О.Б. Сельдюгаев, Д.А. Казимова, И.А. Самойлова

РОБОТООРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Караганда
2023

КАРАГАНДИНСКИЙ УНИВЕРСИТЕТ
ИМЕНИ АКАДЕМИКА Е.А. БУКЕТОВА

О.Б. Сельдюгаев, Д.А. Казимова, И.А. Самойлова

РОБОТООРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Учебник

Караганда
2023

УДК 004.896 (075.8)
ББК 32.816 я73
С29

Рекомендовано к печати Ученым советом
Карагандинского университета имени академика Е.А. Букетова
(Протокол № 16 от 30.05.2023 г.)

Рецензенты

Ж.Ж. Байгунчеков, д-р техн. наук, проф., акад. НАН РК, Казахский национальный университет имени аль-Фараби;

Р.Б. Муканов, д-р PhD, ассоц. проф. кафедры транспортной техники и логистики, Торайгыров университет;

М.Б. Шащанова, канд. техн. наук, проф., Академия «Bolashaq»;

Л.А. Зиновьев, канд. хим. наук, вед. науч. сотр. научно-исследовательской лаборатории «Прикладная механика и робототехника», Карагандинский университет имени академика Е.А. Букетова

Сельдюгаев О.Б.
С29 Роботоориентированное программирование : учебник / О.Б. Сельдюгаев, Д.А. Казимова, И.А. Самойлова. — Караганда : Издательство Карагандинского университета имени академика Е.А. Букетова, 2023. — 237 с.

ISBN 978-601-362-151-7

Учебник включает в себя описание конструктивных элементов робототехнических наборов на базе Lego, Robot Kit и Bioloid; содержит примеры программирования данных типов роботов, что позволяет обучающимся быстро понять методику их программирования.

Предназначен для обучающихся технических, инженерных и естественно-научных направлений.

УДК 004.896 (075.8)
ББК 32.816 я73

ISBN 978-601-362-151-7

Введение

В настоящее время реконструкция специального высшего образования является одной из важнейших задач нашего государства. Поэтому важно создавать и распространять новые учебники и вспомогательные учебные материалы, посвященные новым техническим достижениям; также очень важным аспектом является разработка учебно-методической литературы на высоком научно-методическом уровне.

В связи с этим для обучающихся технических, инженерных и естественно-научных направлений высших учебных заведений, а также и преподавателей, разработан данный учебник по робототехнике, посвященный программированию роботов Lego EV-3, Ultimate Robot Kit и Bioloid на базе платформ программирования Lego Mindstorms_EV3, MBLOCK и RoboPlus. Данные типы робототехнических систем получили большое распространение в образовательной системе Республики Казахстан (особенно Lego EV-3), но в связи с тем, что данные системы изготовлены за пределами стран СНГ наблюдается острая нехватка учебных материалов для обучения управлению данными робототехническими системами.

Учебник «Роботоориентированное программирование» включает в себя описание технических характеристик робототехнических наборов Lego Mindstorms_EV3, Ultimate Robot Kit и Bioloid. Кроме того, в разработанном учебнике приведено большое количество примеров программирования данных роботов с подробным объяснением работы каждой изложенной программы. Большое внимание уделяется функциональному назначению каждого графического знака в операторах программирования. Методика изложения программирования роботов Lego EV-3, Ultimate Robot Kit и Bioloid в разработанном учебнике позволяет обучающимся быстро освоить начальное управление данными роботами и на основе усвоенных знаний самостоятельно разрабатывать более сложные модели и программы управления.

Первая глава учебника посвящена базовым знаниям о комплектующих робототехнического набора Lego

Mindstorms_EV3 и в ней также подробно рассмотрены алгоритмы программ по управлению данным типом робота. В приведенных программах производится постепенное усложнение действий, производимых роботом: начинается глава примерами простых движений, а заканчивается программой прохождения лабиринта с подробным изложением логики движения робота. Материал, изложенный в первой главе, позволяет учащимся понять основы управления роботами: какие моторы включить, на какое время включаются моторы, в каком направлении моторы вращаются или с какой скоростью моторы должны вращаться. Сделав все упражнения, приведенные в первой главе, обучающийся получает общее представление о том, на что способны роботехнические системы.

Вторая глава знакомит с управлением робота Ultimate Robot Kit: подробно рассмотрено предназначение каждого из элементов набора, техническая схема расположения моторов, манипуляторов и плат электронных компонентов. Показано функциональное назначение каждого порта контроллера Ultimate Robot Kit: подробно рассмотрены алгоритмы приведенных во второй главе программ по управлению гусеницами, манипулятором и клешней. Программы идут в порядке возрастания сложности – от простых движений, производимых каждым мотором в отдельности, к одновременному управлению всех моторов (одновременно в движении находятся гусеницы, манипулятор и клешня). Также во второй главе изложена схема подключения и программирования робота Ultimate Robot Kit к среде программирования MBLOCK.

Если в первой главе обучающиеся получали общие навыки управления простейшими легкими неинерциальными движущимися системами при помощи очень простого графического языка, то для программирования Ultimate Robot Kit необходимо применять более сложную систему программирования Скретч, так как механика данной системы более совершенная, чем в Lego EV-3. Робот Ultimate Robot Kit значительно более тяжелый, чем Lego EV-3 и программирование данного типа робота позволяет учащимся получить навык управления тяжелыми, обладающими инерцией системами. При

этом Ultimate Robot Kit использует значительно более мощные моторы, движения можно производить значительно более точные. Среда программирования MBLOCK позволяет также программировать движения при помощи Си- подобного языка и, освоив программирование на данном языке Скретч, обучающиеся могут использовать контроллер Ultimate Robot Kit для управления реальными объектами.

Большой интерес представляет собой третья глава, в которой подробно излагается и иллюстрируется материал о работе с человекообразным роботом Bioloid. Показано функциональное назначение всех кнопок и разъемов на корпусе контроллера Bioloid, рассмотрено механическое устройство человекообразного робота Bioloid. Подробно приведена методика работы с программной средой RoboPlus, что значительно облегчает обучающимся освоение данной программной среды. Показаны и прокомментированы программы по управлению роботом Bioloid: сначала представлены примеры по программированию отдельного мотора (в том числе с использованием датчиков) и далее подробно рассмотрены алгоритмы программ по сложному движению одной и двух рук одновременно с использованием нескольких моторов одновременно.

Глава 1 Основные материальные объекты и методика программирования Lego Mindstorms_EV3

1.1 Основные материальные объекты Lego набора EV3

Базовый набор содержит всё необходимое для обучения с помощью технологий Lego Mindstorms_EV3. Он позволяет обучающимся конструировать, программировать и тестировать их решения, используя настоящие технологии робототехники. Процесс работы с набором включает в себя сборку и программирование робота. Программирование осуществляется в специальном программном обеспечении, которое скачивается бесплатно с сайта Lego Education.

Набор Lego Mindstorms_EV3 изображен на рисунке 1.1 [1].

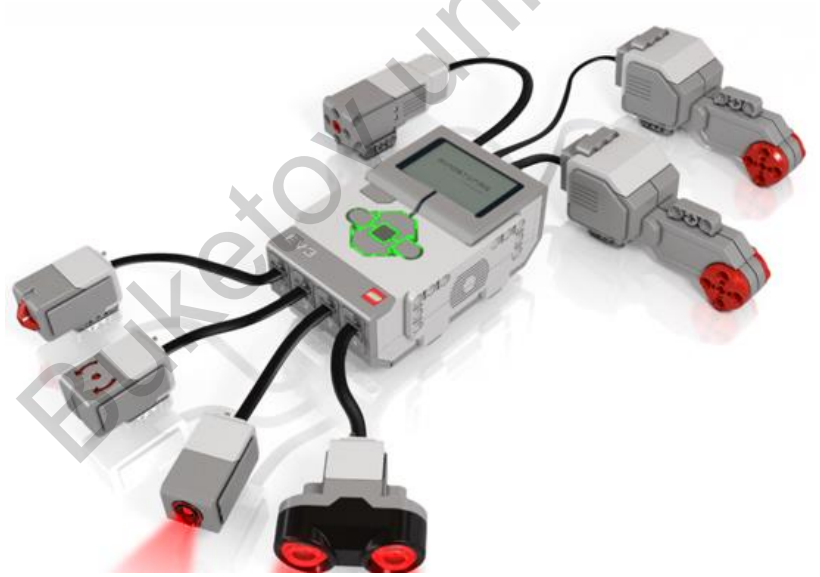


Рисунок 1.1 Подключение компонентов EV3

Состоит из следующих комплектующих:

1 программируемый блок (рисунок 1.2)

Рисунок 1.2 Программируемый блок Lego Mindstorms EV3

3 мотора:

1 средний мотор (рисунок 1.3)
2 больших мотора (рисунок 1.4)

Рисунок 1.3 Блок «Средний мотор»

Рисунок 1.4 Блок «Большой мотор»

5 датчиков:

1 ультразвуковой датчик расстояния (рисунок 1.5)

Рисунок 1.5 Ультразвуковой датчик

2 датчика касания (рисунок 1.6)

1 датчик цвета (рисунок 1.7)

1 гироскоп (рисунок 1.8)

Аккумуляторная батарея для робота

528 деталей Lego Technic.

Рисунок 1.6 Датчик касания

Рисунок 1.7 Световой датчик – различает семь цветов

Рисунок 1.8 Датчик поворота – гироскоп

Для того, чтобы моторы и датчики работали, они должны быть подключены к модулю EV3. **Модуль EV3 —** это центр управления, который приводит в действие робота Lego (рисунок 1.2) [2]. Данный модуль имеет порты с двух сторон корпуса: порты для подключения моторов и порты для подключения датчиков. Порты с буквенной латинской нумерацией **(A, B, C, D)** предназначены для подключения Больших и Средних моторов (рисунок 1.9).

Рисунок 1.9 Порты для подключения Больших и Средних моторов

Порты с цифровой нумерацией (**порт 1, порт 2, порт 3, порт 4**) предназначены для подключения датчиков (рисунок 1.10).

Рисунок 1.10 Это порты для подключения датчиков

При написании программ можно вручную назначить цифровые порты для подключения датчиков. По умолчанию порты будут назначены следующим образом:

порт 1: датчик касания;

порт 2: гироскопический датчик/датчик температуры;

порт 3: датчик цвета;

порт 4: ультразвуковой датчик/инфракрасный датчик.

Под портами 1, 2 находится гнездо подключения питания для аккумуляторов робота Lego.

Если во время программирования модуль EV3 подключен к компьютеру, программное обеспечение автоматически определит, какой порт используется для каждого датчика или мотора. С помощью плоских черных соединительных кабелей подключите моторы к модулю EV3, используя порты вывода A, B, C и D. По умолчанию порты для моторов будут назначены следующим образом:

порт A: средний мотор;

порты B и C: два больших мотора;

порт D: большой мотор.

Кроме того, модуль EV3 имеет встроенный экран, содержащий четыре основных раздела (рисунок 1.11).

На рисунке 1.12 показано расположение всех портов контроллера Lego EV-3. Под экраном располагаются шесть кнопок управления.

В верхней части экрана можно видеть 4 основных раздела интерфейса управления роботом Lego EV-3:

- В первом и втором разделах представлены все записанные на этом программируемом модуле программы.
- Третий раздел позволяет произвести диагностику портов, моторов, памяти робота.
- Четвертый раздел служит для настройки работы программируемого модуля - в нем указано время отключения при простое робота, уровень громкости звукового сигнала, там же прописана возможность подключения удаленного управления через Bluetooth или Wi-Fi.

Рисунок 1.11 Встроенный экран Lego EV-3

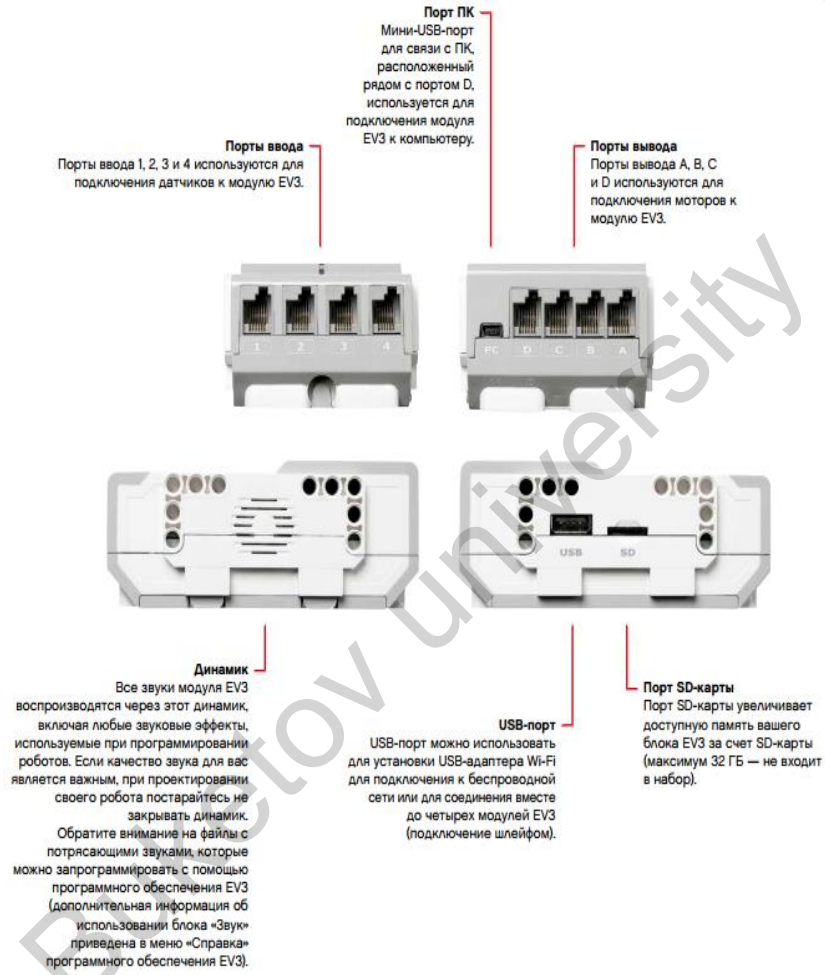


Рисунок 1.12 Порты контроллера Lego EV-3

На лицевой панели робота Lego имеется 6 кнопок управления. Кнопки включения и выключения указаны стрелками, остальные нужны для перемещения курсора управления вправо-влево, вверх-вниз (рисунок 1.13).

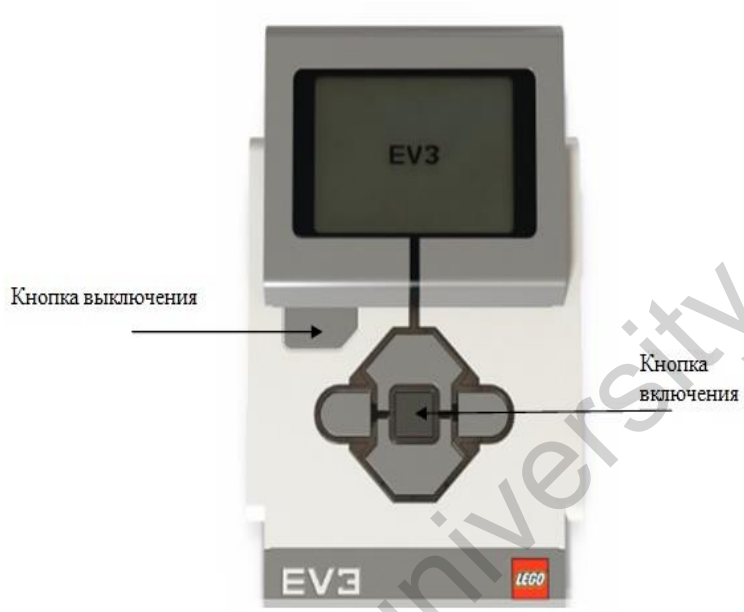


Рисунок 1.13 Кнопки управления

Для включения робота нужно нажать кнопку включения на 1 сек.

Кнопка выключения также используется для выхода из текущего раздела в первый раздел меню управления. Если нажать на кнопку выхода в первом разделе на экране появится рисунок, предлагающий выключить робота – для выключения робота при этом нужно выбрать кнопку с галочкой / справа/. Для отказа от выключения робота еще нажимаем на кнопку выключения.

Для удаления программы из памяти робота Lego надо выбрать, какую именно программу нужно удалить /**навести на нее курсор**/ – далее три раза нажать кнопку включения – появится значок, предлагающий удалить программу – **выбрать действие удалить.**

1.2 Программирование робота Lego Mindstorms_EV3

Для создания программ управления роботом Lego используется программная среда **Lego MindstormsE Education EV3 StudentEdition** [3]. Данный программный комплекс содержит в себе самоучитель для работы с роботом Lego, кроме того, в данный комплекс входит конструктор команд, позволяющий написать любую программу для управления роботом Lego.

Производим следующие действия:

1. Запускаем приложение **Lego Mindstorms Education EV3 StudentEdition.** Открывается страница, в центре которой изображены различные собранные модели робота Lego**.**

В левой части экрана находятся 5 кнопок управления:

- расширенный набор;
- основной набор;
- краткое руководство;
- файл;
- самоучитель.

2. Открываем вкладку **«Самоучитель»**.

На открывшейся странице открываем вкладку «**Основы»**.

3.Выбираем пункт «**Настройка конфигурации блоков**».

4.Нажимаем кнопку «**Открыть**» в правой части экрана. Открывается поле «**Настройка конфигурации блоков**», в котором можно конструировать любые комбинации команд для робота Lego.

Снизу открывшейся страницы находятся 7 разноцветных вкладок, в которых содержится полный набор всех команд управления роботом Lego:

- **действие** /зеленая папка/;
- **управление операторами** /оранжевая папка/;
- **датчик** /желтая папка/;
- **операции с данными** /красная папка/;

- **дополнения** /синяя папка/**;**
- **мои блоки** /голубая папка/.

Выбираем папку **действие** /зеленая папка/. В данной папке содержатся команды, определяющие какой тип управления конкретного механизма робота мы собираемся применить (рисунок 1.14). На данном рисунке имеется 7 значков, значения которых приведены ниже (слева направо):

- средний мотор;
- большой мотор;
- рулевое управление;
- независимое управление моторами;
- экран;
- звук;
- индикатор состояния модуля.

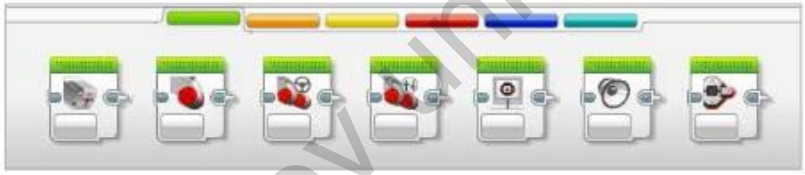


Рисунок 1.14 Основное меню настройки блоков конфигурации

Комплект робота Lego, помимо контроллера, содержит два вида различных моторов – **Большой мотор** 2 штуки (рисунок 1.4**)** и **Средний мотор** 1 штука (рисунок 1.3). Большие моторы вращают колеса робота, а средний мотор применяется, например, для работы движущегося манипулятора.

1.2.1 Запись программы с компьютера в процессор

1. Подключаем контроллер Lego EV-3 через порт PC к компьютеру через USB-порт (рисунок 1.15).

Рисунок 1.15 Подключение к компьютеру контроллера Lego

На экране в рабочей программе Lego Mindstorms_EV3 имеется окно (рисунок 1.16).

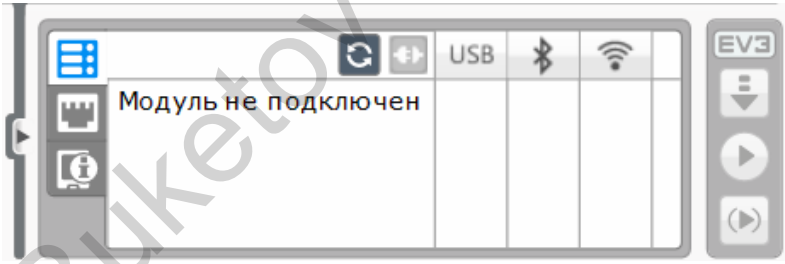


Рисунок 1.16 Окно программы Lego Mindstorms Education EV3

В правом нижнем углу программы Lego MindstormsE Education EV3 Student Edition активируется меню, позволяющее произвести запись разработанной программы в контроллер Lego (рисунок 1.17) [4].

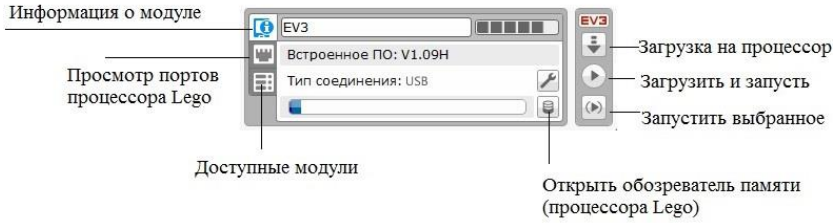


Рисунок 1.17 Меню записи программы в контроллер Lego

А) При нажатии на кнопку **Загрузить и запустить** происходит запись в оперативную память контроллера Lego и сразу происходит выполнение данной программы. Контроллер Lego при этом подключен USB-кабелем к управляющему компьютеру.

Б) Запустить выбранное. Производится активация команды управления, выбранной на экране компьютера.

В) Кнопка «**Загрузка в контроллер**» Разработанная программа записывается в модуль.

Д) При нажатии «**Просмотр портов**» на экран выводится информация о том, какие порты активированы в данный момент в контроллере Lego.

2. Нажимаем кнопку **«Загрузка в контроллер»**, в результате на экране процессора Lego, в разделе №2 экрана появляется название записанной программы. Под этой ссылкой появляется надпись **«Program».** Наводим курсор на «**Program**» и нажимаем на кнопку «**Пуск**» на контроллере Lego. Это приводит к активации выбранной программы.

3. На экране модуля EV3 нажимаем на название загруженной программы.

Любая программа для робота Lego начинается с графического символа:

 , означающего начало новой программы. Все команды программы записываются после данного символа и выполняются поочередно. Если стоит задача начать перемещение робота, то необходимо дать указание Большим моторам как двигаться – а именно необходимо указать время работы моторов и с какой скоростью должны работать моторы. Кроме того, необходимо указать, как должен двигаться робот – только прямо (вперед или назад), по кругу определенного радиуса, с поворотом на определенное количество градусов. Для этого используется один из 2 режимов управления Большими моторами.

Большие моторы могут управляться в режиме **Рулевое управление** и **Независимое управление моторами.**

1.2.2 Блок «Рулевое управление»

Блок «**Рулевое управление**» (рисунок 1.18) может заставлять вашего робота двигаться вперед, назад, поворачиваться или останавливаться. Вы можете регулировать рулевое управление, чтобы заставить вашего робота двигаться прямо, по дуге или делать резкие повороты [5].

Используйте блок «**Рулевое управление**» для программируемых роботизированных средств, в которых **имеются два больших мотора**, где один мотор управляет левой стороной транспортного средства, а второй мотор управляет правой стороной. Блок «**Рулевое управление**» управляет обоими моторами одновременно, чтобы ваш робот двигался в выбранном вами направлении. Рекомендуется использовать данный блок «**Рулевое управление**» при задании движения по прямой линии.

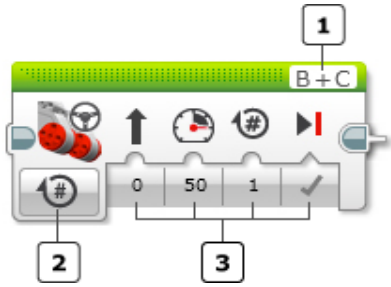


Рисунок 1.18 Блок «Рулевое управление»

Рассмотрим, что означают обозначения на рисунке 1.18.

1 Выбор порта (в данном случае показывается, что **Большие моторы** подключены к портам B и C).

2 Выбор режима движения (можно приказать роботу сделать поворот вокруг своей оси; поворот на определенное количество градусов колес; поворот или прямолинейное движение в течение заданного интервала времени; поворот на определенное количество полных оборотов колес). Для выбора нужного режима наводим на кнопку 2 – появляется выпадающее меню, в котором можно выбрать нужный режим (рисунок 1.19):

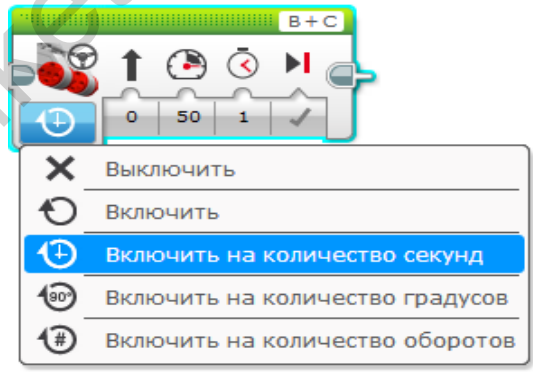


Рисунок 1.19 Выпадающее меню

На приведенном рисунке 1.19 с показанным выпадающим меню задана команда, приказывающая двигаться роботу **вперед** в режиме **Рулевое управление,** со скоростью вращения обоих Больших моторов **50**, в течение 1 секунды. По истечении 1 секунды срабатывает команда остановки робота (значок ).

На рисунке 1.19 прописана команда движения робота вперед в режиме **Рулевое Управление**, на мощности обоих Больших моторов 50, в течение одного полного оборота колес 

.

Данное обозначение  означает набор команд для режима «**Рулевое управление»**.

Этот значок означает, что робот будет двигаться только прямо.

Цифра 50 на данном значке показывает текущую скорость вращения обоих моторов (максимальная скорость – 100); от этого параметра зависит скорость движения робота. При этом робот движется **вперед**. Если задать скорость вращения обоих моторов со знаком минус робот будет двигаться назад.


Данный значок означает приказ на остановку робота.

Данный значок означает движение по кругу в режиме Рулевое управление. Эта команда показывает, что робот при движении вперед (скорость вращения обоих моторов +50), уклоняется вправо, двигаясь по кругу определенного радиуса. Радиус круга определяется числом, указанным в иконке

(в данном случае это число 35). Чем больше это число, тем меньше радиус описываемого роботом круга. Если в поле иконки задана цифра 180 – робот будет вращаться на месте **вправо**. Направление вращения и радиус вращения задаются при помощи регулятора  .

При необходимости режим **Рулевое управление** можно выключить (рисунок 1.20).

Рисунок 1.20 Режим «Выключить рулевое управление»

На рисунке 1.20 показана команда, приказывающая роботу выключить режим **Рулевое управление** и приказ на остановку робота Lego.

 Это кнопка выключения режима **Рулевое управление**. Данная функция применяется в тех случаях, когда необходимо включить режим **Бесконечное движение** (при этом нет никаких ограничений на время движения робота). Для этого нажимаем на кнопку 2 (выбор режима – рисунок 1.19) и в открывшемся меню выбираем функцию **Включить.**

 – кнопка включения режима **Рулевого управления**. При ее активации на экране появляется команда, приказывающая роботу бесконечно двигаться вперед**,** скорость вращения обоих моторов 50 (рисунок 1.21). Режим бесконечное движение удобен для программирования робота при использовании датчиков.

Рисунок 1.21 Режим «Бесконечное движение на рулевом управлении»

Существует возможность включить Большие моторы на определенное количество секунд (**включаются оба мотора на 1 сек**.) и робот при этом будет двигаться прямо (рисунок 1.22) со скоростью 50.

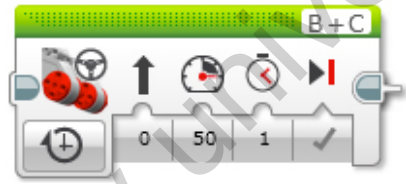


Рисунок 1.22 Прямолинейное движение в режиме ограничения времени

Включить моторы на мощности 50 для поворота колес на определенное количество градусов (**в данном случае оба колеса поворачиваются на 360°**) и робот при этом будет двигаться прямо (рисунок 1.23).

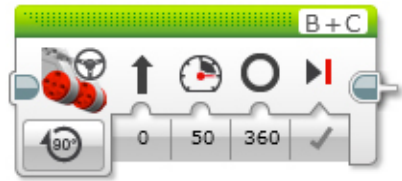


Рисунок 1.23 Прямолинейное движение, определяемое углом поворота колес робота

Включить моторы со скоростью вращения 50 для поворота колес на определенное количество оборотов (**в данном случае колеса поворачиваются на 1 полный оборот**) и робот при этом будет двигаться прямо и после одного поворота колес остановится (рисунок 1.24).

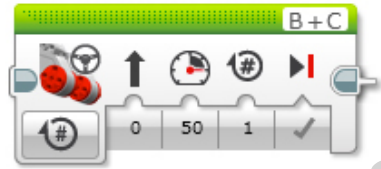


Рисунок 1.24 Прямолинейное движение, определяемое количеством полных оборотов колеса

Для осуществления поворотов удобнее применять режим **«Независимое управление моторами»**, при котором каждый мотор программируется независимо от другого. При одинаковой скорости вращения моторов в одном направлении в режиме «независимое управление» повороты невозможны.

1.2.3 Блок «Независимое управление моторами»

Блок «Независимое управление моторами (на каждый мотор задаются собственные параметры).

Блок «**Независимое управление моторами**» (рисунок 1.25) может заставлять вашего робота двигаться вперед, назад, поворачиваться или останавливаться. Используйте «Независимое управление моторами» для программируемых роботизированных средств, в которых имеются два больших мотора, где один мотор управляет левой

стороной транспортного средства, а второй мотор управляет правой стороной [5]. Можно заставить два мотора вращаться с разными скоростями или в разных направлениях, чтобы ваш робот поворачивался.

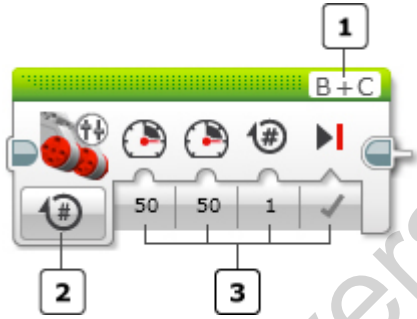
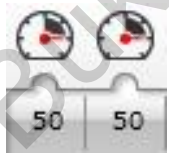


Рисунок 1.25 Режим независимое управление моторами

Рассмотрим значения каждого поля на представленном рисунке 1.25.

Данный значок показывает, что для управления роботом Lego используется режим «**Независимое управление моторами**».



Этот значок показывает, что оба Больших мотора имеют одинаковую скорость вращения 50 (напомним, максимальная скорость вращения каждого мотора 100).

Этот значок показывает, что будет произведен один полный оборот колес в режиме **Независимое управление моторами**.


Галочка, установленная в данном значке под 
означает, что после осуществления всех действий, прописанных в данном командном блоке, робот Lego остановится.

1 Выбор порта - данное поле показывает какие порты контроллера **Lego** используются для управления моторами (используются порты B и C).

2 Выбор режима - в данном поле можно задать определенные действия (поворот робота вокруг своей оси: поворот на определенное количество градусов определенного колеса, поворот в течение определенного времени, поворот на определенное количество полных оборотов колеса). Для выбора нужного режима наводим на кнопку 2 – появляется выпадающее меню, в котором можно выбрать нужный режим:

В данном примере (рисунок 1.26) в режиме **Независимое управление моторами** левый и правый большие моторы работают на полную скорость и вращаются в разные стороны, что приводит к повороту робота на месте.

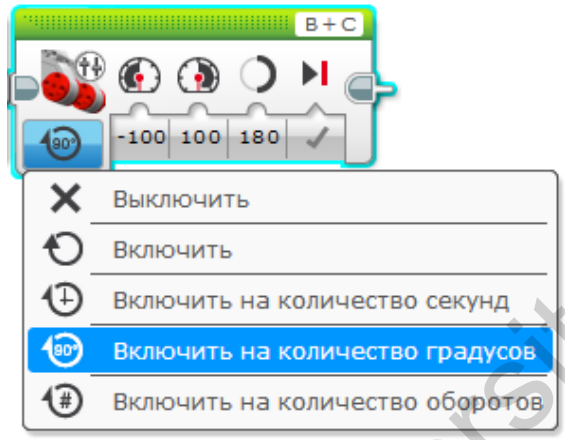


Рисунок 1.26 Настройка параметров режима независимое управление моторами

Колеса при этом должны повернуться на **180 градусов**. Затем робот останавливается. В выпадающем меню можно также выбрать поворот на определенное количество оборотов колес, или производить поворот в течение заданного времени. В данном примере в выпадающем меню выбран **режим поворота колес** Больших моторов на определенное количество градусов.

Примеры использования режима **Независимое управление моторами:**

Рисунок 1.27 Используется режим независимое управление моторами

На рисунке 1.27 показана команда использования режима «**Независимое управление моторами**» на **бесконечное** прямолинейное движение, мощность каждого мотора 50. Для подключения Больших моторов используются порты **B** и **C**.

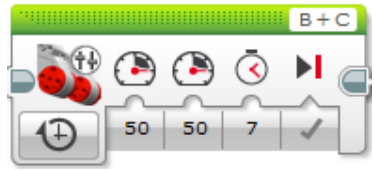


Рисунок 1.28 Режим Независимое управление моторами с подсчетом времени

На рисунке 1.28 приведена команда режима независимое управление Большими моторами «**Включить моторы на 7 секунд**»: левый и правый Большие моторы имеют скорость вращения 50, и через 7 секунд работы Большие моторы выключаются.

- данный значок показывает, что в данном командном блоке время работы обоих Больших моторов равно 7 секундам.

Рисунок 1.29 Режим Независимое управление моторами с подсчетом градусов поворота

На рисунке 1.29 приведена команда режима независимое управление моторами **«повернуть колеса на 360 градусов».**

Рисунок 1.30 Команда выключить режим Независимое управление моторами

На рисунке 1.30 приведена команда «**Выключить Независимое управление моторами**» и остановиться.

 - это кнопка выключения режима **Независимого управление моторами.**

При одинаковой скорости вращения моторов в одном направлении в режиме **«Независимое управление» повороты невозможны.**

Рассмотрим программу движения, состоящую из нескольких командных блоков (рисунок 1.31).

Рисунок 1.31 Пример использования режима Независимое управление моторами

1) Первый командный блок приказывает роботу двигаться **вперед 1 секунду** в режиме **Рулевое управление** на скорости 50 и затем остановиться.

2) Далее включается 2 командный блок, в котором прописана команда перехода в режим **Независимое управление моторами** – в данном режиме Большие моторы работают в противоположных направлениях на скорости вращения **100** – при этом каждое колесо поворачивается **на 90 градусов.** В результате робот на месте повернулся на 30 градусов и затем робот останавливается.

3) Далее включается 3 командный блок – снова происходит переход управления в режим **рулевое управление** и робот движется **прямо** в течение **1 секунды** на скорости 50 – затем робот останавливается.

1.2.4 Блок «Большой мотор»

Робот Lego движется на одном Большом моторе – равновесие соблюдается при этом за счет наличия пассивного колеса.

- данный значок означает набор команд для режима **Большой мотор** [6].

