекомендации.

Тема: «Вычисление ускорения свободного падения. Построение графиков»

Цель: изучить процесс создания и программирования робота для изучения гравитационного взаимодействия.

Задачи:

- изучить и закрепить на практике процесс создания робота по изучению гравитации

- изучить особенности управления роботом данной конструкции

- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий.

Задание 1 (Уровень А)

Соберите устройство согласно инструкции, представленной ниже

Задание 2 (Уровень В)

Создайте две программы, с помощью которых можно управлять и собирать данные для построения графика в режиме реального времени.

Пример решения.

- Используя знания по созданию программ для управления сервомотором и датчиком касания робота, а также получения данных с гироскопа CyberPi и работой с его дисплеем, получим алгоритм для проведения исследования. В программе будет задействован алгоритм, который состоит из двух параллельно работающих частей. Первая часть связана с устройством, которое запускает в свободное падение тело. В нашем случае – это CyberPi. Вторая часть – это считывание значений с гироскопа и

построение графика на дисплее CyberPi.

Задание 1

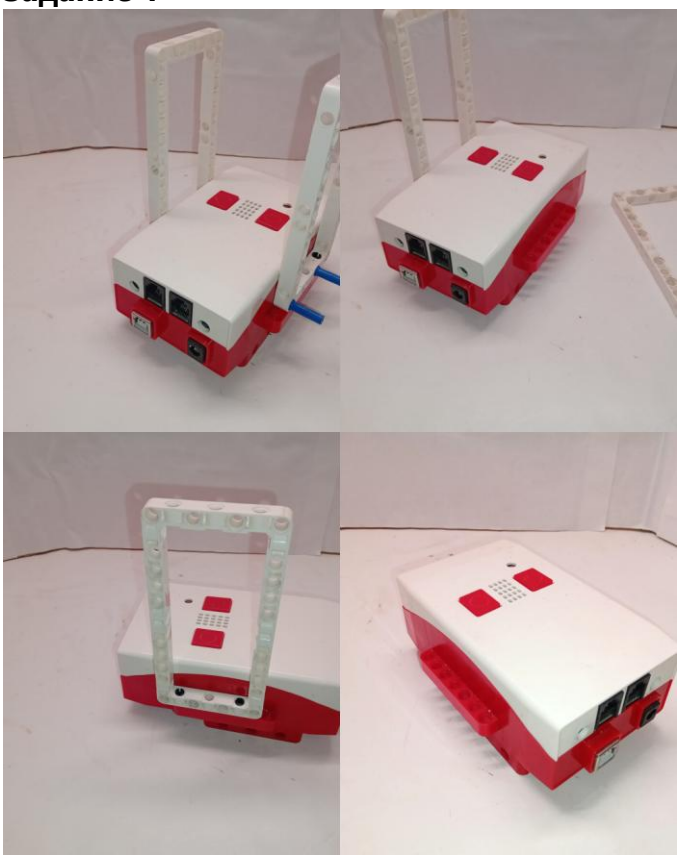


Рисунок 1. Гравитация

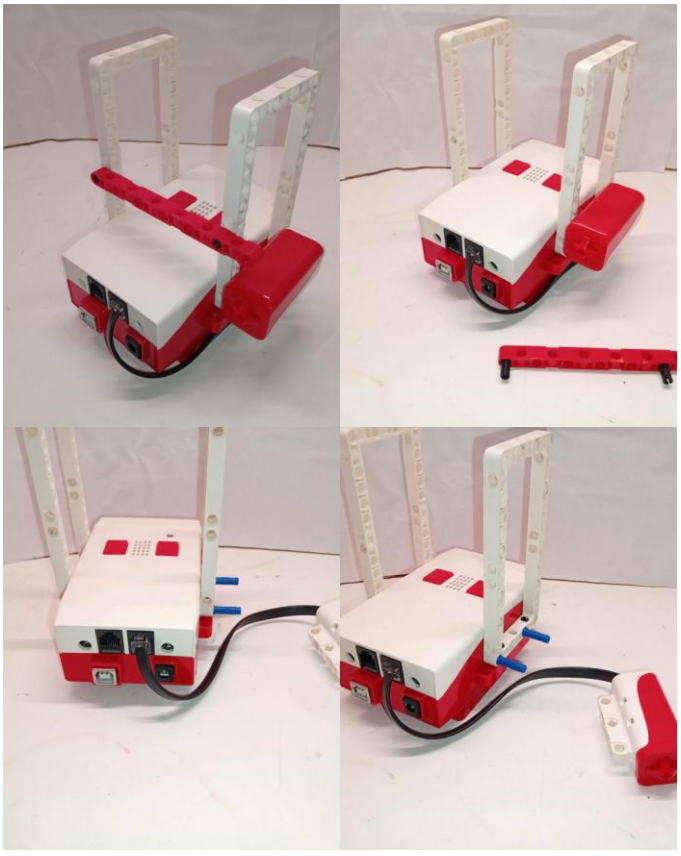


Рисунок 2. Гравитация

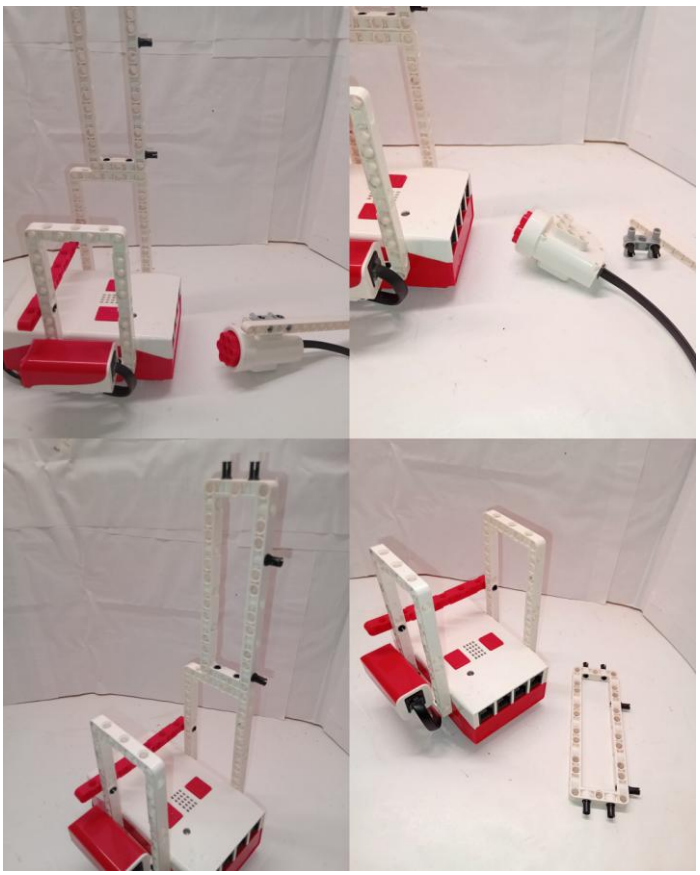


Рисунок 3. Гравитация

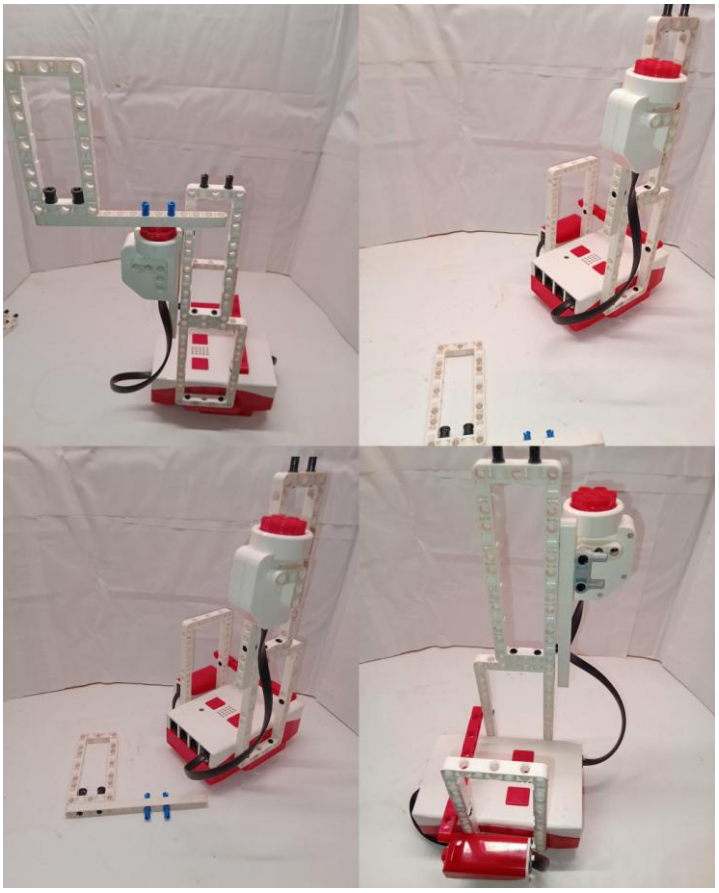


Рисунок 4. Гравитация

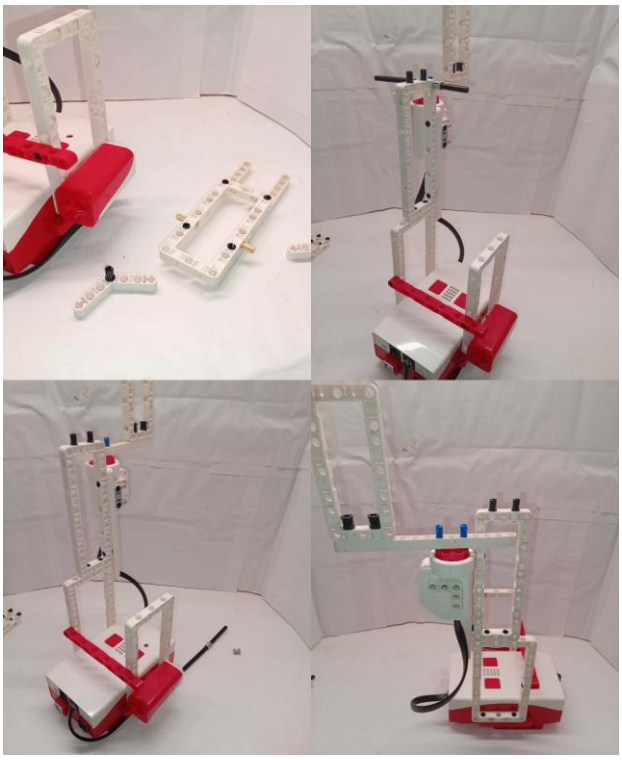


Рисунок 5. Гравитация

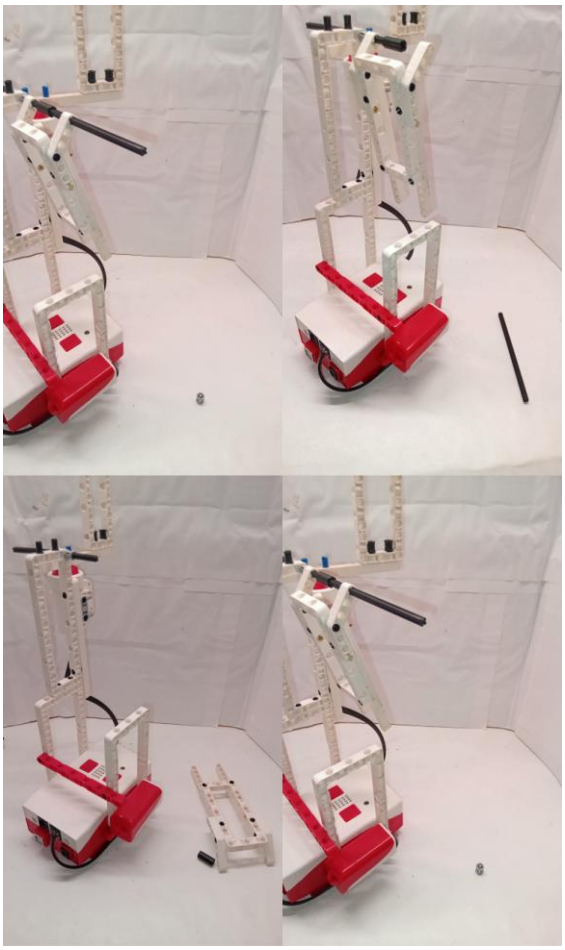


Рисунок 6. Гравитация

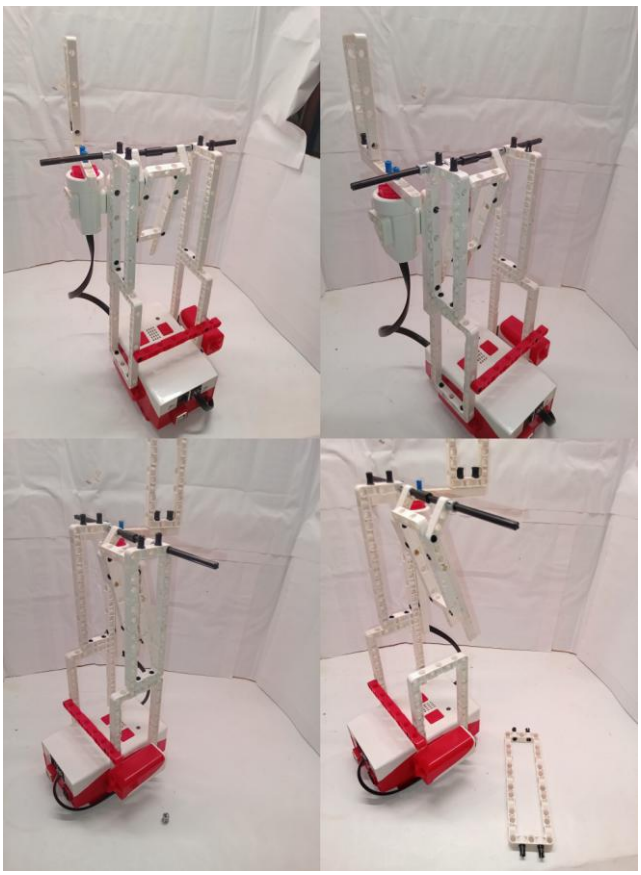


Рисунок 7. Гравитация

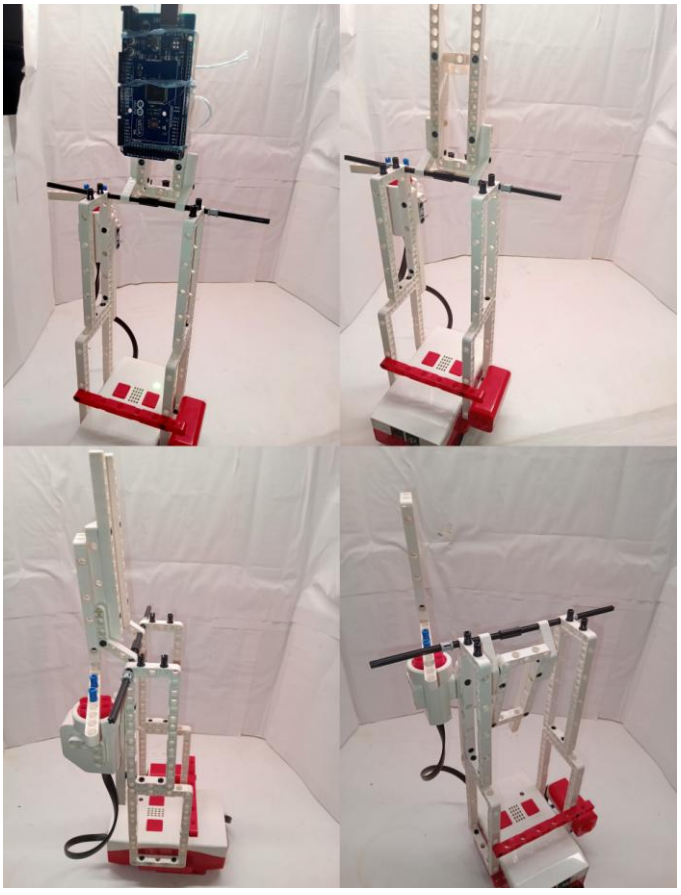
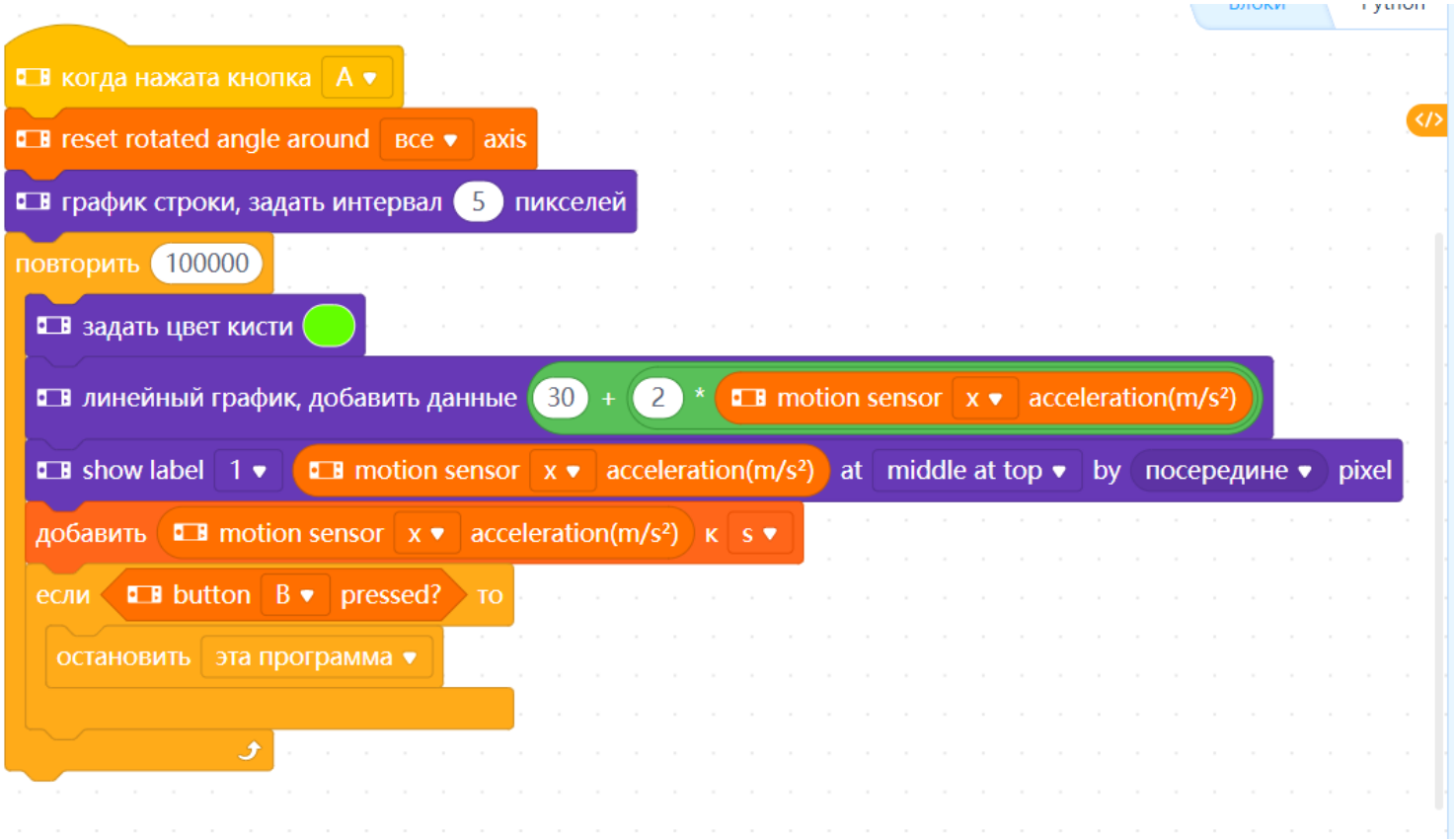


Рисунок 8. Гравитация

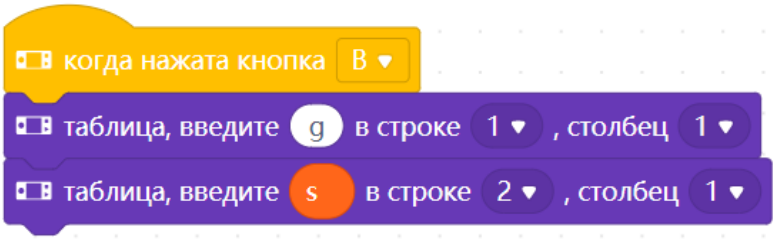
Задание 2.

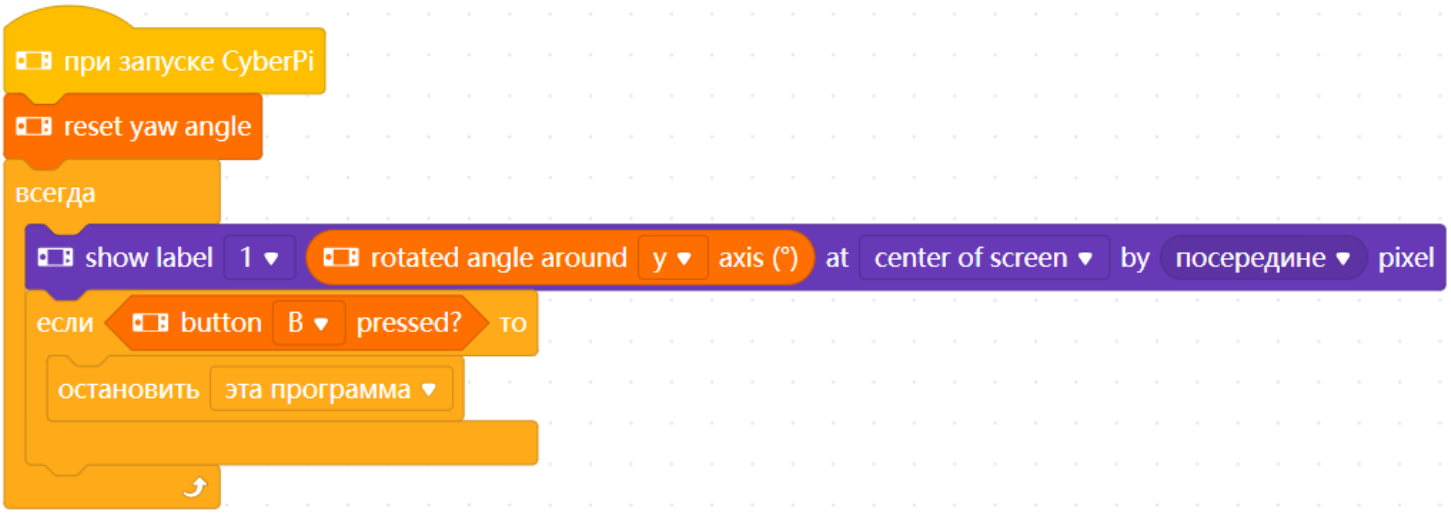
Создадим программу для CyberPi. Она должна получать данные с гироскопа и вычислять ускорение по осям. Данные значения будет откладывать на дисплее и строить график.



Как видно из рисунка, процес снятия показаний начнётся по нажатию кнопки A на панели CyberPi. Параллельно показаниям будет строится график зависимости ускорения и времени. Ускорение CyberPi рассматривается по оси X.

Данное ускорение будет зависеть от угла наклона гироскопа. Чтобы это проверить добавим ещё две программы. Одна будет показывать угол наклона CyberPi на нашей установке, а другая при нажатии B выведет значения ускорения, которые снял гироскоп.





Как только процесс запуска программы на CyberPi начался, то необходимо запустить устройство. Задача программы - отпустить держатель подвижной части, где крепится CyberPi. Держатель снимается с помощью сервопривода, подключённого к шестому порту и датчика касания, подключённого ко второму порту.

13.2. Вычисление угловой и линейной скоростей вращающегося тела.

Методические рекомендации.
Тема «Робокарусель. Управления с помощью двух датчиков»
Цель: Отработка навыков программирования и конструирования.
Задачи:
- получение и закрепление знаний умений и навыков в области конструировании простых механизмов с применение разных типов соединения и передач.
- закрепление работы с датчиком касания и ir модулем.
Результат занятия: автоматизированные карусель с различным видом управления.
Время урока: 90 мин.

Методические рекомендации.

Тема: «Вычисление угловой и линейной скоростей вращающегося тела»

Цель: изучить процесс создания и программирования робота для изучения криволинейного движения.

Задачи:

- изучить и закрепить на практике процесс создания робота по изучению криволинейного движения
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий.

Задание 1 (Уровень А)

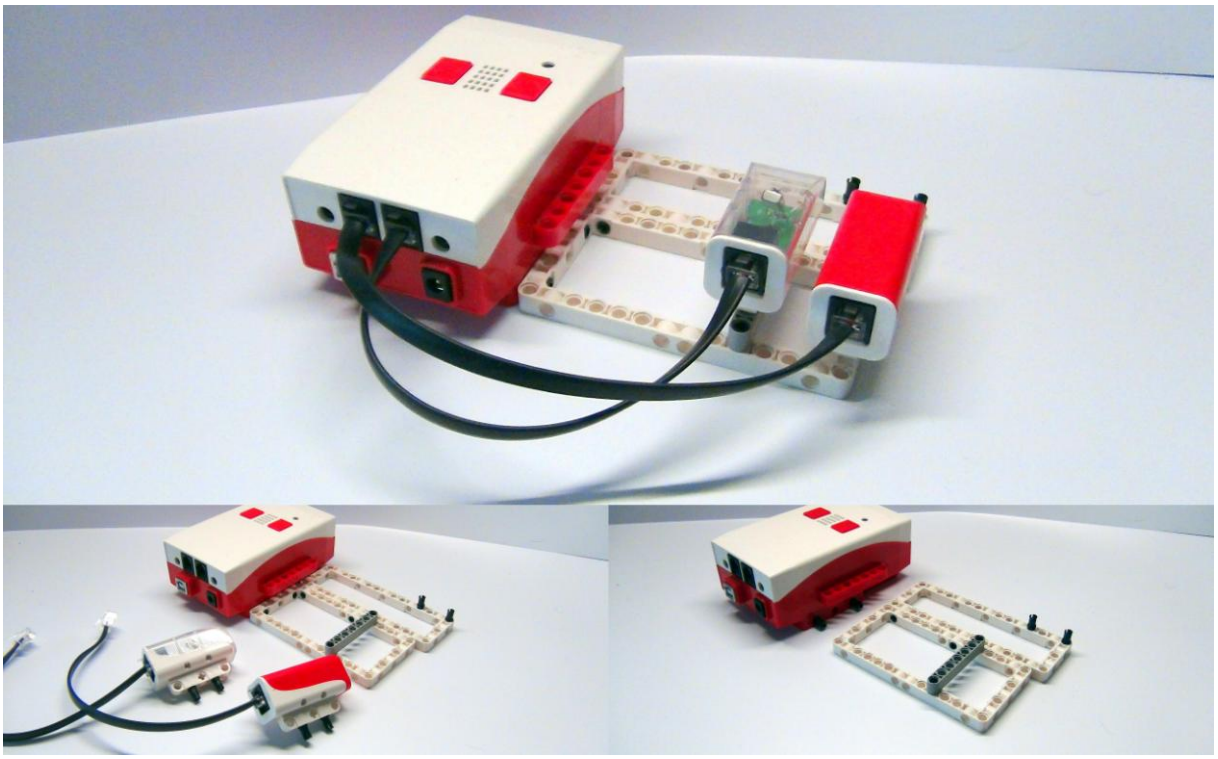
Соберите устройство согласно инструкции, представленной ниже

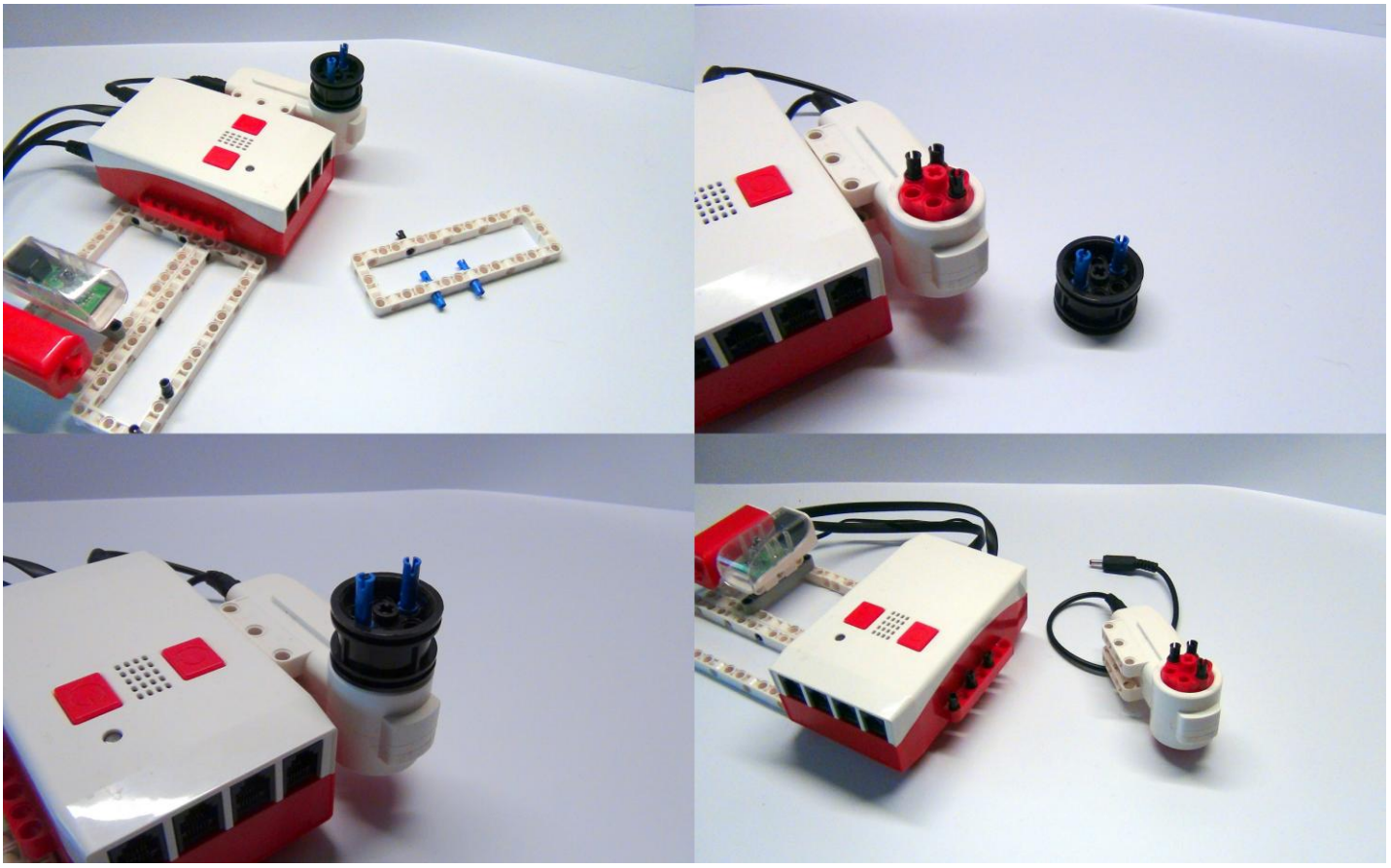
Задание 2 (Уровень В)

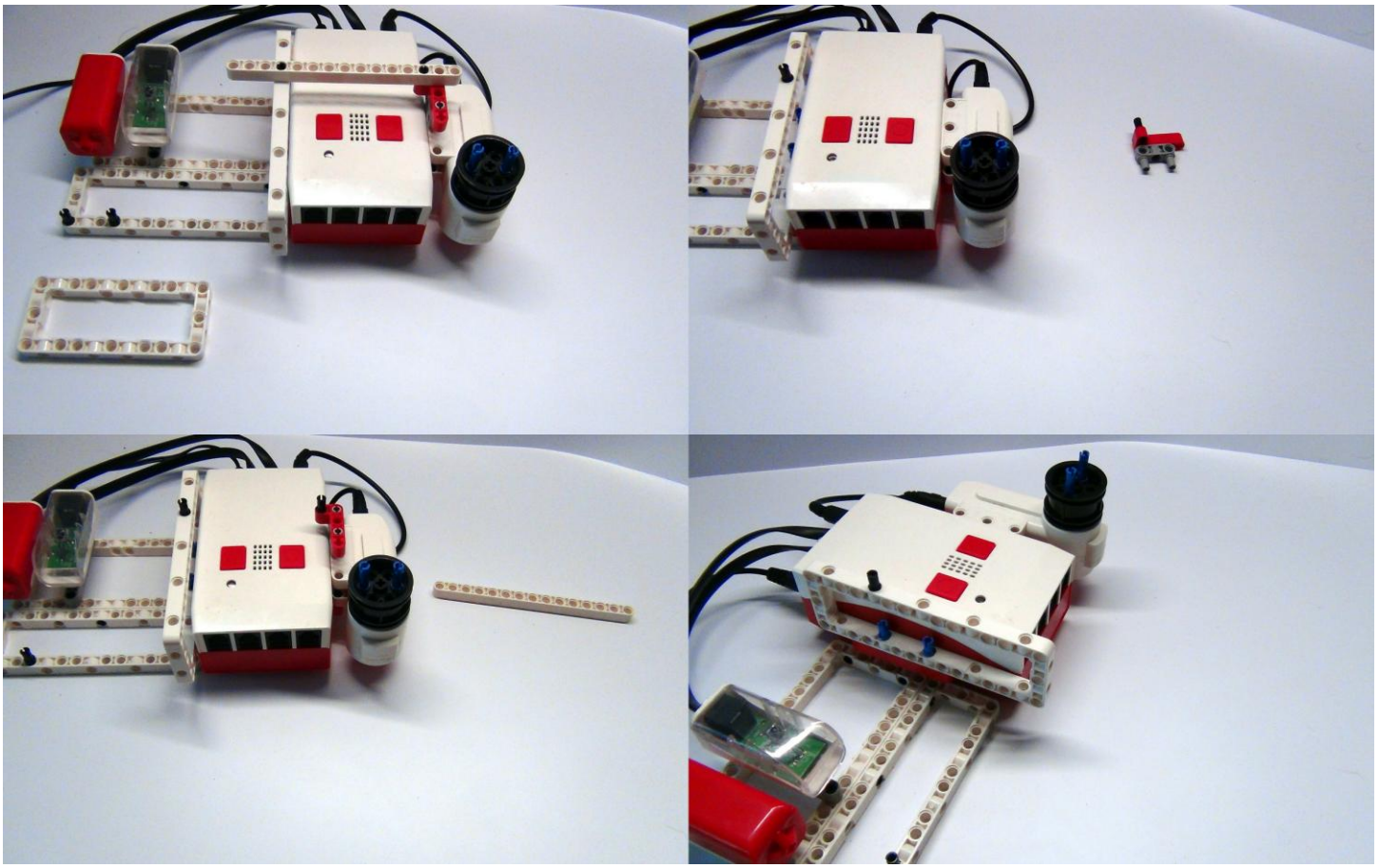
Создайте две программы, с помощью которых можно управлять и собирать данные для построения графика в режиме реального времени.