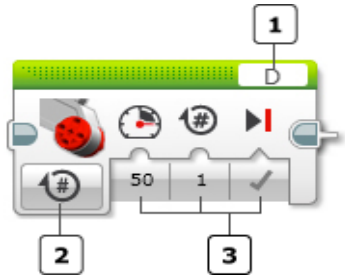


Рисунок 1.32 Режим Большой мотор

1 Выбор порта (в данном случае Большой мотор подключен к порту **D)**.

- цифра **50** на рисунке 1.32 означает текущую скорость вращения **Большого** мотора (максимальная скорость вращения – 100);


- установленная галочка на этой кнопке приказывает **Большому мотору** остановиться в конце движения.

- цифра **1** на рисунке 1.32 означает количество оборотов колеса, которое приводится во вращение **Большим мотором.**

Программа на рисунке 1.32 читается так**:** Большой мотор, подключенный к порту D, работает в течение одного полного оборота колеса на скорости вращения 50, и после того, как Большой мотор повернется на один полный оборот произойдет выключение Большого мотора.

2 - кнопка выбора режима (рисунок 1.33) (поворот вокруг своей оси: поворот колеса на определенное количество градусов, поворот в течение определенного времени, поворот на определенное количество полных оборотов **колеса)**. Для выбора

нужного режима наводим на кнопку 2 – появляется выпадающее меню, в котором можно выбрать нужный режим.

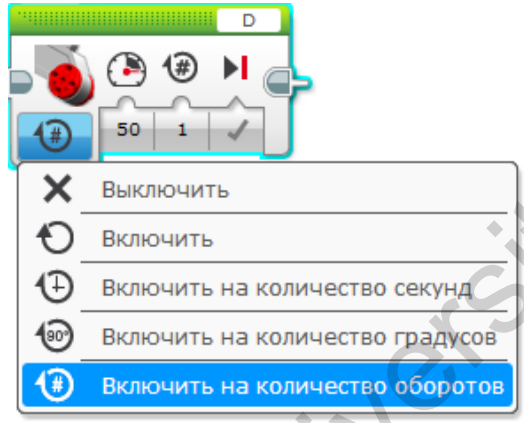


Рисунок 1.33 Подменю кнопки Выбор режима

При нажатии кнопки «Выключить» (рисунок 1.33) данный командный блок **Большой мотор** выключается и появляется значок, показанный на рисунке 1.34.

Рисунок 1.34 Команда отключения режима Независимое управление моторами

При нажатии на крестик на рисунке 1.34 опять появляется выпадающее меню (рисунок 1.33), позволяющее включить режим **Большой мотор** в одном из нескольких вариантов использования:

Рисунок 1.35 Бесконечное движение в режиме Независимое управление моторами

Бесконечное движение на заданной скорости вращения Большого мотора – максимальная скорость - 100.

На рисунке 1.36 изображен командный блок, приказывающий роботу бесконечно двигаться вперед при скорости вращения Большого мотора 50.

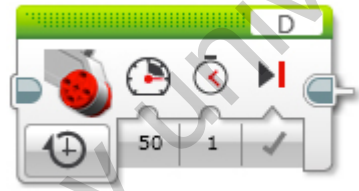


Рисунок 1.36 Включить Большой мотор на заданное количество секунд

На рисунке 1.36 изображен командный блок, приказывающий роботу двигаться вперед на мощности **Большого мотора** 50 в течение 1 секунды. По истечению 1 секунды робот должен остановиться.

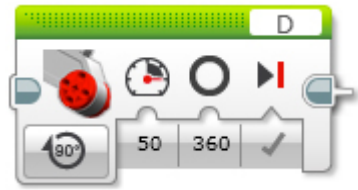


Рисунок 1.37 Включить Большой мотор на заданное количество градусов поворота колеса

 - этот значок означает, что используется режим поворота колеса на определенное количество градусов.

 - этот значок означает количество градусов, на которое провернется колесо, движимое Большим мотором (в данном случае колесо поворачивается на 360 градусов).

Рисунок 1.37 Включить Большой мотор на заданное количество полных оборотов.

 - цифра **1** на рисунке 1.37 означает количество оборотов колеса.

1.2.5 Датчики

Для получения информации об окружающем пространстве используются специальные устройства – **датчики**: они позволяют определять цвет, освещенность, расстояние до ближайших предметов, наличие препятствий на пути, наличие внешнего ультразвукового сигнала и многое другое.

В данном типе робота в комплект входят 2 датчика касания, ультразвуковой датчик, датчик поворота/гироскоп, датчик определения цвета и освещенности [7]. Все датчики подключаются к портам 1, 2, 3, 4 программируемого блока **Lego Mindstorms EV3** (рисунок 1.38)**.** По умолчанию под определенный тип датчика используется зарезервированный номер порта.

Рисунок 1.38 Порты контроллера Lego предназначенные для подключения датчиков

Порт 1: **датчик касания**

Датчик касания – это цифровой датчик, который может определять, когда красная кнопка датчика нажата, а когда отпущена. Это означает, что датчик касания можно запрограммировать для действия в зависимости от трех условий: нажатие, отпускание и щелчок (нажатие и отпускание).

Порт 2: **Датчик поворота робота Lego (Гироскоп)**

Гироскопический датчик – это цифровой датчик, который обнаруживает вращательное движение по одной оси. Если вы будете вращать гироскопический датчик в направлении стрелки на корпусе датчика, датчик может определить скорость вращения в градусах в секунду.

Гироскопический датчик EV3 измеряет вращательное движение робота и изменение его положения, обладает следующими техническими свойствами.

- может использоваться для определения текущего направления вращения;
- точность: +/- 3 градуса на 90 градусов оборота (в режиме измерения наклона);
- может определить максимум 440 градусов/с (в режиме гироскопа);
- частота работы: 1 кГц.

Гироскоп должен устанавливаться рисунком вверх.

Порт 3: **Датчик цвета**

Датчик цвета – это цифровой датчик, который может определять цвет или яркость света,

поступающего в небольшое окошко на лицевой стороне датчика. Данный датчик может работать в трех разных режимах: в режиме «Цвет», в режиме «Яркость отраженного света» и в режиме «Яркость внешнего освещения».

Датчик цвета EV3 различает 7 цветов и может определить отсутствие цвета. Как и в прошлой версии, он может работать как датчик освещенности. Технические свойства:

- измеряет отраженный красный свет и окружающее освещение;
- способен определять различия между белым и черным цветом, или цветами: синим, зеленым, желтым, красным, белым и коричневым;
- частота работы: 1 кГц.

Порт 4: **Ультразвуковой датчик расстояния**

Ультразвуковой датчик – это цифровой датчик, определяющий расстояние до находящегося перед ним объекта. Он делает это, посылая звуковые волны высокой частоты и измеряя время, за которое звук отразится назад к датчику.

К основной функции ультразвукового датчика EV3 добавилась еще одна — он также может «слушать» ультразвуковые колебания, испускаемые другими датчиками ультразвука.

Технические свойства следующие

- может измерять расстояние в диапазоне 3–250 см;
- точность измерений: +/- 1 см;
- дискретность результата измерений: 0.1 см;
- может быть использован для поиска других активных ультразвуковых датчиков (в режиме прослушивания);

- красная led-подсветка вокруг «глаз».

Немигающий световой индикатор вокруг «глаз» сенсора говорит о том, что датчик находится в режиме «Измерение».

Мигающий световой индикатор сообщает, что датчик находится в режиме «Присутствие». В режиме «Присутствие» этот датчик может обнаруживать другой ультразвуковой датчик, работающий поблизости.

1.2.6 Детальная инструкция по работе с датчиками

Датчик касания

Датчик касания определяет, нажата ли кнопка на передней поверхности датчика. Датчик касания можно использовать, например, для того, чтобы определить, когда робот наезжает на что-либо. Также на датчик касания можно надавить пальцем, чтобы он сработал.

Датчик касания может показывать, что он либо нажат, либо нет. Он не может измерить, как далеко или насколько сильно вдавлена кнопка. Датчик касания дает логические данные (**истина или ложь**) или логический **ноль** или логическая **единица**. Положение кнопки датчика касания называется ее состоянием, при этом истина соответствует нажатию, а ложь соответствует отсутствию нажатия (освобождению).

Существует три режима использования датчика касания.

1) Когда кнопка не нажата на порте датчика касания установлен **логический ноль**. Соответственно, когда кнопка нажата, устанавливается **логическая единица**.

Датчик касания также может отслеживать была ли раньше нажата кнопка и затем отпущена. Это называется «Щелчок» и

применяется, например, для обнаружения нажатия пальцем.

Для использования датчиков нужно сообщить роботу о наличии этих датчиков при помощи командного блока  «**логическое ожидание**».

При нажатии на кнопку логического ожидания выпадает меню, которое позволяет выбрать необходимый тип датчика и необходимый режим использования данного датчика. На рисунке 1.39 показан командный блок **датчик ожидания** с выпадающим меню, в котором произведен выбор датчика касания в режиме «**сравнение – состояние**». В данном состоянии на порте 1, к которому подключен датчик касания, находится состояние «**логический ноль**», которое изменяется на состояние «**логическая единица**» при нажатии датчика касания.

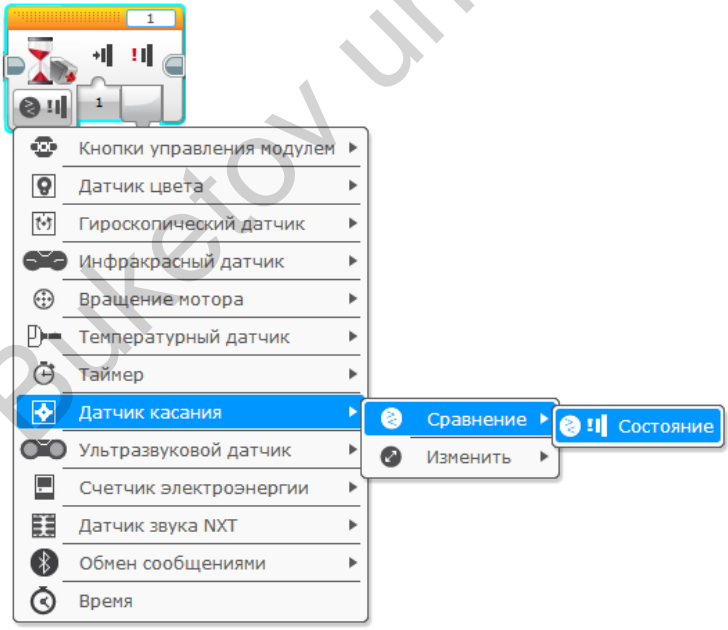


Рисунок 1.39 Меню Командного блока ожидания

 Данный значок означает, что **командный блок ожидания** использует **датчик касания**.

 Данный значок означает, что при нажатии, на порте, к которому подключен датчик касания, устанавливается **логическая единица**.

 Данный значок показывает, что блок ожидания использует порт 1 процессора EV3.

Программа на рисунке 1.40 состоит из трех командных блоков.

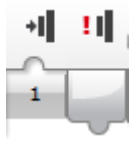
Первый блок приказывает в режиме рулевого управления двигаться **бесконечно долго прямо**.

Рисунок 1.40 Пример использования датчика касания

Далее **командный блок ожидания** с подключенным датчиком касания указывает, что при нажатии на датчик касания на порте 1 меняется логическое состояние с **логического нуля** на **логическую единицу**. Пока датчик касания не сработает,

командный блок ожидания не включится, и робот будет двигаться бесконечно долго прямо.

После срабатывания датчика касания происходит выключение рулевого управления, и робот останавливается.



- цифра **1** в поле означает, что движение прекратится при нажатии на датчик. Пока датчик не нажат, на входе порта **1 логический ноль**, при нажатии на датчик на порт **1** поступает **логическая единица** и происходит срабатывание блока ожидания.

2) Помимо стандартного режима, когда по умолчанию на порте программного блока установлен **логический ноль** и срабатывание командного блока ожидания происходит при изменении логического ноля на **логическую единицу**, существует **обратный режим** – по умолчанию на порте установлен **логическая единица** и при отпускании датчика касания происходит смена логической единицы на **логический ноль**.

На рисунке 1.41 представлена программа, в которой смена логического состояния на порте 1 происходит при отжатии датчика. По умолчанию на порте 1 была логическая единица при нажатом датчике касания, при этом робот производил бесконечное движение прямо.

Рисунок 1.41 Использование датчика касания в Обратном режиме

При отжатии датчика на порте 1 установился логический ноль, сработал **командный блок ожидания,** и робот остановился. Важно помнить, что все командные блоки, находящиеся в программе после командного блока ожидания неактивны (не работают), **пока не сработает датчик.**

- данный значок означает, что **логический ноль** на порте 1 устанавливается после отжатия датчика касания.

3) Существует также режим срабатывания датчика касания, когда регистрируется **щелчок**, то есть датчик касания должен нажаться и отжаться.

- цифра **2** в поле означает, что срабатывание блока ожидания от датчика касания произойдет при щелчке (надо нажать и отпустить).

1.2.7 Применение ультразвукового датчика в Lego EV-3

Ультразвуковой датчик может измерять расстояние до находящегося перед ним объекта.

Он делает это, посылая звуковые волны и измеряя время, которое требуется, чтобы отраженный звук вернулся к датчику. Частота звука слишком высока, чтобы звук можно было услышать («ультразвук»).

Расстояние до объекта можно измерить либо в дюймах, либо в сантиметрах. Вы можете воспользоваться этим, например, для того, чтобы заставить вашего робота остановиться на определенном расстоянии до стены.

Также вы можете использовать ультразвуковой датчик для обнаружения другого ультразвукового датчика, работающего поблизости. Например, вы можете использовать его для обнаружения присутствия поблизости другого робота, который использует ультразвуковой датчик. В этом «пассивном» режиме датчик прослушивает, но не посылает звуковые сигналы [8].

Немигающий световой индикатор вокруг «глаз» сенсора говорит о том, что датчик находится в режиме **«Измерение»**.

Мигающий световой индикатор сообщает, что датчик находится в режиме **«Присутствие»**. В режиме **«Присутствие»** этот датчик может обнаруживать другой ультразвуковой датчик, работающий поблизости.

Данные ультразвукового датчика

Ультразвуковой датчик может выдавать следующие данные:

Данные	Тип	Интервал	Примечания
Расстояние в сантиметрах	Числовое значение	0 – 255	Расстояние до объекта в сантиметрах.
Расстояние в дюймах	Числовое значение	0 – 100	Расстояние до объекта в дюймах.
Ультразвуковое обнаружение	Логическое значение	Истина/Ложь	Истина, если обнаружен другой ультразвуковой датчик.

Для использования ультразвукового датчика в программе нужно установить **командный блок ожидания**, в котором необходимо выбрать ультразвуковой датчик (рисунок 1.42). Выбираем ультразвуковой датчик – далее используем режим **сравнение** и указываем расстояние в сантиметрах (можно в дюймах). Действия, прописанные после командного блока

ожидание, произойдут только, если выполнится прописанное в командном блоке **ожидание** условие**.**

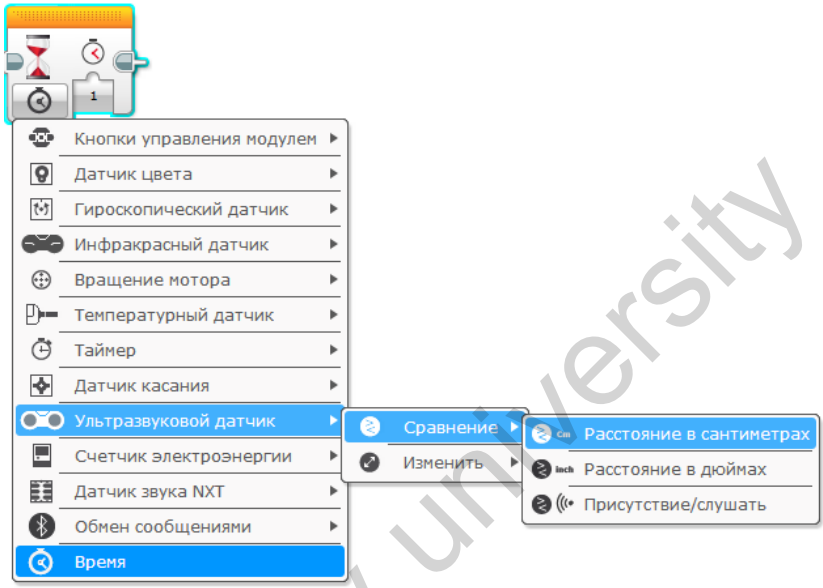


Рисунок 1.42 Установка использования ультразвукового датчика

В отличие от контактного датчика ультразвуковой датчик позволяет получить информацию о наличии препятствия задолго до столкновения робота с препятствием (до 2,5 метра).

Примеры использования ультразвукового датчика

Пример 1. Остановиться на определенном расстоянии перед стеной.

Программа на рисунке 1.43 заставляет робота Lego перемещаться бесконечно вперед до тех пор, пока ультразвуковой датчик не обнаружит что-либо на расстоянии ближе 35 сантиметров, затем робот останавливается.

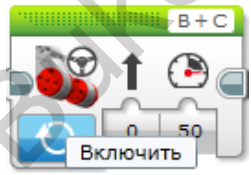
Рисунок 1.43 Пример использования ультразвукового датчика

Данная программа состоит из 3 командных блоков:

Первый командный блок приказывает роботу в режиме **Рулевое управление** бесконечно двигаться вперед на мощности обоих моторов 50 (максимальная мощность 100).

Второй командный блок **Ожидание** приказывает использовать **ультразвуковой датчик** в режиме ожидание. Робот будет двигаться пока расстояние до препятствия меньше 35 см. Показано, что ультразвуковой датчик подключен к процессору через **порт 4.**

Третий командный блок срабатывает только при условии срабатывания второго командного блока и производит отключения режима **Рулевое управление** и производит затем остановку робота.



- данный командный блок приказывает роботу Lego двигаться бесконечно вперед в режиме **Рулевое управление** со скоростью 50 (для подключения Больших моторов используются порты процессора **B** и **C**).

- командный блок **Ожидание** показывает, что используется «Ультразвуковой датчик – в режиме **Сравнение** – Расстояние в сантиметрах».

 данный значок показывает, что используется командный блок **Ожидание** (в котором можно выбрать тип используемого датчика).

 данный значок прямо указывает, что командный блок **Ожидание** использует ультразвуковой датчик.

 это логотип режима **Сравнение** при использовании ультразвукового датчика.

данный значок указывает, что ультразвуковой датчик сработает, когда расстояние до препятствия станет меньше 35 см.

данный значок показывает, что для подключения к процессору используется **порт 4**.

Рассмотрим еще один пример использования ульразвукового датчика.

Программа на рисунке 1.44 приказывает роботу Lego двигаться прямо бесконечно, пока робот не встретит препятствие (препятствие начинает определяться с расстояния 250 см) и не пройдет после этого 60 см. Например, ультразвуковой датчик определил, что через 250 см находится стена и после этого определения робот еще пройдет 60 см и остановится.

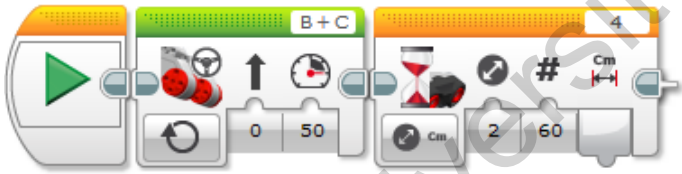
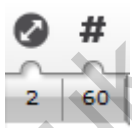


Рисунок 1.44 Пример 2 использования ультразвукового датчика

Для подключения **Больших моторов** используются порты **B** и **C**, для подключения ультразвукового датчика **порт 4**, оба Больших мотора работают со скоростью вращения 50 в режиме **Рулевое управление**.



- данный значок означает, что после обнаружения препятствия робот еще будет двигаться 60 см в том же направлении.

Как уже указывалось ранее ультразвуковой датчик в режиме **Присутствие** способен обнаружить работу другого ультразвукового датчика. При отсутствии ультразвукового сигнала на порте процессора будет **логический ноль**, при появлении сигнала - **логическая единица**.

На рисунке 1.45 установлено использование ультразвукового датчика в режиме **Сравнение -Присутствие.**

Рисунок 1.45 Датчик в режиме Присутствие

 - данный значок показывает, что ультразвуковой датчик включен в режиме **Сравнение** - **Присутствие (**обнаружение другого ультразвукового сигнала**).**

 — это логотип режима ультразвукового датчика **Присутствие.**

 — это логотип режима **Сравнение** при использовании ультразвукового датчика.

Напоминание – все команды прописанные после **командного блока ожидания**, использующего какой-либо датчик, выполняются только при срабатывании данного датчика.

1.2.8 Применение датчика поворота (гироскопа)

Гироскопический датчик определяет вращательное движение. Если вы вращаете гироскопический

датчик по направлению стрелок на корпусе датчика, то датчик сможет определить скорость вращения в градусах в секунду. Вы можете использовать скорость вращения для определения, например времени поворота части вашего робота или времени его переворота.

Кроме того, гироскопический датчик регистрирует общий угол вращения в градусах. Вы можете использовать этот угол вращения для того, чтобы определить, например, насколько повернулся ваш робот.

Гироскопический датчик может выдавать следующие данные:

Данные	Тип	Примечания
Угол	Числовое значение	Угол вращения в градусах. Измерено с прошлого сброса. Сброс блока гироскопического датчика в режиме «Сброс».
Скорость	Числовое значение	Скорость вращения в градусах в секунду.

Рассмотрим пример использования гироскопического датчика поворота (**поворот направо**) (рисунок 1.46).

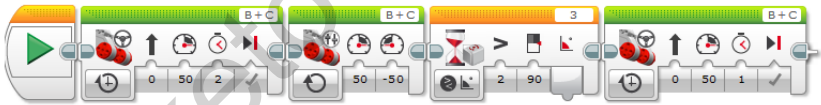
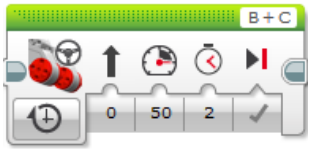


Рисунок 1.46 Пример использования гироскопа

Программа на рисунке 1.46 содержит четыре командных блока:

1. Первый блок **Рулевое управление**

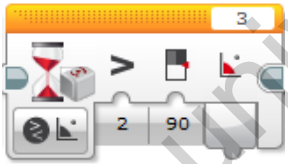


приказывает роботу двигаться прямо вперед в течение 2 секунд и затем остановиться. На обоих

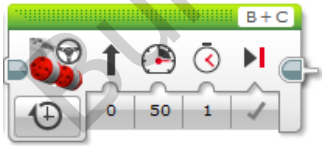
Больших моторах скорость вращения 50 (при максимуме 100). Большие моторы подключены к портам **B** и **C**.

2. Второй командный блок 

Независимое управление Большими моторами необходим для совершения поворота **направо** всего робота. Большие моторы в данном режиме вращаются в разные стороны /каждый скоростью вращения 50/ — это позволяет производить поворот робота **направо** на месте.

3. Осуществление поворота всего робота на заданное количество градусов контролируется командным блоком 

Ожидание, при помощи **гироскопического датчика.** Как только угол поворота робота превысит 90 градусов, Большие моторы будут остановлены и режим Независимое управление моторами будет выключен. Гироскопический датчик подключен к **порту 3.**

4. После осуществления поворота всего робота на заданное количество градусов / на 90 в нашем случае/ робот продолжит прямолинейное движение в режиме **Рулевое** управление 

в течение 1 секунды со скоростью 50 и затем остановится.

На рисунке 1.47 представлено выпадающее меню в командном блоке **Ожидание,** позволяющее выбрать нужный тип датчика /в нашем случае выбран **Гироскопический датчик,** в режиме **Сравнение.**

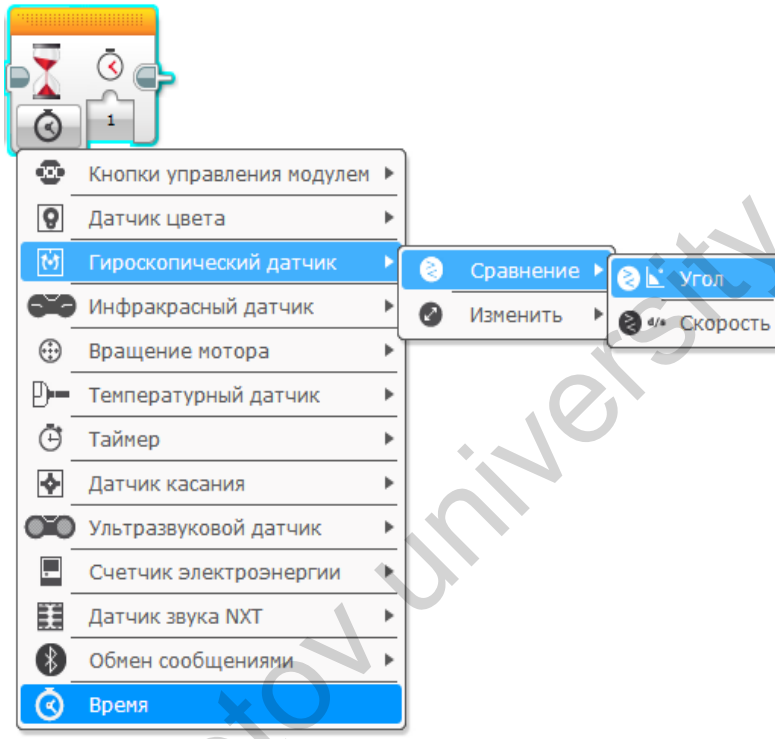


Рисунок 1.47 Меню в командном блоке Ожидание
в режиме Сравнение

— это логотип использования **гироскопического датчика** в командном блоке Ожидание.

- здесь **цифра 2** означает номер арифметического знака **больше** >.

Цифра 0 (ноль) означает операцию **равно** =.

Цифра 4 означает операцию **меньше** <.

Если в программе присутствует больше одного поворота в одну и ту же сторону (например, вправо), то угол поворота надо продолжать отсчитывать от первоначального направления движения (гироскоп в датчике поворота при всех поворотах всегда направлен в сторону первоначального движения) [9]. Например, если первоначальный поворот был произведен вправо на 90 градусов, а затем надо повернуть еще раз вправо на 50 градусов, указывают, что угол второго поворота равен 140 градусов.

Контрольные вопросы:

1. Назовите основные материальные объекты Lego набора EV3?
2. Для создания программ управления Lego какие программные среды используются?
3. Какие специальные устрйоства используются для получения информации об окружающем пространстве?
4. Что такое датчик касания?
5. Что измеряет гироскопический датчик EV3?
6. Для чего предназначен датчик цвета?
7. Приведите примеры использования ультразвукового датчика.
8. Опишите технологию работы с датчиками.
9. Что определяет гироскопический датчик?
10. Какой датчик регулирует общий угол вращения в градусах?

1.3 Дополнительные датчики

1.3.1 Температурный датчик

В комплект датчиков Lego входит температурный датчик, который можно активировать, используя командный **Блок ожидания.**

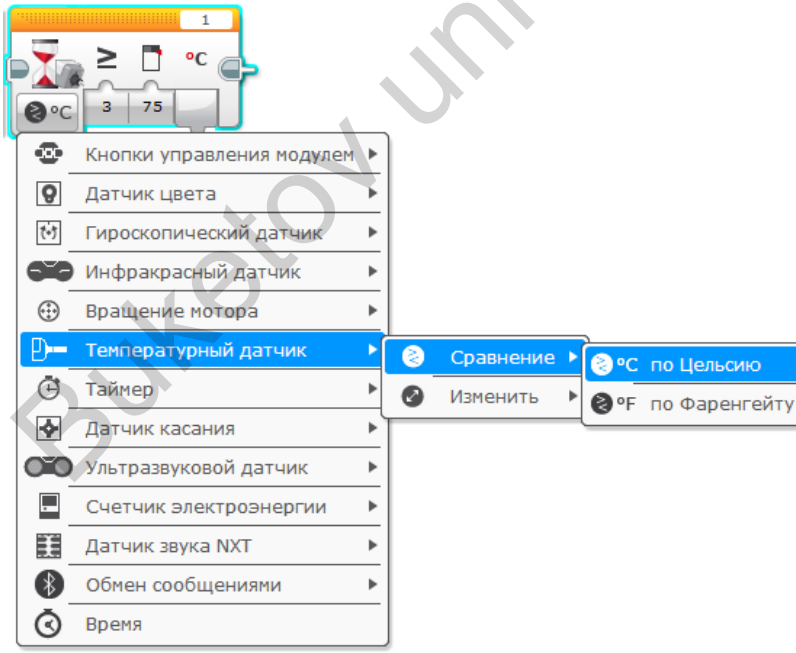
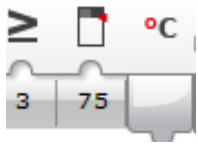


Рисунок 1.48 Использование температурного датчика в блоке ожидания

На рисунке 1.48 показано, что температурный датчик, подключенный к порту №1, сработает, когда температура по Цельсию превысит или станет равна 75^0C. Температурный датчик работает в режиме **Сравнения,** в котором измеряемая температура сравнивается с некоторой заранее задаваемой температурой.

- показано, что используется датчик температуры.



- показано, что температурный датчик должен сработать, когда температура станет больше или равно заранее заданной температуры 75^0C.

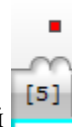
На рисунке 1.49 приведена программа, в которой показано, что робот Lego будет бесконечно двигаться вперед со скоростью **100** пока температурный датчик, установленный на нем, не покажет, что температура стала равна или больше **75^0C**. При срабатывании температурного датчика режим прямолинейного движения вперед /**Рулевое управление**/ выключается, и робот Lego останавливается.

Рисунок 1. 49 Использование температурного датчика

 данный логотип показывает, что используется температурный датчик в режиме **Сравнение;** показано, что температура измеряется в градусах Цельсия.

1.3.2 Использование датчика Цвета

На рисунке 1.50 показан путь перевода **датчика Цвета** в режим обнаружения определенного цвета  .

Датчик Цвета на рисунке 1.50 подключен к порту №3 контроллера Lego и настроен для обнаружения красного цвета  , закодированного в условии цифрой **5** и квадратиком красного цвета над цифрой  .

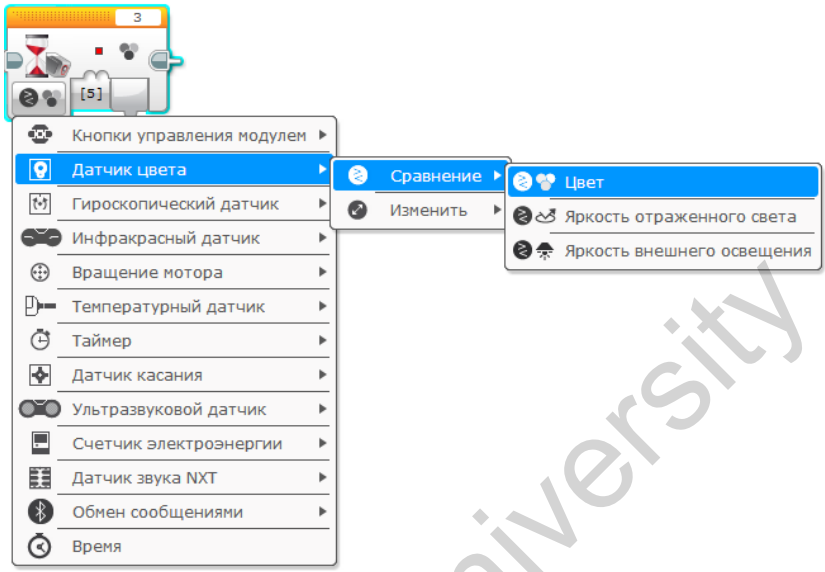
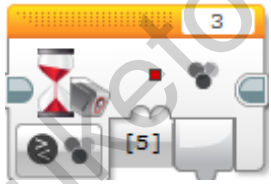


Рисунок 1.50 Настройка Датчика Цвета в режим обнаружения определенного цвета

На рисунке 1.51 приведена программа, при работе которой робот Lego движется вперед со скоростью **100** пока **датчик Цвета**  не обнаружит источник **Красного цвета**.

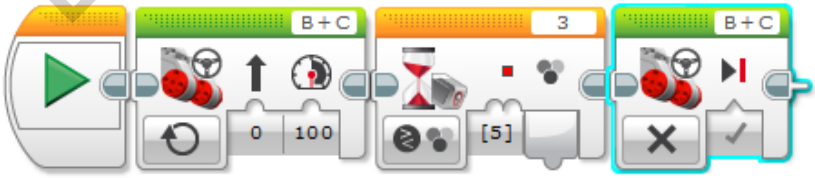


Рисунок 1. 51 Использование Датчика Цвета

Датчик Цвета, работающий в режиме **сравнения Цветов**  , подключен к порту **3** контроллера Lego. При обнаружении источника красного цвета робот Lego прекратит движение в режиме **Рулевое Управление** и остановится  .

Программа позволяет настроить датчик Цвета реагировать на несколько цветов одновременно.

данный рисунок означает, что датчик настроен на желтый и красный цвет 

(4 –номер желтого, 5 – номер красного цвета).

1.3.3 Использование датчика Цвета как датчика освещенности (фотометра)

На рисунке 1.52 показан путь перевода **датчика Цвета** в режим определения освещенности  (определения яркости внешнего освещения).

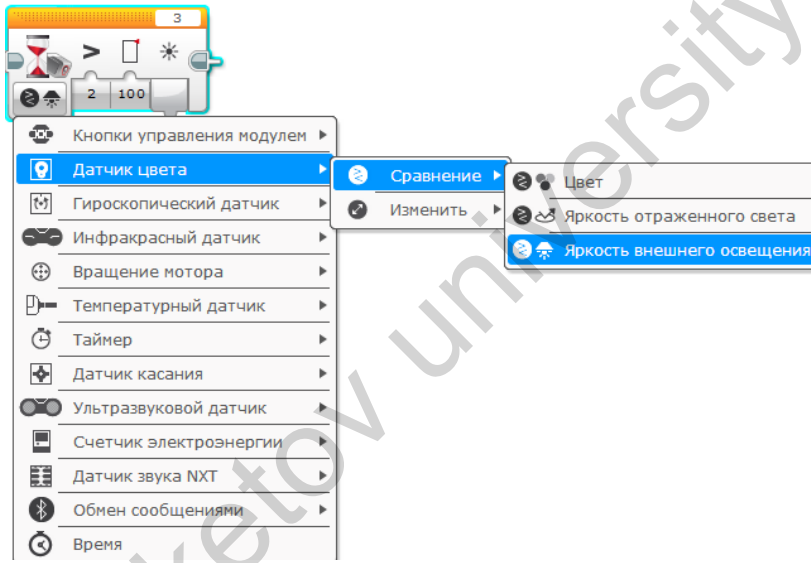
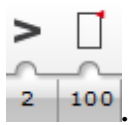


Рисунок 1. 52 Настройка Датчика Цвета в режим измерения освещенности

На рисунке 1.53 представлена программа, приказывающая роботу из состояния покоя начать движение вперед со скоростью **100,** если освещенность помещения превысит **100 люксов** 

.

Рисунок 1. 53 Использование Датчика Цвета в качестве фотометра

 - **датчик Цвета,** работающий в режиме измерения освещенности.