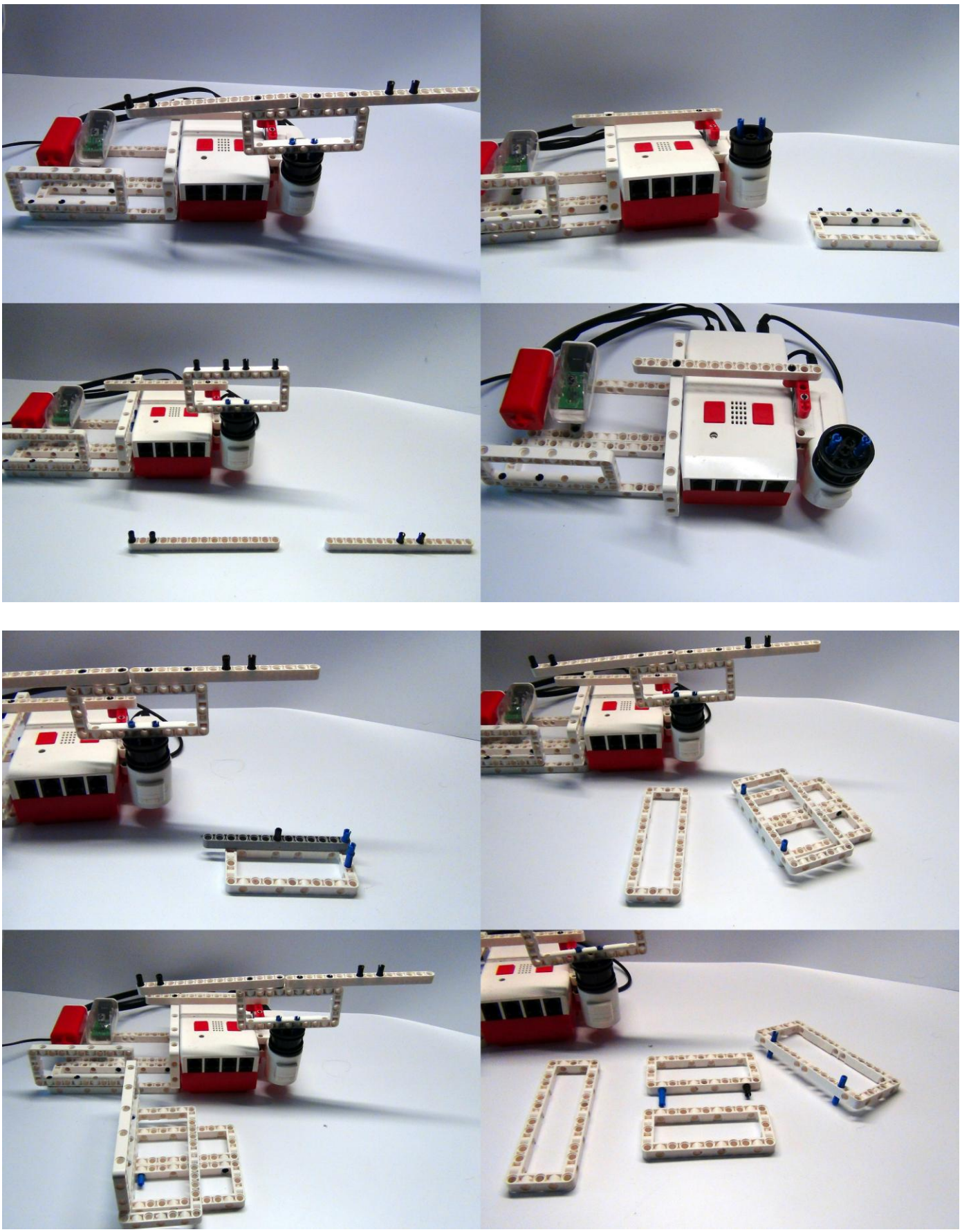
Пример решения.

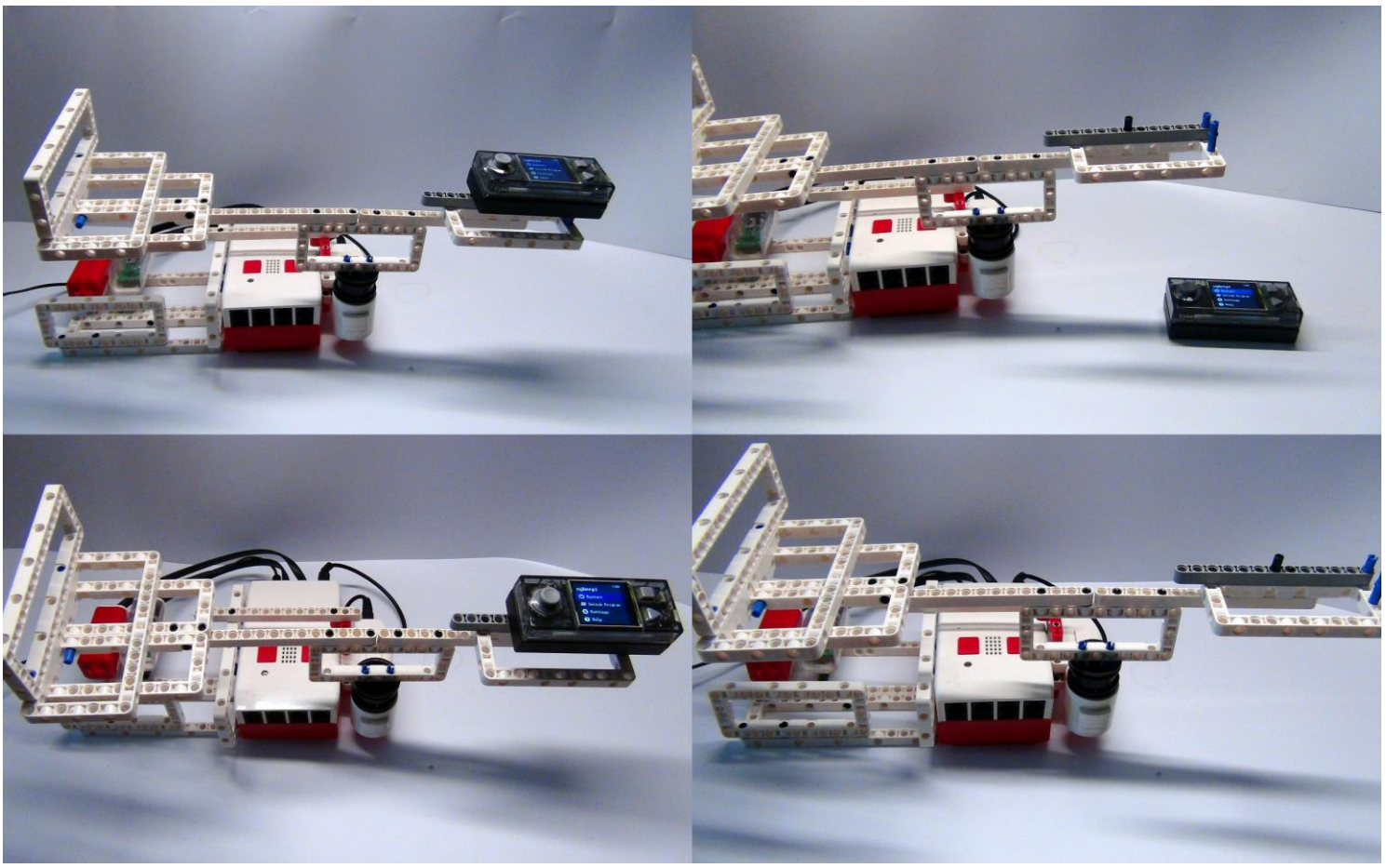
- Используя знания по созданию программ для управления сервомотором и датчиком касания робота, а также получения данных с гироскопа CyberPi и работой с его дисплеем, получим алгоритм для проведения исследования. В программе будет задействован алгоритм, который состоит из двух параллельно работающих частей. Первая часть связана с устройством, которое запускает карусель робота. Вторая часть – это считывание значений с гироскопа и построение графика на дисплее CyberPi.







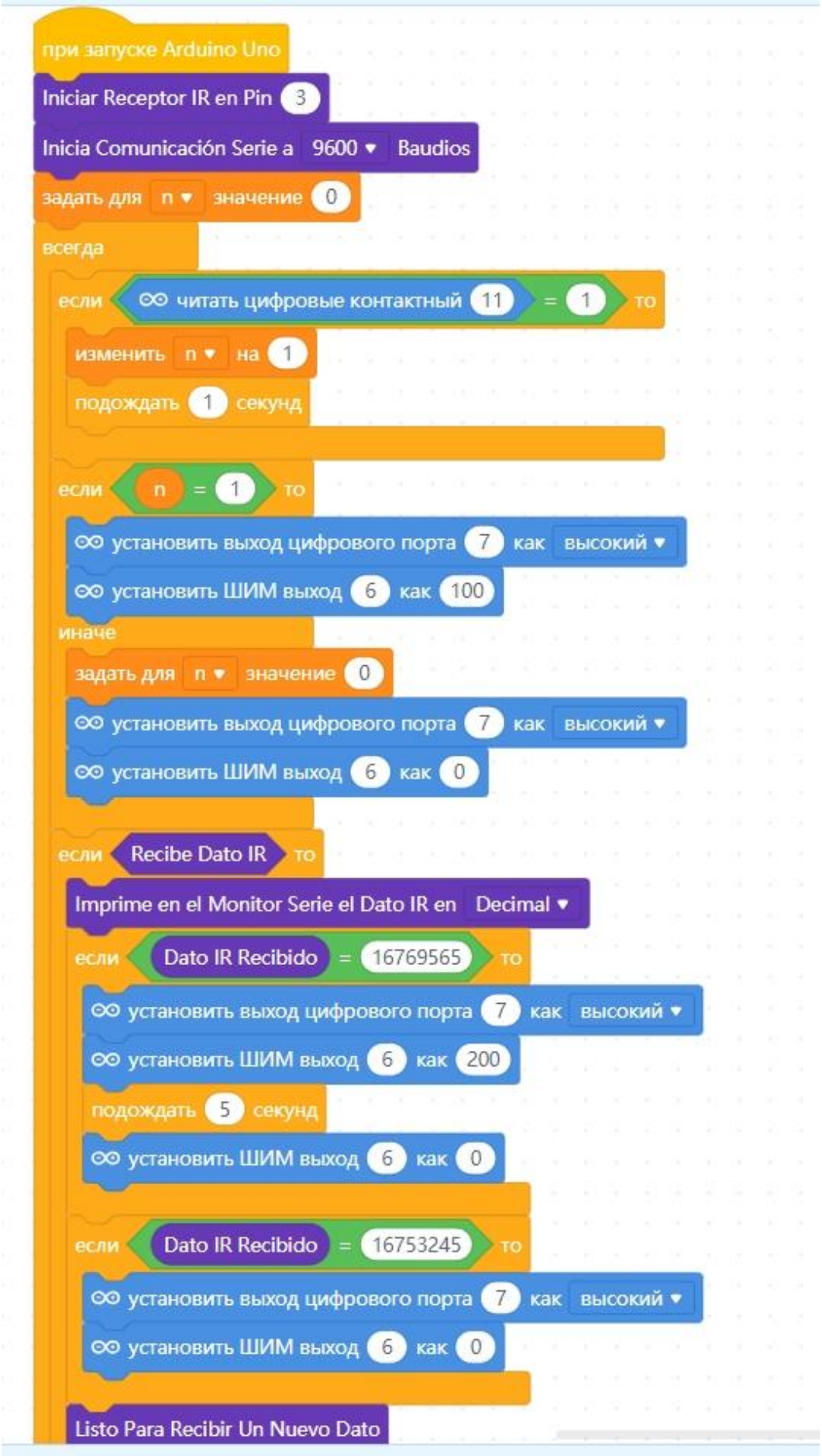




Задание 2.

Создадим программу для управления каруселью. Данным устройством можно управлять с Помощью ir модуля и датчика касания. IR модуль подключен ко второму порту, датчик касания к первому порту блока управления.

Кликая один раз на датчик касания, мы активируем запуск мотора, который вращает подвижную конструкцию с CyberPi. Повторное нажатие на датчик касания приведёт к полной остановке мотора.
По такому же принципу работает и управление через IR модуль. В этом случае нам необходимо нажать кнопку CH+ на пульте, чтобы запустить мотор и CH-, чтобы остановить мотор. Используя информацию из пункта 6.6. можно настроить и другие кнопки.


при запуске Arduino Uno
Iniciar Receptor IR en Pin 3
Inicia Comunicación Serie a 9600 Baudios
задать для n значение 0
всегда
если читать цифровые контактный 11 = 1 то
изменить n на 1
подождать 1 секунд
если n = 1 то
установить выход цифрового порта 7 как высокий
установить ШИМ выход 6 как 100
иначе
задать для n значение 0
установить выход цифрового порта 7 как высокий
установить ШИМ выход 6 как 0
если Recibe Dato IR то
Imprime en el Monitor Serie el Dato IR en Decimal
если Dato IR Recibido = 16769565 то
установить выход цифрового порта 7 как высокий
установить ШИМ выход 6 как 200
подождать 5 секунд
установить ШИМ выход 6 как 0
если Dato IR Recibido = 16753245 то
установить выход цифрового порта 7 как высокий
установить ШИМ выход 6 как 0
Listo Para Recibir Un Nuevo Dato

После того как настроили управление каруселью, нам понадобиться составить программу для CyberPi для снятия показаний. Мы будем считывать угловую скорость и центростремительное ускорение со встроенного гироскопа, а линейную скорость вычислять по формуле.

$$\vartheta = \frac{\omega}{R}$$

Где R – радиус окружности (плечо карусели). Согласно нашей схеме равен 26 см или 0,26 м. А ω – угловая скорость. Угловая скорость измеряется в $\frac{\text{рад}}{\text{с}}$.

Гироскоп измеряет угловую скорость в $\frac{^\circ}{\text{с}}$. Следовательно, чтобы перевести в радианы, нам нужно воспользоваться выражением:

$$\text{рад} = \frac{3.14 * \alpha^\circ}{180}.$$

Чтобы отобразить данные отдельно на CyberPi, воспользуемся кнопками A, B и Джойстиком. Для хранения всех полученных значений используем списки.

При нажатии кнопки B мы получаем вывод трёх значений в режиме реального времени. При Нажатии A, мы прерываем сбор данных и смотрим собранные данные по циклической частоте.

При сдвиге вверх джойстика мы получаем вычисленные данные по линейной скорости.

Пример программы выглядит так.

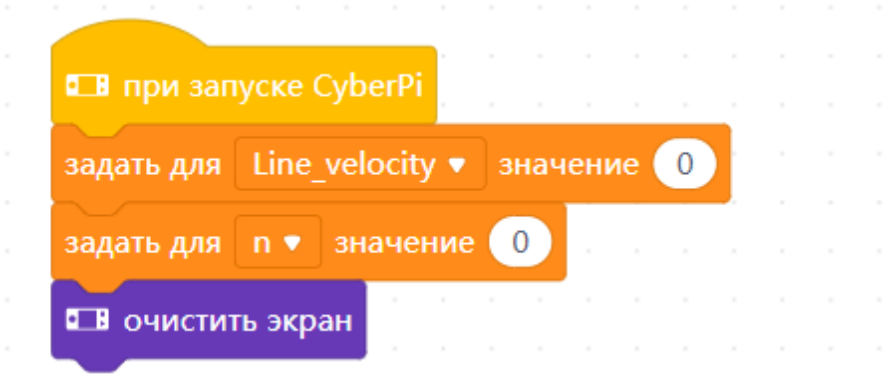


Figure 1 Первая часть программы

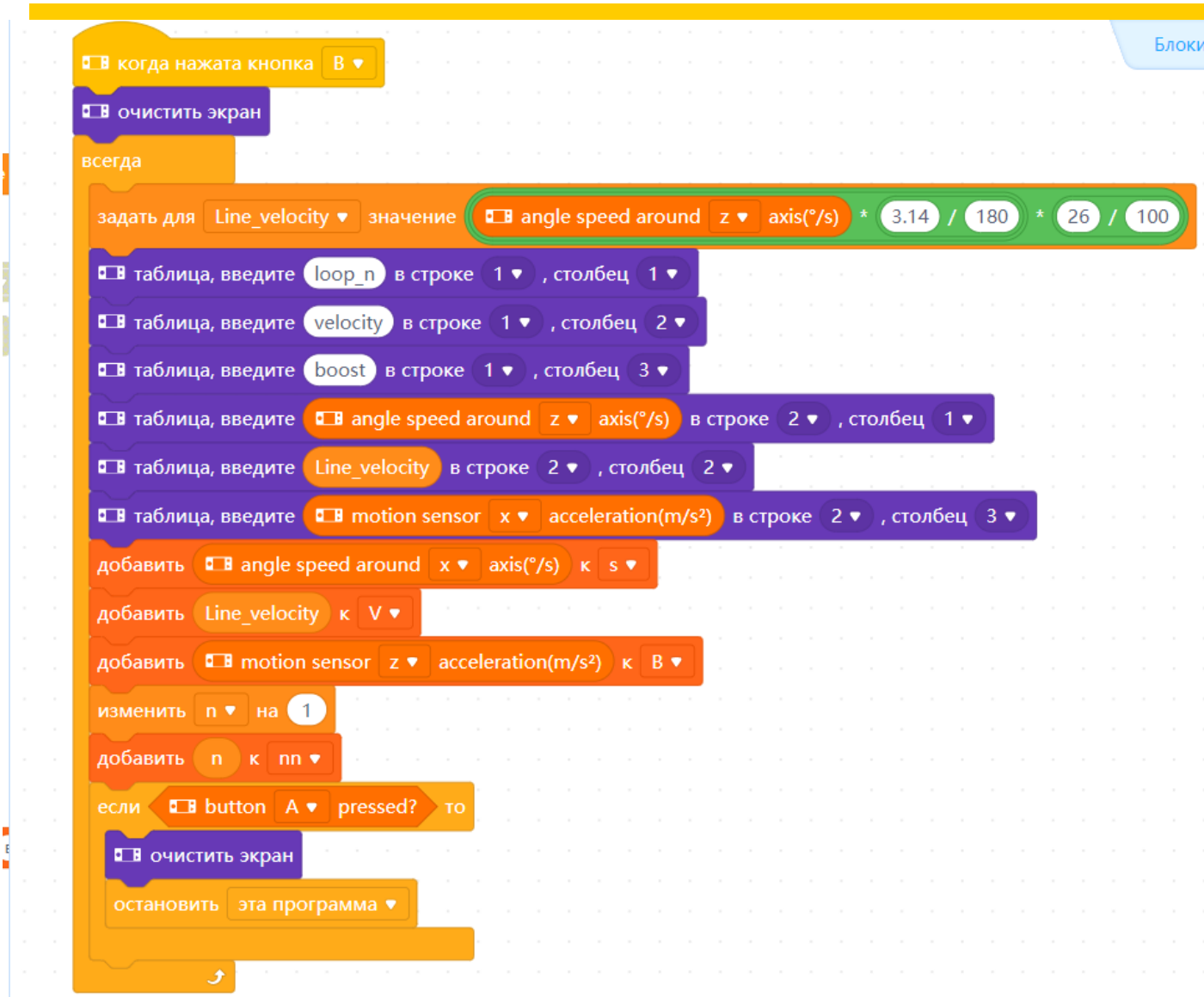


Figure 2. Вторая часть программы

Figure 3. Третья часть программы

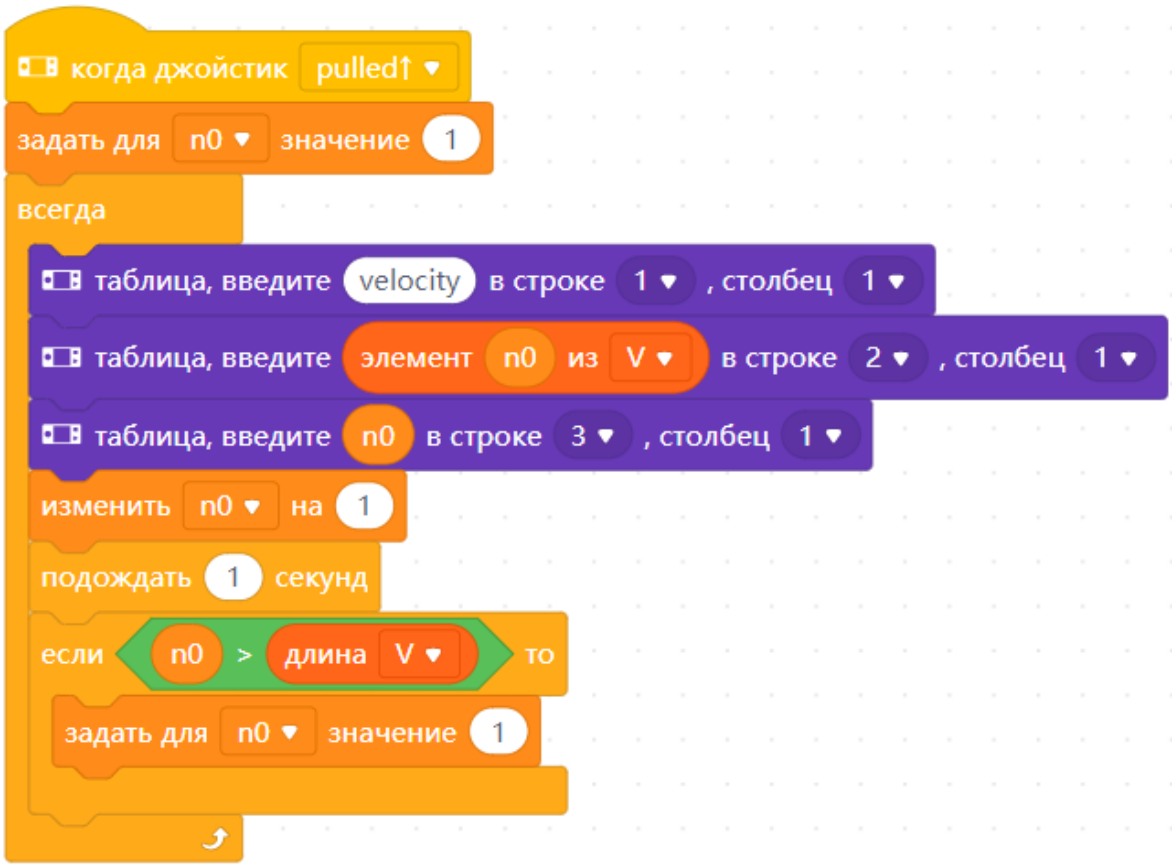


Figure 4. Четвёртая часть программы

Как можно заметить, радиус мы перевели в метры. Все три показания заносятся в таблицу.

13.3. Мобильный робот картограф.

Методические рекомендации.

Тема: «Запись данных о маршруте и построение карты»

Цель: изучить процесс создания и программирования робота для построения карты движения.

Задачи:

- изучить и закрепить на практике процесс создания робота по исследованию пространства и построению карты маршрута
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий.

Задание 1 (Уровень А)

Соберите устройство согласно инструкции, представленной ниже

Задание 2 (Уровень В)

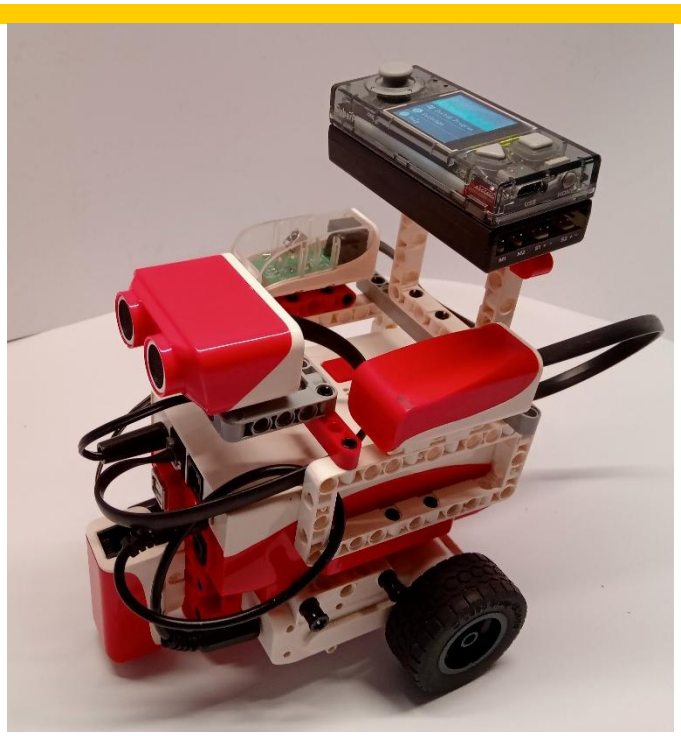
Создайте две программы, с помощью которых можно управлять и собирать данные для построения графика в режиме реального времени.

Пример решения.

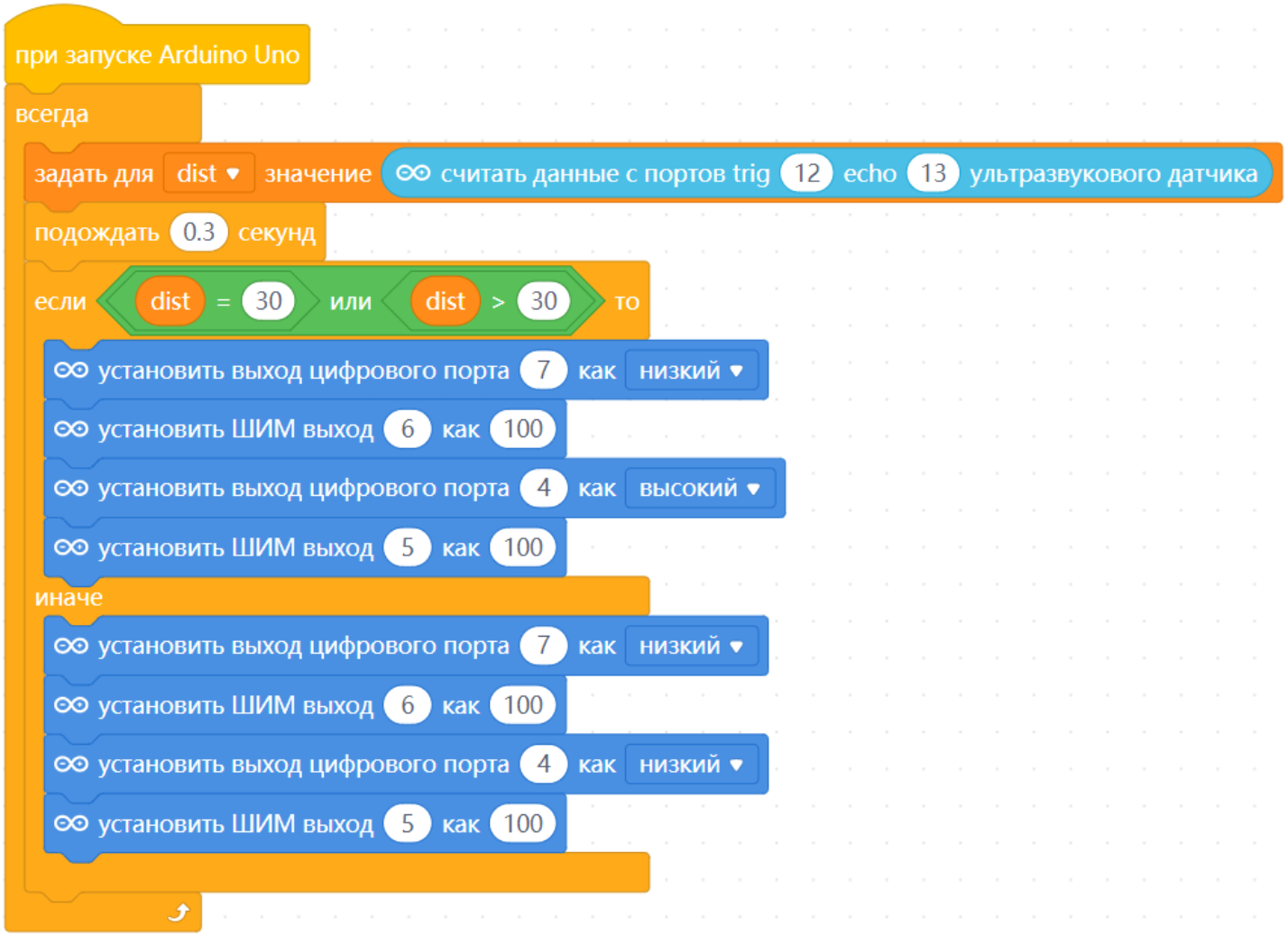
- Используя знания по созданию программ для управления поведением мобильного робота, на основе ультразвукового датчика, написать код, при котором, робот будет перемещаться в пространстве и объезжать препятствия. Для того чтобы робот параллельно записывал своё положение относительно начало своего движения мы будем использовать CyberPi.

Задание 1.

Для построения мобильного робота, способного объезжать препятствия можно воспользоваться инструкцией из пункта 8.1.1. В нашем случае, к данной конструкции необходимо прикрепить CyberPi по такому примеру.



Задание 2.
Напишем программу для мобильного робота. Можно взять пример из пункта 8.1.1.



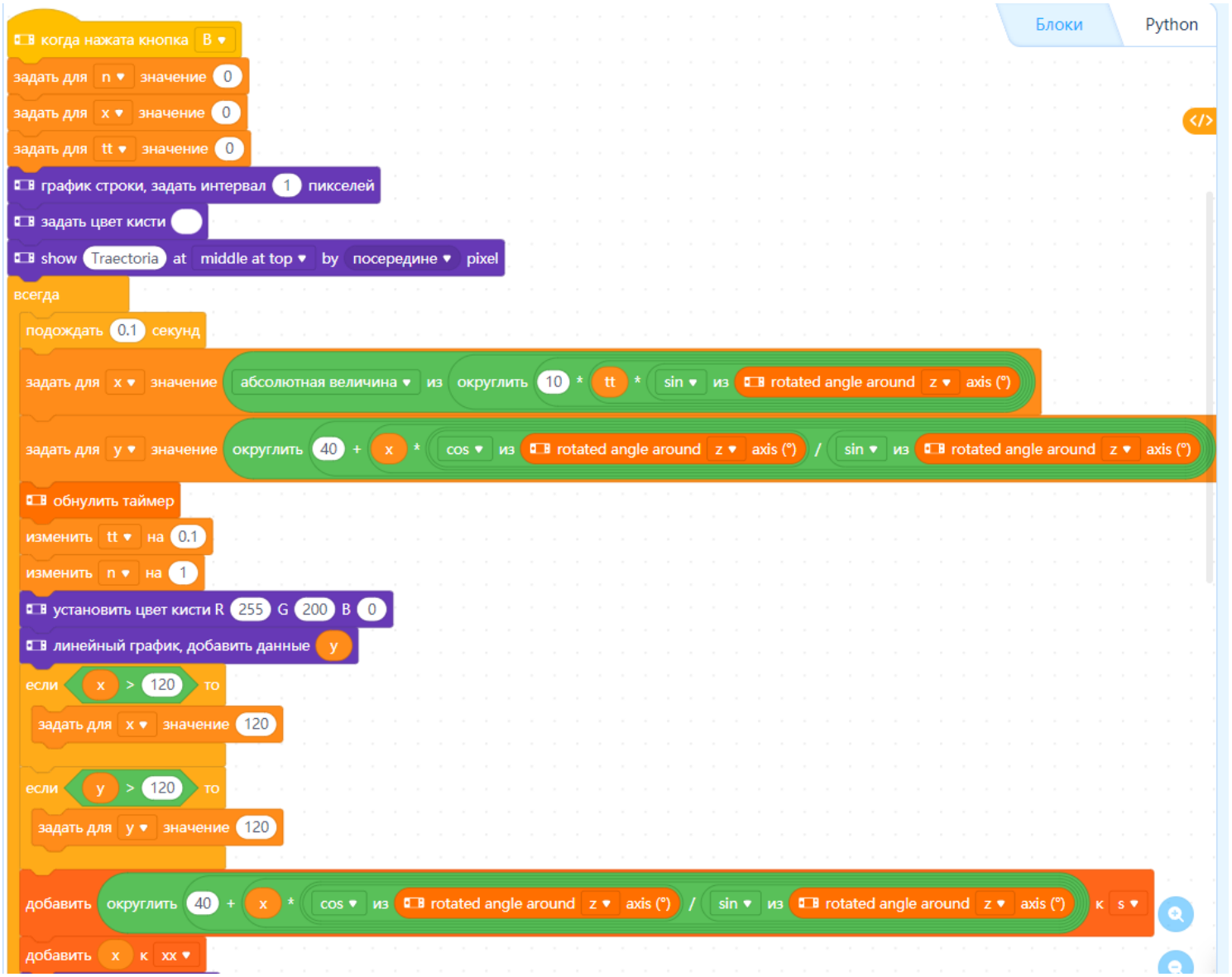
Протестируйте работу программы. Если робот объезжает препятствия и не врезается в них, то всё сделано правильно. Если есть ошибки, то можете увеличить диапазон определения расстояния до препятствия.

Теперь перейдём к программированию CyberPi.B зависимости от того, как мы разместим контроллер, будет зависеть по каким осям гироскопа нам вычислять.

Она будет состоять из двух подпрограмм. Первая подпрограмма вычисляет координаты положения с учётом скорости и угла поворота.