 - режим измерения освещенности **датчика Цвета** (освещенность измеряется в **Люксах** - Лк). Освещенность в комнате вечером обычно не превышает 200 Люксов. Освещенность в зимний солнечный день около 20000 люксов.

1.3.4 Инфракрасный датчик

Инфракрасный датчик цвета имеет три разных режима: «**Приближение**», «**Маяк**» и «**Дистанционное управление**».

Режим «Приближение»

Режим «**Приближение**»

В режиме «**Приближение**» инфракрасный датчик посылает свой инфракрасный сигнал и может определить отражение этого сигнала от объекта, находящегося перед датчиком. Сила отраженного сигнала может использоваться для определения приближенности (расстояния до объекта).

Режим «**Маяк**»

В режиме «**Маяк**» ИК-маяк передает специальный постоянный сигнал маяка, и инфракрасный датчик может определить приблизительное положение маяка, находящегося перед датчиком.

Режим «**Дистанционное управление**»

В режиме «**Дистанционное управление**» инфракрасный датчик может определять нажатие кнопок на ИК-маяке. Вы можете использовать режим «Дистанционное управление» для дистанционного управления, например, вашим роботом.

На рисунке 1.54 показан путь настройки блока Инфракрасного датчика в режим **Приближение**  .

Логический блок **Ожидание** сработает, когда **Инфракрасный датчик,** подключенный к порту **№4** контроллера Lego, обнаружит тепловой объект на расстоянии менее 50 см.

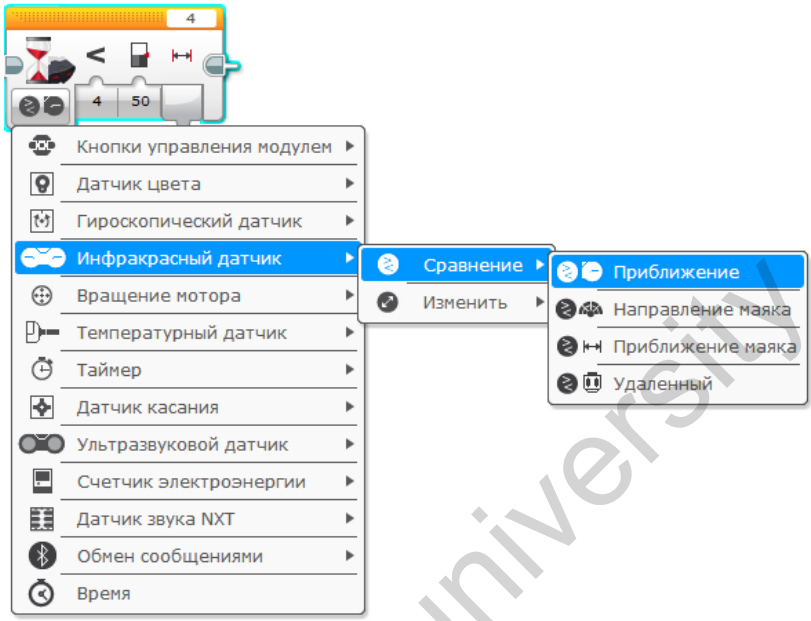
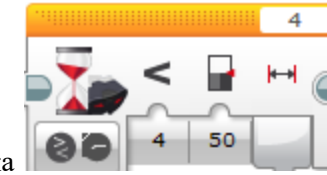


Рисунок 1. 54 Настройка Инфракрасного датчика в режиме **Приближение**

На рисунке 1.55 представлена программа, при выполнении которой робот находится в покое 

до момента пока инфракрасный датчик, подключенный к порту №4 в режиме **Приближение**  не обнаружит теплый объект

на расстоянии менее **50 см** от датчика 

(ИК датчик установлен в передней части робота).

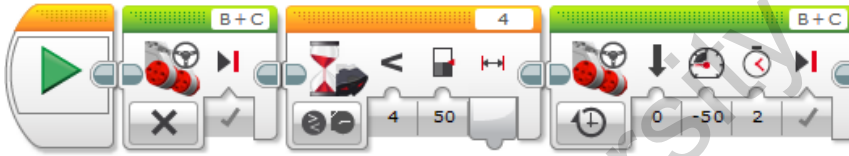


Рисунок 1.55 Инфракрасный датчик позволяет удерживать расстояние до объекта

Обнаружив теплый объект, робот включает задний ход и начинает удаляться от теплого объекта со скоростью **50** в течение **2** секунд 

.

Данная программа может быть полезна при программировании пожарной техники, когда огонь приближается к пожарному роботу, тогда робот включает двигатели и отходит от линии огня на безопасное расстояние.

1.4 Обход периметра полигона с использованием ультразвукового датчика и гироскопа

Данная задача заключается в прохожденнии колесного робота Lego вдоль периметра прямоугольного полигона (рисунок 1.56). Для осуществления данной задачи робот Lego оснащен гироскопом, контролирущим угол поворота, который производится при обнаружении препятствия на пути следования

робота (поворот в данной примере всегда осуществляется направо). Также для определения препятствия на пути движения робот оснащен курсовым ультразвуковым датчиком, установленным на передней поверхности контроллера: датчик срабатывает, когда расстояние до препятствия становится меньше 25см.

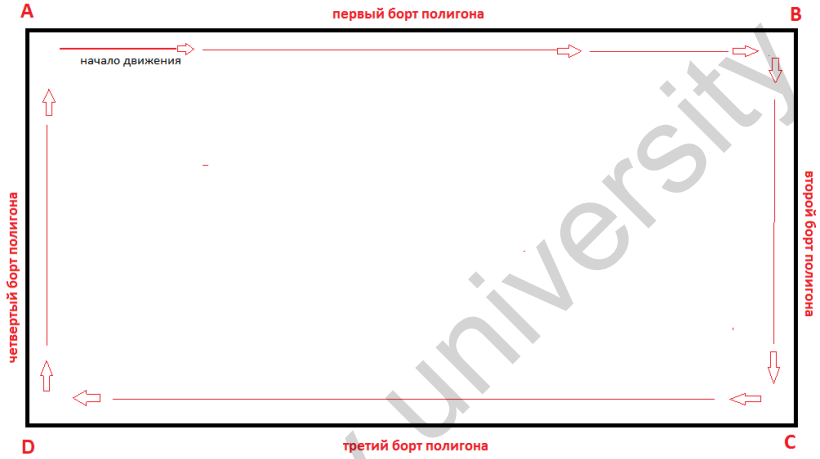


Рисунок 1.56 Движение робота вдоль бортов прямоугольного полигона

1. Обход периметра полигона с использованием ультразвукового датчика и гироскопа

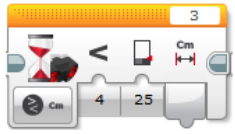
На рисунке 1.57 показан программный код, при выполнении которого робот двигаясь вдоль бортов прямоугольного полигона (рисунок 1.56) последовательно выполняет три поворота. Теоретически повороты должны быть по 90 градусов от направления, по которому робот двигался до обнаружения препятствия, но учитывая плохое качество механических деталей набора Lego (кривые колеса) реальный угол поворота робота может находиться в пределах от 84 до 95 градусов.

Рисунок 1.57 Программный код движения робота Lego,
Обход периметра Полигона

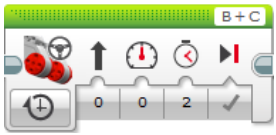
Движение вдоль первого борта полигона

1. 

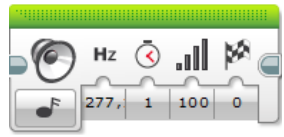
Робот Lego двигается вдоль первого борта полигона в режиме **Рулевое Управление** в режиме бесконечного прямолинейного движения.

2. Движение вперед прекращается при обнаружении курсовым ультразвуковым датчиком препятствия на расстоянии меньше **25см** 

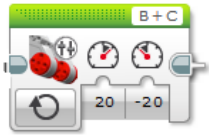
. Ультразвуковой датчик подключен к порту номер **3**.

3. 

Робот Lego прекращает прямолинейное движение в режиме **Рулевое Управление** и не движется в течение двух секунд.

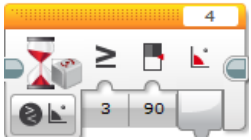
4. Остановившись, робот издает звуковой сигнал в течение 1 секунды на частоте 277 Гц 

предупреждая, что сейчас будет произведен поворот.

5. В режиме 

Независимое Управление Моторами (моторы подключены к портам (**B + C**) происходит поворот робота Lego **направо,** пока угол, на который повернулся робот, не станет равен **90** градусам. Для поворота

направо необходимо чтобы правое колесо вращалось против часовой стрелки (указана скорость -20), а левое колесо робота вращалось по часовой стрелке (указана скорость +20).

В данном командном блоке **Ожидание** 

указано, что используется гироскопический датчик  , который подключен к порту **4** контроллера робота Lego. Значок  показывает, что командный блок **Ожидание** работает в режиме **Сравнение – угол.** В данном режиме в процессе поворота робота Lego происходит сравнение угла, на который уже повернулся робот с некоторым заранее заданным углом – в данном случае угол равен **90** градусов.

Угол поворота робота отсчитывается гироскопическим датчиком от направления первоначального движения робота, которое устанавливается в момент включения программы.

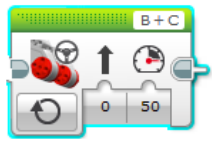
Используемая в командном блоке цифра 

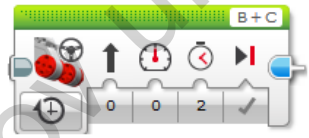
показывает, что используется тип сравнения **3 (**больше или равно**).** Значок  указывает, что измерения угла поворота робота производятся в градусах; значок  показывает пороговое значение, при котором происходит срабатывание командного блока **Ожидание.** При разработке данной программы реальный угол поворота указывался **85** градусов — это угол работающий непосредственно на роботе Lego, на котором испытывалась данная программа. Теоретически надо указывать при повороте угол 90 градусов, но при этом надо учитывать неидеальное качество механических деталей (например, кривые пластиковые колеса) в робототехническом наборе Lego, что приводит к

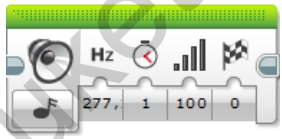
искажениям направления движения – поэтому реальный угол поворота может лежать в пределах от 83 до 90 градусов).

Итак, в точке **B** совершен первый поворот на **90** градусов при движении робота вдоль периметра полигона. Далее происходит движение вдоль второго борта полигона к точке **C**, в которой произойдет еще один поворот на 90 градусов).

Движение вдоль второго борта полигона

Команда 

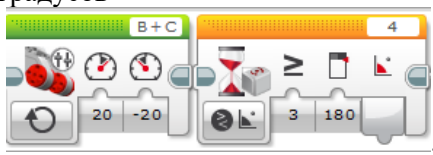
приказывает роботу Lego бесконечно двигаться вдоль второго борта полигона – это движение будет остановлено командным блоком **Ожидание,** когда ультразвуковой датчик определит, что до третьего борта полигона осталось 25 см. После остановки робот будет не двигаться 2 секунды 

. Остановившись, робот издает звуковой сигнал в течение 1 секунды на частоте **277** Гц 

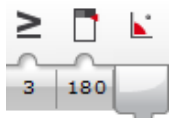
предупреждая, что сейчас будет произведен поворот.

Для того, чтобы начать двигаться вдоль третьего борта полигона робот Lego в точке **C** должен произвести еще один поворот на **90** градусов. При этом робот уже повернулся на **90** градусов в точке **B** относительно первоначального направления движения. Все углы отсчитываются от первоначального направления движения робота, заданного при включении контроллера – поэтому в командном блоке **Ожидани**е, который

контролирует второй поворот в точке С, надо указывать угол **180** градусов



. Этот угол складывается из угла поворота направо в точке **B** (направо 90 градусов) и угла поворота в точке **C** (еще 90 градусов направо).

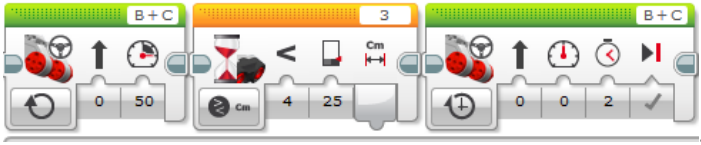


- в данном условии показано, что угол поворота робота в точке **C** должен быть больше или равен **180** градусов относительно первоначального направления движения робота. Когда угол поворота достигнет **180** градусов **Большие моторы** прекратят движение в режиме **Независимое управление** и поворот прекратится.

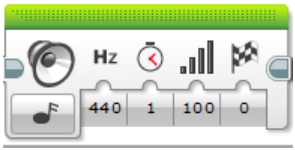
Робот прошел вдоль первого и второго борта полигона и совершил при этом два поворота по 90 градусов – после поворота в точке **C** робот должен двигаться вдоль третьего борта полигона до точки **D.**

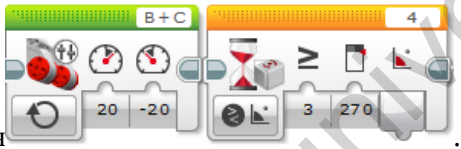
Движение вдоль третьего борта полигона

Так же как при движении вдоль первого и второго борта полигона при перемещении вдоль третьего борта робот двигается со скоростью **50** и на расстоянии **25 см** от четвертого борта ультразвуковой датчик определяет наличие препятствия (это четвертый борт полигона) и командный блок **Ожидание** производит остановку робота на 2 секунды



.

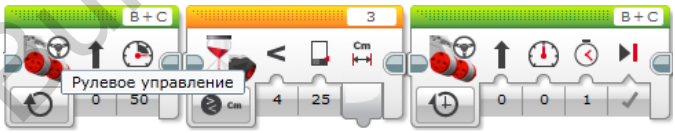


Далее срабатывает звуковой сигнал на частоте **440** герц, который предупреждает, что сейчас будет произведен поворот **направо на 90** градусов в сторону первого борта (к точке А). Это третий поворот направо который совершил робот при движении вдоль периметра прямоугольного полигона. Учитывая, что все углы поворота отсчитываются от первоначального направления движения робота в командном блоке **Ожидание** в данном случае, поворот будет прекращен, когда угол поворота робота достигнет **270** градусов от первоначального направления движения 

.

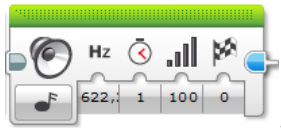
Итак, робот дошел до точки **D**, которая находится на стыке третьего и четвертого бортов, развернулся на 90 градусов направо и должен вернуться с исходную точку отправления точку **А**, двигаясь вдоль четвертого борта.

Движение вдоль четвертого борта полигона (возвращение в исходную точку начала движения)



-

Двигаясь на скорости **50** вдоль четвертого борта робот на расстоянии 25 см от точки А (точка начала движения) робот останавливается и срабатывает звуковой сигнал на частоте **622**

герца 

.

Задача выполнена, робот прошел вдоль периметра полигона вдоль четырех бортов, и при этом совершил три поворота направо по 90 градусов (в точках B, C, D).

2. Обход периметра полигона с использованием курсового датчика касания и гироскопа

В данной задаче для обнаружения препятствия на пути следования робота Lego используется курсовой **датчик касания**. При движении вдоль борта полигона робот наезжает на препятствия / в данном случае это будет другой борт полигона/, датчик касания срабатывает меняя свое логическое состояние с логического нуля на логическую единицу, и робот останавливается после чего движется назад в течении 1 секунды. Движение назад необходимо для того чтобы появилось пространство для поворота робота на 90 градусов вправо.

На рисунке 1.58 представлен алгоритм обхода периметра полигона роботом, использующим курсовой датчик касания и гироскоп.

данная логическая комбинация приказывает роботу Lego бесконечно двигаться вперед в режиме **Рулевое Управление** со скоростью 50; движение происходит вперед пока датчик касания не наедет на препятствие в результате чего произойдет **Щелчок** 

и логическое состояние порта **1** к которому подключен датчик

касания не сменится с логического нуля на логическую единицу

( - данная символика показывает, что при нажатии на порте к которому подключен датчик касания устанавливается логическая единица).

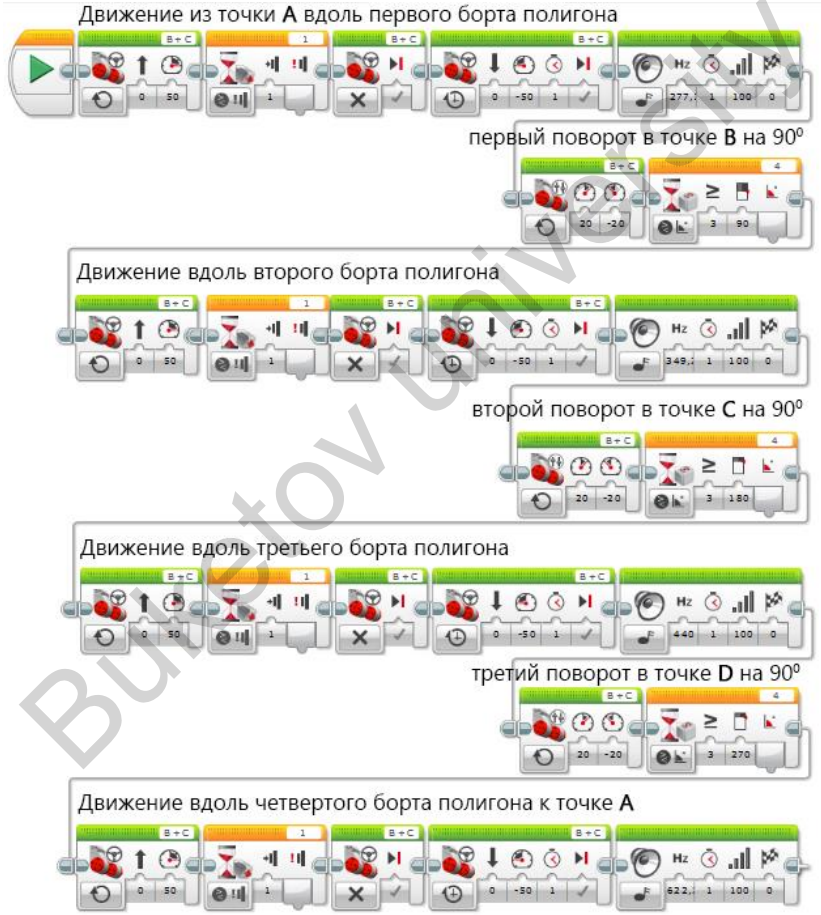


Рисунок 1.58 Обход периметра полигона вторым способом

В результате командный блок **Ожидание**  произведет выключение режима  **Рулевое управление** и робот Lego остановится.

Так как робот остановился прямо около борта у него нет места для осуществления разворота – поэтому далее происходит срабатывание команды  приказывающей роботу двигаться назад в течении 1 секунды. Прекратив движение назад робот издает звуковой сигал , осуществляет поворот на 90 градусов вправо и начинает двигаться вдоль другого борта.

1.5 Лабиринт с гироскопом

Прохождение лабиринта - известна последовательность направлений поворота робота и углы поворота, но неизвестны расстояния, которые робот должен пройти от одной точки поворота до другой точки поворота. На рисунке 1.59 изображен вид лабиринта сверху с помеченными туннелями 1–4 и угловыми точками (A–E, L–N, Z).

Для прохождения лабиринта используются два ультразвуковых датчика: 1 курсовой датчик, устанавливается спереди корпуса робота, и еще один ультразвуковой датчик устанавливается на правом борту ближе к корме робота. Курсовой ультразвуковой датчик останавливает робота, если расстояние до препятствия меньше 10 см (курсовой датчик в данном случае подключен к порту 3). Бортовой ультразвуковой

датчик останавливает робота, если расстояние до препятствия становится больше 20 см (бортовой датчик подключен к порту №2).

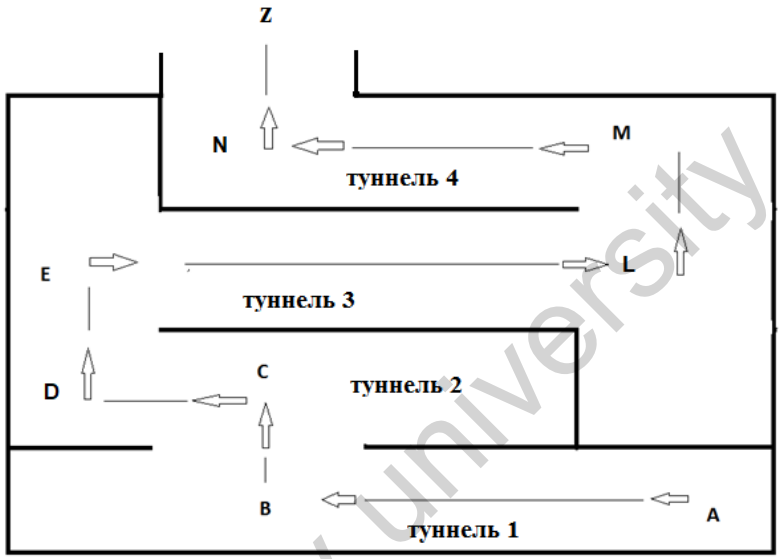


Рисунок 1.59 Вид лабиринта сверху

Для прохождения лабиринта используются два ультразвуковых датчика: 1 курсовой датчик, устанавливается спереди корпуса робота, и еще один ультразвуковой датчик устанавливается на правом борту ближе к корме робота. Курсовой ультразвуковой датчик останавливает робота, если расстояние до препятствия меньше 10 см (курсовой датчик в данном случае подключен к порту 3). Бортовой ультразвуковой датчик останавливает робота, если расстояние до препятствия становится больше 20 см (бортовой датчик подключен к порту №2).

Лабиринт состоит из четырех туннелей и проходов между ними. Туннели лабиринта имеют ширину 30 см, проходы между туннелями также имеют ширину 30 см.

Прохождение данного лабиринта состоит из 7 поворотов и 8 прямолинейных участков движения. Направления поворотов в

каждой точке поворота известны. Неизвестны расстояния, которые должен пройти, робот от одной точки поворота до другой точки поворота. Например, неизвестно расстояние от точки E до точки L. При этом известно, что в точке E робот должен повернуть направо на 90 градусов от предыдущего направления, а в точке L робот должен повернуть налево на 90 градусов.

1. Движение начинается в точке A - до точки B робот движется прямолинейно в режиме **Рулевое управление**. Расстояние от точки A до точки поворота B неизвестно.
2. В точке B робот производит поворот на 90 градусов направо (по часовой стрелке).
3. Произведя поворот, робот движется прямолинейно до точки C - расстояние между точками B и C неизвестно.
4. В точке C робот производит поворот на 90 градусов налево (против часовой стрелки).
5. Далее робот движется прямолинейно до точки D – расстояние между точками C и D неизвестно.
6. В точке D робот производит поворот на 90 градусов направо (по часовой стрелке).
7. Далее робот начинает двигаться прямолинейно до точки поворота E – расстояние между точками D и E неизвестно.
8. В точке E робот осуществляет еще один поворот на 90 градусов направо (по часовой стрелке). При этом общий угол отклонения от первоначального движения составляет 180 градусов.
9. Далее робот производит прямолинейное движение по длинному туннелю до точки поворота L.
10. В точке L робот производит поворот на 90 градусов налево (против часовой стрелки).
11. Далее робот движется прямолинейно до точки поворота M – расстояние между точками L и M неизвестно.
12. В точке M робот производит еще один поворот налево на 90 градусов (против часовой стрелки).
13. Произведя поворот, робот движется прямолинейно до точки поворота N – расстояние между точками M и N неизвестно.
14. В точке N робот производит последний поворот – поворот направо на 90 градусов (по часовой стрелке).

15. И наконец двигаясь прямолинейно робот выходит из лабиринта и останавливается в точке Z (точка Z – это точка выхода из лабиринта).

Программа прохождения по лабиринту представлена на рисунках 1.60 и 1.61 и разбита на две части: начало программы (лист 1) и продолжение программы (лист 2).

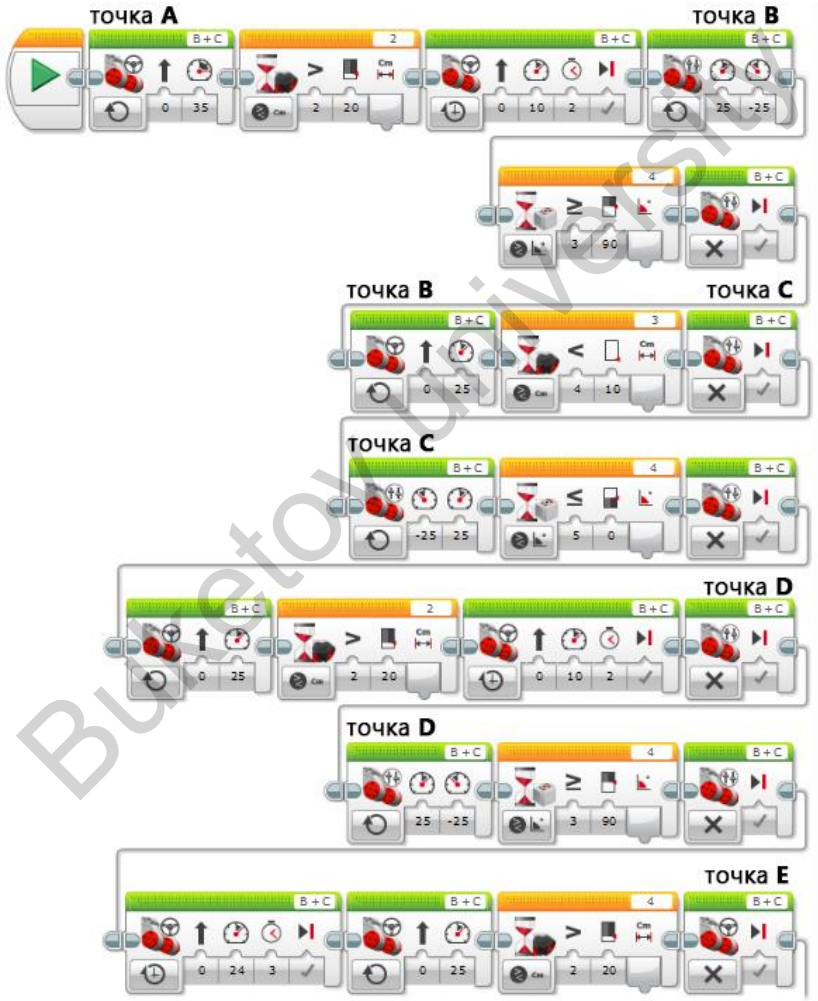


Рисунок 1.60 Лист 1 программы прохождения лабиринта

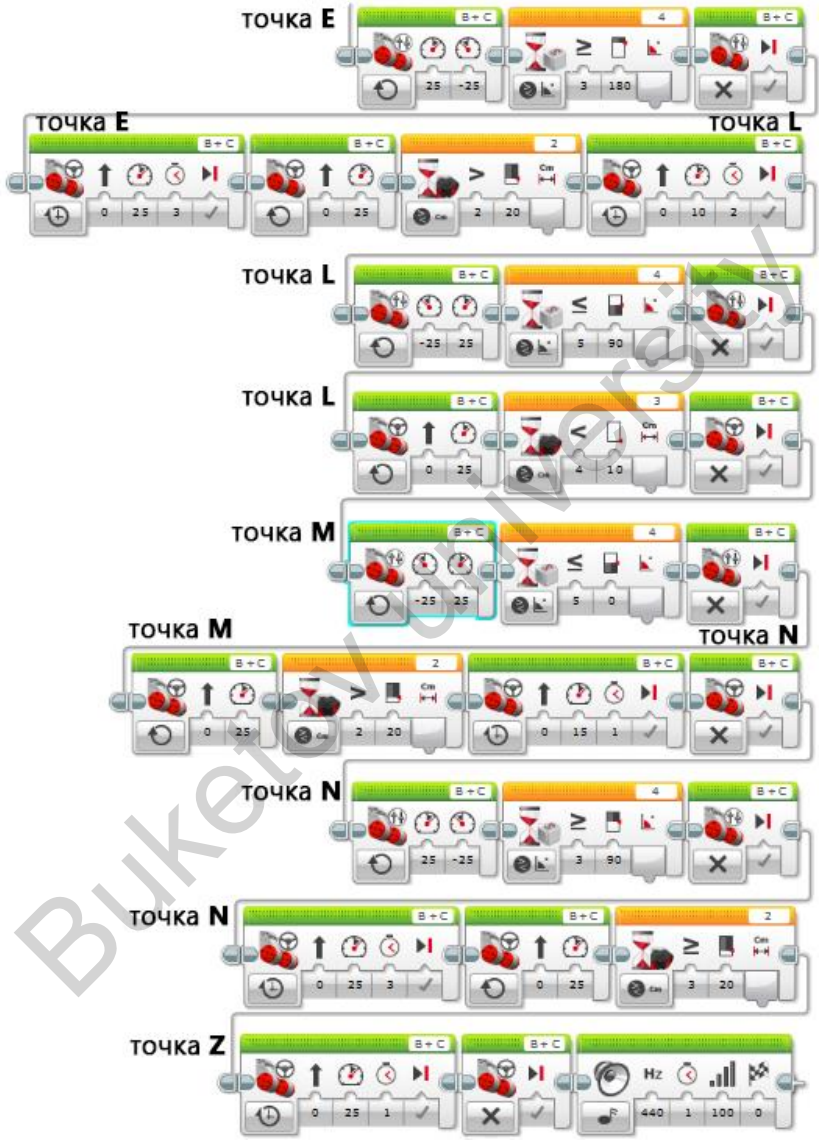
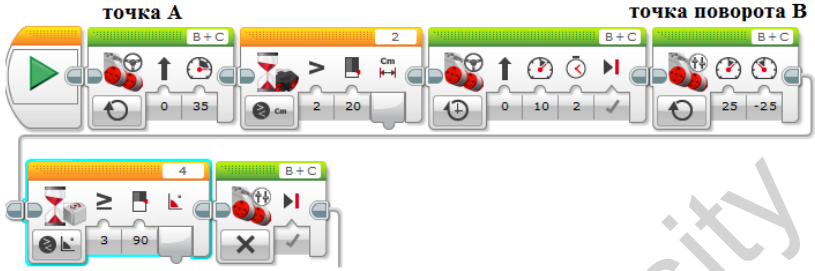


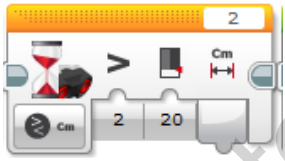
Рисунок 1.61 Лист 2 программы прохождения лабиринта

1. Движение по первому туннелю.



Робот начинает прямолинейное движение из точки **А** со скоростью **35**. Робот будет двигаться прямолинейно пока боковой ультразвуковой датчик, установленный на правом борте робота, не определит, что расстояние до препятствия /боковой стены/ стало больше 20 см – что означает в стене лабиринта есть проход.

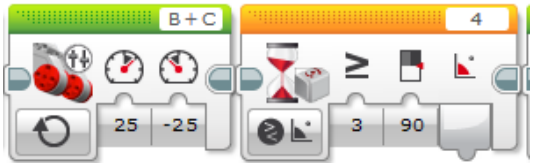
Срабатывание бокового ультразвукового датчика



прекращает бесконечное движение робота вперед со скоростью **35,** и робот в течении 2-х секунд продолжает движение со скоростью **10**: движение с замедленной скоростью необходимо в связи с тем ширина прохода в стене ограничена (30 см) , и также в связи с тем, что если сразу осуществить поворот после обнаружения ультразвуковым датчиком прохода в стене, робот после поворота врежется в боковую стенку туннеля перед точкой **В** – ультразвуковой датчик установлен на боковой стенке робота и после обнаружения прохода часть робота еще находится в туннеле между точками **А** и **В**. Если установить ультразвуковой датчик позади процессора робота, то вследствие увеличения длины робот в точке **В** при повороте заденет стенку, находящуюся напротив прохода задней частью и проход по лабиринту станет

невозможным (стандартный лабиринт с шириной между стенками 30 см узковат для роботов типа Lego EV-3).

После **2-х** секунд движения со скоростью **10** робот производит в точке **B** поворот направо на **90 градусов**

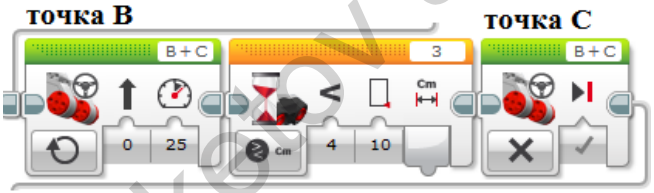


.

Произведя поворот в точке **B** робот останавливается

.

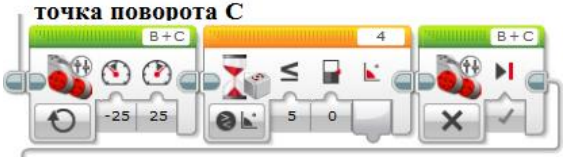
2. Теперь робот должен через проход выйти из первого туннеля во второй туннель лабиринта (перейти из точки поворота **B** в точку **C**).



Робот в режиме **Рулевое управление** двигается бесконечно прямолинейно из точки **B** со скоростью **25** пока ультразвуковой курсовой датчик не обнаружит препятствие перед роботом на расстоянии 10 см перед датчиком – обнаружив препятствие перед собой робот останавливается в точке **C.**

3. Теперь надо произвести поворот в точке **C** налево для того, чтобы робот мог двигаться вдоль второго туннеля лабиринта до точки поворота **D.**

точка поворота C



точка D

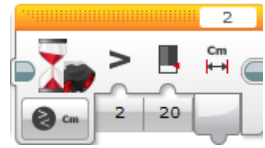
Робот в точке **C** начинает в режиме **независимое управление моторами** производит поворот **налево**

со скоростью **25** на каждом моторе. Поворот будет производиться до тех пор, пока угол отклонения от первоначального направления движения робота не станет меньше или равно ноль

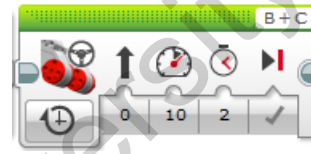
градусов . В данном условии указано, что гироскоп подключен к **порту 4** контроллера робота (в *точке **C** перед началом поворота угол отклонения робота от первоначального направления составлял 90 градусов*).

Произведя поворот налево в точке C робот начинает двигаться вдоль второго туннеля лабиринта со скоростью **25**

пока ультразвуковой датчик на правом борту контроллера не обнаружит, что расстояние до борта лабиринта

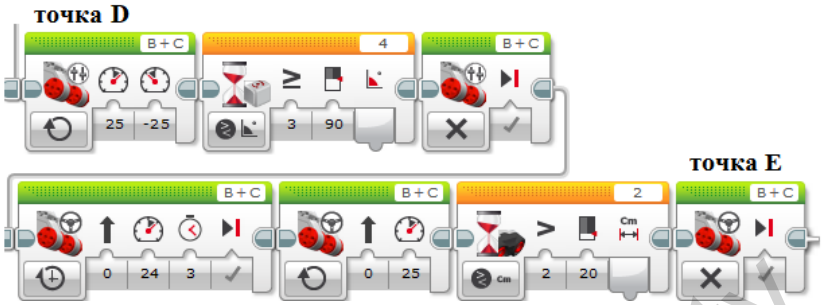
стало больше 20 см 

, что означает наличие прохода в стене второго туннеля - робот должен зайти в этот проход (*в команде указано, что боковой ультразвуковой датчик подключен к* **порту 2**).