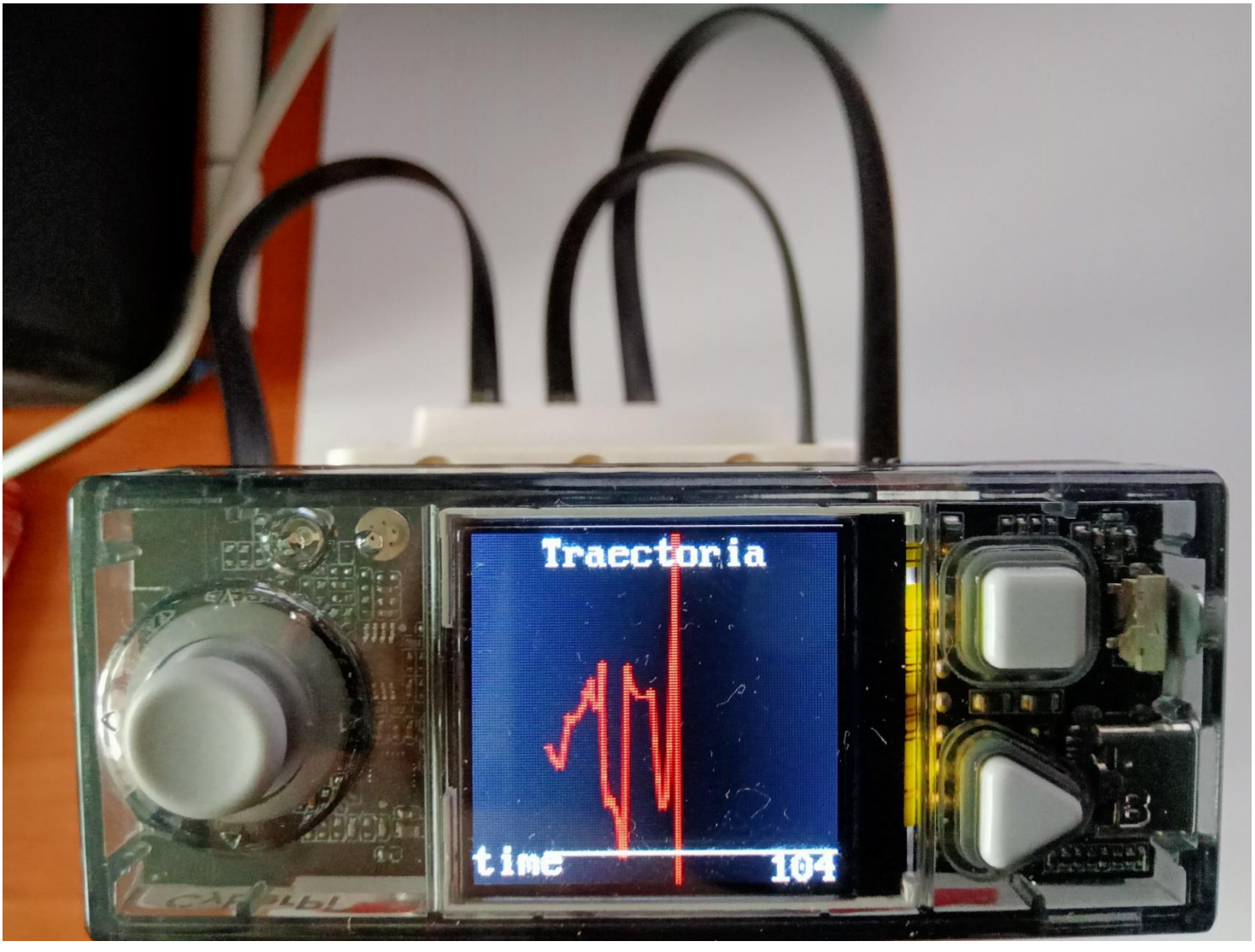
$$\begin{cases} x = \vartheta * t * \sin(\alpha) \\ y = \vartheta * t * \cos(\alpha) \end{cases}$$

Данные значения сохраняются в списки xx и s.



```
задать цвет кисти
show label 2 ▾ time at bottom left corner ▾ by посередине ▾ pixel
линейный график, добавить данные 10
show label 1 ▾ n at x: n y 115 by посередине ▾ pixel
если button A ▾ pressed? то
    остановить эта программа ▾
```

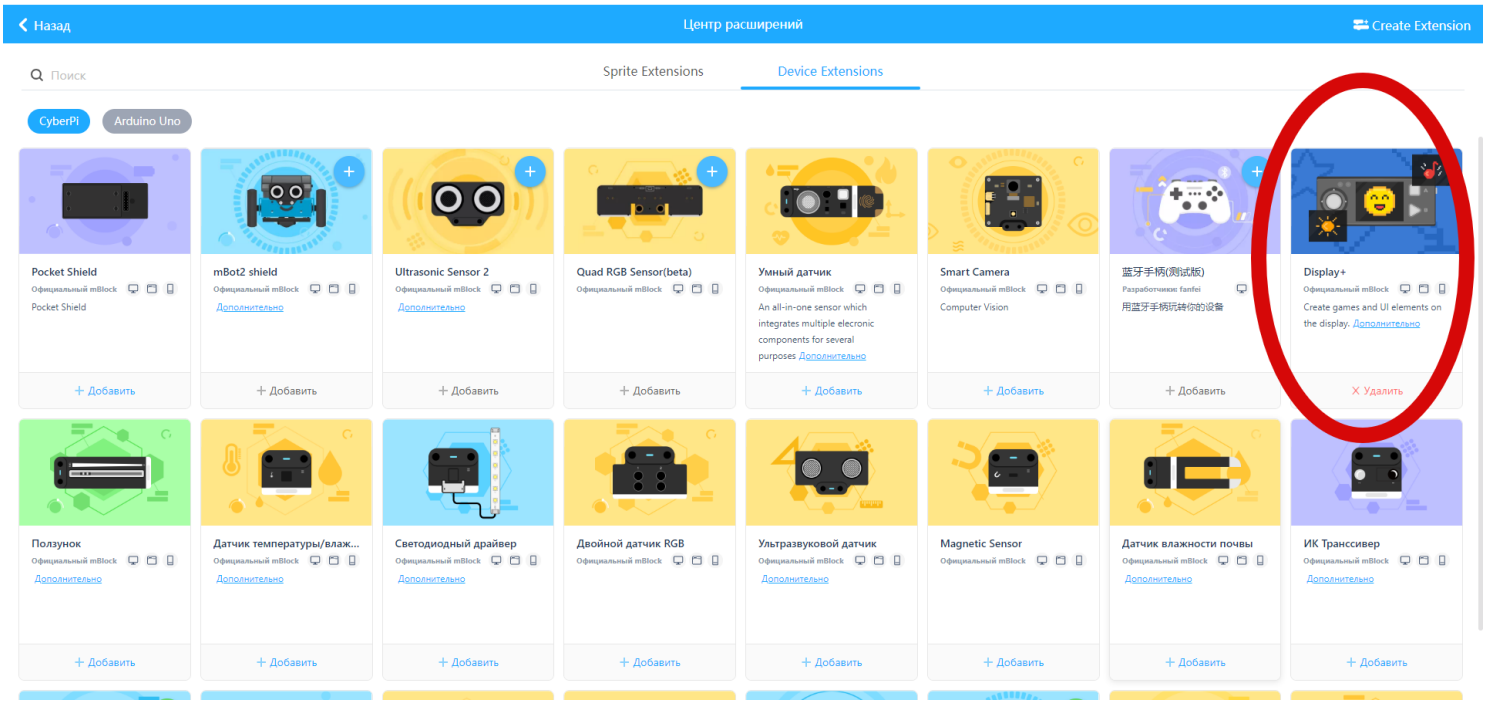
Как видно, значения x и y округлены для корректного отображения точек на дисплее. Параллельно вычислению координат мы можем наблюдать построение графика перемещения с течением времени.



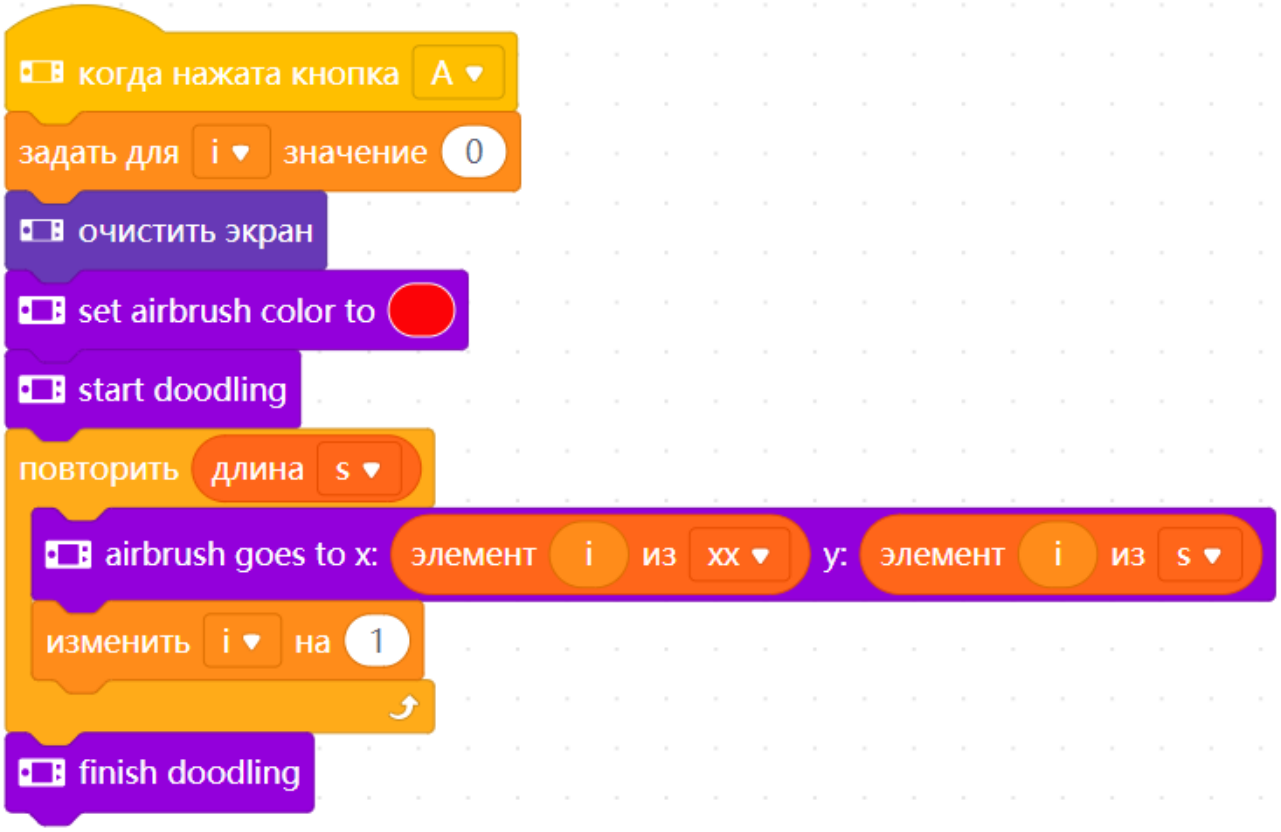
Здесь мы можем наблюдать и время работы устройства. Точнее количество шагов, которые при умножении на 0.1 секунду покажет время работы.

Если мы кликнем по клавише A, то сотрётся предыдущий рисунок и построится траектория движения робота в примерном масштабе экрана.

Для реализации данного алгоритма необходимо подгрузить расширение для CyberPi.

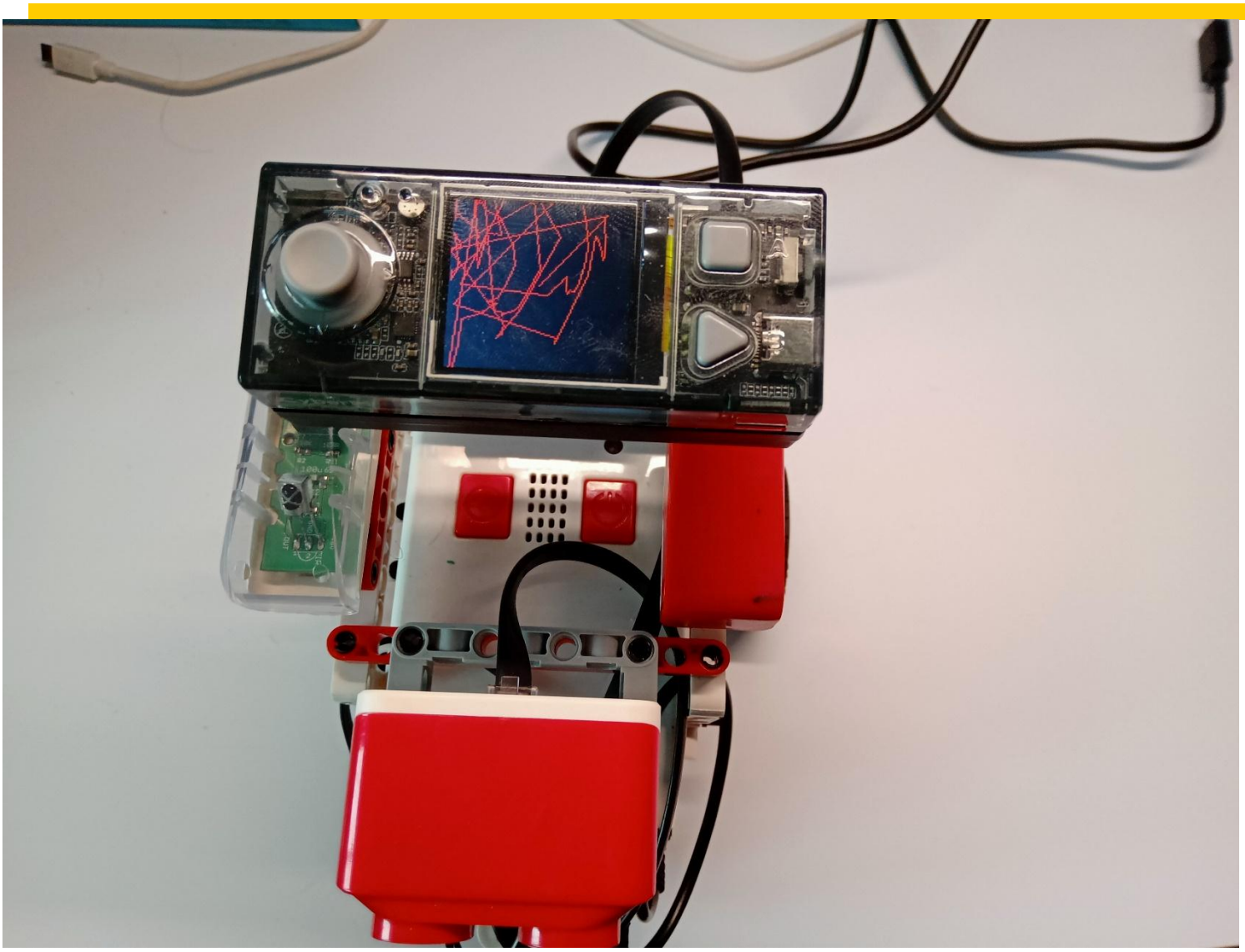


После, можем приступить к написанию второй подпрограммы. В ней мы должны пробегать каждый из двух списков и присваивать их значения координатам точки, которая рисуется с помощью кисти. Мы должны пробежаться по всему списку, поэтому нам нужен конечный цикл по его длине.

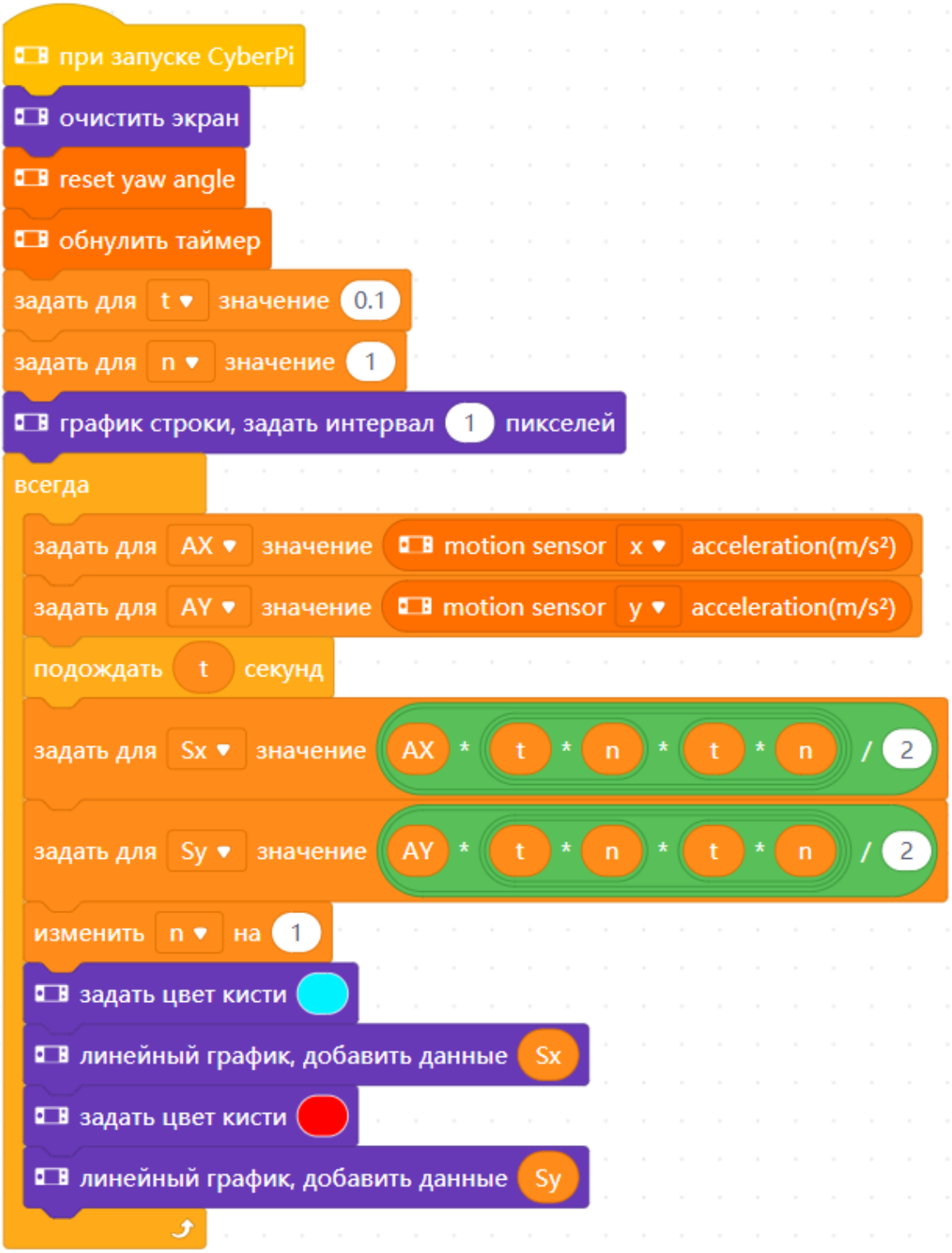


Если запустить программу на CyberPi, нажатием на клавишу B, то получим выполнение первой подпрограммы. Затем запустим мобильного робота, который начнёт перемещаться по помещению и объезжать препятствия.

Если мы хотим закончить эксперимент, то просто останавливаем робота и кликаем по кнопке A на CyberPi. И В результате будем наблюдать построение траектории движения.



0,


при запуске CyberPi
очистить экран
reset yaw angle
обнулить таймер
задать для t значение 0.1
задать для n значение 1
график строки, задать интервал 1 пикселей
всегда
задать для AX значение motion sensor x acceleration(m/s²)
задать для AY значение motion sensor y acceleration(m/s²)
подождать t секунд
задать для Sx значение AX * t * n * t * n / 2
задать для Sy значение AY * t * n * t * n / 2
изменить n на 1
задать цвет кисти
линейный график, добавить данные Sx
задать цвет кисти
линейный график, добавить данные Sy

13.4. Робот исследователь.

Методические рекомендации.

Тема: «Автономное исследование окружающей среды»

Цель: изучить процесс создания и программирования робота для проведения исследования пространства.

Задачи:

- изучить и закрепить на практике процесс создания робота по исследованию пространства:, положение
- изучить особенности управления роботом данной конструкции
- получить и закрепить на практике знания, умения и навыки в области кинематики робота и создания программ для него.

Примеры заданий.

Задание 1 (Уровень А)

Соберите устройство согласно инструкции, представленной ниже

Задание 2 (Уровень В)

Создайте две программы, с помощью которых можно управлять и собирать данные для построения графика в режиме реального времени.

Пример решения.

- Используя знания по созданию программ для управления поведением мобильного робота, на основе ультразвукового датчика, написать код, при котором, робот будет перемещаться в пространстве и объезжать препятствия. Для того чтобы робот параллельно записывал своё положение относительно начало своего движения мы будем использовать CyberPi.

Задание 1.