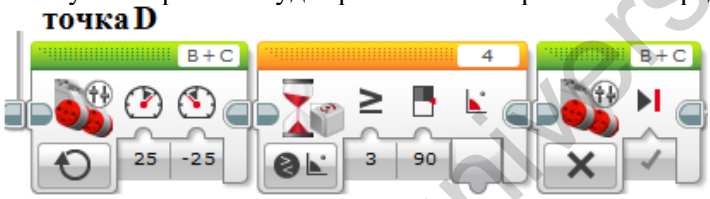
Обнаружив проход в боковой стене второго туннеля, робот уменьшает скорость до **10** и медленно в течении **2** секунд выходит на середину прохода 

(середина прохода во втором туннеле **это точка D**). Медленное движение на середину прохода позволяет избежать столкновения робота со стенками туннеля после поворота – необходимо помнить, что боковой ультразвуковой датчик находится не позади робота, а на боковой поверхности контроллера. Если сразу осуществить поворот направо после обнаружения прохода во втором туннеле робот при прохождении прохода заденет ультразвуковым датчиком о стенку – то есть произойдет авария.

4. В точке **D** робот должен произвести поворот на 90 градусов вправо и начать двигаться из второго туннеля лабиринта в третий туннель до точки поворота **E**. Точка **E** находится перед входом в третий туннель. При этом надо помнить, что после поворота в точке **D** робот еще находится во втором туннеле, и его боковой ультразвуковой датчик обращен в сторону второго туннеля из которого робот и пришел – поэтому нельзя включать боковой ультразвуковой датчик пока робот находится во втором туннеле : боковой ультразвуковой датчик надо включить когда робот полностью перейдет из второго туннеля в третий туннель, но не раньше этого момента.

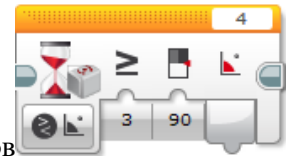


В точке D робот начинает производить поворот направо пока угол порота на будет равен или не превысит 90 градусов.



Поворот осуществляется в режиме **Независимое управление моторами** со скоростью **25** на каждом моторе бесконечно долго, но моторы при этом вращаются в разные стороны

. Поворот будет производиться пока датчик поворота /гироскоп/ не определит, что угол поворота равен или



больше 90 градусов (датчик поворота подключен к порту 4). Датчик поворота работает в режиме

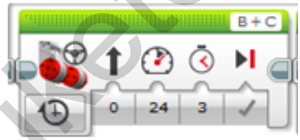
сравнения с определенным углом . После срабатывания датчика поворота оба мотора работающие в **Режиме**

независимое управление моторами выключатся и поворот робота прекратится 

.

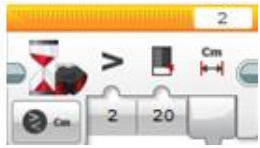
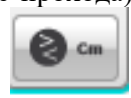
Произведя поворот в точке **D** необходимо вывести робота из второго туннеля лабиринта в третий туннель. При этом пока робот находится во втором туннеле нельзя включать боковой ультразвуковой датчик (который определяет наличие прохода в стенках лабиринта), так как датчик увидит туннель, из которого только что пришел робот. Ультразвуковой датчик определения прохода в стене необходимо включить только когда робот окажется в центре третьего туннеля.

точка E

Для выполнения данного задания необходимо чтобы робот в режиме **Рулевое Управление** двигался вперед в течение заданного времени (в приведенном примере 

робот двигается вперед со скоростью **24** в течение **3** секунд): в результате робот выйдет из туннеля № 2 лабиринта в туннель № 3. Далее включается режим бесконечного прямолинейного движения 

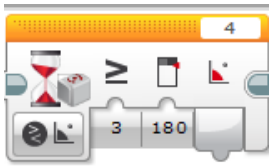
который прекратится, когда бортовой ультразвуковой датчик обнаружит, что справа от робота отсутствует стенка лабиринта (то есть обнаружит вход в туннель

№ 3):  - в данной команде указано, что ультразвуковой датчик подключен к порту **2**, и что ультразвуковой датчик сработает когда обнаружит что расстояние до препятствия (стенки лабиринта) превышает **20 с**м, что означает расположение прохода). При этом ультразвуковой датчик работает в режиме  **Сравнения расстояния в сантиметрах**. После срабатывания бокового ультразвукового датчика робот останавливается в точке **E** (при этом угол поворота робота от первоначального направления, заданного при включении робота, равен 90 градусов). Точка **E** находится в самом начале третьего туннеля лабиринта и далее надо развернуть робота на **90** градусов вправо и провести его по этому туннелю до точки **L**.

5. В точке **E** разворачиваем вправо робот еще на **90** градусов (при этом общий угол поворота достигнет **180** градусов движения от первоначального направления).

точка E

в данном блоке первая команда приказывает в режиме **Независимое управление моторами** роботу начать разворот вправо (левый мотор вращается по часовой стрелке, а правый мотор вращается против часовой стрелки). Разворот будет производиться до момента времени, когда датчик поворота не зафиксирует, что робот развернулся на **180** градусов от направления первоначального

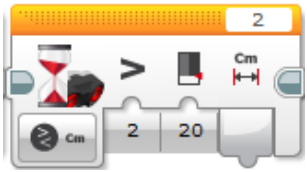


движения . При срабатывании датчика поворота происходит выключение моторов, которые производили поворот, и робот

остановится . Робот после остановки по-прежнему находится в точке **E**, но развернут для вхождения в третий туннель лабиринта. Необходимо помнить, в точке E нельзя включать боковой ультразвуковой датчик, так как робот еще не зашел в сам туннель № 3, а боковой ультразвуковой датчик при включении в точке **E** обнаружит проход – это тот проход из которого робот приехал в точку **E** из точки **D**. Боковой ультразвуковой датчик будет включен, когда робот полностью зайдет в туннель 3. Включение ультразвукового бокового датчика в туннеле № 3 необходимо чтобы определить место окончания туннеля – туннель оканчивается, прямолинейное движение прекращается и робот поворачивается в точке **L**.

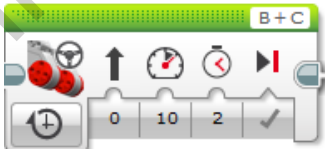
В данном блоке первая команда приказывает роботу двигаться вперед в течение **3** секунд со скоростью **25** в режиме **Рулевое Управление.** Данное время необходимо чтобы завести робот полностью внутрь туннеля № 3. Внутри туннеля срабатывает

команда приказывающая роботу двигаться бесконечно вперед со скоростью 25. Робот будет двигаться вперед пока не сработает боковой ультразвуковой датчик



- срабатывание датчика означает, что справа от робота нет стенки (по плану лабиринта известно, что стенки туннеля №3 имеют одинаковую длину. То есть если нет стенки справа от робота в туннеле №3, то значит нет и стенки слева от робота в туннеле№ 3).

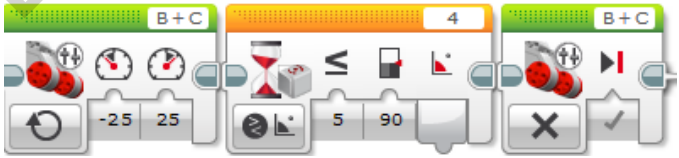
Производить поворот влево сразу по окончании туннеля № 3 нельзя, так как робот имеет реальные размеры и, если повернуть влево сразу после окончания туннеля № 3 робот врежется в стенку, разделяющую туннель №3 и туннель № 4. **необходимо проехать до точки L,** которая находится на середине расстояния между местом окончания туннеля №3 и наружной стенкой лабиринта.

Для этого используется команда 

приказывающая роботу двигаться вперед с малой скоростью **10** в течение **2** секунд. Теперь робот находится в точке **L**, и при этом он развернут на **180** градусов от направления первоначального движения, которое было задано при включении робота.

Далее необходимо развернуть робота в точке **L** на **90** градусов влево (при этом угол поворота робота относительно первоначального направления будет **90** градусов).

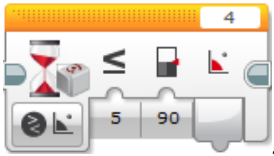
точка L



- первая команда данного блока приказывает роботу начать поворот влево.

точка L

(левый мотор вращается против часовой стрелки, правый мотор вращается по часовой стрелке). Робот будет поворачиваться влево, пока датчик поворота не обнаружит, что угол отклонения от первоначального направления стал **меньше или равно 90** градусов



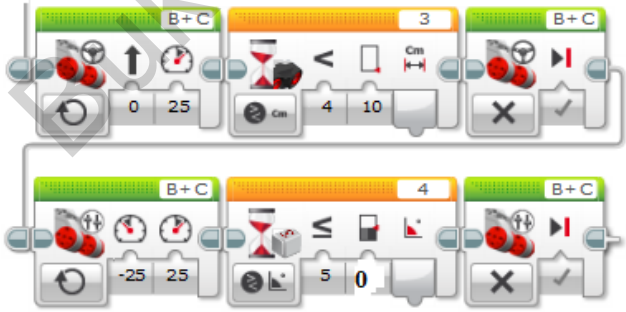
.

При срабатывании датчика поворота моторы прекратят

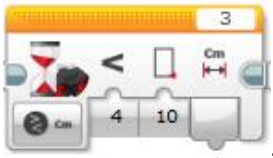
работать и поворот робота налево прекратится (угол отклонения робота при этом от первоначального направления **90** градусов; находится в точке **L**).

6. Теперь необходимо чтобы робот переместился из точки **L** туннеля №3 в точку **M** туннеля № 4.

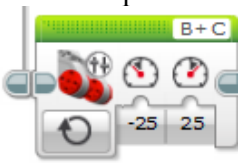
точка L



точка M

Первая команда данного блока приказывает роботу двигаться вперед со скоростью пока ультразвуковой датчик, подключенный к порту №3 не определит, что расстояние до стенки лабиринта станет меньше 10 см 

. После срабатывания ультразвукового датчика робот остановится в точке **M** (при этом робот развернут от направления первоначального движения на 90 градусов). Для того чтобы робот зайти в туннель №4 робот необходимо развернуть его в точке **M** против часовой стрелки на **90** градусов.

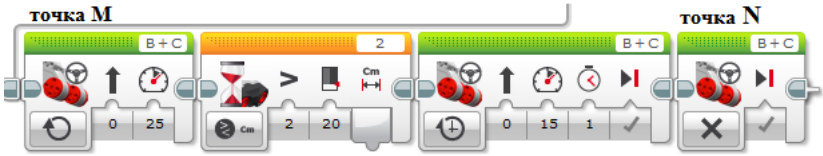


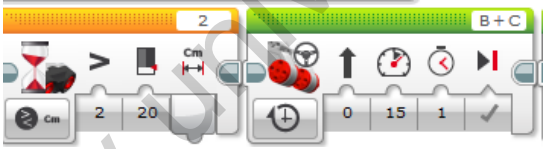
точка M данная команда приказывает роботу начать поворот налево – оба мотора при этом работают со скоростью 25, но в разных направлениях. Поворот будет производиться пока датчик поворота 

не определит, что угол отклонения от первоначального направления не превышает нуля градусов ( — это графическое обозначение датчика поворота). После срабатывания датчика поворота оба мотора работающие в режиме **Независимое управление моторами** выключатся и поворот прекратится 

. Робот при этом по-прежнему находится в точке **M** и его надо завести в туннель №4, где он должен переместиться до точки **N.**

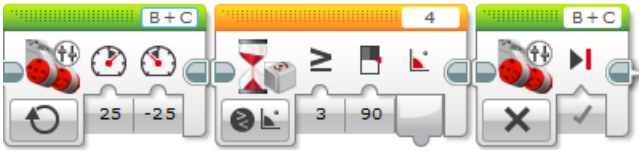
7.

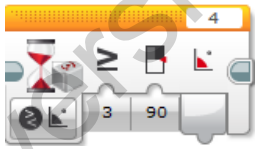


Первая команда данного блока приказывает роботу в режиме **Рулевое управление** бесконечно двигается вперед по туннелю №4 из точки **M** со скоростью **25** – его задача достигнуть точки **N.** Движение вперед будет продолжаться пока боковой ультразвуковой датчик подключенный к порту №2 не обнаружит, что справа от робота отсутствует стенка (расстояние от робота до стенки туннеля стало больше 20 см). Срабатывание бокового ультразвукового датчика прекратит бесконечное движение вперед и включится команда приказывающая роботу двигаться вперед  в течении 1 секунды со скоростью **25**. Замедленное движение вперед необходимо чтобы весь робот прошел мимо окончания стены туннеля №4 после обнаружения в данной стене прохода (необходимо чтобы робот после обнаружения прохода в боковой стене четвертого туннеля прошел до точки **N** – сброс скорости после обнаружения прохода позволяет произвести подход к точке **N** более точно; реальные размеры робота и туннелей лабиринта точка **N** оптимально подходит для осуществления поворота). Достигнув точки **N** робот останавливается

(угол отклонения робота при этом от начального направления движения равен **0** градусов). Необходимо в точке **N** повернуть робот на **90** градусов направо, чтобы ввести робот в проход, являющийся выходом из туннеля №4 лабиринта.

точка N

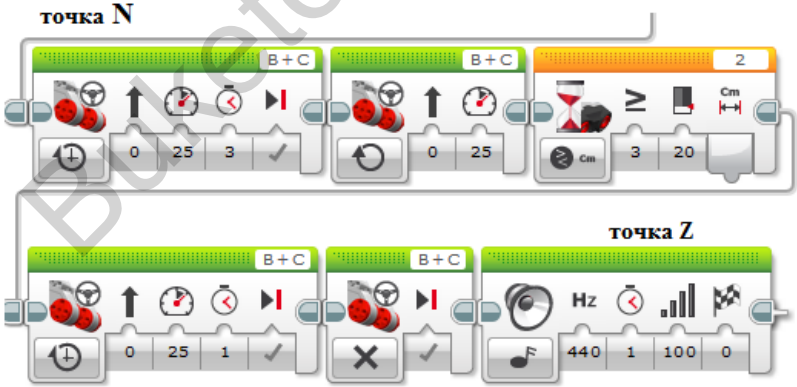


первая команда данного блока приказывает роботу начать поворот направо (правый мотор вращается против часовой стрелки, левый мотор вращается по часовой стрелке): поворот будет происходить направо пока датчик поворота не определит, что угол отклонения робота от направления первоначального движения *больше или равен нулю* 

. Срабатывание датчика поворота приказывает моторам, работающим в режиме **Независимое управление** выключиться, что приводит к прекращению поворота. Робот находится в точке **N** и угол отклонения робота от направления первоначального движения равен 90 градусов. **Робот готов к выходу из лабиринта.**

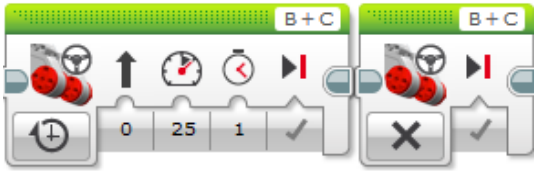
8.

точка N



Первая команда данного блока приказывает роботу двигаться из точки **N** прямолинейно по направлению к выходу из лабиринта (к точке **Z**) – движение происходит со скоростью 25 в течении 3

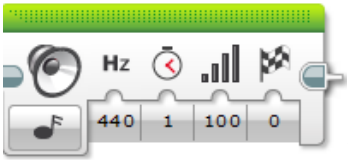
секунд : по истечении 3 секунд робот переходит в режим бесконечного прямолинейного движения со скоростью 25, который будет отключен когда бортовой ультразвуковой датчик подключенный к порту №2 обнаружит , что стенки лабиринта нет . Для того чтобы полностью выйти из лабиринта робот будет далее двигаться еще 1 секунду со скоростью 25



и остановится в точке **Z**, которая является точкой выхода из лабиринта.

Робот остановится в точке **Z** и издаст звуковой сигнал сигнализирующий, что лабиринт пройден

точка Z



.

Контрольные вопросы:

1. Опишите технологию работы с дополнительными датчиками.
2. В каких режимах работает температурный датчик?
3. В каких проектах может быть использован температурный датчик?
4. Особенности использования датчика цвета.
5. С помощью каких датчиков может быть решена задача обхода роботом периметра полигона?
6. Постановка задачи о прохождении лабиринта с неизвестным расстоянием, которое робот должен пройти от одной точки поворота до другой.
7. Минимальный набор датчиков для прохождения роботом лабиринта.
8. Как можно сделать более эффективным прохождение роботом лабиринта?

Глава 2 Makeblock ULTIMATE ROBOT KIT

2.1 Базовый набор оборудования ULTIMATE ROBOT KIT

Робот построен на базе гусеничной тележки с платой Arduino и с модулем Bluetooth (рисунок 2.1). Имеет двухканальный выход на двигатели (один двигатель на левую гусеницу и один на правую). Гусеницы с сильным понижающим редуктором, так что движение медленное. Для продления срока жизни используется алюминиевая обшивка. Время катания ограничено энергией батареек (аккумуляторов).

Набор **Makeblock Ultimate Robot Kit** включает в себя основу для сборки роботов, механических устройств и конструкций разного дизайна и концепций [10].

Рисунок 2.1 Общий вид собранного робота Ultimate Robot Kit

Каждая деталь в наборе изготовлена из прочного алюминия, что обеспечивает конструкции практически неограниченный срок эксплуатации, требуя лишь периодической замены батареек. Все элементы окрашены в нейтральный голубой цвет, который интересно сочетается не только с внутренними электронными механизмами, но и со светодиодными лентами, переливающимися разнообразными цветами.

Семейство деталей Makeblock включает в себя алюминиевые рейки, зубчатые колёса, покрышки, моторы, ремни передач, шкивы, втулки, переходники, винты, заклёпки и другие элементы. Рейки обладают жёлобом с резьбой, что позволяет производить крепление элементов в произвольном месте вдоль всей длины. Отверстия на элементах также имеют нарезанную резьбу. Всё это практически избавляет от необходимости использовать гайки.

Комплектация набора Ultimate Robot Kit

2 x луч 0824-64
Временной шкив 18T
31 x винт M4 x 8
2 x луч 0824-80
Временной шкив 62T
42 x винта M4 x 14
3 x луч 0824-96
Временной шкив 90T
10 x винтов M4 x 22
2 x луч 0824-128
2 x временной шкив 90T
10 x винтов M4 x 30
2 x луч 0824-144
1 x шестеренка 16
28 x гаек M4

4 x луч 0808-88
шестеренка 8
2 x гайки M8

8 x стержень тяги
1 x временной пояс 240 MXL
1 x крестовая отвертка
2 x кронштейн P3
6 x резиновое кольцо
1 x гаечный ключ
1 x колесо
2 x D ось 4 x 56 мм
1 x шестигранный ключ 2,5 мм
4 x пластина 45°
2 x D ось 4 x 128 мм
1 x шестигранный ключ 1,5 мм
1 x диск D72
6 x резьбовая ось 4 x 39 мм
16 x пластмассовая заклепка R4060
2 x мотора -25 кронштейн
10 x осевое кольцо 4 мм
12 x пластмассовая заклепка R4120

4 x луч 0808-184	1 x мотор -37 кронштейн
2 x временной соединитель	8 x подшипник 4 x 8 x 3 мм
30 x нейлоновый зажим для гайки	2 x базовая скобка
4 x кронштейн 3 x 3	3 x стержневой соединитель
4 x шины 68,5 x 25	4 x нейлоновый подшипник
10 x винтов M3 x 8	1 x держатель для батареи (6)AA
1 x кронштейн 3 x 6	2 x мотор - 25
40 x след	2 x микро свитч
16 x установочный винт M3 x 5	1 x makeblock orion
4 x пластины 3 x 6	1 x мотор - 37
40 x осевой след	2 x микро свитч-кабель
6 x нейлоновый стержень M4 x 30	1 x ультразвуковой сенсор
1 x пластина 7x9	3 x блок терминала
1 x временной пояс 90 MXL	4 x самореза 2 x 10
1 x шестигранник	1 x двойной проводной двигатель
1 x звуковой сенсор	3 x моторный кабель
1 x RG25 адаптер	50 x пластмассовое кольцо 4 x 7 x 2 мм
1 x LED RGB лентовая полоса	1 x bluetooth модуль
3 x RJ 25 кабель 20 см	3 x ферритовое кольцо
3 x RJ кабель 30 см	1 x микро свитч-скобка
1 x USB кабель	1x последовательность направления

Сборка электронных составляющих производится без использования паяльника. Все приводы и механизмы в наборе поставляются заведомо собранными, что обеспечивает им максимальную надежность, функциональность и простоту (рисунок 2.2). В конструктор включено железо и моторы. Но для того, чтобы вдохнуть жизнь в собранный механизм, необходима управляющая электроника, такая как Arduino, и драйвер моторов, такой как Motor Shield. А для того, чтобы механизм

стал разумным, стоит добавить немного сенсоров и соответствующей логики в программу микроконтроллера.

Рисунок 2.2 Компоненты Ultimate Robot Kit

2.1.1 Подключение модулей

Все модули в наборе оборудованы **портами RJ25** для упрощения подключения. Чтобы собрать робота, вам всего лишь потребуется соединить провода. Все провода подключаются просто, благодаря использованию стандартизированных физических сетевых интерфейсов **RJ25** (рисунок 2.3) [11].

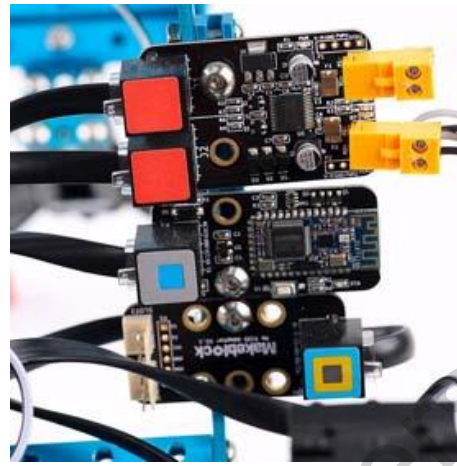


Рисунок 2.3 Модули подключенные через интерфейс RJ25

Основой электроники Makeblock Ultimate Robot Kit служит микроконтроллер, программируемый в среде разработки Arduino (рисунок 2.4).

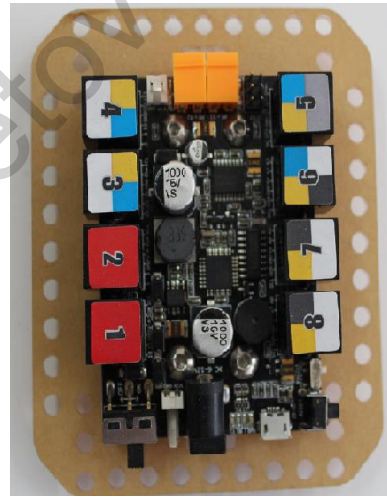


Рисунок 2.4 Общий вид контроллера Ultimate Robot Kit

К разъемам RJ-25 контроллера Arduino, помеченных красным цветом (рисунки 2.5, 2.6), подключаются устройства с напряжением питания 6-12В, моторы движения манипулятора и клешни (рисунки 2.7, 2.8).

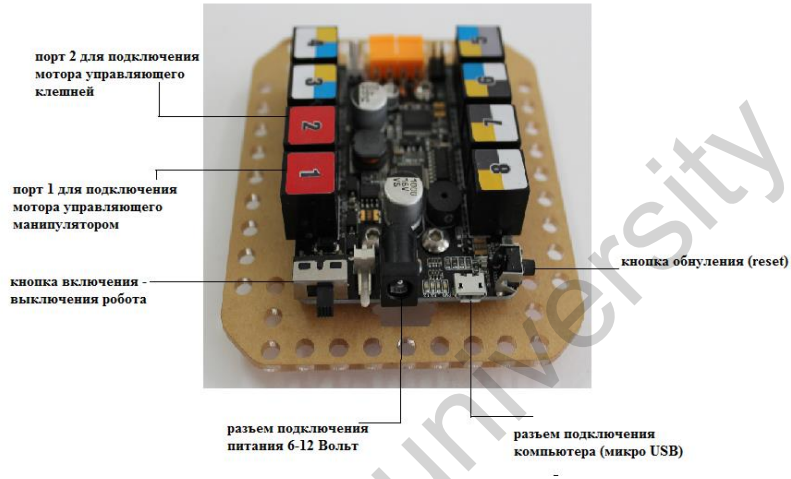


Рисунок 2.5 Контроллер Ultimate Robot Kit – вид спереди

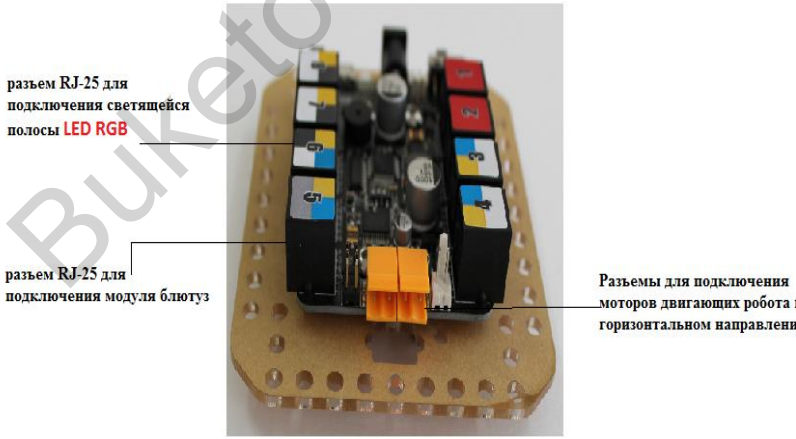


Рисунок 2.6 Контроллер Ultimate Robot Kit – вид сзади

Рисунок 2.7 Мотор DC- 6, приводящий в движение гусеницы робота

Рисунок 2.8 Мотор DC- 12, приводящий в движение манипулятор робота

2.1.2 Bluetooth

Данный модуль предназначен для возможности удаленного управления роботом – расстояние до оператора не более 10 м (рисунок 2.9) [12].

Рисунок 2.9 Модуль Блютуз

Спецификация:

• рабочее напряжение: 5 В DC;

• версия: Bluetooth v2.0 и v4.0;

• уровень выходного напряжения: 5V / высокий, 0В / низкий;

• размеры: 52 х 24 х 16 мм (длина х ширина х высота);

• вес: 10 граммов.

2.1.3 Модуль управления двумя моторами (клешни и манипулятора)

Двойной драйвер двигателя (Me Dual DC Motor Driver) является улучшение версия Me Motor Driver V2.1 (рисунок 2.10) [13]. Благодаря порту RJ25 вы можете управлять двумя двигателями с силой тока 1A (пиковый ток 2A). Двойной драйвер двигателя (Me Dual DC Motor Driver) собран на плате TB6612, которая сочетает в себе высокую эффективность MOSFET-транзисторов с низкими тепловым потерями. Встроенный предохранитель остановит Me Dual DC Motor Driver, чтобы не сгорел двигатель.

Особенности:

• TB6612PNG с высокой эффективностью MOSFET-транзисторов;

• поставляется с библиотекой Arduino для облегчения программирования;

• питание двигателя в диапазоне от **6** В до **12** В;

• максимальный ток на двигатель 1A (пиковый ток 2A);

• защита от перегрузок по току;

• простота подключения через RJ25 интерфейс;

• маркирован красным цветом для удобного подключения к Me–Base Shield;

•расстоянием между контактами 2.54 мм;

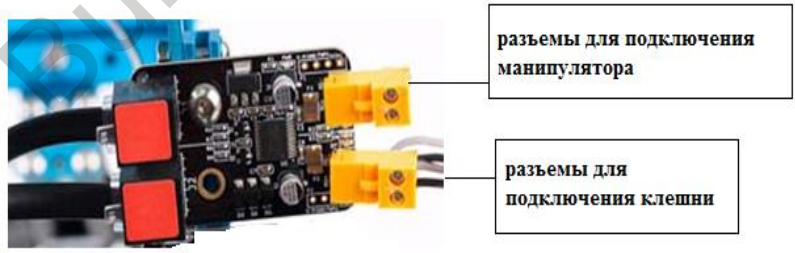


Рисунок 2.10 Модуль управления двумя моторами

•16 мм расстояние между монтажными отверстиями M4, для совместимости с элементами Makeblock.

Технические характеристики:

- подключение моторов: 2;
- минимальное рабочее напряжение: 6 В;
- максимальное рабочее напряжение: 12 В;
- длительный выходной ток на канал: 1А;
- пиковый выходной ток на канал: 2А;
- размеры: 2.4см x 4.8см;
- вес:15 граммов.

Желтые клеммы непосредственно соединены с моторами, управляющими движением манипулятора (стрелы робота) и движением клешни. Если при проверке программы клешня или манипулятор совершают движение в противоположную сторону от задуманного вами движения **(*например, манипулятор вместо движения вниз двигается вверх*) надо поменять подключение проводов на желтых клеммах данного контакта. Контакты, помеченные красным цветом (разъем RJ-25), подключаются к красным контактам (разъем RJ-25) на основной плате контроллера.**

2.1.4 Ультразвуковой датчик (Me Ultrasonic Sensor V3.0)

Рисунок 2.11 Ульразвуковой датчик

Ультразвуковой датчик (Me Ultrasonic Sensor V3.0) является улучшенной версией Ультразвуковой датчик (Me Ultrasonic Sensor V2.0), усовершенствовав качество и стабильность работы. **Ультразвуковой датчик (Me Ultrasonic Sensor V3.0)** может быть использован для измерения расстояния или уклонение от препятствий в диапазоне от 3см до 4м [14].

Особенности [15]:

- поставляется с Arduino библиотека для простого программирования;
- защита от перегрузок по току;
- простое подключение через 6-контактный интерфейс (RJ25);
- **имеет желтую маркировку для легкого подключения к Me–Base Shield;**
- расстоянием между контактами 2.54 мм;
- 16 мм расстояние между монтажными отверстиями M4, для совместимости с элементами Makeblock;
- LED индикатор для отладки и обратной связи;
- рабочий диапазон: **3см - 4м**, при угле в 30 градусов;
- Вес: 22 грамма.

2.1.5 Адаптер (Me RJ25 Adapter V2.1)

Адаптер (Me RJ25 Adapter V2.1) - это улучшенная версия модуля Адаптер (Me RJ25 Adapter V2.0). Служит для подключения через один коннектор 6P6C RJ25 два сигнальных интерфейса, которые включают в себя сигнальные интерфейсы и питание. Это позволяет подключать модули различных производителей к модулям Me серии.

Особенности:

- два распространённых разъема: два сигнальных, которые включают в себя сигнальные интерфейсы и питание;
- подключение модулей различных производителей к модулям Me серии;
- защита от перегрузок по току;
- совместимость с Me–Base Shield;

• I2C контакты;
• расстоянием между контактами 2.54 мм;
• 16 мм расстояние между монтажными отверстиями M4, для совместимости с элементами Makeblock;
• красный LED индикатор питания.

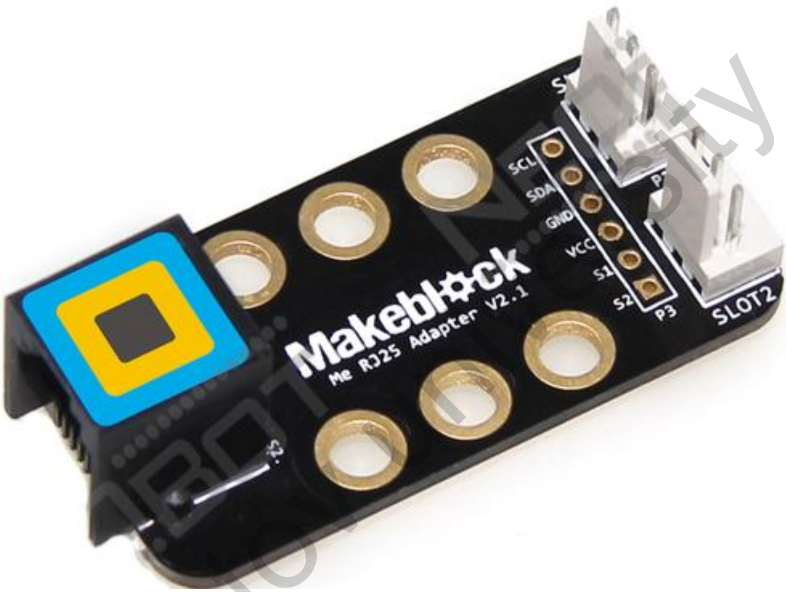


Рисунок 2.12 Адаптер

2.1.6 Датчик линии (Me Line Follower V2.2)

Модуль Me Line Follower V2.2 предназначен для линейного входа робота. Он имеет два датчика на модуле и каждый датчик содержит две части - светодиод, излучающий ИК-излучение, и фототранзистор, чувствительный к инфракрасному излучению (рисунок 2.13). Он может выводить цифровой сигнал на Arduino, чтобы робот мог надежно следовать черной линии на белом фоне или наоборот.

Особенности:

• поставляется с библиотекой Arduino для упрощения программирования;

• простое подключение с 6-контактным интерфейсом RJ25;

• маркируется синей биркой и совместим с Me – Base Shield.2.54мм контактные штыри для соединения с перемычками;

• монтажные отверстия M4 с интервалом 16 мм, совместимые с балками Makeblock;

• светодиодный индикатор питания и два светодиодных индикатора состояния датчика.

Дальность обнаружения: 1~2см

Рисунок 2.13 Датчик линии

2.1.7 Датчик звука Me Sound Sensor V1.0

Модуль Me Sound Sensor V1.0 способен измерять интенсивность звука в окружающем пространстве (рисунок 2.14). Благодаря этому модулю Вы можете создавать

интерактивные звуковые проекты, например систему, управляемую голосом, или можно создать танцующего робота. Модуль Me Sound Sensor V1.0 имеет усилитель мощности LM386 и электретный микрофон, что позволяет выводить аналоговые значения в диапазоне от 0 до 1023. А благодаря потенциометру, чувствительность можно настроить в зависимости от конкретной ситуации.

Рисунок 2.14 Датчик звука

Особенности:

• управляемая чувствительность потенциометра;
• индикатор состояния и индикатор питания;
• 16 мм интервал между монтажными отверстыми M4, для совместимости с балками из наборов Makeblock;
• 2.54 мм расстояние между гнездами на контактной площадке;
• простое подключение с помощью 6-контактного интерфейса – RJ25.

Спецификация:

• номинальное напряжение: **5 В**;
• тип сигнала: **Аналоговый (контакт А0 показывает 483** (при покое), и от **483 до 1023** (при шуме); контакт А1 показывает **0** (при покое), и от 0 до **1024** (при шуме); чувствительность контакта А0 выше, чем у контакта А1;
• размеры (ДхШхВ): 24х48х 32 мм.