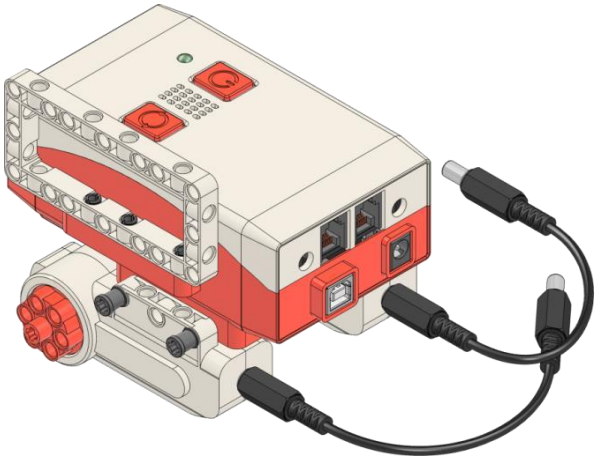



Рис.49 Движение по линии

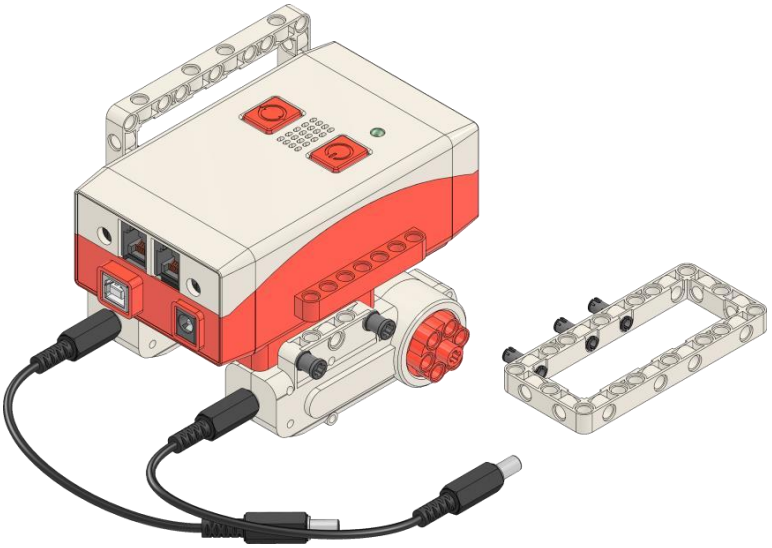


Рис.50 Движение по линии

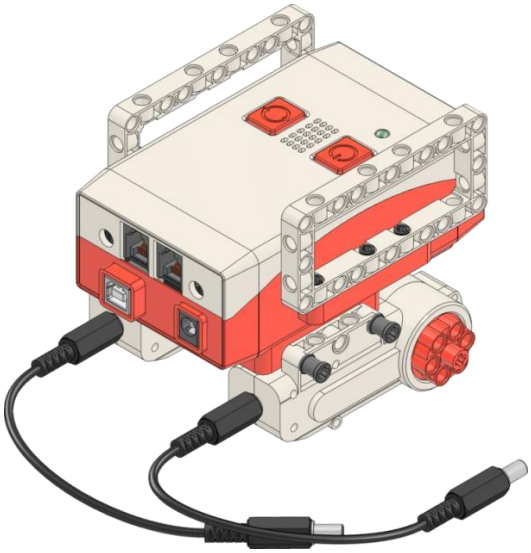


Рис.51 Движение по линии

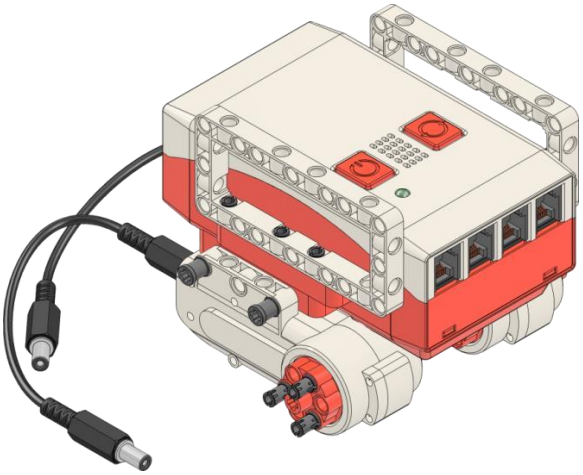


Рис.52 Движение по линии

Рис.53 Движение по линии

Рис.54 Движение по линии

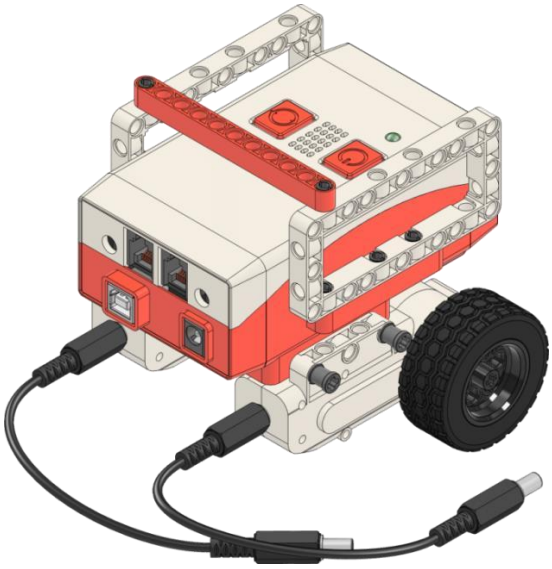


Рис.55 Движение по линии

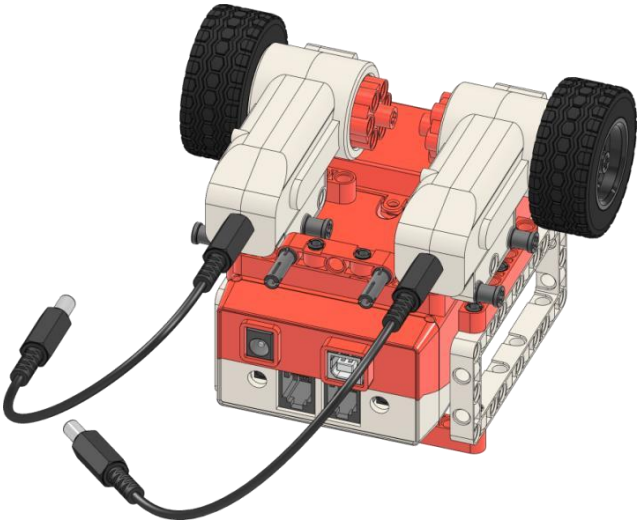


Рис.56 Движение по линии

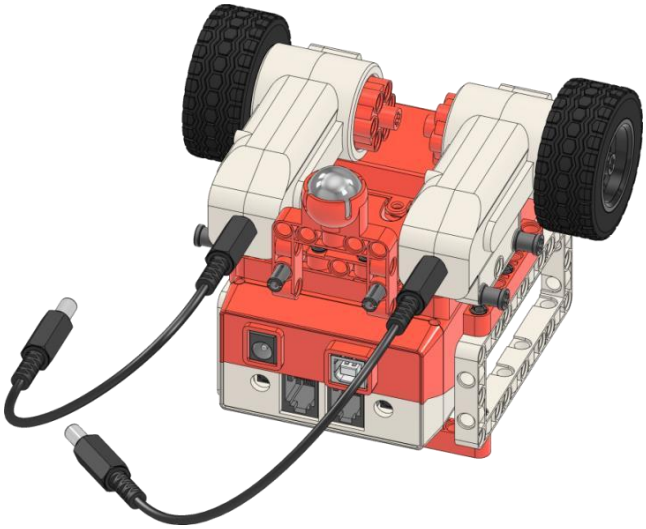


Рис.57 Движение по линии

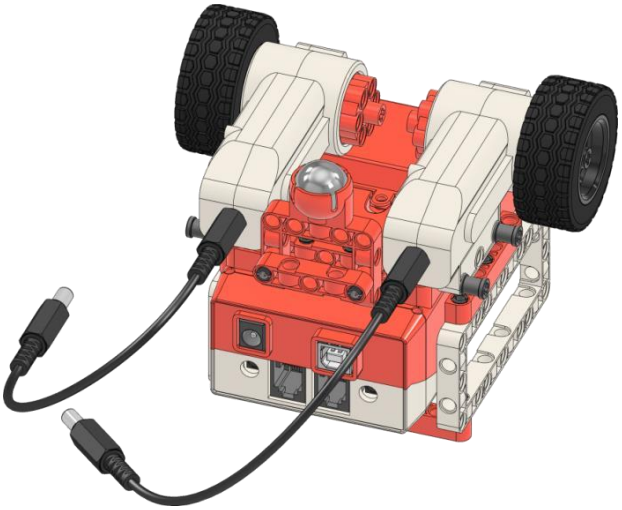


Рис.58 Движение по линии

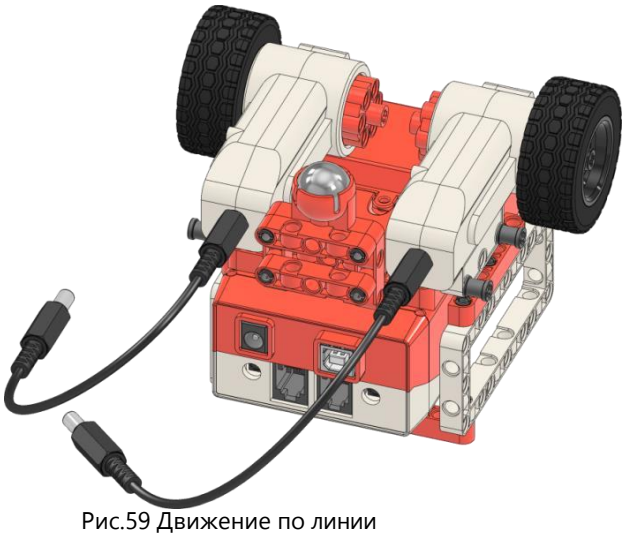


Рис.59 Движение по линии

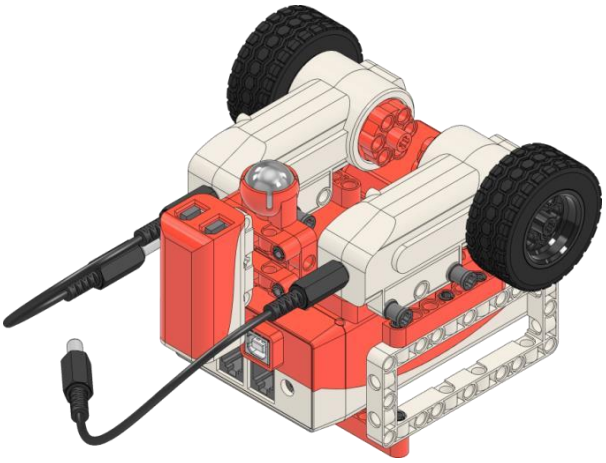


Рис.60 Движение по линии

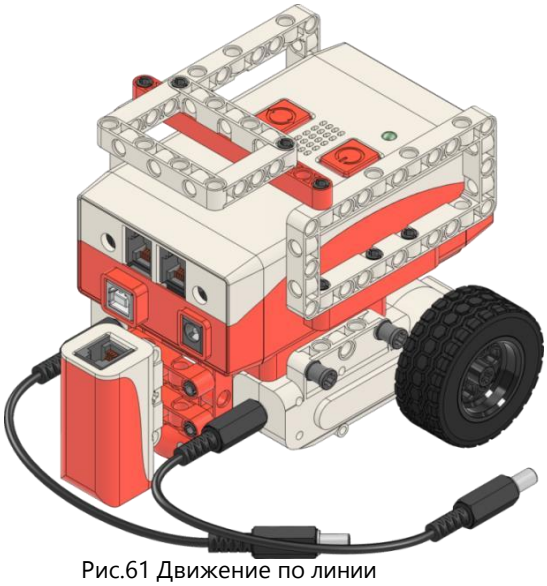


Рис.61 Движение по линии

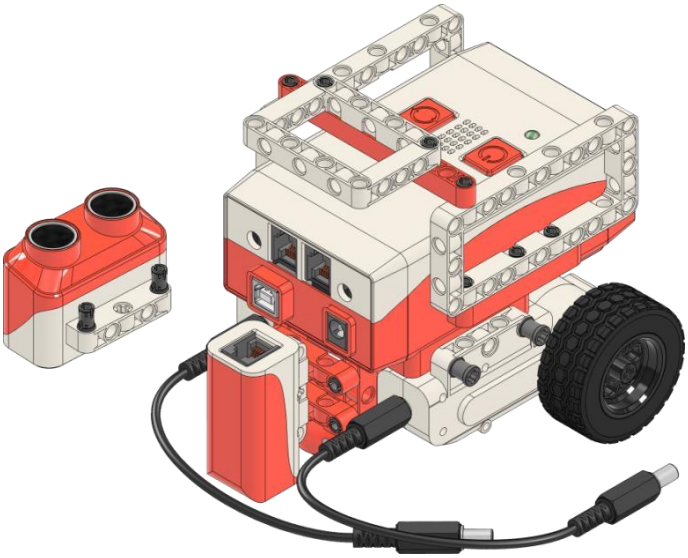


Рис.62 Движение по линии

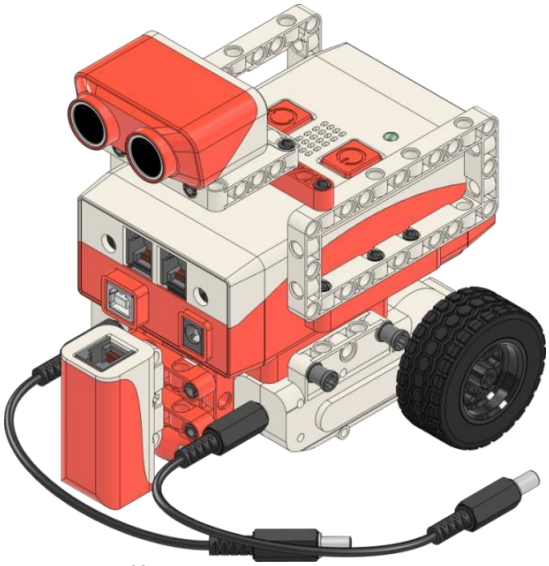


Рис.63 Движение по линии

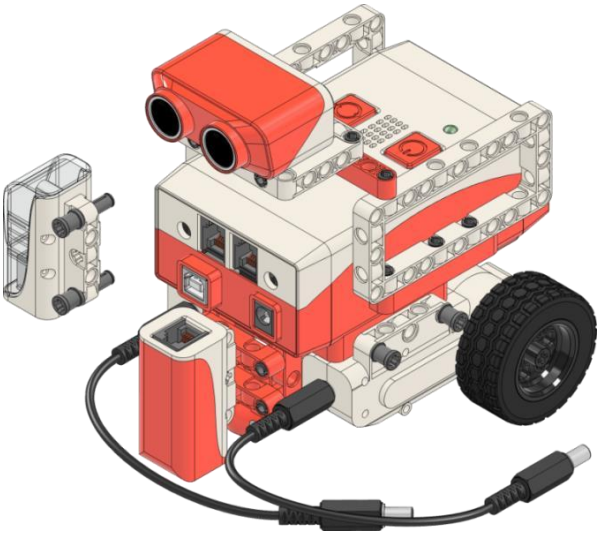


Рис.64 Движение по линии

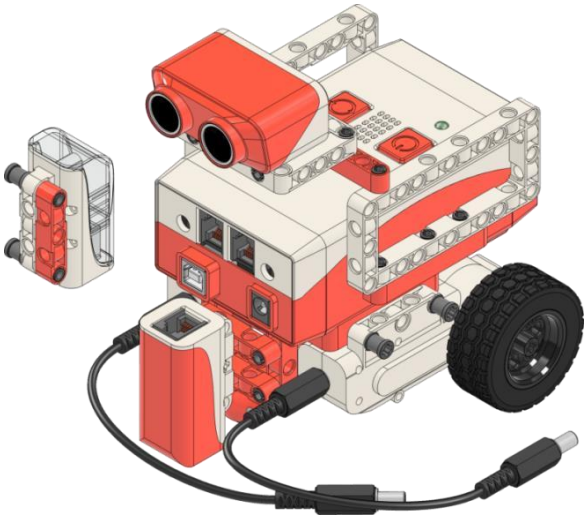


Рис.65 Движение по линии

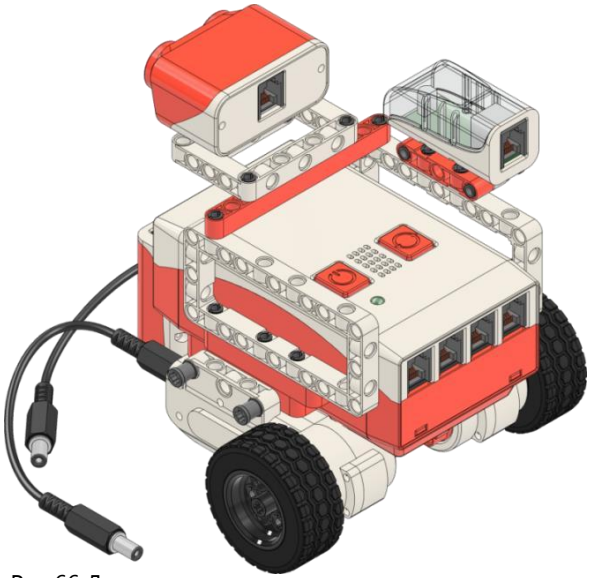


Рис.66 Движение по линии

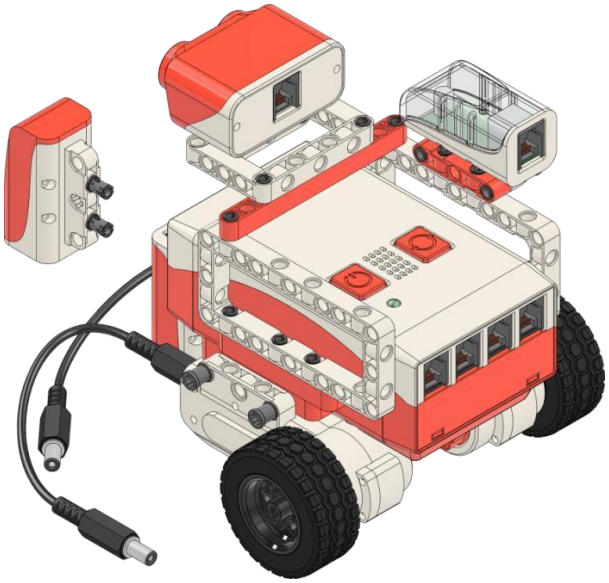


Рис.67 Движение по линии

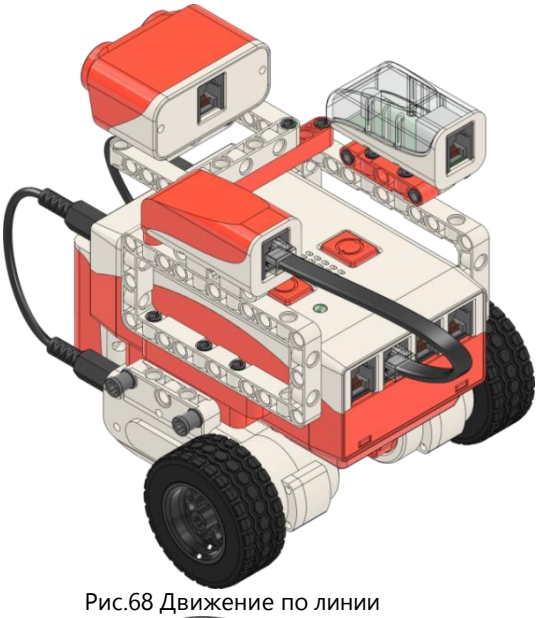


Рис.68 Движение по линии

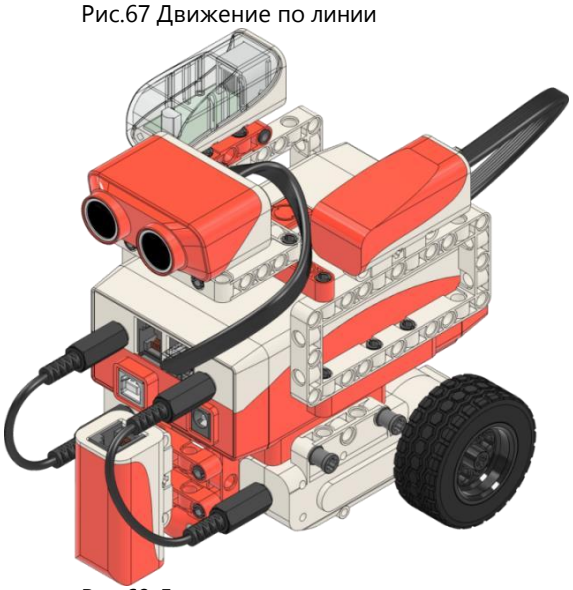


Рис.69 Движение по линии

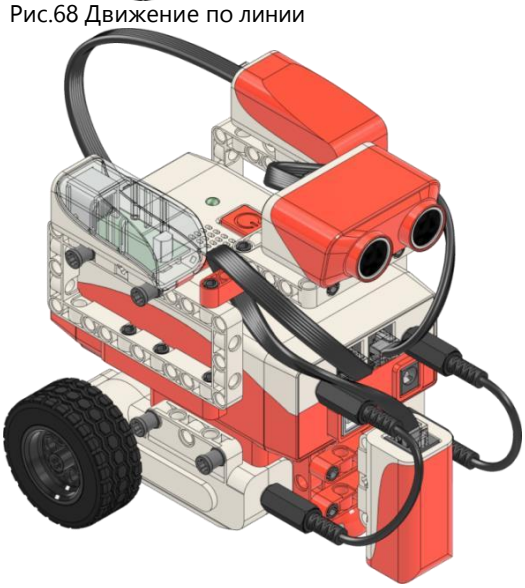


Рис.70 Движение по линии

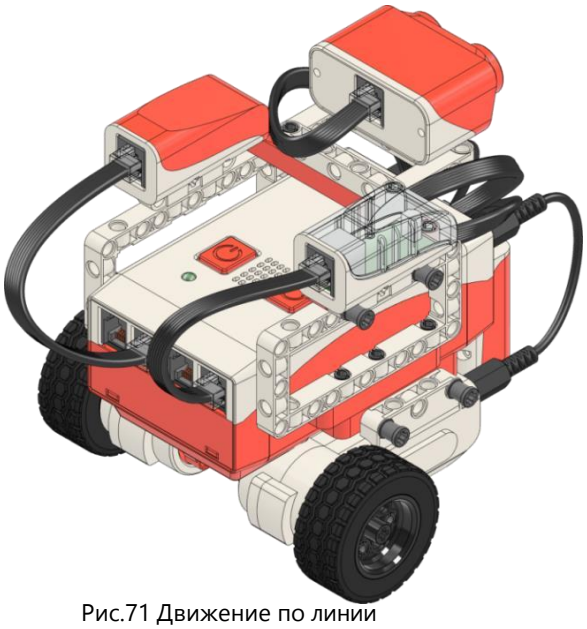
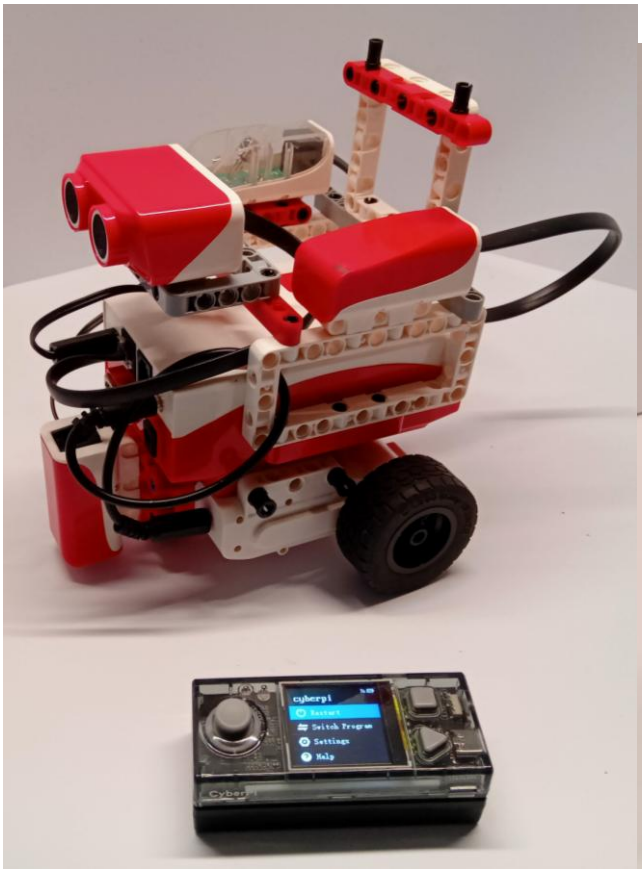
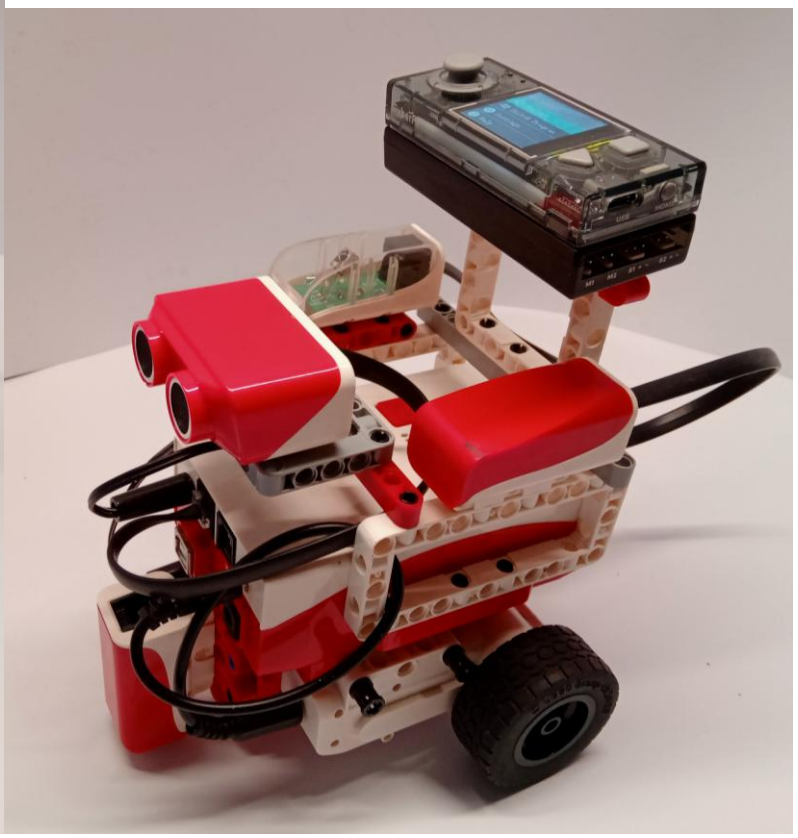


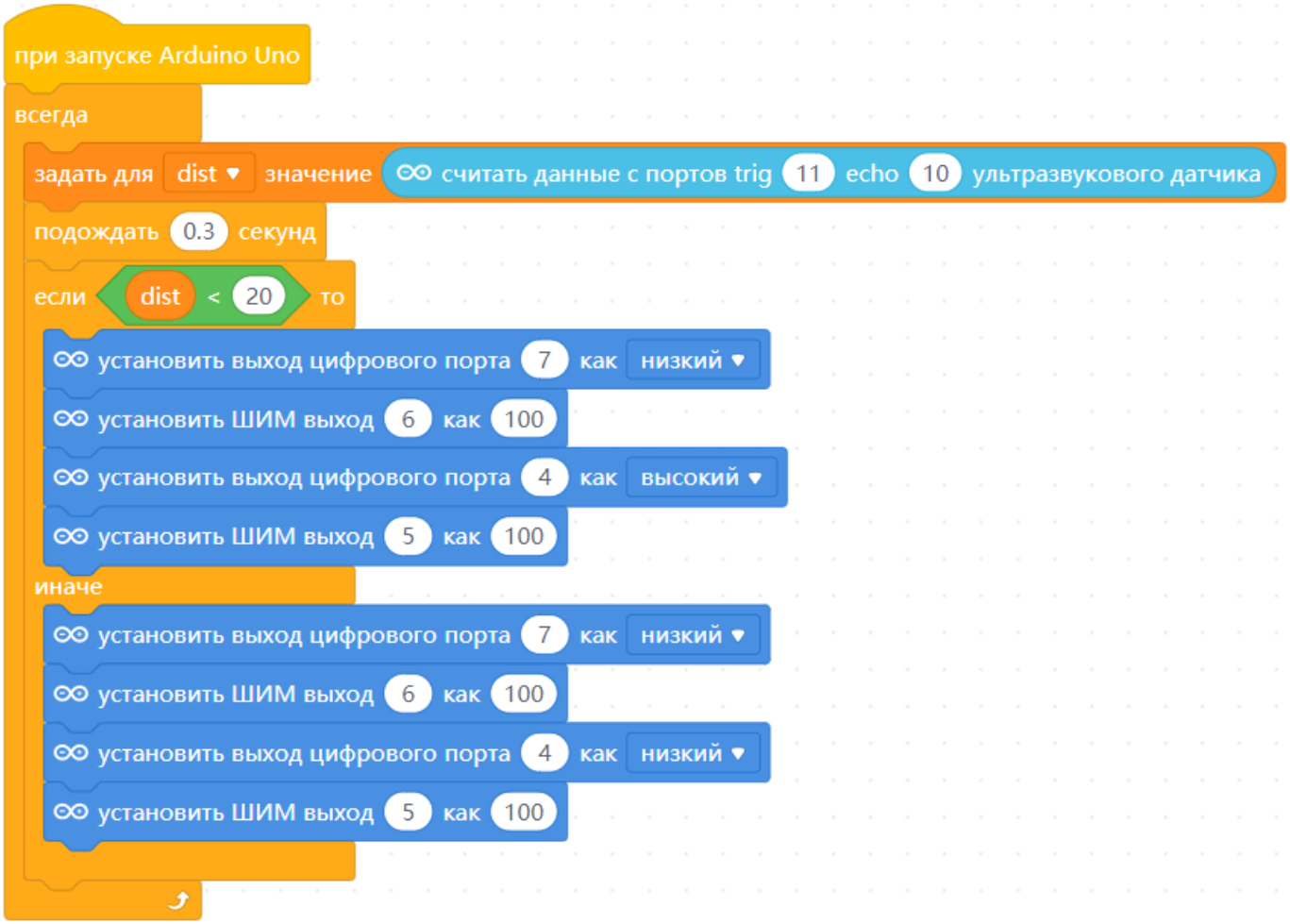
Рис.71 Движение по линии





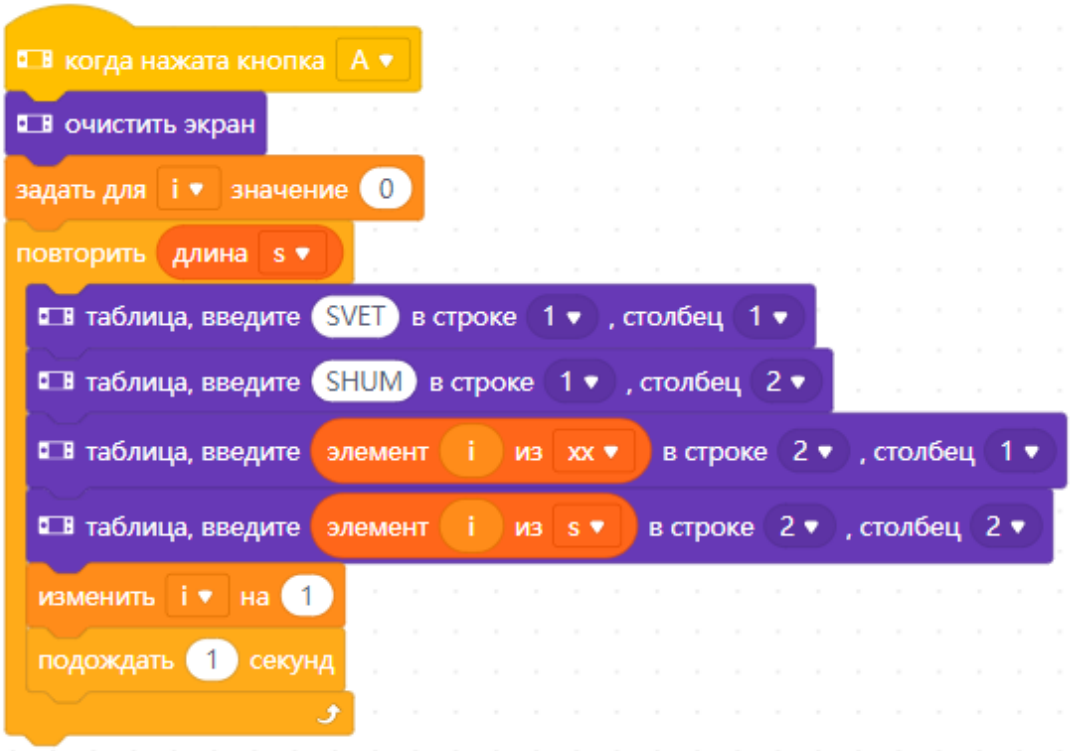
Задание 2.

Напишем программу для ориентации мобильного робота в пространстве. Ориентироваться робот будет с помощью ультразвукового датчика расстояния. Если робот видит препятствие, то совершает поворот в течении 0.3 секунды.



Теперь перейдём к написанию программы для CyberPi.
Она будет состоять из двух подпрограмм. Первая подпрограмма вычисляет уровень шума и освещение окружающей среды. Вторая подпрограмма выводит полученные значения в таблице.
Каждую секунду данные из списка пробегают на шаг. В результате, мы можем провести анализ по данным признакам о наличии, например, признаков жизни, неисправности устройств, влияние света на интенсивность животного мира.

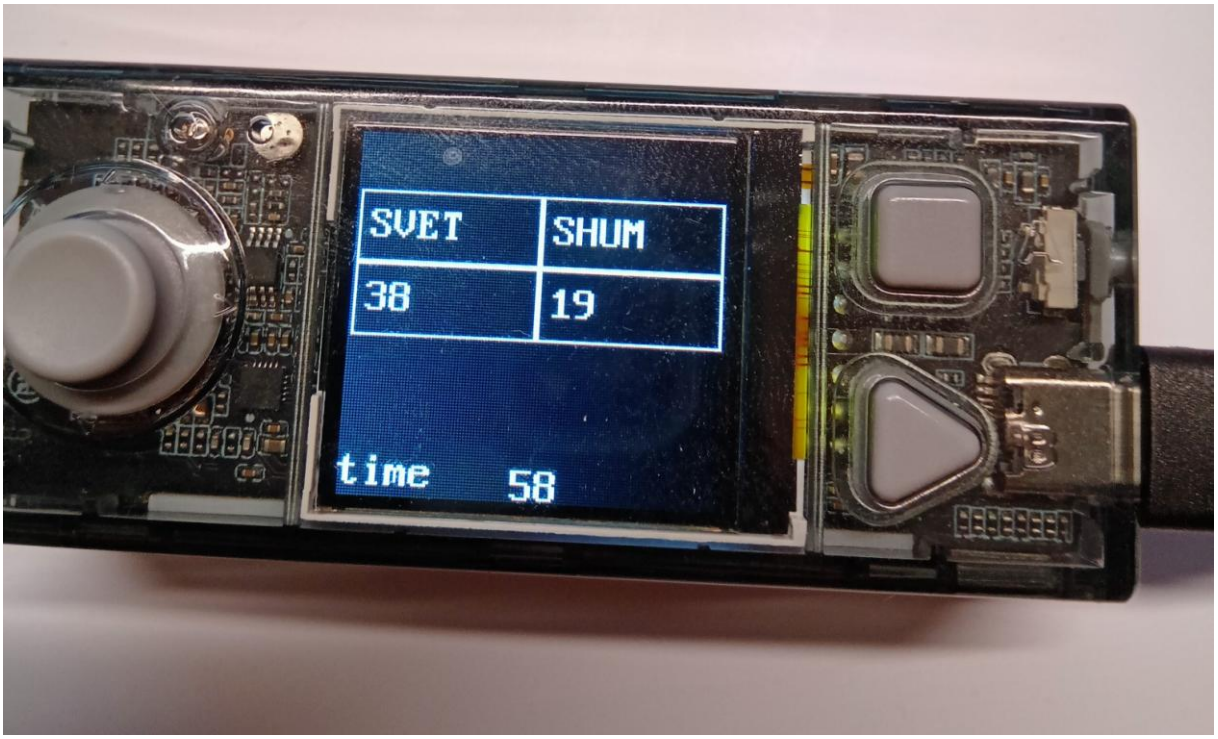

когда нажата кнопка B
очистить экран
задать для n значение 0
задать для x значение 0
задать для tt значение 0
график строки, задать интервал 5 пикселей
задать цвет кисти
show SVET at middle at top by посередине pixel
всегда
подождать 0.1 секунд
задать для x значение яркость окружающего света
задать для y значение громкость
обнулить таймер
изменить tt на 0.1
изменить n на 1
установить цвет кисти R 255 G 200 B 0
линейный график, добавить данные y
задать цвет кисти
линейный график, добавить данные x
добавить y к s
добавить x к xx
задать цвет кисти
show label 2 time at bottom left corner by посередине pixel
линейный график, добавить данные 10
show label 1 n at x: n y 115 by посередине pixel
если button A pressed? то
остановить эта программа



Здесь мы можем наблюдать и время работы устройства. Точнее количество шагов, которые при умножении на 0.1 секунду покажет время работы.

Если мы кликнем по клавише A, то сотрётся предыдущий рисунок и загрузится таблица собранных данных.


SVET
time 41


SVET
SHUM
38
19
time
58

14. Послесловие

Дорогие педагоги и ученики. Вы изучили основы работы с набором КЛИК и рассмотрели различные возможности применения его в разных направлениях, как предметных, так направлений в области робототехники.

Возможности arduino платформ обширны, а вместе с защитным корпусом и деталями для конструирования данный набор становиться заветной мечтой многих образовательных учреждений. Теперь вы не ограничены в деталях, в датчиках, и в программном обеспечении. Комбинируйте с деталями из конструкторов других производителей, создавая различное множество неограниченных в креплениях и способах соединений прототипов и моделей роботизированных конструкций. Вы можете с лёгкостью подключить любой датчик с которым может работать плата arduino, а их насчитывается более ста моделей. Программируя в разнообразных средах, вы обучаете ребёнка вариативности, расширяете языковую базу в области программирования, учите видеть одни те же алгоритмы на разных языках.

Данная методика позволит выйти на новый уровень преподавания робототехники.

Учебно–тематическое планирование

Учебная пара – 90 минут.

№ п/п	Тема	Кол – во часов
1	Введение в робототехнику	13
2	Введение в конструирование и программирование	11
3	Юный робототехник	13
4	Физические эксперименты	4
Итого		**41**

Приложение

Календарно – тематическое планирование

№ п/п	Тема занятий	Краткое описание содержания занятия	Кол – во часов
1. Введение в робототехнику			
1	Инструктаж по технике безопасности. Идея создания роботов. История робототехники. Что такое робот. Виды современных роботов. Применение роботов в современном мире. Конкурсы, состязания в мире робототехники	Инструктаж по технике безопасности. Применение роботов в современном мире: от детских игрушек, до серьезных научных исследовательских разработок. Демонстрация передовых технологических разработок. История робототехники от глубокой древности до наших дней. (Презентации, с использованием ИКТ)	1
2	Знакомство с конструктором КЛИК	Знакомство с основными составляющими частями среды конструктора. Работа с классификацией деталей. Знакомство с видами соединений и особенностями подключения электроники. Умения слушать инструкцию педагога	1
3	Краткий обзор программного обеспечения	Знакомство с четырьмя средами программирования Arduino ide, ArduBlock, MBlock3, MBlock5	1
4	Программирование в среде mBlock5. **Панель инструментов: возможности и функции**	Знакомство детей с панелью инструментов, функциональными командами; составление программ в режиме Конструирования.	1
5	Программирование в среде mBlock5. **Линейные алгоритмы**	Получение знаний, умений и навыков в создании программ с линейным алгоритмом	1
6	Программирование в среде mBlock5. **Ветвления и вложенные ветвления**	Получение знаний, умений и навыков в создании программ с алгоритмом ветвления	1
7	Программирование в среде mBlock5. **Циклы: конечные и бесконечные**	Получение знаний, умений и навыков в создании программ с циклическими алгоритмами	1
8	Программирование в среде mBlock5. **Вложенные циклы**	Получение знаний, умений и навыков в создании программ с вложенными циклами	1
9	Программирование в среде mBlock5. **Комбинированные**	Получение знаний, умений и навыков в создании программ с	1

	алгоритмы	комбинированными алгоритмами	
10	Программирование в среде Arduino ide. **Плата Arduino uno**. **Панель инструментов Arduino ide: возможности и функции**	Обзор платы Arduino uno: технические возможности, подключения, параллельное и последовательное соединение, разновидность пинов. Получение знаний умений и навыков при работе в среде Arduino ide	1
11	Программирование в среде Arduino ide. **Особенности конструкции кода. Основные функции и операторы: int, pinMode(), digitalWrite(), Serial(), delay(). Линейный алгоритм**	Получение знаний, умений и навыков при работе в среде Arduino ide. Знакомство с базовыми функциями Arduino api	1
12	Программирование в среде Arduino ide. **Ветвление и вложенные ветвления**	Получение знаний, умений и навыков в создании программ с алгоритмом ветвления в среде Arduino ide	1
13	Программирование в среде Arduino ide. **Циклы и вложенные циклы**	Получение знаний, умений и навыков в создании программ с циклическими алгоритмами в среде Arduino ide	
2. Введение в конструирование и программирование			
Основы управления			
14	DC Моторы	Получение знаний, умений и навыков в подключении и настройки работы моторов	1
15	Сервопривод	Получение знаний, умений и навыков в подключении и настройки работы сервоприводов	1
16	Ультразвуковой датчик расстояния	Получение знаний, умений и навыков в подключении и настройки работы ультразвукового датчика расстояния.	1
17	Датчики линии	Получение знаний, умений и навыков в подключении и настройки работы датчика линии	1
18	Датчик цвета	Получение знаний, умений и навыков в подключении и настройки работы датчика цвета.	1
19	IR приёмник	Получение знаний, умений и навыков в подключении и настройки работы IR модуля	1
20	Bluetooth модуль	Получение знаний, умений и навыков в подключении и настройки работы Bluetooth модуля	1
21	Пьезоэлемент	Получение знаний, умений и навыков в подключении и настройки работы	1

		пьезоэлемента	
Механика конструкции			
22	Зубчатая передача	Получение знаний, умений и навыков в разработке и применении зубчатых передач	1
23	Гусеничная передача	Получение знаний, умений и навыков в разработке и применении гусеничной передачи	1
24	Кулачковая передача	Получение знаний, умений и навыков в разработке и применении кулачковой передачи	**1**
3. Юный робототехник			
Мобильная робототехника			
25	Робоплатформа NikiRobot	Отработка и закрепление навыков в области конструирования и программирования колёсных роботов.	**1**
26	Объезд препятствий	Отработка и закрепление навыков в области конструирования и программирования колёсных роботов.	**1**
27	Поиск объекта	Отработка и закрепление навыков в области конструирования и программирования колёсных роботов	1
28	Захват объекта	Отработка и закрепление навыков в области конструирования и программирования колёсных роботов	1
29	Движение по линии	Отработка и закрепление навыков в области конструирования и программирования колёсных роботов	1
30	Управление по IR	Отработка и закрепление навыков в области конструирования и программирования колёсных роботов	1
31	Управление по Bluetooth	Отработка и закрепление навыков в области конструирования и программирования колёсных роботов	1
Инженерная робототехника			
32	Сортировщик цвета	Отработка и закрепление навыков в области конструирования и программирования роботов с определённой инженерной задачей	**1**

33	Манипулятор	Отработка и закрепление навыков в области конструирования и программирования роботов с определённой инженерной задачей	**1**
34	Роботанк	Отработка и закрепление навыков в области конструирования и программирования роботов с определённой инженерной задачей	**1**
35	Робот Муравей	Отработка и закрепление навыков в области конструирования и программирования роботов с определённой инженерной задачей	**1**
36	Ультразвуковой терменвокс	Отработка и закрепление навыков в области конструирования и программирования роботов с определённой инженерной задачей	**1**
37	Автоматизированные часы	Отработка и закрепление навыков в области конструирования и программирования роботов с определённой инженерной задачей	**1**
4. Физические эксперименты			
38	Равномерное прямолинейное движение	Получение знаний, умений и навыков в области проведения физических опытов с использованием роботизированного набора	**1**
39	Равноускоренное прямолинейное движение	Получение знаний, умений и навыков в области проведения физических опытов с использованием роботизированного набора	**1**
40	Колебания	Получение знаний, умений и навыков в области проведения физических опытов с использованием роботизированного набора	**1**
41	Криволинейное движение	Получение знаний, умений и навыков в области проведения физических опытов с использованием роботизированного набора	**1**
CyberPi			
42	Знакомство с CyberPi	Строение устройства. Обзор по портам и датчикам. Расширения к программированию. Примеры.	1
43	Звуковая машина	Получение знаний в области программирования мелодии с CyberPi и использовании RGB светодиодов	1

44	Диктофон	Получение знаний и навыков в области программирования для записи звука и голоса с дальнейшим воспроизведением	1
45	Итерация диктофона	Отработка навыков по работе с диктофоном. Углубление в программирование CyberPi	1
46	Игровой контроллер	Получение знаний и опыта в программировании CYberPi в качестве контроллера. Применение контроллера для управления спрайтами в mBlock5	1
47	Данные с датчиков	Получение знаний и опыта в области программирования CyberPi для использования встроенных датчиков шума и освещённости	1
48	Цветовой микшер	Получение знаний и опыта в программировании цвета по RGB схеме	1
49	Измерение силы встряски	Получение знаний и опыта в области программирования CyberPi для использования встроенного гироскопа	1
50	Подарок с сигнализацией	Отработка навыков программирования гироскопа и динамика в CyberPi	1
Комбинированная робототехника			
51	Свободное падение тела. Построение графика	Получение знаний и опыта в области сбора данных с экспериментальной установки и дальнейший их анализ и построение графика	1
52	Вычисление угловой и линейной скоростей вращающегося тела	Получения знаний и опыта в области программирования CyberPi для получения данных с гироскопа и на их основе вычислять взаимосвязанные физические величины.	1
53	Мобильный робот картограф	Отработка навыков программирования CyberPi для записи данных по положению робота в пространстве. И Отработка в области конструирования мобильных роботов с учётом их одометрии	1
54	Робот исследователь	Отработка нывыком сборки мобильного робота и программирования CyberPi для проведения исследовательской работы по сбору данных с окружающей среды	1
итого			54

Список используемой литературы:

1. КЛИК. Методический сборник по образовательной робототехнике. Корягин А.В.
2. Физические эксперименты и опыты с LEGO MINDSTORMS Education EV3. Корягин А.В., Смольянинова Н.М. – М.: ДМК Пресс, 2020 г.
3. Игровая робототехника для юных программистов и конструкторов MBOT и MBLOCK. А.Т. Григорьев, Ю.А. Винницкий – СПб.: БХВ-Петербург, 2019 г.
4. Образовательная робототехника. Сборник методических рекомендаций и практикумов. Корягин А.В. Смольянинова Н.М. – М. : ДМК Пресс, 2015 г.
5. Образовательная робототехника. Рабочая тетрадь. Корягин А.В. Смольянинова Н.М. – М.: ДМК Пресс, 2015 г.

Программное обеспечение

1. mBlock5
2. Arduino IDE