2.2 Платформа ARDUINO. Принцип работы микроконтроллера

На рисунке 2.15 представлена Arduino - Open Source платформа для прототипирования устройств, которая построена на микроконтроллерах Atmel AVR ATmega328, ATmega168, ATmega2560, ATmega32U4, ATTiny85 с частотой тактирования 16 или 8 МГц [16, 17]. В старых изделиях применялись ATmega8, ATmega1280 и другие [18, 19]. Такие микроконтроллеры можно программировать с помощью ISP (In-System Programming, внутрисхемный программатор) программатора на языке Си. Arduino позволяет программировать находящийся на ней микроконтроллер с помощью встроенного в неё программатора. Arduino и Arduino-совместимые платы спроектированы таким образом, чтобы их можно было при необходимости расширять, добавляя в устройство новые компоненты. Эти платы расширений подключаются к Arduino посредством установленных на них штыревых разъёмов.

Платы семейства могут отличаться базовым микропроцессором и набором навесного оборудования, но при этом внутри сохраняется совместимость по разъемам расширения и средствами программирования и отладки. В идеале, программа или устройство расширения, созданные для одной платы, будут работать и на всех остальных. Такая совместимость, а также открытость платформы сделали Arduino очень популярной.

Микроконтроллеры для **Arduino** (рисунок 2.15) отличаются наличием предварительно прошитого в них загрузчика (**bootloader**). С помощью этого загрузчика пользователь загружает свою программу в микроконтроллер без использования аппаратных программаторов [20]. Загрузчик соединяется с компьютером через USB или с помощью отдельного переходника UART-USB. Поддержка загрузчика встроена в Arduino IDE и выполняется в один щелчок мыши.

На случай затирания загрузчика или покупки микроконтроллера без загрузчика разработчики предоставляют возможность прошить загрузчик в микроконтроллер

самостоятельно. Для этого в Arduino IDE встроена поддержка нескольких популярных дешевых программаторов, а большинство плат Arduino имеет штыревой разъем для внутрисхемного программирования (ICSP для AVR [21], JTAG для ARM [22]).

Для управления манипулятором используются порты контроллера с номерами **11 и 10** (рисунок 2.15). Через порт 10 задается направление движения манипулятора (**вверх** или **вниз**), а через порт 11 задается скорость движения манипулятора (например, 150). Порты **10 и 11** входят в состав разъема **RJ-25** на плате контроллера Arduino – разъем **красного цвета** имеет номер **1**.

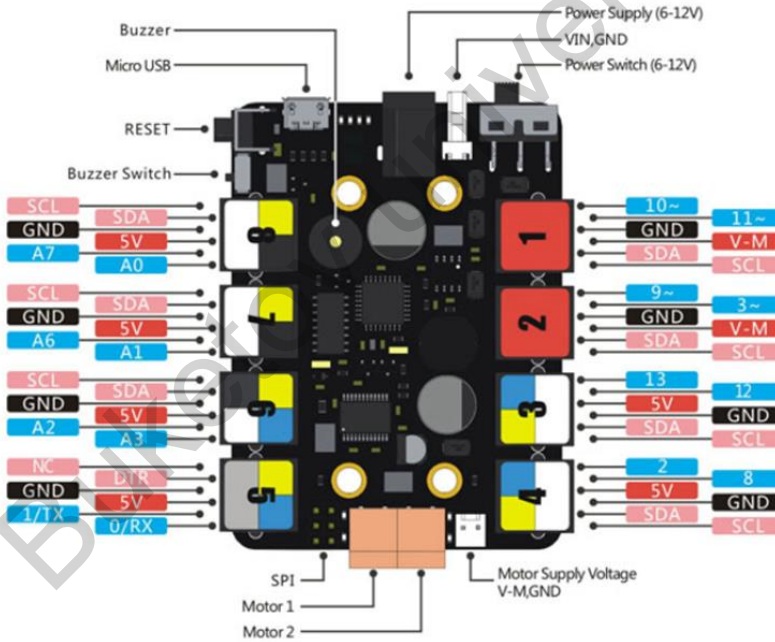


Рисунок 2.15 Распределение портов подключения процессора Arduino

Для управления клешней используются порты контроллера с номерами **3 и 9**. Через порт 9 задается направление движения клешни (**разжиматься** или **сжиматься**), а через порт 3 задается скорость движения клешни (например, 150). Порты **9 и 3** входят в состав разъема **RJ-25** на плате контроллера Arduino – разъем **красного цвета** имеет номер **2.**

В представленных ниже таблицах (таблицы 2.1, 2.2) указано каким портам платы Arduino подключаются те или иные датчики или моторы [23].

Таблица 2.1

Типы подключаемых к **ULTIMATE ROBOT KIT** устройств

Цвет этикетки	**Подключаемые устройства**
	**DC драйвер мотора**, Серво драйвер, Stepper драйвер мотора, Encoder драйвер мотора (6-12 Вольт)
	**DC драйвер мотора**, Серво драйвер, Stepper драйвер мотора, Encoder драйвер мотора (6- 12 Вольт)
	Линейный искатель, Bluetooth, инфракрасный приемник, ультразвуковой датчик(сенсор), концевой выключатель, Rocker, Кнопка
	Bluetooth модули высоко предпочтительны для этого порта, Линейный искатель, Инфракрасный приемник, Ультразвуковой датчик(сенсор), Концевой выключатель, Кнопка
	Bluetooth, Линейный искатель, Инфракрасный приемник, Ультразвуковой датчик (сенсор), Концевой выключатель, Кнопка
	Линейный искатель, Bluetooth, Инфракрасный приемник, Ультразвуковой датчик (сенсор),

	Концевой выключатель, Рокер, Кнопка
	Линейный искатель, Bluetooth, Инфракрасный приемник, Ультразвуковой датчик (сенсор), Концевой выключатель, Рокер, Кнопка
	Линейный искатель, Bluetooth, Инфракрасный приемник, Ультразвуковой датчик (сенсор), Концевой выключатель, Рокер, Кнопка

Порты 1 и 2 предназначены для подключения моторов 6-12 Вольт.

Таблица 2.2

Цвета маркировки разъемов контроллера **ULTIMATE ROBOT KIT**

Цвет	**Функции**	**Устройство**
	Для подключения драйверов моторов с напряжением 6-10 Вольт	
	Одиночный цифровой порт	Ультразвуковой датчик
	Последовательный порт для управления внешних устройств	Блютуз, вай-фай
	Двойной цифровой порт	Датчик движения, датчик линии
		Гироскоп (датчик поворота)
	Порт для подключения аналоговых устройств	Звуковой датчик, джостик, потенциометр

2.3 Программа MAKEBLOCK

MBLOCK – это программа, позволяющая создавать программы управления роботами типа ULTIMATE KIT (рисунок 2.16).

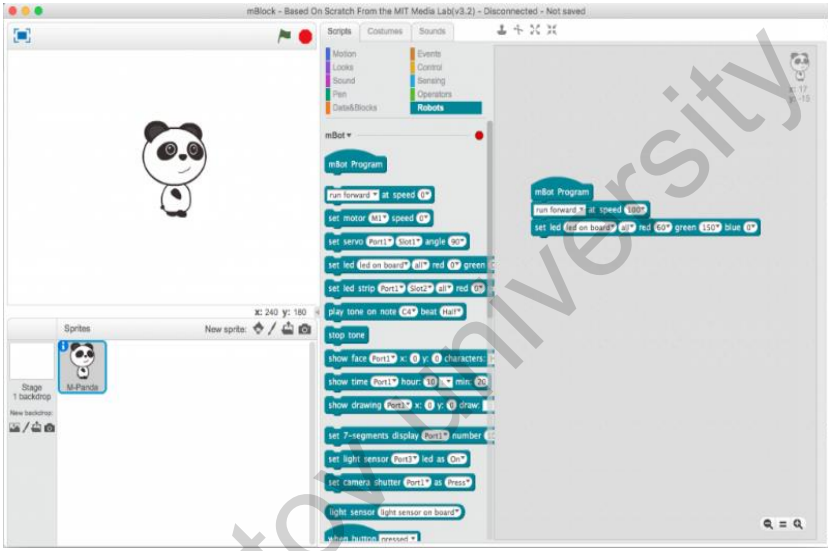


Рисунок 2.16 Интерфейс программы MBot

Большая часть роботов MBLOCK разработана на базе SCRATCH – признанного стандарта в графическом программном обеспечении, который преподают во многих заведениях. Вот неполный перечень его функций [24]:

- Вы можете «написать» свою собственную программу, просто перетаскивая блоки.

- Вы можете управлять такими роботами, как MBOT, MAKEBLOCK STARTER KIT, MAKEBLOCK ULTIMATE KIT и другими роботами, которые построены на основе проводной системы MAKEBLOCK RJ-25.

- Вы можете программировать Ваших роботов MAKEBLOCK так, чтобы они могли работали самостоятельно.

- Вы можете проявить себя с помощью программно-аппаратных комбинированных проектов, таких как игры с использованием человеческих тел в качестве контроллеров. Интерфейс программы MBlock представлен на рисунке 2.16.

Шаг 1. Скачать MBLOCK (mobogeniemini_1002_10006 (1).exe)

В первую очередь необходимо загрузить и установить программу **MBLOCK.** Данная программа позволяет создавать программы по управлению роботом Ultimate Robot Kit и загружать эти программы в данный робот. Доступны версии как для WINDOWS, так и для MAC.

Шаг 2. Установить драйвер USB

Если Вы устанавливаете и открываете MBLOCK впервые, Вам нужно установить драйвер USB. В противном случае USB-кабели роботов не будут работать. Выберите «Подключить» и «Установить драйвер Arduino», как показано на рисунке 2.17.

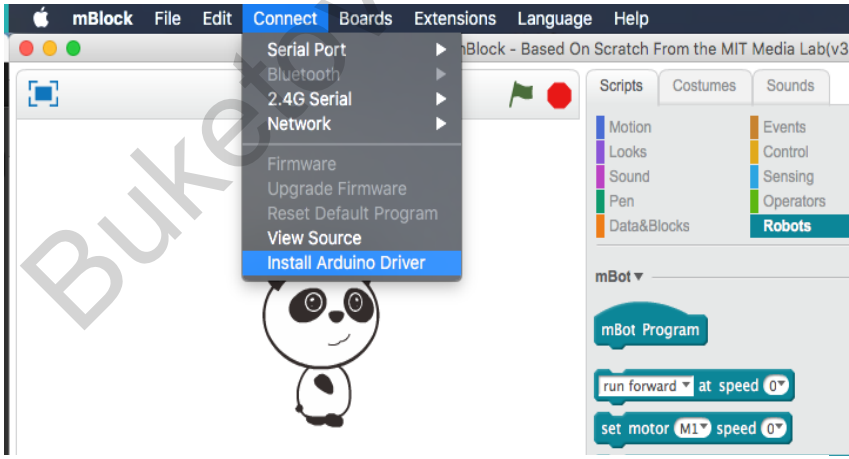


Рисунок 2.17 Установка драйверов Arduino

2.4 Подключение робота ULTIMATE KIT к компьютеру

MBLOCK позволяет подключить робота к компьютеру двумя способами:

- с **помощью USB кабеля**. Этот способ рекомендован новичкам и также используется, когда необходимо обновить прошивку и переустановить программу;

- если Ваш робот поддерживает Bluetooth, Вы можете подключить Вашего робота через Bluetooth.

Подключение робота ULTIMATE KIT через кабель USB

Чтобы подключить робота через кабель USB необходимо:

1) Подключить кабель USB для подключения между Вашим роботом и платой контроллера и компьютером (рисунок 2.18).

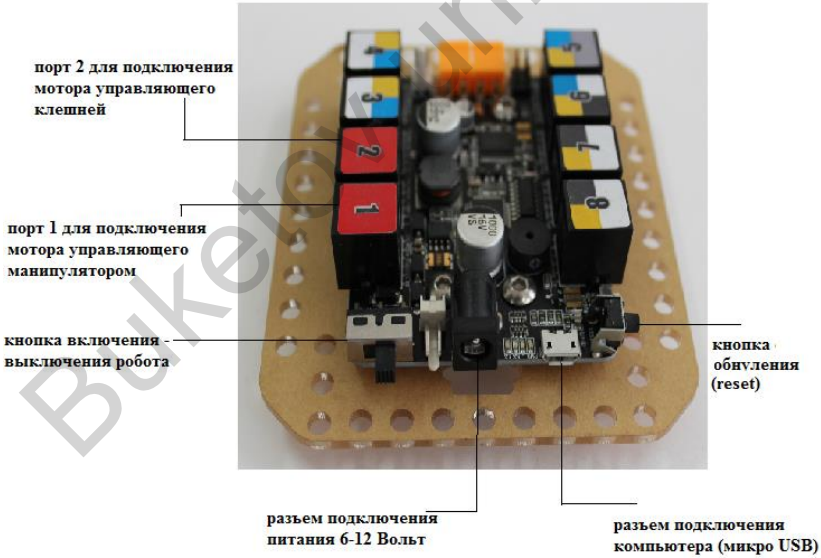


Рисунок 2.18 Порты микроUSB

На панели управления **Mblock** появятся зеленые индикаторы в виде небольших зеленых кругов, подтверждающие наличие связи компьютера с роботом. На роботе подключение производится к разъему микроUSB (рисунок 2.19).

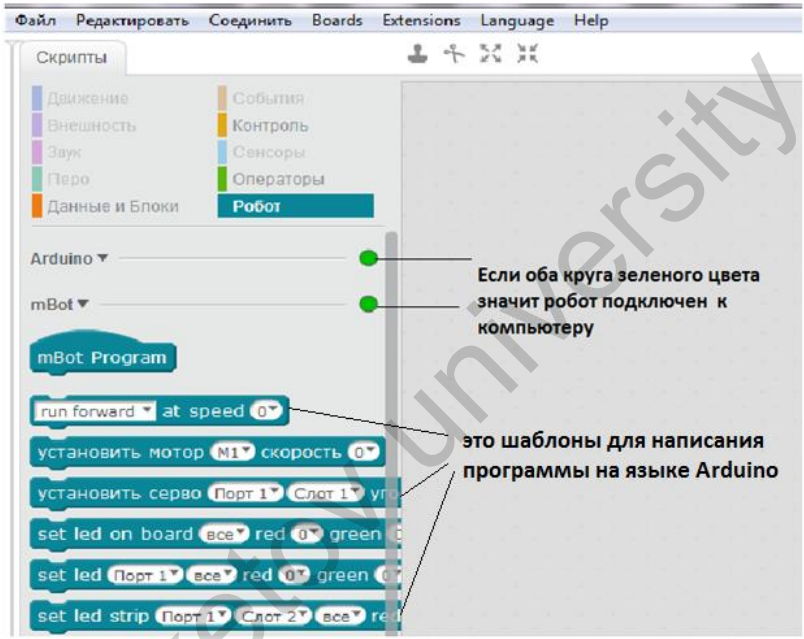


Рисунок 2.19 Общий вид интерфейса программы Mblock, с подключенным роботом (*круги зеленые*)

2) Убедиться, что робот **выключен** (с помощью кнопки питания). При этом на роботе снизу горят сигнальные лампочки красного цвета – у вашего робота выключены моторы, но память робота готова к записи программы управления, по которой в дальнейшем будет происходить работа моторов робота.

3) На панели управления программы Mblock необходимо выбрать кнопку **«Подсоединить»** («Connect»), в выпавшем контекстном меню выбрать позицию «**Последовательный**

порт» («Serial Port»), а затем опцию, соответствующую Вашему роботу (обычно **COM 2)** (рисунок 2.20).

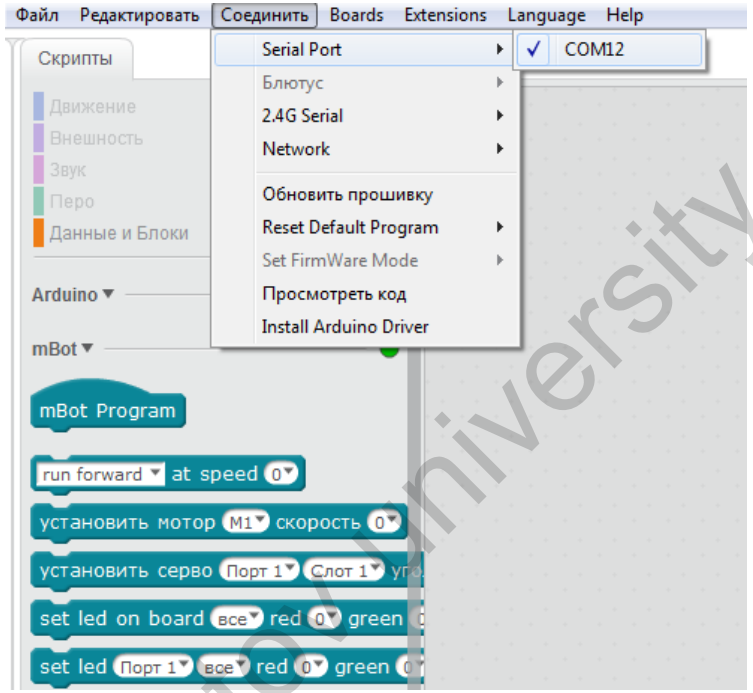


Рисунок 2.20 Подключение робота через порт COM 2

4) Если Вы используете WINDOWS, это должно быть что-то вроде "COM" и номер. Вы можете попробовать разные варианты. Если программа не может обнаружить порт соединения с контроллером, необходимо проверить плотность соединения кабеля с контактом USB на плате контроллера Arduino.

Подключение робота через BLUETOOTH

Если Ваш компьютер поддерживает Bluetooth и у Вас есть модули BLUETOOTH для MBOT или ORION, Вы можете

управлять роботом и программировать его по беспроводной сети.

Для операционной системы WINDOWS просто включите Вашего робота, выберите Подключить > Bluetooth > Найти. В открывшемся списке выберите свое устройство и можно начинать работу (рисунок 2.21).

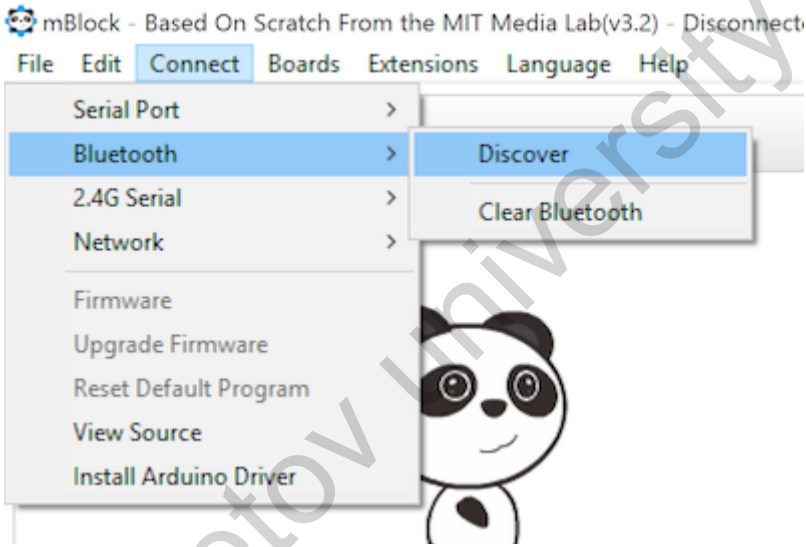


Рисунок 2.21 Подключение робота через Bluetooth

2.5 Выбор типа Вашего продукта / платы контроллера

Пользователям MBOT нужно выбрать Панель установки > MBOT; пользователям START / ULTIMATE или DIY с платой контроллера ORION – Панель установки - > ME ORION (рисунок 2.22).

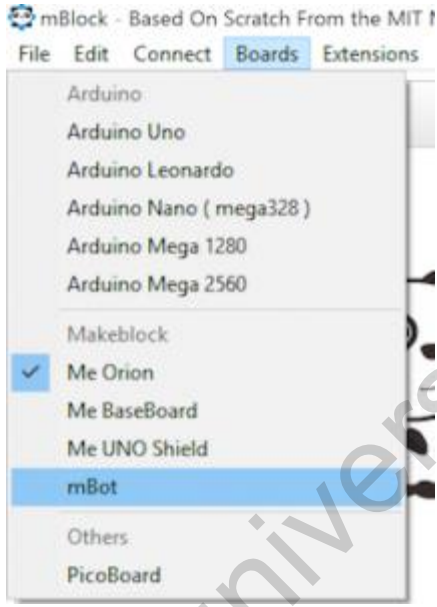


Рисунок 2.22 Подключение к плате контроллера

Обновить встроенное программное обеспечение

Перед началом использования робота MBLOCK, Вам необходимо загрузить / обновить прошивку вашего MBOT или ORION на панели управления. Выберите Подключение > Обновление встроенного программного обеспечения для завершения этого шага.

2.6 Первая программа для Ваших роботов

Теперь Вы можете производить управление вашими роботами с MBLOCK. Большинство роботов MBLOCK построены на языке программирования SCRATCH**.** Он включает в себя все базовые командные блоки и может поддерживать SCRATCH программы. Все программы, связанные с роботами, расположены в разделе **«Роботы».** В левой части расположены

шаблоны управляющих команд, в которые надо вставить конкретные параметры движения (двигаться вперед или назад, указать скорость движения и время движения). Сначала вы выбираете нужный шаблон – затем перетаскиваете его вправо на рабочее поле и указываете в выбранном шаблоне нужные вам параметры движения. Затем выбираете следующий шаблон и уже в нем выставляете данные. Любая программа должна начинаться с блока 

. Шаблоны показаны на (рисунок 2.23).

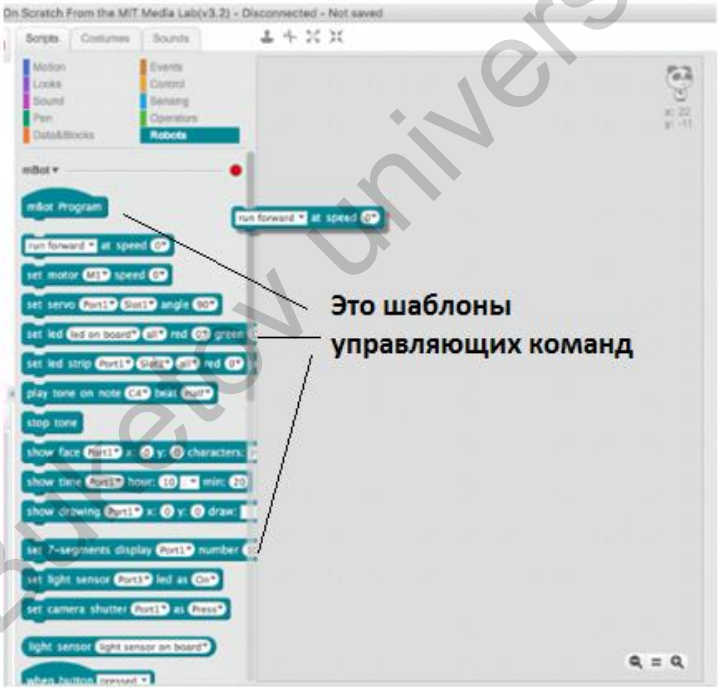
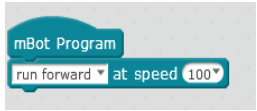


Рисунок 2.23 Набор шаблонов управляющих команд для создания программы на Mblock



- это простейшая команда для робота: данная команда приказывает роботу Ultimate двигаться вперед со скоростью 100.

2.7 Загрузка Вашей программы в программное обеспечение робота

Программы, написанные в MBLOCK, могут быть загружены в память роботов и работать без компьютера. Это особенно удобно, если вы хотите построить робота, который работает самостоятельно. Шаги для загрузки программы просты:

1) В панели управления программы MBLOCK нажать кнопку **«Редактировать»** (рисунок 2.24).

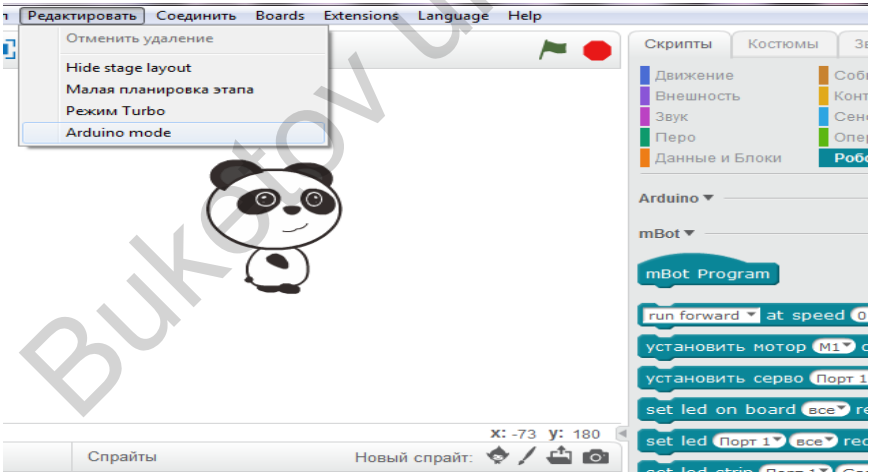


Рисунок 2.24. Подключение блока перевода Arduino

2) В выпавшем меню выбрать позицию «**Arduino mode**» (рисунок 2.24)**.** Это необходимо, чтобы перевести программу,

написанную на языке программирования Ардуино, в машинные коды контроллера Robot Kit. В результате в правой части экрана появится перевод программы, написанной на языке **Arduino,** на язык **С(си)**, с которого очень легко создавать машинные коды контроллера (рисунок 2.25).

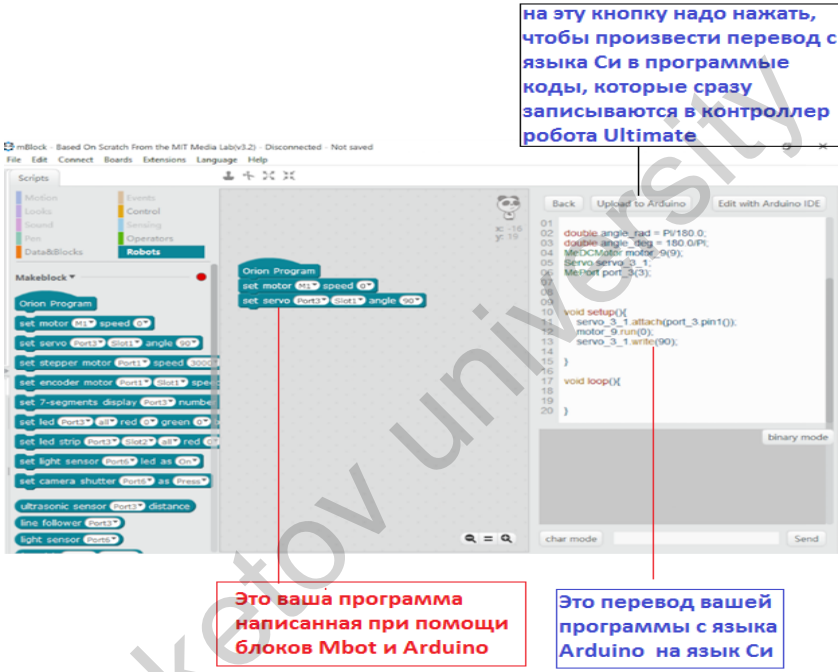


Рисунок 2.25. Запись программы на постоянную память робота Ultimate

Затем вы увидите код ARDUINO (рисунок 2.25), сгенерированный MBLOCK. ARDUINO представляет собой язык программирования, используемый для управления электронными устройствами, которые создают сами владельцы.

3) Далее необходимо записать разработанную программу в постоянную память ULTIMATE ROBOT KIT. Для этого в правой верхней части экрана нажимаем кнопку **Upload Arduino.**

После этого начинается перевод разработанной вами программы в *машинный программный код*, который сразу же записываются в память робота ULTIMATE ROBOT KIT. Программа может вывести сообщение, что отсутствует порт подключения – в этом случае надо еще раз на панели управления нажать кнопку **Соединить и** установить **порт COM12 (**рисунок 2.24)**.**

При нормально протекающей загрузке на экране появляется изображение загрузки программы с названием **Start uploading** (рисунок 2.26).

Рисунок 2.26 Выполнение загрузки программы на ПО робота

Данную кнопку необходимо закрыть после полной записи кодов в робота (рисунок 2.27). Окончание записи кодов контролируется по появлению в нижней правой части экрана записи **Reading** с последующим текстом. Нажимаем З**акрыть** и программа записана в робота.

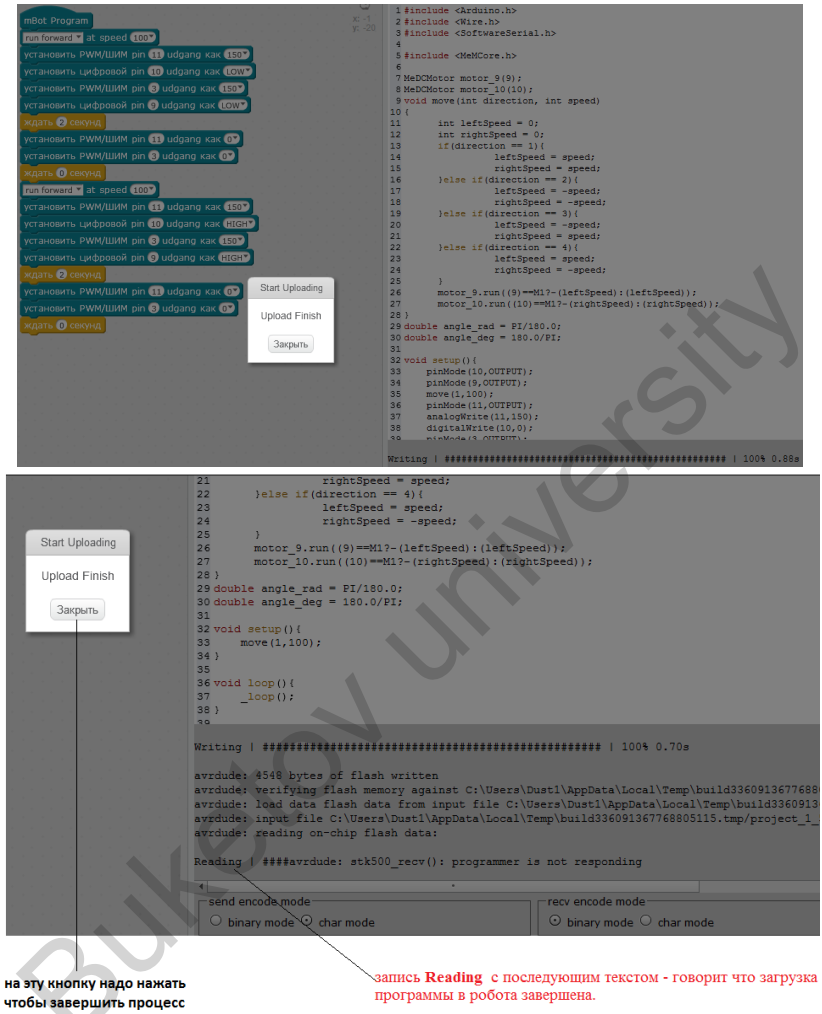


Рисунок 2.27 Загрузка программы выполнена – необходимо нажать кнопку «закрыть».

Затем включаем робота и в случае необходимости нажимаем кнопку **Reset** (на контроллере робота, рисунок 2.27), которая начинает выполнение работы с самого первого оператора.

2.8 Примеры программ управления для робота ULTIMATE KIT

2.8.1 Однократное движение манипулятора «вниз» и «вверх» (порты 11, 10)

Программа на рисунке 2.28 приказывает манипулятору робота Ultimate сначала двигаться вниз со скоростью 150 в течение 2 секунд.

Рисунок 2.28 Пример программы управления манипулятором

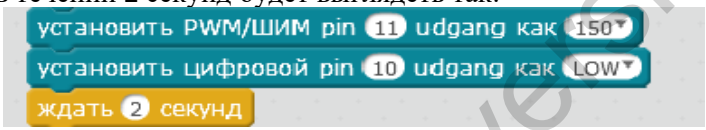
Затем идет приказ манипулятору двигаться вверх со скоростью 150 в течение 2 секунд.

1) Скорость движения **150** манипулятора задается через порт **11** при помощи команды:

2) Направление движения LOW (вниз) манипулятора задается командой при помощи порта **10**:

3) Эта команда приказывает выполнять движение манипулятора вниз в течение 2 секунд:

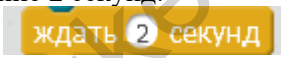
Таким образом мы видим, что для задания какого действия необходимо указать тип действия (движение вниз), скорость действия 150 и время дейстия 2 секунды. Таким образом, программый блок движения манипулятора вниз со скоростью 150 в течении 2 секунд будет выглядеть так:



Далее необходимо запрограмировать движение манипулятора вверх.

4) Данная команда приказывает манипулятору двигаться HIGH (вверх**),** используя порт **10**:

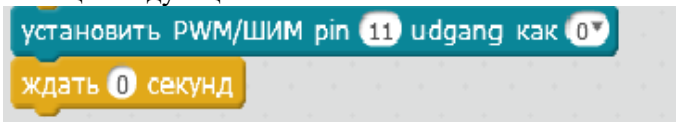
5) Эта команда приказывает выполнять движение вверх в течение 2 секунд:



6) Данная команда устанавливает скорость движения вверх равной 150:

Для стабильной работы программ робота необходимо, чтобы прекращение какого-либо действия было прямо прописано в программе. Например, в нашей программе надо четко указать, что после движения вниз, а затем вверх

манипулятор полностью прекращает движение. Это делается при помощи следующего блока:



.

В данном блоке прописано, что скорость движения манипулятора прописывается как ноль в течение нуля секунд, что фактически означает прекращение данного действия. Данную операцию необходимо прописывать в программе для гарантированной очистки ячеек памяти контроллера робота.

2.8.2 Однократное расширение и сжатие клешни (порты 3, 9)

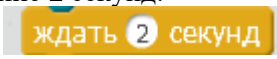
Рисунок 2.29 Пример программы управления клешней

На рисунке 2.29 приведена программа, которая приказывает клешне робота Ultimate сначала раздвигаться со скоростью 150 в течение 2 секунд. Затем программа приказывает клешне сжиматься со скоростью 150 в течение 2 секунд со скоростью 150.

1) Скорость раздвигания клешни **150** задается через порт **3** при помощи команды :

2) Это команда приказывает клешне раздвигаться, используя порт **9** и команду **HIGH**:

3) Эта команда приказывает клешне раздвигаться в течение **2** секунд:



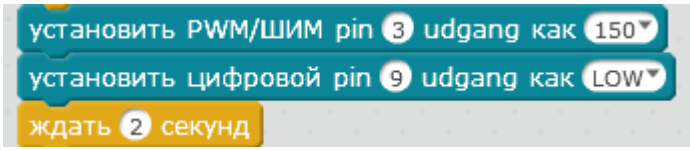
Таким образом, **БЛОК РАЗЖИМАНИЯ КЛЕШНИ** со скоростью **150** в течение **2** секунд будет выглядеть так:

Далее необходимо запрограммировать сжатие клешни.

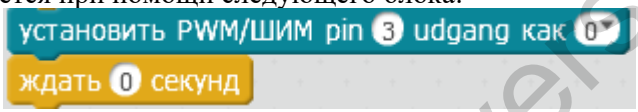
4) Данная команда приказывает клешне сжиматься, используя порт **9, и** команду **LOW:**

5) Эта команда приказывает клешне сжиматься в течение 2 секунд:

6) В нижеизложенном примере БЛОК СЖИМАНИЯ КЛЕШНИ со скоростью **150** в течение **2** секунд будет выглядеть следующим образом:



Для стабильной работы программ робота необходимо, чтобы прекращение какого-либо действия было прямо прописано в программе. Например, в нашей программе надо четко указать, что после раздвижения клешни, а затем сжатия клешни необходимо полностью прекратить движение. Это делается при помощи следующего блока:



В данном блоке прописано, что скорость движения клешни прописывается как ноль в течение нуля секунд, что фактически означает прекращение данного действия. Использование данного блока необходимо для гарантированной очистки ячеек памяти контроллера.

2.8.3 Последовательное использование двух движений

Программа на рисунке 2.30 выполняет 5 последовательных действий:

1) Сначала поступает приказ манипулятору через порты **11** и **10** двигаться со скорость **150** вверх в течение **2** секунд.

2) Затем поступает приказ манипулятору через порты **11** и **10** двигаться вниз со скоростью **150** в течение **2** секунд.

3) Для гарантии прекращения движения манипулятора манипулятору поступает приказ двигаться со скоростью **ноль** в течении **0 секунд .**

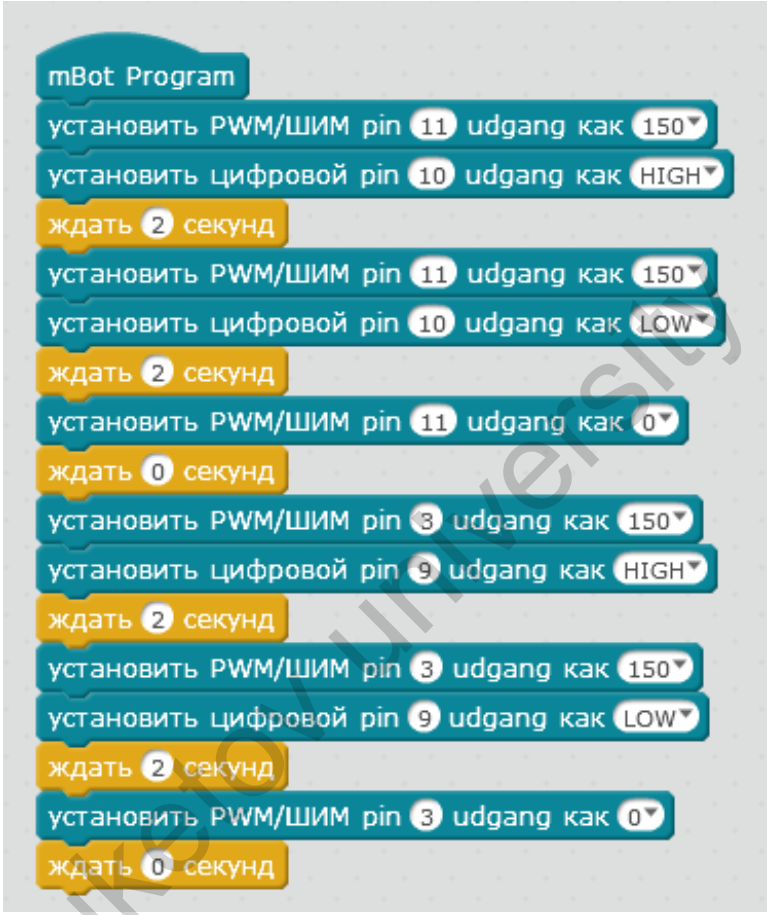
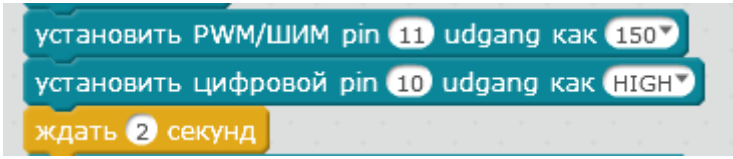


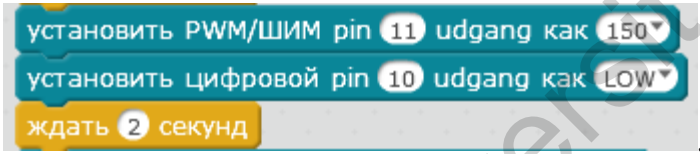
Рисунок 2.30 Пример программы по выполнению последовательных движений

4) Далее поступает приказ клешне на расширение в течение **2** секунд, через порты **3** и **9**.

5) Заключительная команда приказывает клешне сжиматься в течение **2** секунд, через порты **3** и **9**.

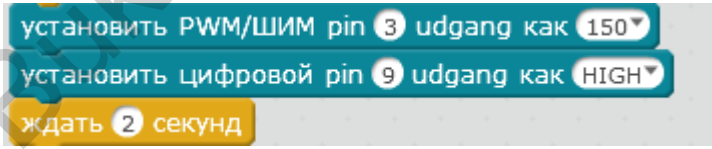


Данный блок команд приказывает манипулятору двигаться вверх **HIGH** (используется порт **10**) со скоростью 150 (используется порт **11**). Движение осуществляется в течение **2** секунд.

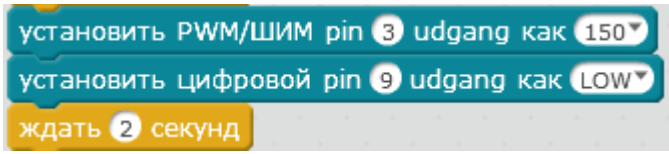


Данный блок команд приказывает манипулятору двигаться вниз **LOW** (используется порт **10**) со скоростью **150** (используется порт **11**). Движение осуществляется в течение **2** секунд.

Данная команда полностью прекращает движение манипулятора, приказывая ему двигаться со скоростью **ноль** в течение **0** секунд.



Данный блок команд приказывает клешне расширяться HIGH (используется порт **9**) со скоростью **150** (используется порт **3**). Движение осуществляется в течение **2** секунд.



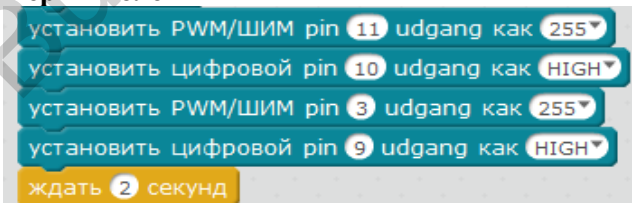
Данный блок команд приказывает клешне сжиматься LOW (используется порт **9**) со скоростью **150** (используется порт **3**). Движение осуществляется в течение **2** секунд.

Данная команда полностью прекращает движение клешни, приказывая ей двигаться со скоростью ноль в течение 0 секунд.

2.8.4 Одновременное выполнение двух движений: движение манипулятора и клешни

При выполнении программы, показанной на рисунке 2.31 робот одновременно поднимает манипулятор и при этом раздвигает клешню в течение **2** секунд при скорости **255**. Затем одновременно опускается манипулятор и клешня сдвигается в течение **2** секунд со скоростью **255**. Эти операции повторяются 2 раза.

Первый блок



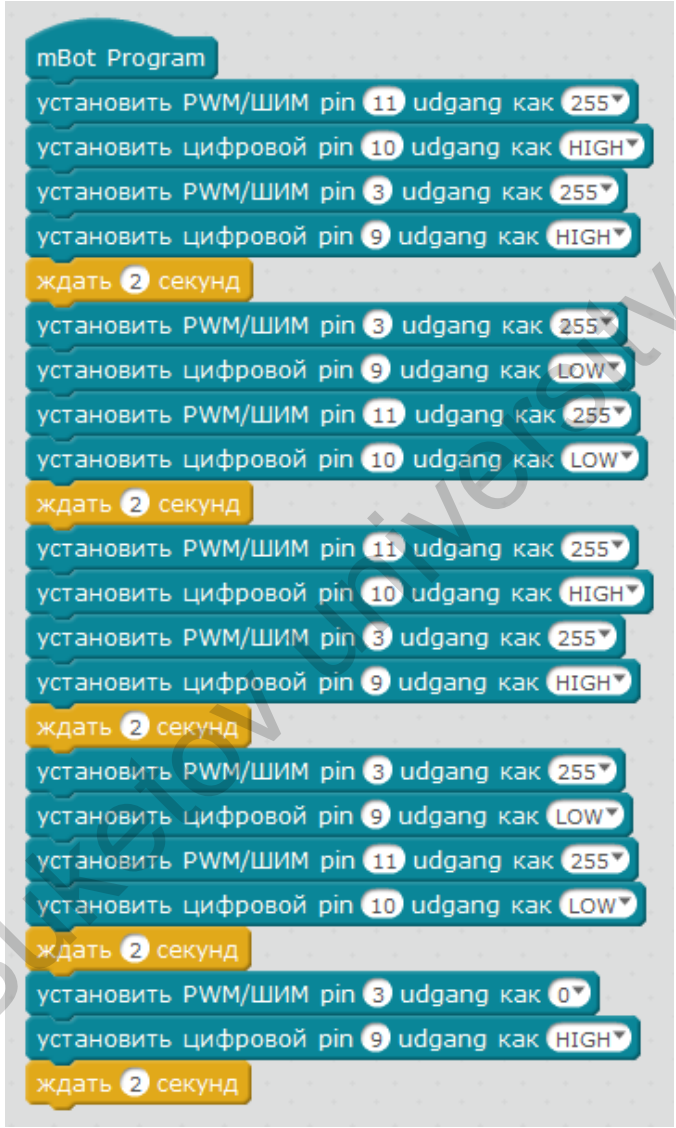
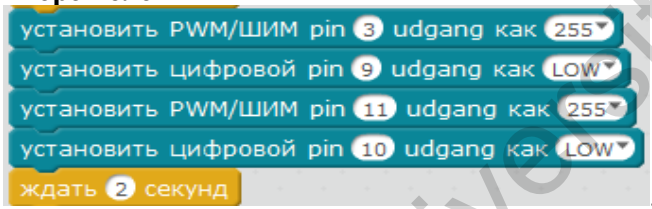


Рисунок 2.31 Пример программы по выполнению одновременных движений

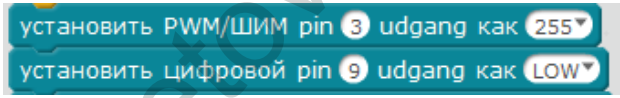
Вышеизложенбный блок приказывает манипулятору через порт **10** двигаться вверх со скоростью **255 (порт 11)** и при этом клешня разжимается (**порт 9**) со скоростью **255 (порт 3)**.

Оба действия (поднятие манипулятора и разжимание клешни) одновременно выполняются в течение 2 секунд.

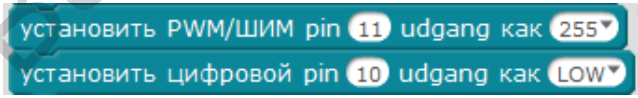
Второй блок



Второй блок команд приказывает манипулятору через порты **11** и **10** двигаться вниз со скоростью **255** и приказывает клешне через порт **3** и **9** сжиматься со скоростью **255** в течение **2** секунд.



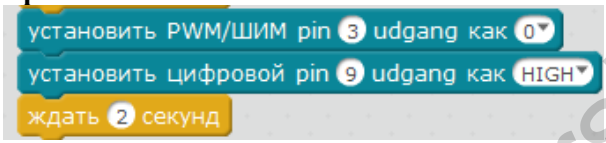
Данные команды из **блока 1** через порт **9** приказывают клешне сжиматься и через порт **3** поступает приказ клешне сжиматься со скоростью **255**.



Данные команды из **блока 1** через порт **10** приказывают манипулятору двигаться вниз и через порт **11** приказывают опускаться со скоростью **255**.

После выполнения второго блока команд происходит выполнение первого блока команд (манипулятор поднимается – клешня разжимается) и снова выполнение второго блока команд (манипулятор опускается – клешня сжимается). Далее происходит выполнение третьего блока команд, необходимого для стабильной работы робота.

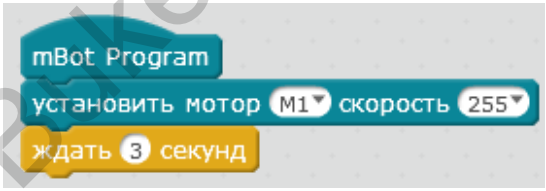
Третий блок



.

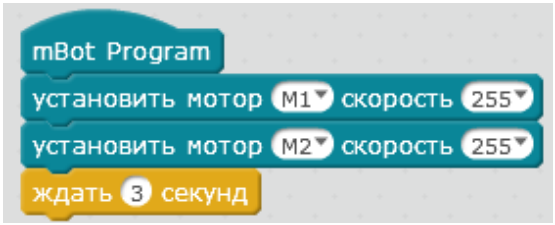
В третьем блоке прописано, что скорость движения манипулятора и клешни прописывается как **ноль** в течение **нуля** секунд, что фактически означает прекращение движение манипулятора и клешни. Данное действия является необходимым для стабильной работы программы управления роботом Ultimate.

2.8.5 Примеры программ по управлению гусеницами (используются моторы M1 и M2)

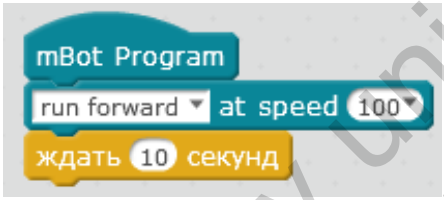


Данная программа приказывает одной гусенице, управляемой мотором **M1,** двигаться со скоростью **255** вперед в течение **3-х** секунд.

- команда приказывает выполнять данное действие **три** секунды.

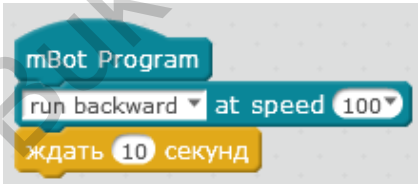


Данная программа заставляет вращаться вперед обе гусеницы со скоростью **255** в течение **трех** секунд.



.

Данная программа приказывает роботу двигаться **вперед и прямо** на скорости **100** в течение **10** секунд. При этом работают оба мотора **M1** и **M2**.



.

Данная программа приказывает роботу двигаться **назад и прямо** на скорости **100** в течение **10** секунд. При этом работают оба мотора **M1** и **M2**.

Программа на рисунке 2.32 сначала приказывает роботу двигаться **вперед и прямо** в течение **10** секунд.

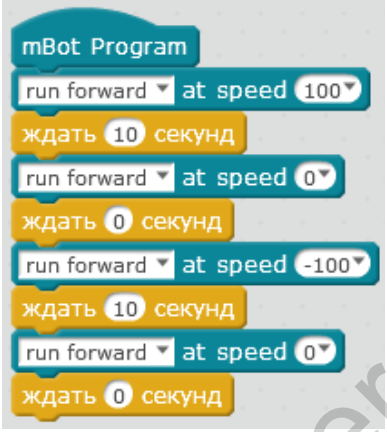


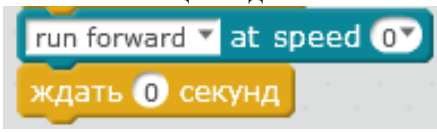
Рисунок 2.32 Пример программы по управлению моторами M1 и M2

Затем поступает приказ роботу двигаться **назад и прямо** в течение **10** секунд. Рассмотрим алгоритм движения поподробнее:

- блок команд, который приказывает роботу двигаться **вперед и прямо**

в течение **10** секунд

.

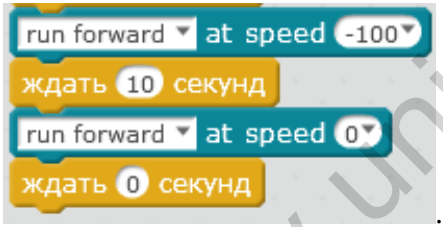
В конце данного блока стоят две команды 

, необходимые для стабильной работы робота.

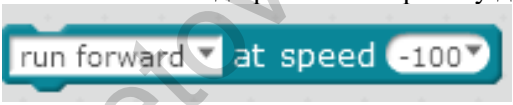
Вставка команд 

и 

необходима для стабильной работы робота (данные команды подтверждают электронике робота, что данное действие закончено).

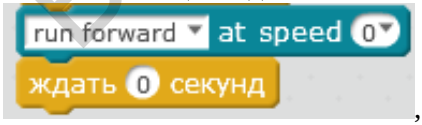


.

Данный блок команд приказывает роботу двигаться **назад и прямо** 

со скоростью **-100 (минус 100)** в течение **10** секунд 

.

В конце данного блока стоят две команды 

, необходимые для стабильной работы робота.

Вставка команд 

и 

необходима для стабильной работы робота

(данные команды подтверждают электронике робота, что данное действие закончено).

2.8.6 Осуществление поворота

Для осуществления поворота необходимо чтобы гусеницы вращались в разные стороны. Чем быстрее двигаются гусеницы, тем быстрее идет поворот (рисунок 2.33).

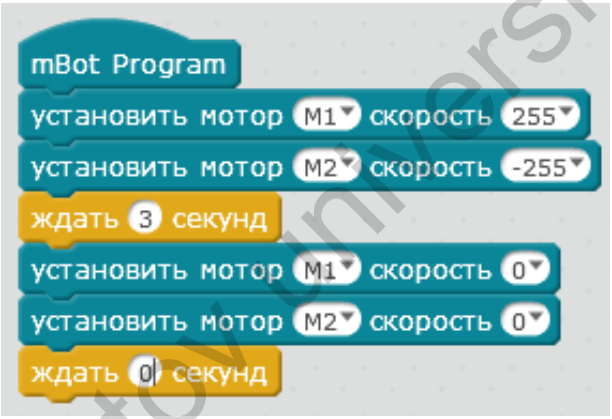


Рисунок 2.33 Пример программы по повороту робота на гусеницах

Программа на рисунке 2.33 осуществляет поворот робота в течение **3-х** секунд. В конце программы стоит блок, приказывающий повороту закончиться.

При выполнении данной программы в течение

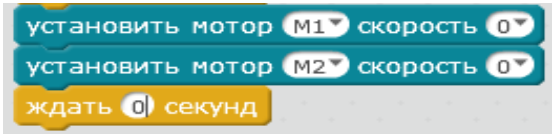
3-х секунд мотор **M1** вращается **вперед** со скоростью **255**

,

а мотор **M2** вращается **назад** со скоростью **-255 (минус 255)**

.

В результате гусеницы робота вращаются в разные стороны, и робот поворачивается на месте.



.

Данный блок команд необходим для стабильной работы робота – команды в блоке приказывают моторам M1 и M2 двигаться со скоростью **ноль** в течение **нуля** секунд.

Программа на рисунке 2.34 приказывает роботу двигаться **вперед и прямо** со скоростью **100** в течение **5** секунд.

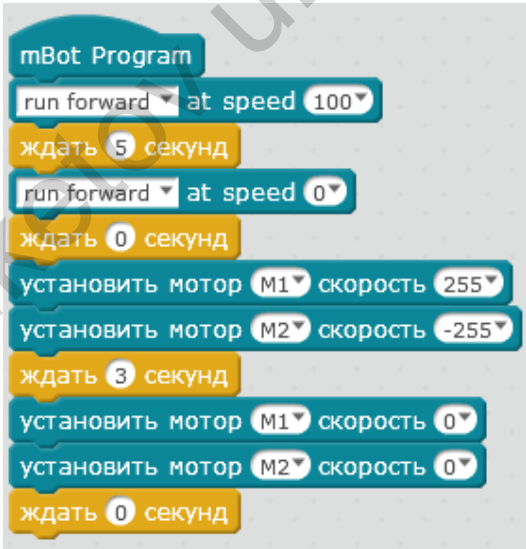
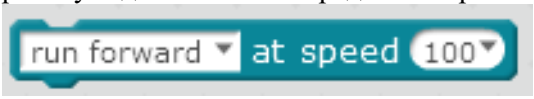


Рисунок 2.34 Пример программы по сложному движению робота на гусеницах

Затем поступает приказ осуществлять поворот в течение **3-х** секунд, после чего робот получает приказ остановиться.

- блок приказывает роботу двигаться вперед и прямо со скоростью **100**



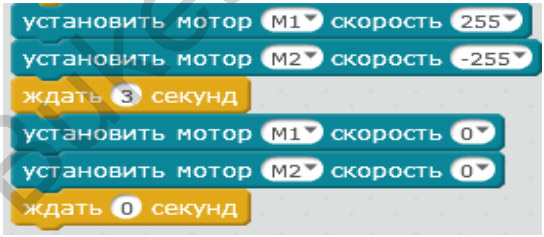
в течение **5** секунд

.

Затем роботу поступает команда двигаться со скоростью ноль

в течение 0 секунд

.



- блок приказывает роботу производить поворот

в течение **3-х** секунд

, после чего моторы **M1** и **M2** должны

двигаться со скоростью ноль

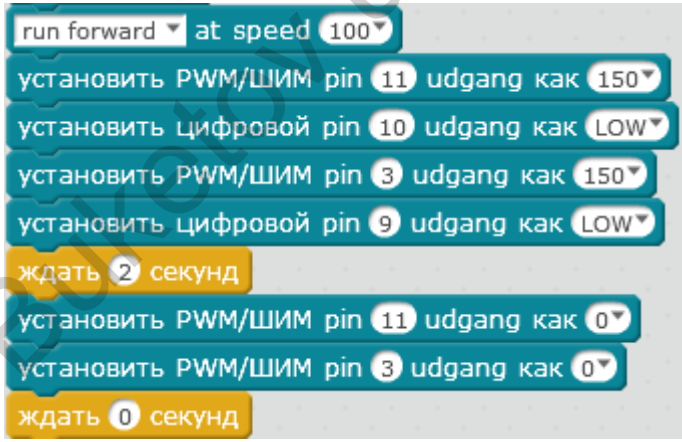
в течение **0** секунд

(команда окончательной остановки).

2.8.7 Одновременное движение гусеницами, манипулятором и клешней

Рассмотрим движение робота, когда он двигается вперед или назад, и при этом происходит движение манипулятором и клешней.

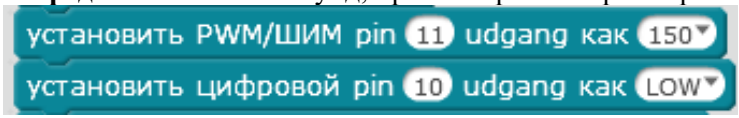
Программа на рисунке 2.35 приказывает роботу двигаться вперед и при этом производить действие с манипулятором и клешней.



- это **командный блок-1.**

Рисунок 2.35 Пример программы управлению тремя группами моторов одновременно

1) Командный блок-**1** приказывает роботу **двигаться вперед** в течение **2-х** секунд; при этом робот через порты **10** и **11**



отдает приказ манипулятору **LOW (двигаться вниз)** со скоростью **150** в течение **2-х** секунд.

По истечении **2-х** секунд робот отдает приказ

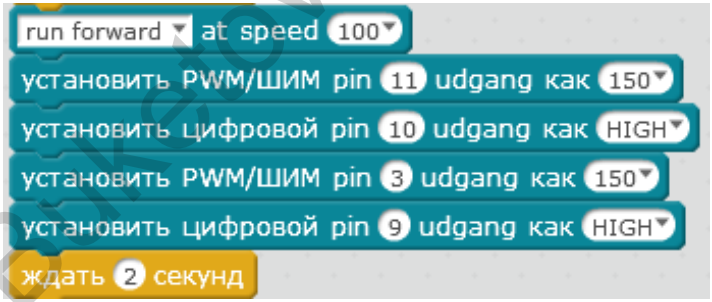
прекратить движение манипулятором.

Также, в течение **2-х** секунд робот через порты **3** и **9** отдает приказ **LOW (сжимать клешню)** со скоростью **150**

.

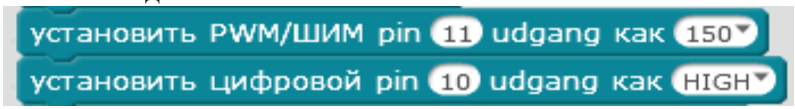
По истечении **2-х** секунд робот отдает приказ прекратить движение клешней

Далее выполняется командный **блок-2**, приказывающий роботу двигаться вперед со скоростью **100** в течение **2-х** секунд. И двигать **манипулятором вверх со** скоростью **150**, **раздвигая** при этом клешню со скоростью **150**:



— это командный **блок-2.**

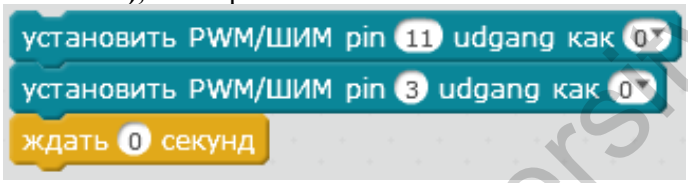
Команды



через порты **10** и **11** приказывают манипулятору робота **HIGH** (**двигаться вверх**) со скоростью **150**.

Команды

через порты **3** и **9** приказывают клешне робота **HIGH (разжиматься),** со скоростью **150**.



Данные команды приказывают роботу **полностью прекратить движение** клешней и манипулятора: поступает приказ на порт **11** двигаться манипулятору со скоростью **ноль**, также поступает приказ на порт **3** двигаться клешне со скоростью **ноль (**общее время движения **ноль** секунд).

В результате действия вышеизложенной программы робот:

1) в течение первых 2-х секунд будет опускать манипулятор вниз, сжимая при этом клешню;

2) затем в течение следующих 2-х секунд робот будет поднимать манипулятор вверх, раздвигая при этом клешню.

При этом в течение всех 4-х секунд робот движется вперед и прямо. По истечении 4 секунд все действия робота прекратятся.

2.9 Программирование датчиков ULTIMATE ROBOT KIT

2.9.1 Подключение датчиков к контроллеру Arduino

Для получения информации об окружающем пространстве используются специальные устройства – **датчики**: они

позволяют определять наличие горючих газов, наличие звукового сигнала, расстояние до ближайших предметов, наличие препятствий на пути, наличие внешнего ультразвукового сигнала, позволяют определить на какой угол произошло изменение направления движения и многое другое.

В данном типе робота рассмотрим работу с ультразвуковым датчиком, датчиком поворота (гироскопом), датчиком наличия горючих газов, датчиком наличия звукового сигнала.

Для работы с датчиками необходимо в программе **Mblock** использовать логические операторы (если условие, записанное в заголовке логического оператора, выполняется, то происходит выполнение первого блока действий, если условие, записанное в заголовке логического оператора, не выполняется происходит выполнение второго блока действий).

Все датчики подключаются к портам **3, 4, 5, 6, 7, 8 контроллера Arduino** (рисунок 2.36)**.**

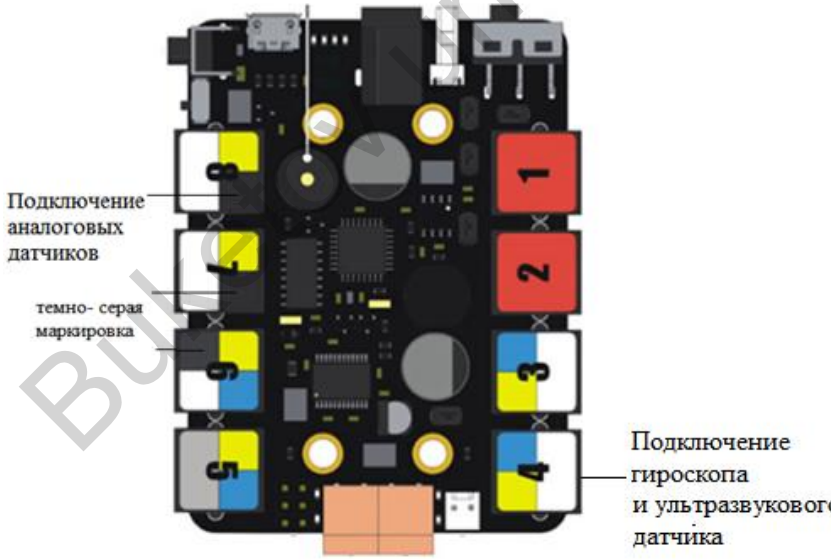


Рисунок 2.36 Порты подключения датчиков

Датчики бывают аналоговые и с цифровым выходом (например, датчики касания имеют цифровой интерфейс).

К портам 1 и 2 датчики контроллера Arduino не подключаются**. Порты с 3 по 8** имеют цветовую маркировку, по которой можно определить тип датчика, подключаемого к данному порту. Например, аналоговые датчики должны подключаться к портам, на которых есть темно-серая маркировку (*к 8, 7, 6 портам контроллера Arduino).*

Гироскопы подключаются к разъемам, имеющим белую маркировку /*порты 8, 7, 6, 4, 3/.*

Для работы с датчиками необходимо использовать операторы контроля, которые находятся в разделе основного меню КОНТРОЛЬ (рисунок 2.37). Цвет операторов контроля - болотный.

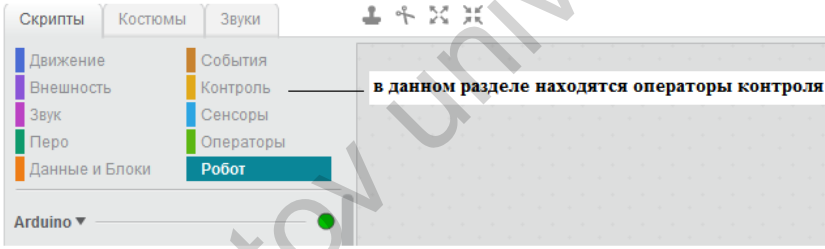


Рисунок 2.37 Меню программы MBlock

При открытии раздела **КОНТРОЛЬ** открывается окно, в котором находятся шаблоны операторов контроля (рисунок 2.38).

Для подключения датчиков используем оператор контроля **ВСЕГДА** (рисунок 2.39) и оператор контроля ЕСЛИ_**ТО, ИНАЧЕ**.

Для создания логической конструкции необходимо оператор контроля **ЕСЛИ_ТО, ИНАЧЕ** вставить внутрь оператора контроля **ВСЕГДА** (рисунок 2.40).

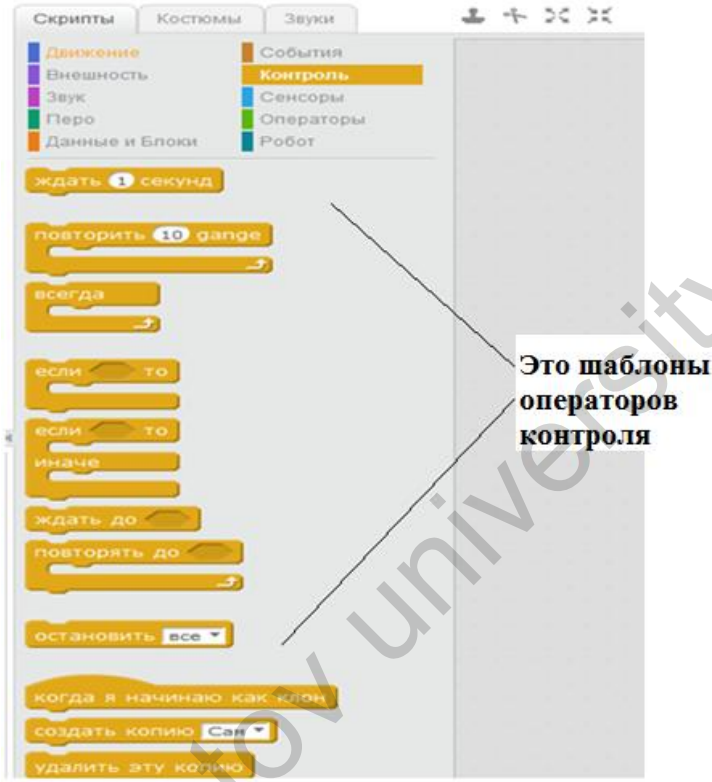


Рисунок 2.38 Шаблоны операторов контроля

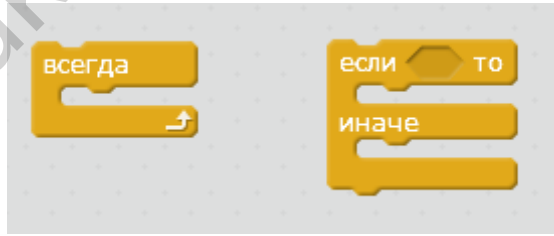


Рисунок 2.39 Операторы контроля ВСЕГДА и ЕСЛИ_ТО, ИНАЧЕ

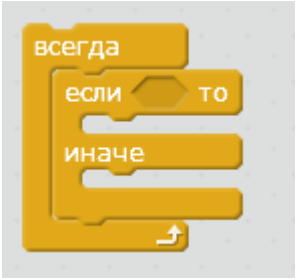
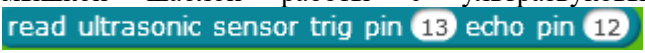


Рисунок 2.40 Логическая конструкция необходимая для работы с датчиками

Далее необходимо уточнить логическое условие, с которым будет работать датчик ( больше,  меньше,  равно). Все логические операторы находятся в разделе основного меню **ОПЕРАТОРЫ** (выделен зеленым цветом). При входе в раздел **ОПЕРАТОРЫ** появляется меню шаблонов логических операторов (рисунок 2.41).

Выбираем необходимый логический шаблон и вставляем данный шаблон в логическую конструкцию, состоящую из операторов контроля **ВСЕГДА и ЕСЛИ_ТО, ИНАЧЕ** (рисунок 2.42)**.**

Теперь в логическую конструкцию необходимо вставить тип используемого датчика и указать условие срабатывания данного датчика. Например, необходимо при помощи ультразвукового датчика определить момент, когда расстояние до препятствия будет меньше 0,5 метра. Для этого переходит в основном меню в раздел **РОБОТ (**рисунок 2.19) и выбираем шаблон работы с ультразвуковым датчиком. Захватываем мышкой шаблон работы с ультразвуковым датчиком 

и вставляем этот шаблон в логическую конструкцию внутрь логического

оператора  (меньше), слева от знака неравенства (рисунок 2.43).

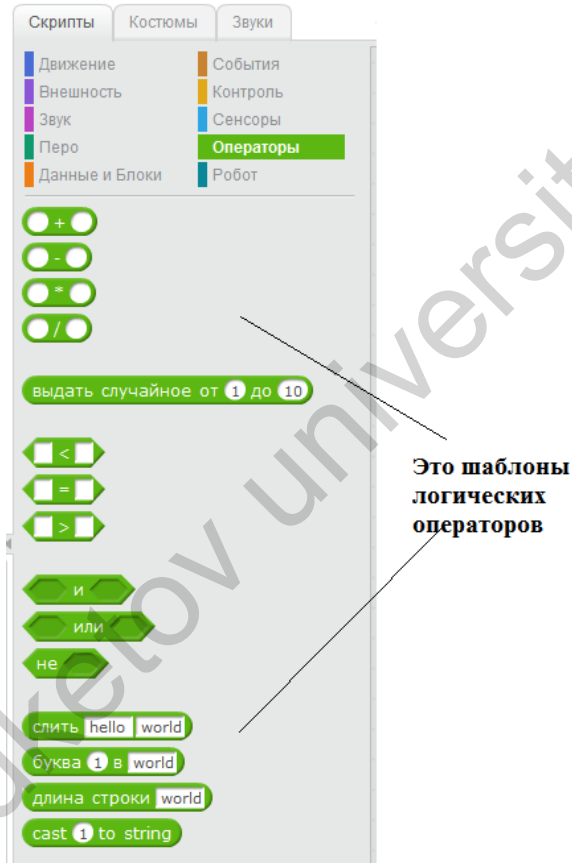


Рисунок 2.41 Шаблоны логических операторов

При использовании в шаблоне управления ультразвукового датчика **pin 13** и **pin 12** ультразвуковой датчик должен быть подключен к плате контроллера Arduino к разъему под номером **3** (рисунок 2.15).

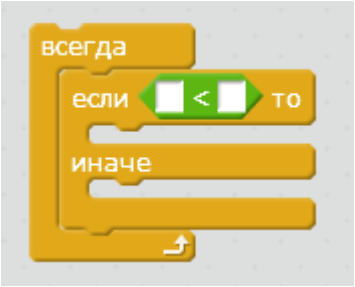


Рисунок 2.42 Логическая конструкция с уточняющим логическим условием

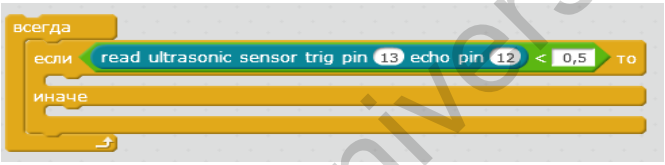
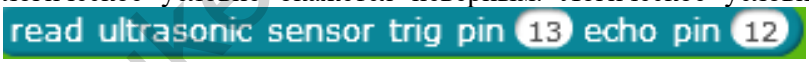
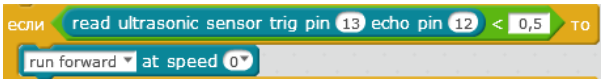


Рисунок 2.43 Логическая конструкция с вставленным шаблоном управления ультразвуковым датчиком

Далее необходимо определить действия, которые будут производиться роботом, если логическое условие, записанное в логической конструкции, окажется верным и действие, если логическое условие окажется неверным. Логическое условие 

записано в логической конструкции между **ЕСЛИ** и **ТО.** Например, в логическом условии может быть прописано указание о том, что при обнаружении препятствия на расстоянии меньше **0,5** метра робот останавливается – если это условие не выполняется робот, продолжает движение со скоростью **100**. Тогда приказ об остановке робота 

будет вставлен внутрь оператора управления **ЕСЛИ_ТО, ИНАЧЕ** непосредственно под логическим условием



в верхнюю строку.

Если же логическое условие не выполняется (препятствий на пути нет), происходит выполнение приказа о прямолинейном движении со скоростью **100**:

.

В результате программа по обнаружению препятствий ультразвуковым датчиком будет иметь вид, показанный на рисунке 2.44 .

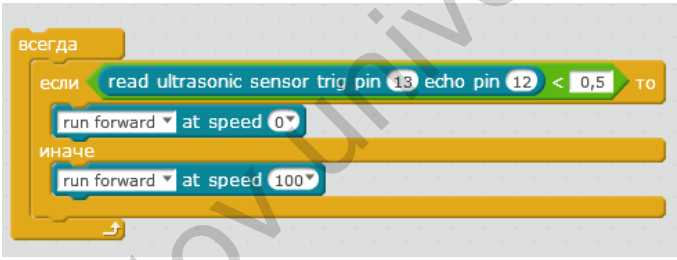


Рисунок 2.44 Логическая конструкция с вставленным шаблоном управления ультразвуковым датчиком

2.9.2 Датчик определения наличия горючих газов. Разъем 8

Датчик определения наличия горючих газов /углеводородов/ MQ-**2** определит концентрацию углеводородных газов (**пропан, метан, н-бутан**), дыма (взвешенных частиц, являющихся результатом горения) и водорода в окружающей среде (рисунок 2.45) [25]. На задней

поверхности датчика имеется надпись MeGasSensor (MQ2) V1.0. ДатчикMQ-2 имеет аналоговый **A0** и цифровой **D0.**

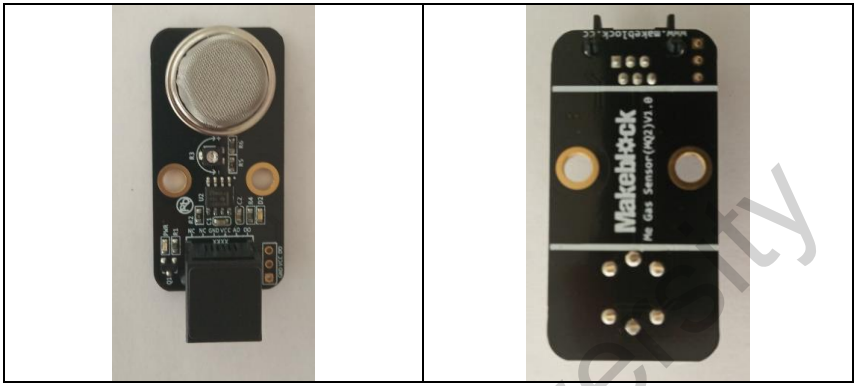


Рисунок 2.45 Датчик обнаружения углеводородных газов MQ-2

При использовании аналогового интерфейса **A0** датчик должен быть подключен к контроллеру Arduino через **разъем 8 (**рисунок 2.15). При срабатывании датчика обнаружения горючих газов MQ-2 на датчике загорается **синий индикатор** (при этом красный индикатор сигнализирует о наличии питания). Внутри датчика под сеткой находится нагревательный элемент и перед использованием датчик должен быть прогрет не менее **20 секунд** (желательно час).

Диапазон измерений датчика газов MQ-2:

1) Пропан: **200–5000** ppm;
2) Бутан: **300–5000** ppm;
3) Метан: **500–20000** ppm;
4) Водород: **300–5000** ppm.

Имеющиеся датчики MQ-2 протестированы в диапазоне **от 60 до 900**. Не используйте в программах значения уровня сигнала датчика **MQ-2 меньше** 60!

Характеристики датчика газов MQ-2:

- Напряжение питания нагревателя: **5 В.**

- Напряжение питания датчика: **3,3–5 В.**
- Потребляемый ток: 150 мА.
- Габариты: 25,4×25,4 мм.

Рассмотрим работу программы по обнаружению горючих углеводородных газов (рисунок 2.46).

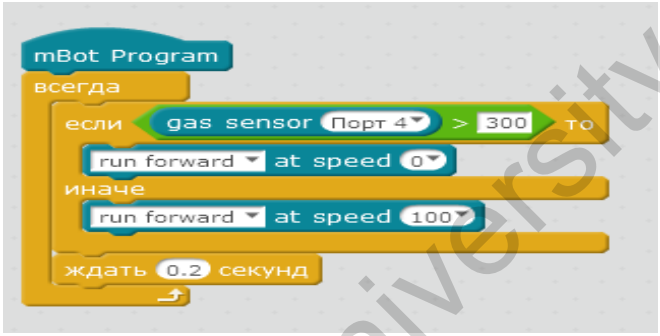
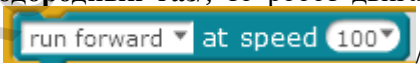


Рисунок 2.46 Программа по контролю уровня газа

Используется аналоговый **разъем 8** на плате контролера А**0.** Для данного разъема и датчика газов работает **обратная логика** – если сигнал на датчике горючих газов **более 300** 

/обнаружен горючий углеводородный газ/, то робот двигается вперед со скоростью **100** 

/оба маршевых двигателя работают/. Если на датчике газов сигнал менее 300 – **фактически сигнала нет /газ не обнаружен/**, то робот стоит на месте 

– скорость его равна нулю.

При использовании газового датчика на аналоговом разъеме 8 контролера необходимо читать логический оператор снизу вверх.

2.9.3 Акселератор и Датчик поворота всего робота Ultimate–Гироскоп. Разъем 8

Гироскопический датчик — это цифровой датчик, который обнаруживает изменение направления движения по одной из 3-х осей (рисунок 2.47):

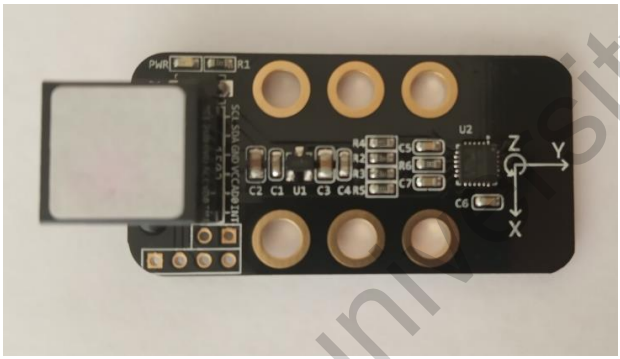


Рисунок 2.47 Акселератор и Датчик поворота

Ось X – (акселерометр) учитывает отклонение датчика от вертикали типа **качание плоскости движения.** При использовании данного режима необходимо строго ориентировать гироскоп – при движении вперед ось X на гироскопе должна указывать налево и в программе необходимо указывать углы поворота в градусах: при этом, если **правый** край датчика поворота начнет опускаться, это будет соответствовать **положительному отклонению** – углы поворота при этом задаются положительными значениями градусов /например **30/**.

1-X. На рисунке 2.48 показана программа, приказывающая роботу двигаться вперед со скоростью **100** /оба маршевых двигателя работают на одинаковой скорости 100/ если угол отклонения по оси X больше **30** градусов

. Датчик – акселератор, при этом отклонен от вертикального положения больше, чем на 30 градусов.

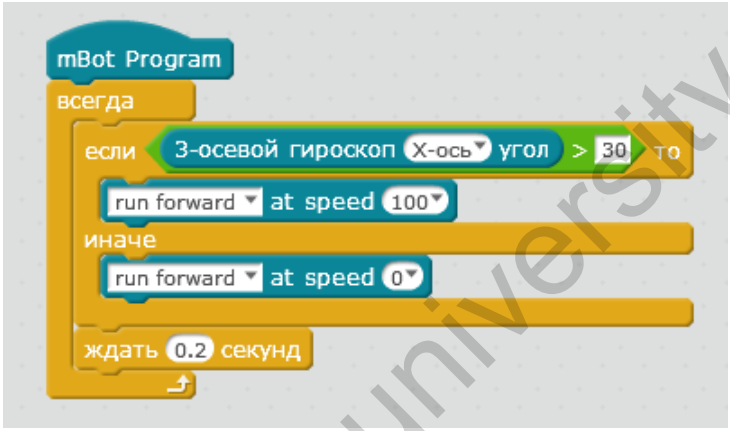


Рисунок 2.48 Программа подключения акселератора

Если же угол отклонения от вертикали меньше **30** градусов, то роботу поступает команда не двигаться /оба маршевых двигателя выключены/ - скорость вперед указана равной нулю

.

Датчик при этом отклонен от вертикального положения меньше, чем на 30 градусов.

Таким образом, на рисунке 2.48 записана программа, приказывающая двигаться вперед роботу при условии, что робот отклонился от строго горизонтального положения более чем на **30** градусов – если же это условие не выполняется робот не движется вперед /его скорость равна нулю/.

Команда 

показывает, что положение датчика-акселератора проверяется каждые **0,2** секунды.

2-X. Если же опускается **левый** край датчика поворота - это соответствует **отрицательному** отклонению: углы поворота в программе при этом задаются отрицательными значениями градусов, например, **-30. Разъем 8**

На рисунке 2.49 показана программа, приказывающая роботу двигаться вперед при отклонении от вертикали вниз левого края датчика-акселератора на 55 градусов / -55градусов/, если отклонение левого края вниз меньше чем

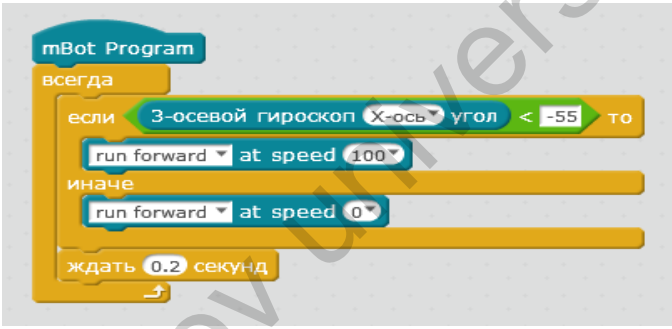


Рисунок 2.49 Акселератор с отрицательным покачиванием

55 градусов робот не двигается. **При работе с отрицательными углами необходимо учитывать, что чем больше отрицательный угол, тем меньше его числовое значение**, т. е. число характеризирующее, например угол -30 больше числа соответствующего углу -55. Необходимо также в программе при работе с отрицательными углами менять логику **/заменить знак > на знак <.** Команда

означает, что как только числовое знечение угла отклонения акселератора станет меньше определенного числа, соответсвующего углу отклонения -55 градусов произойдет

срабатывание условия и маршевые двигатели заработают **вперед со скоростью 100.**

1X) Если угол отклонения акселерометра положительный, используется логический оператор *больше* >  .

2X) Если угол отклонения акселератора отрицательный, используется логический оператор *меньше* <  .

Ось У – учитывает отклонение робота в вертикальной плоскости от горизонтали – вверх или вниз. При использовании данного режима необходимо строго ориентировать гироскоп – при движении вперед ось X на гироскопе должна указывать **налево** и в программе необходимо указывать углы поворота в градусах в положительных цифрах. При отклонении задней грани датчика вверх / - *на эту поверхность указывает стрелка Y/* - робот при этом движется вниз/ необходимо указывать положительный угол в градусах, например 30. Когда робот **движется вверх**, задняя грань гироскопа - *на эту поверхность указывает стрелка Y/* **опускается вниз** и нужно указывать углы отклонения с отрицательным знаком /, например -30. **Разъем 8.**

1-У)

На рисунке 2.50 представлена программа, которая приказывает роботу двигаться вперед со скоростью 100 

если угол отклонения датчика-гироскопа от вертикали превышает 30 градусов 

.

/задняя грань гироскопа движется вверх/.

Если угол отклонения датчика – гироскопа от горизонтали не превышает 30 градусов, робот не двигается /его скорость вперед равна нулю/

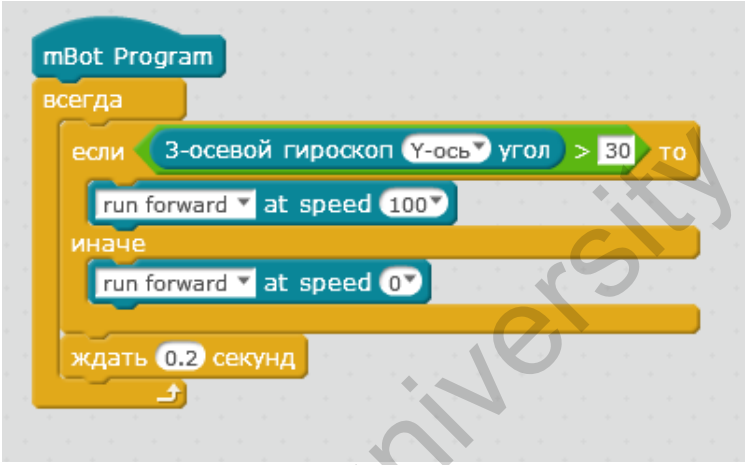
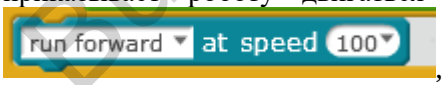


Рисунок 2.50 Программа подключения датчика вертикального отклонения

Команда 

показывает, что положение датчика-гироскопа проверяется каждые 0,2 секунды.

2-У) На рисунке 2.51 представлена программа, которая приказывает роботу двигаться вперед со скоростью 100 

, если угол отклонения датчика-гироскопа от вертикали становится больше **минус 30 градусов (-30)**

.

/задняя грань гироскопа движется вниз/.

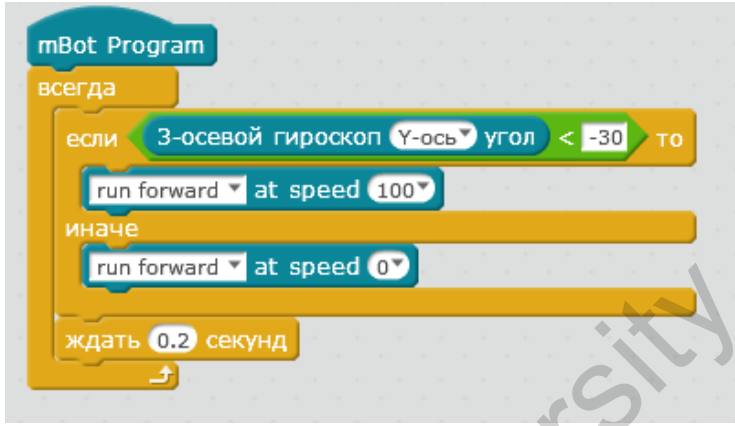
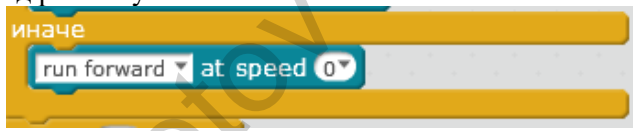


Рисунок 2.51 Программа отклонения вертикального датчика с отрицательными углами

Если угол отклонения датчика–гироскопа от горизонтали не превышает **минус 30 градусов**, робот не двигается /его скорость вперед равна нулю/



.

Команда показывает, что положение датчика-гироскопа проверяется каждые 0,2 секунды.

При работе с отрицательными углами необходимо учитывать, что чем больше отрицательный угол, тем меньше его числовое значение, то есть число, характеризирующее, например **угол -15,** больше числа, соответствующего **углу -30**. Необходимо также в программе при работе с отрицательными углами менять логику **/заменить знак > на знак < /.**

Команда

означает, что как только числовое знечение угла отклонения гироскопа станет меньше определенного числа, соответсвующего углу отклонения -30 градусов, произойдет срабатывание условия и маршевые двигатели заработают **вперед со скоростью 100.**

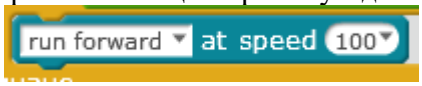
1-У) Если угол отклонения гироскопа положительный, используется логический оператор *больше* > 

.

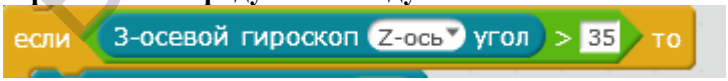
2-У) Если угол отклонения гироскопа отрицательный, используется логический оператор *меньше* < 

.

Ось Z учитывает отклонение робота в горизонтальной плоскости – влево или вправо. Учет отклонения происходит в градусах. Отклонение в течении работы всей программы отсчитывается от первоначального направления, заданного при включении робота */при плохой работе в данном режиме необходимо жестко закрепить контакт в разъеме датчика-поворота/*.

Гироскопы подключаются к разъемам, имеющим белую маркировку */порты 8,7,6,4,3/*. На рисунке 2.52 показана программа, приказывающая роботу двигаться вперед со скоростью **100** 

/оба маршевых двигателя включены/, если произошло отклонение датчика поворота в горизонтальной плоскости **против часовой стрелки** на **35 градусов по модулю**



от направления первоначального движения. Робот при этом двигается **влево в горизонтальной плоскости.**

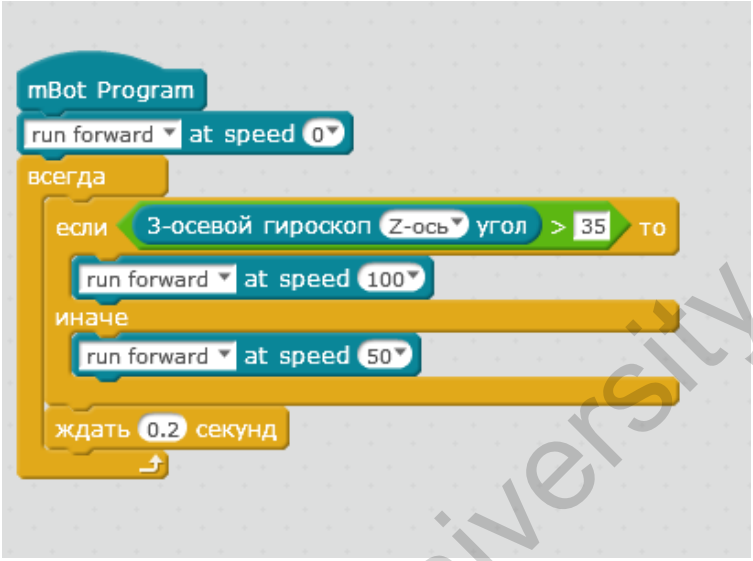
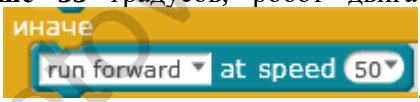


Рисунок 2.52 Программа подключения датчика поворота

Если угол отклонения датчика поворота против часовой стрелки **меньше 35** градусов, робот двигается вперед со **скоростью 50** 

. */оба маршевых двигателя включены/*.

Программа, приведенная на рисунке 2.52, читается:

– Если угол отклонения датчика-поворота от первоначального направления больше **35 градусов,** робот двигается со **скоростью 100**.

– Если угол поворота датчика-поворота меньше **35 градусов,** робот двигается **со скоростью 50**.

На рисунке № 2.53 показана программа, приказывающая роботу двигаться вперед со скоростью **100**

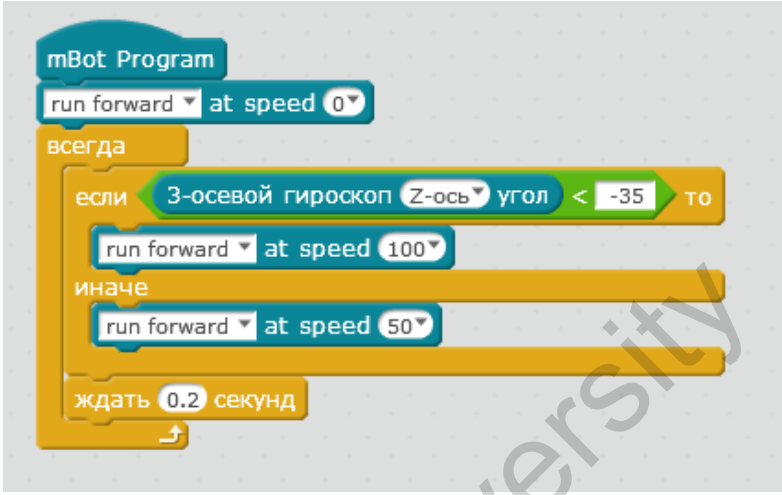
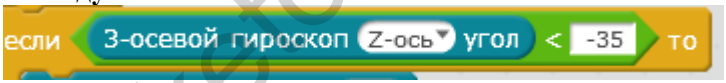


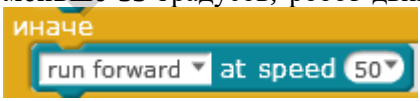
Рисунок 2.53 Программа подключения гироскопа

/оба маршевых двигателя включены/, если произошло отклонение датчика поворота в горизонтальной плоскости **по часовой стрелке** на **35 градусов по модулю**



от направления первоначального движения. Робот при этом двигается **вправо в горизонтальной плоскости.**

Если угол отклонения датчика поворота по часовой стрелке **меньше 35** градусов, робот двигается вперед со **скоростью 50**



/оба маршевых двигателя выключены/.

При работе с отрицательными углами надо учитывать, что чем больше отрицательный угол, тем меньше его числовое

значение */числовое значение угла (-15градусов) больше числового значения угла (-35 градусов)/. Условие, записанное на рисунке, надо читать так: Как только числовое значение угла, на который произошел поворот от первоначального направления, станет меньше числового значения угла (-35 градусов), начнется движение вперед со скоростью 100. Если числовое значение угла поворота больше числового значения угла (-35 градусов), скорость движения вперед равна 50.*

3-Z) Если угол отклонения гироскопа положительный,

используется логический оператор *больше* > . *Робот при этом поворачивает налево - задняя часть датчика поворота движется **против** часовой стрелке.*

3-Z) Если угол отклонения гироскопа отрицательный,

используется логический оператор *меньше* < . *Робот при этом поворачивает направо - задняя часть датчика поворота движется **по** часовой стрелке.*

Перед запуском программы, в которой используется датчик поворота в горизонтальной плоскости, необходимо нажать кнопку **RESET** на контроллере. При плохой работе в данном режиме необходимо жестко закрепить контакт в разъеме датчика- поворота/.

2.9.4 Ультразвуковой датчик расстояния. Разъем 4

Ультразвуковой датчик — это цифровой датчик, который определяет расстояние до находящегося перед ним объекта. Он делает это, посылая звуковые волны высокой частоты и измеряя время, за которое звук отразится назад к датчику.

На рисунке 2.54 представлена программа, приказывающая роботу, оснащенному ультразвуковым датчиком расстояния,

направленным по курсу движения, двигаться вперед */оба маршевых двигателя работают вперед со скоростью 100/*, если в пределах **0,5 метра** от него по курсу движения **нет препятствия**. Если на расстоянии менее **0,5 м**етра **возникает препятствие** ультразвуковой датчик его обнаруживает и роботу поступает приказ прекратить движение.

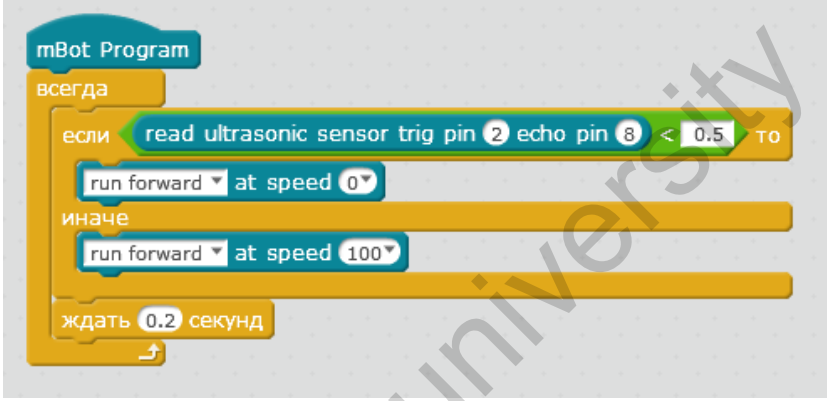
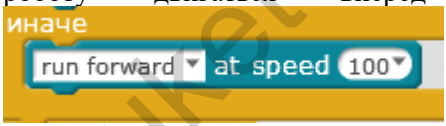


Рисунок 2.54 Порты подключения ультразвукового датчика

На данном рисунке показана программа, приказывающая роботу двигаться вперед со скоростью **100**



/оба маршевых двигателя включены /, если ультразвуковой датчик определения расстояния определяет, что по курсу движения нет препятствий на расстоянии менее **0,5 метра**

Если же ультразвуковой датчик обнаруживает препятствие на расстоянии менее **0,5 метра** по курсу движения, робот

получает приказ прекратить движение 

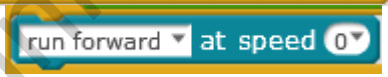
.

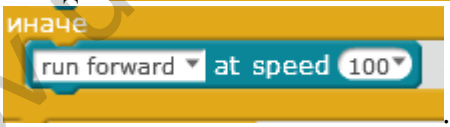
Команда 

производит определение расстояния до препятствия каждые **0,2 секунды**.

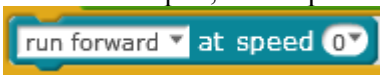
При работе с **портом 4** необходимо в логическом условии использовать **pin 2 и 8.**

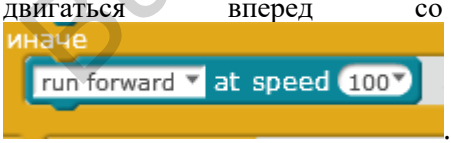
Обратите внимание! При использовании ультразвукового датчика программа читается сверху вниз – при срабатывании логического условия, записанного в операторе 

происходит выполнение команды 



и игнорирование команды .

Если же логическое условие неверно, то происходит игнорирование команды 

и происходит выполнение команды, приказывающей роботу двигаться вперед со скоростью 100 

.

При тестировании работы с ультразвуковым датчиком необходимо использовать в качестве препятствия твердый предмет большей плоскости /*лучше всего книга в твердом переплете*/, не располагать предмет ближе 10 см от датчика, что

связано с особенностями работы данного типа ультразвуковых датчиков.

Контрольные вопросы

1. Назовите базовый набор ULTIMATE ROBOT KIT?
2. Для чего предназначена алюминиевая обшивка?
3. Что такое микроконтроллер?
4. Опишите принцип работы микроконтреллера?
5. Для чего предназначен датчик линии?
6. Какой датчик измеряет интенсивность звука в пространстве?
7. Какая программ позволяет создавать управление роботами типа ULTIMATE ROBOT KIT?
8. Что позволяет подключить адаптер?
9. Что такое акселератор?
10. С помощью какого устройства можно управлять роботами по беспроводной сети?

Глава 3 Человекообразный робот Bioloid

3.1 Контроллер робота Bioloid

3.1.1 Описание системы управления контроллеров CM-530

В данном роботе всё управление осуществляется через контроллер **CM-530.** К данному контроллеру подключаются моторы и датчики, контроллер имеет систему индикации различных режимов, кнопку включения питания и кнопку запуска выполнения программы, находящейся в памяти контроллера [26]. Контроллер **CM-530** имеет **5** портов **DXL** для подключения моторов; **6** портов **GPIO** для подключения датчиков; **1** порт для подключения к компьютеру и разъем подключения питания. На рисунке 3.1 представлен общий вид контроллера CM-530.

Рисунок 3.1 Общий вид контроллера CM-530

Для включения робота **Bioloid** необходимо в случае отсутствия аккумулятора подключить к контроллеру CM-530 блок питания и нажать кнопку включения (рисунок 3.2): после

включения должен светить красным цветом индикатор над кнопкой включения.

Рисунок 3.2 Разъемы контроллера CM -530 для подключения питания и связи с компьютером

Если на панели индикации высвечивается красным светом режим **PLAY** (в этом режиме запускаются записанные в память контроллера программы) необходимо нажать кнопку **START:** после этого произойдет выполнение программы.

Если после включения на панели индикации высвечивается режим **MANAGE** или **PROGRAM**, необходимо нажать кнопку **MODE**, которая производит переключение режимов.

На рисунке 3.3 показана панель управления контроллера CM 530 с обозначенными основными элементами.

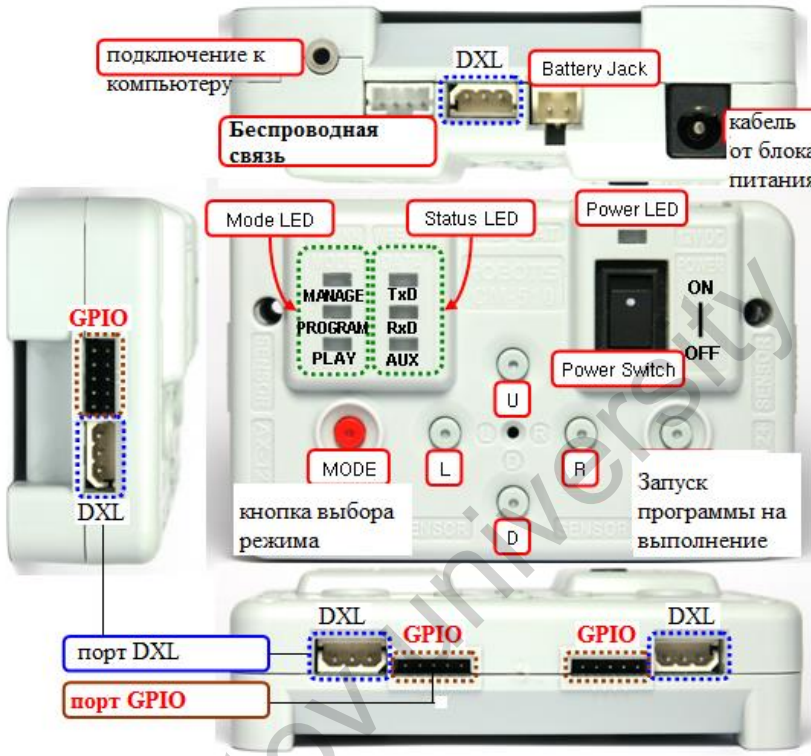


Рисунок 3.3 Панель управления контроллера CM 530

Далее представлена спецификация процессора CM-530:

- Вес: 54 г
- Контроллер: STM32F103RE
- Рабочее напряжение
- Допустимый диапазон: 6V ~ 15V
- Рекомендуемое Напряжение: 11.1V (Li-Po 3Cell)
- Потребляемый электрический ток
- В режиме ожидания: 50 мА
- Внешний вход/выход; Максимальный ток: 300 мА
- Всего Максимальный ток: 10A (предохранитель)
- Рабочая температура: -5 ℃ ~ 70 ℃
- Внутренний ввод/вывод устройства

- Кнопка: 5 (Сброс 1, Порт 5)
- Микрофон (для обнаружения звука): 1
- Датчик напряжения: 1
- Внешний вход/выход устройства
- **GPIO** совместимый 5 контактный I/O 6 портов
- Порт для AX/MX серии Dynamixel: 5

Режимы эксплуатации робота Bioloid

В данной модели предусмотрено 3 режима эксплуатации:

1. PLAY
2. PROGRAM
3. MANAGE

Play – это режим воспроизведения, записанное в память программы. Программа записывается в контроллер CM-530 при помощи установленной на компьютере программы **RoboPlusTask**. Кнопка **Start** должна быть нажата пользователем для выполнения, когда светодиод **Play** мерцает.

Program – это режим записи программы из компьютера в контроллер CM-530. На компьютере при этом используется программа **RoboPlusMotion**. Режим активен, если робот подключен к компьютеру.

Manage – означает управление и:

- показывает, что контроллер находится в режиме управления Dynamixel;
- используется для установки или тестирования операций CM-5x0, AX/MX Dynamixel с помощью RoboPlusManager;
- включается автоматически, когда RoboPlusManager и CM-5x0 подключены.

3.1.2 Панель управления контроллера CM-530

- **PC Link** - связь с ПК (CM-530 USB порт): используется для подключения кабеля интерфейса к CM-5x0 и ПК.

Используется для связи с другим компьютером или загрузки кода задачи.

- **Communication device** - разъём для подключения коммуникационного устройства: используется для беспроводной связи с BT-110A, ZIG-110, IR-приемником, модулями или другими устройствами.
- **Battery Jack** - разъем аккумулятора: используется для подключения аккумулятора.
- Power Jack - разъем питания: используется для подключения блока питания SMPS.
- **Power LED** - индикатор питания: индикатор состояния питания **Вкл**. и **Выкл**.
- **Power Switch** - переключатель питания: используется для включения/выключения робота.
- **MODE** - режим: кнопка используется для выбора режима работы **CM-530**. Позволяет переключать работу контроллера между режимами **MANAGE, PROGRAM, PLAY**.
- **START** - старт: кнопка используется для запуска выбранного режима.
- U / L / D / R кнопки: функции данных кнопок могут быть назначены программно.
- **DXL порт.** 3 pin Dynamixel port - разъем Dynamixel: разъем шины AX/MX используется для подключения сервомоторов Dynamixel в последовательное соединение. **Данных портов 5.**
- 5 pin Auxdevice port (**GPIO)** - порт для подключения периферийных устройств: используется для подключения **датчиков** и других периферийных устройств. **Таких портов 6** – также имеют название. **GPIO** порт ввода/вывода общего назначения, служит для низкоуровневого обмена цифровыми сигналами с внешними по отношению к микроконтроллеру устройствами.
- **Mode Led** - светодиод режима работы: светодиодный индикатор для отображения текущего режима работы CM -530, см. ниже:
- **Status Led** - светодиод состояния: индикатор отображает текущее состояние CM-530:

Значение индикаторов панели **Status Led** следующее:

- **TXD**: активен, пока СМ-530 передаёт данные.
- **RXD**: активен, пока СМ-530 получает данные.
- **AUX**: назначенный индикатор может использоваться пользователем в программе.

Подключение электропитания

- Контроллер может работать от аккумулятора либо от блока питания **SMPS** (рисунок 3.4).
- Батарея должна быть подключена в гнездо аккумулятора **СМ-530.**
- При использовании блока питания необходимо перевести переключатель питания в положение **ON**.
- Если ток поступает, то индикатор включен и один из светодиодов мерцает.

Рисунок 3.4 Подключение контроллера СМ 530
к блоку питания

Включение

▪ **Включение**: для включения питания, переведите переключатель из положения OFF в положение ON.

Если питание не включается, несмотря на перемещение переключателя в положение ON, возможно, разрядилась аккумуляторная батарея. Перезарядите батарею или подключите блок питания.

▪ **Start**: для перехода в режим воспроизведения записанной в контроллер программы управления роботом, используйте кнопку **Mode**, чтобы управлять режимом работы. Нажмите кнопку **START**, когда светодиод **PLAY** мерцает.

При нажатии на кнопку **START** индикатор **PLAY** больше не должен мерцать; это нормальное состояние исполнения задачи.

Если вы хотите остановить исполнение задачи, нажмите кнопку **MODE**, чтобы вернуться в режим ожидания, или выключите питание.

Использование

▪ **RoboPlus Manager** используется для контроля моторов Dynamixel и периферийных устройств. Она может подключаться к Dynamixel (AX / MX Dynamixel и т. д.) и различным компонентам Ollo (сенсорный датчик, светодиодный модуль, IR-датчик и т. д.).

▪ Движения робота могут быть изменены и сохранены через **RoboPlus Motion**. Сохраненные движения (mtn-файл) могут быть всегда выполнены **RoboPlusTask**, а также движение каждой части можно регулировать путем записи кодов задач для управления.

Подключение к ПК

▪ Контроллер **CM-530** подключается к USB порту ПК.

▪ Для подключения используется кабель с разъемом милли USB (рисунок 3.5).

Рисунок 3.5 Подключение контроллера CM 530
к компьютеру

3.1.3 Инфракрасный датчик Robotis

Для работы с контроллером CM-530 используется специальный совместимый ИК-датчик (рисунок 3.6), который подключается к контроллеру через **GPIO порты** (рисунок 3.3). **GPIO** (general-purpose I/O port) - порт ввода/вывода общего назначения, служит для низкоуровневого обмена цифровыми сигналами с внешними по отношению к микроконтроллеру устройствами.

1. SIG1: поддерживает низкий сигнал;
2. GND: контакт заземления;
3. ADC: вывод значения, обнаруженного ИК-приемником в аналоговом сигнале;
4. VCC (3,3 В);

5. SIG2: может включить светодиод, посылая высокие сигналы.

Рисунок 3.6 Инфракрасный датчик

Расстояние обнаружения объекта данным датчиком **- не более 15 см.**

Контроллер **CM-530** имеет **6 портов GPIO** для подключения датчиков.

3.2 Методика программирования

3.2.1 Основы программирования контроллера CM 530

Robotis предоставляет пользователю собственную среду разработки RoboPlus. Данное программное обеспечение является единым для всей продукции Robotis и применяется для семейства OLLO, Bioloid, Expert kit [27].

Для начала процесса программирования открываем программу **RoboPlus**. Появляется окно RoboPlus (рисунок 3.7), после чего выбираем пункт **RoboPlus Task**.

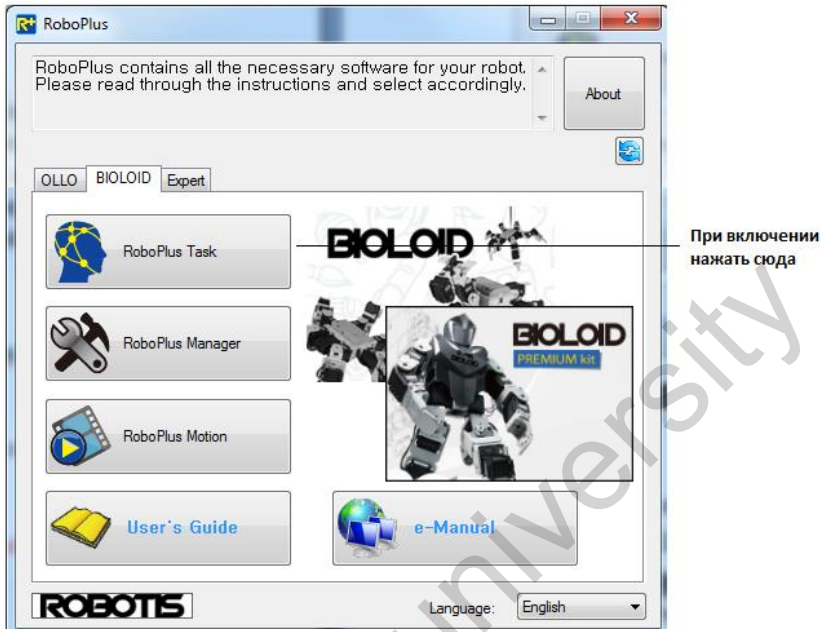


Рисунок 3.7 Окно **RoboPlus**

Открывается окно **RoboPlus Task** с панелью управления и инструментами (рисунок 3.8). В данном окне производится написание программ, производящих управление контролером CM-530 [28].

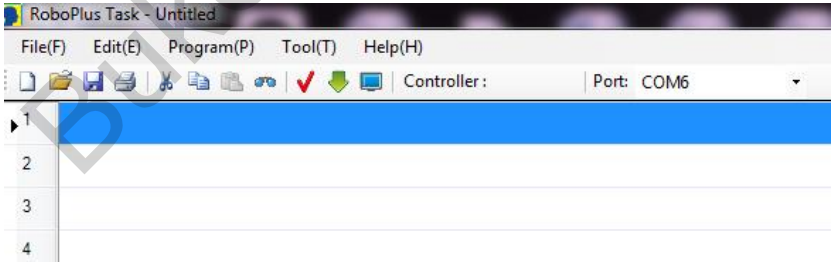


Рисунок 3.8 Окно **RoboPlus Task**

Чтобы осуществить запись программы на устройство, мы подключаем микроконтроллер CM-530 к ПК через USB-кабель

(рисунок 3.9). При этом микроконтроллер должен быть подключен к источнику питания и включен.

Рисунок 3.9 Подключение к контроллеру CM-530 штекера питания и разъема USB.

Затем на панели инструментов выбираем нужный порт Port: COM5 (рисунок 3.10).

Далее необходимо указать какой тип контроллера подключен к программе **RoboPlus**. Для этого на панели управления окна **RoboPlusTask**. нажимаем на кнопку Controller: - появляется окно **Please select the appropriate controller** (рисунок 3.11).

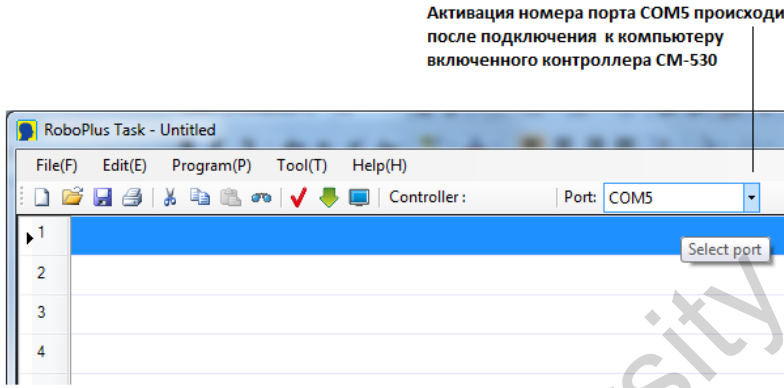


Рисунок 3.10 Выбор порта подключения

Рисунок 3.11 Первая страница меню Выбора контроллера

Для выбора контроллера CM-530 необходимо в окошке **Please select the appropriate controller** нажать на кнопку **1.0** (поставить галочку на ячейку 1.0) В результате появится окно с тем же названием, но с другим списком контроллеров (рисунок 3.12).

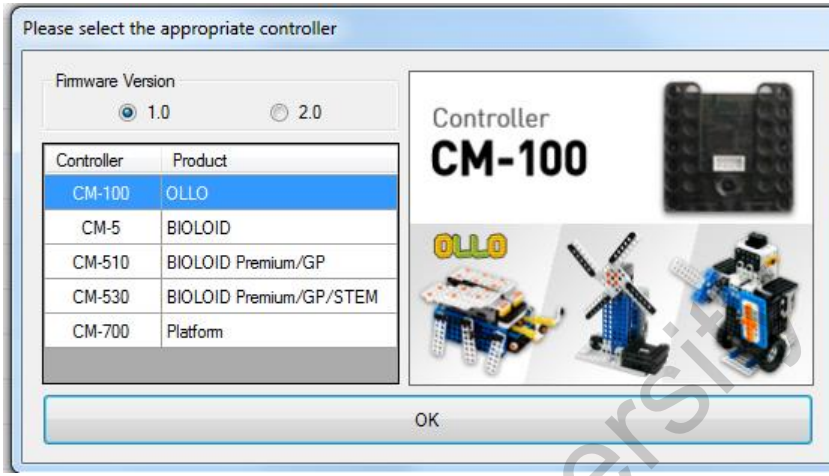


Рисунок 3.12 Вторая страница меню Выбора контроллера

Необходимо выбрать контроллер CM-530: появится окно **Please select the appropriate controller** с выбранным контроллером **CM-530.** Нажимается кнопка **OK** (рисунок 3.13).

Рис 3.13 Выбор контроллера CM-530

В результате на экране появится рабочее окно для написания программы управления под управлением контроллера CM-530 через порт **COM5** (рисунок 3.14).

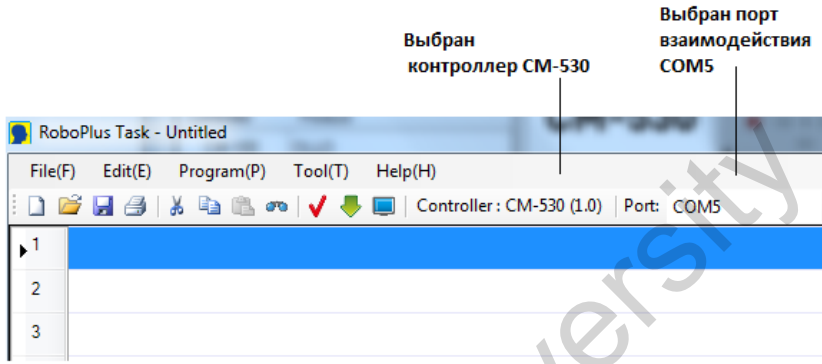


Рисунок 3.14 Окно с выбранным типом контролера CM-530 и портом COM5

В данном окне пишется текст программы управления роботом Bioloid, которая затем записывается в контроллер CM-530 (для этого используется режим **PROGRAM** контроллера CM-530; прерывисто светится синяя лампочка). При этом должен светиться индикатор **RxD.**

Создание программы управления начинаем с двойного щелчка на первой строке поля ввода данных. При этом появляется окно **Select instruction type,** содержащее контекстное меню (рисунок 3.15).

Выбираем команду начала создания программы **Start program** (щелкаем по этой команде 2 раза). На экране появляется команда **START PROGRAM** в строке №1, за которой стоят фигурные скобки, составляющие кодовый блок, внутри которого создается все программное управление (рисунок 3.16).

Далее мы нажимаем на свободную строку №3 **(**рисунок 3.17), расположенную между фигурными скобками и нажатием двойного щелчка выводим контекстное меню и выбираем пункт **?=? (LOAD).**

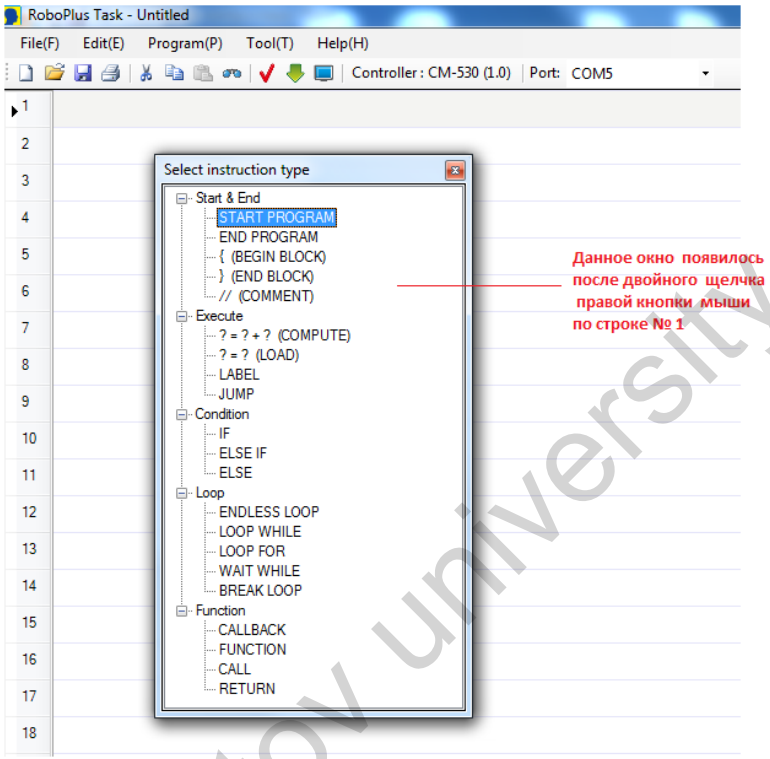


Рисунок 3.15 Окно **Select instruction type**

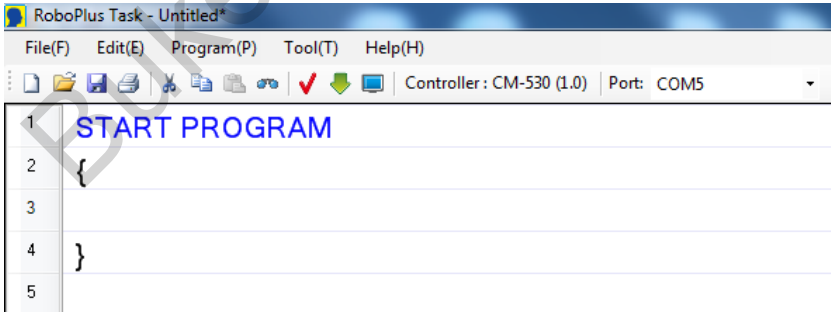


Рисунок 3.16 Показан результат выполнения команды **START PROGRAM**

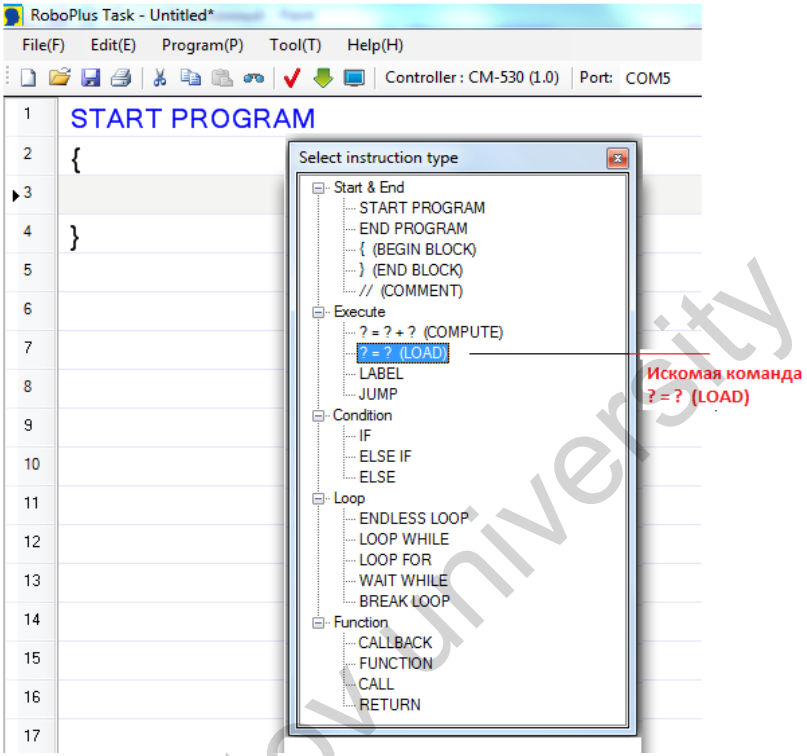


Рисунок 3.17 Выбор команды **? = ? (LOAD)**

На строчке №3 появляются два знака вопроса разделенных арифметическим знаком «равно» (рисунок 3.18).

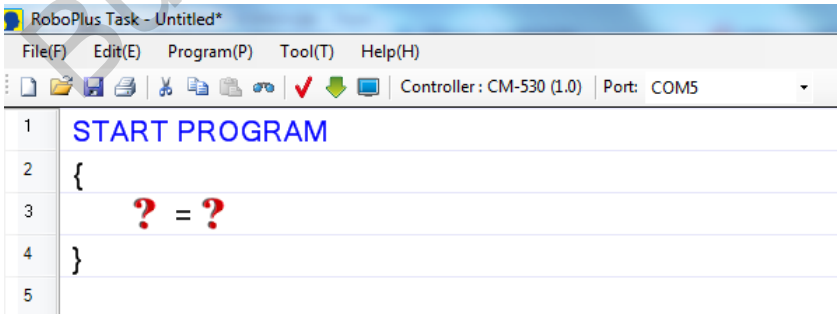


Рисунок 3.18 Результат выбора команды **? = ? (LOAD)**

Слева от знака «=» записываем тип команды (например, управление мотором), а справа - параметры этого управления (значения скорости и направления). Наводим мышь на левый знак вопроса и кликаем два раза – в результате появится служебное окно, показанное на рисунок 3.19.

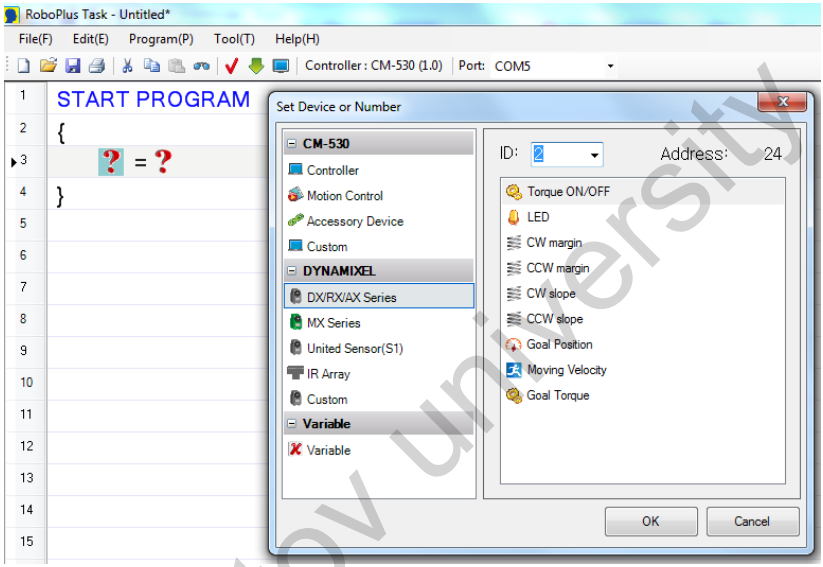


Рисунок 3.19 Служебное окно **Set device or Number**

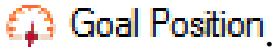
В появившимся окне **Set device or Number** в левой части окна выбираем пункт **DYNAMIXEL** – далее переходим в подменю **DX/RX/AX Series**. В данном подменю устанавливаются рабочие характеристики работы мотора **типа AX-12A**: например, скорость вращения мотора 

, направление вращения мотора (по часовой **CW** или против часовой стрелки **CCW**), номер позиции куда должен вращаться мотор. В правой части окна появляется меню в пункте **ID** ставим цифру **2** — это означает, что используется мотор типа **AX-12A**, имеющий внутренний номер **2** (рисунок 3.20).

Рисунок 3.20 Мотор типа AX-12A

Внутренняя нумерация номера мотора (номер мотора) указана на корпусе под контактами (рисунок 3.21). Спецификация **ID** показывает, что используется мотор типа **AX-12A**. Моторы данной серии имеют внутренний сервопривод и внутренний контроллер, позволяющий вращать двигатель до нужной позиции, и блокирование вращения на указанной позиции. Также на корпусе моторов **типа AX-12A** имеются два разъема – через один из разъемов мотор получает управляющие команды и если команды предназначены не для данного мотора (с другим номером ID), то команды пересылаются через второй разъем на другой мотор.

В этом же окне **Set device or Number** устанавливаем конечную позицию движения мотора **№2** (рисунок 3.21): в

правой части окна – выделяем запись **Goal Position** 

.

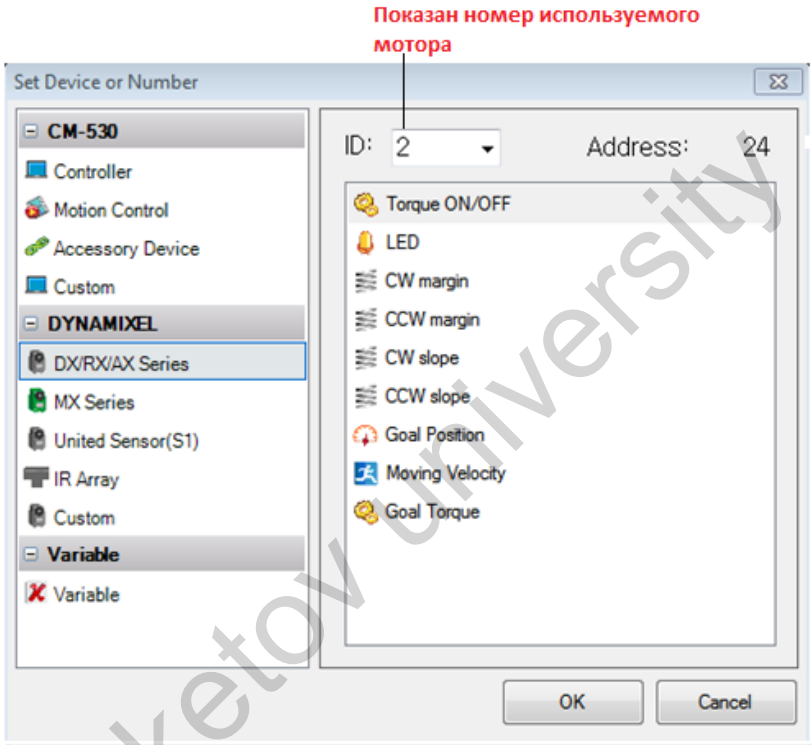


Рисунок 3.21 Выбор номера мотора

Данная команда позволяет при включении мотора **ID: 2** определить к какой позиции должен двигаться мотор при вращении (рисунок 3.22).

Номера позиций могут меняться от **0** до **1023**, мотор при этом может двигаться с заранее заданной скоростью как по часовой, так и против часовой стрелки.

После выбора этой строки нажимаем **кнопку ОК** – в результате на экране появляется окно **RoboPlus Task** данного вида (рисунок 3.23), в котором указано в левой части команды

(перед знаком =), что мотор **№2** должен двигаться к определенной позиции (но номер позиции пока не определен – его надо установить в правой части команды после знака =).

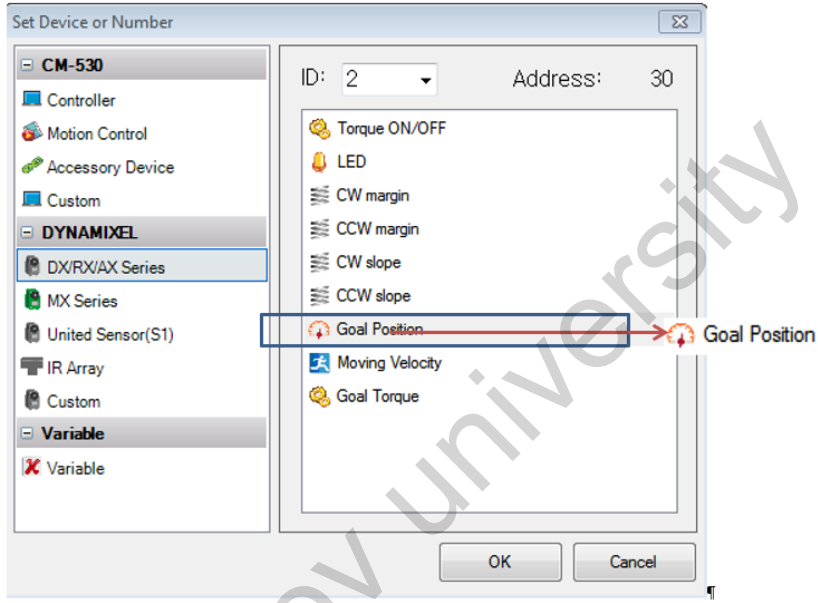


Рисунок 3.22 Окно в выделенной записью **Goal Position**

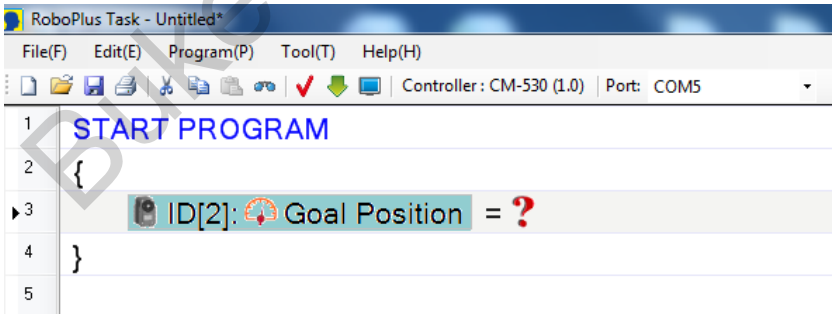


Рисунок 3.23 Окно с указанием в левой части оператора типа команды **Goal Position**

Теперь необходимо выбрать до какой позиции будет вращаться мотор. Нажимаем на правый знак вопроса в строке №3: в появившемся окне **Set device or Number** в левой части окна выбираем пункт **Constant Value** и далее пункт **Number**. В правой части устанавливаем значение позиции, до которой должен двигаться мотор, в нашем случае это **400** (рисунок 3.24).

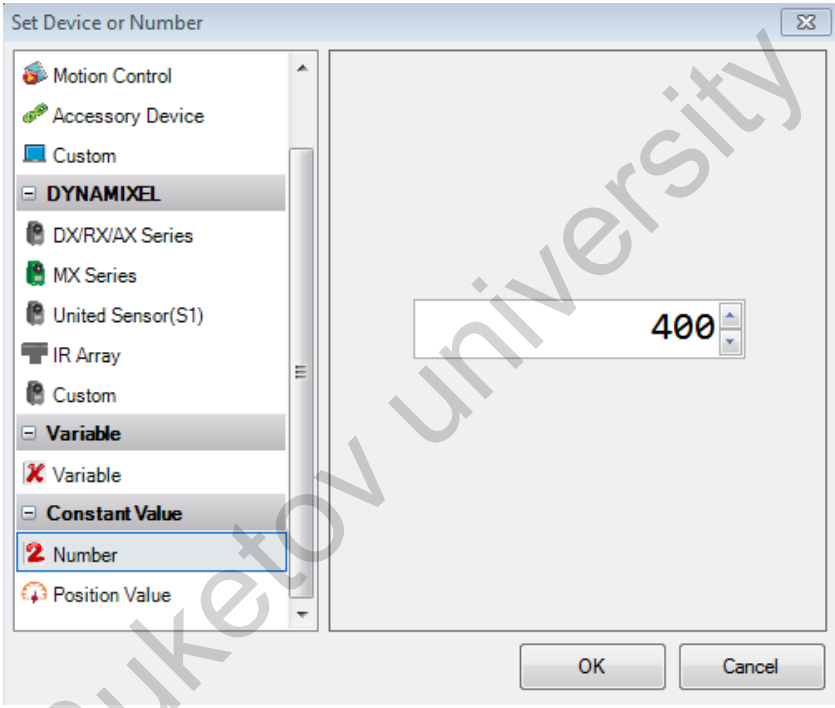


Рисунок 3.24 Окно для установки позиции, к которой будет происходить движение мотора

Далее нажимаем кнопку ОК и на экране появляется окно следующего вида (рисунок 3.25.)

На рисунке 3.25 в данном окне указано, что мотору **№2** надо перейти в позицию 400.

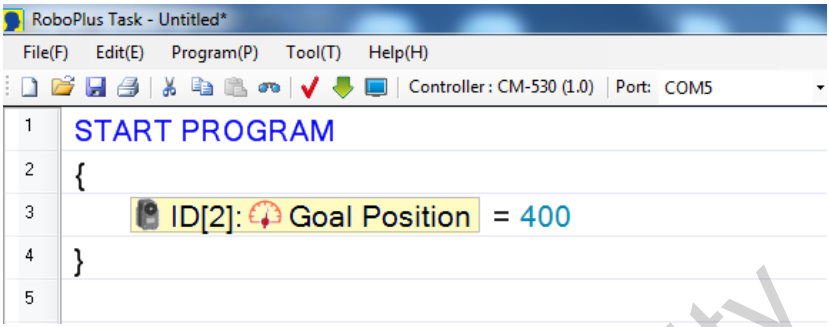


Рисунок 3.25 Оператор **Goal Position**, приказывающий мотору №2 двигаться к позиции 400

Для записи программы в контроллер CM-530 необходимо перейти в окне **RoboPlus Task – Untitled*** в пункт **Program (P).** В раскрывшемся меню выбрать позицию записи программы в контроллер **Download Program (D)** (рисунок 3.26).

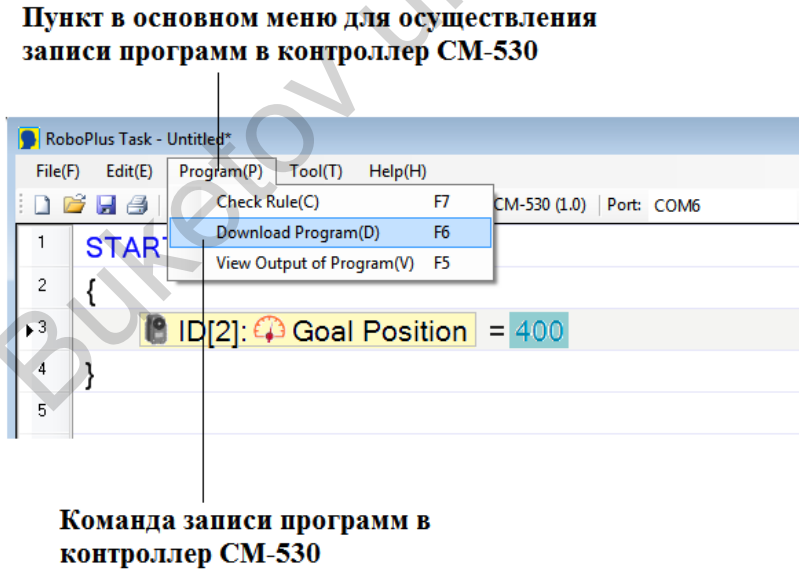


Рисунок 3.26 Алгоритм записи программ на контроллер

Для выполнения этой команды контроллер CM-530 должен быть подключен к компьютеру и должен быть подключен к питанию. Контроллер **CM-530** должен находиться в режиме **PROGRAM** (мигает синяя лампочка на контроллере в столбике PC LINK над надписью **PROGRAM**).

После нажатия команды **Download Program (D)** программа записывается в процессор **CM-530** и на экране компьютера появляется окно предупреждения, в котором надо нажать кнопку OK (рисунок 3.27).

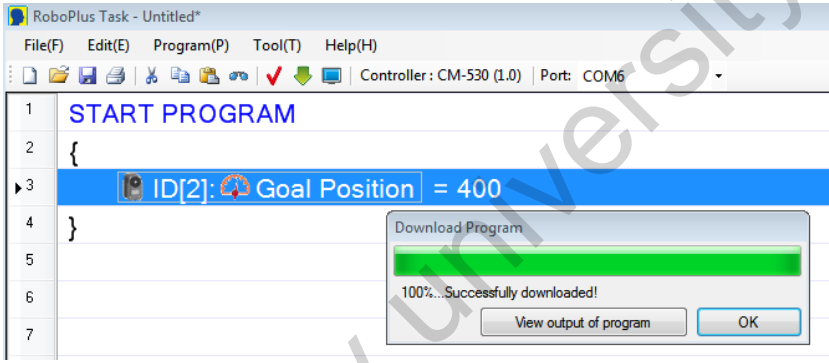


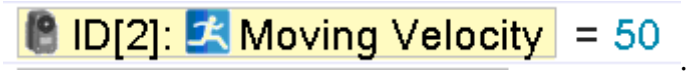
Рисунок 3.27 Результат выбора Download Program (D)

Для выполнения программы мотором №2 необходимо перевести контроллер CM-530 в режим **PLAY** и нажать кнопку **START**.

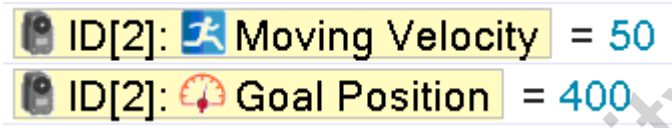
После этого мотор ID:2 включается – вращается и переходит в позицию 400.

3.2.2 Задание моторам нужной скорости вращения

Использование моторов серии **DYNAMIXEL AX-12A** позволяет вращать их с разной скоростью. Для задания скорости вращения мотора надо использовать следующую команду:



Данная команда приказывает мотору №2 вращаться со скоростью 5**0** (скорость вращения мотора можно регулировать).



Данный блок команд приказывает мотору **№2**, вращаясь со скоростью **50**, двигаться к позиции **400** (всего позиций, в которых может находиться мотор, **1023**). При этом неважно в какой позиции при включении данного блока команд находился мотор **№2** (это может быть позиция **0**, а может быть позиция **1000**). В какой бы позиции при включении ни находился мотор, после получения данных команд мотор **№2** начнет вращение к указанной позиции **400** со скоростью 5**0**. Процессом движения при этом будет полностью управлять контроллер, встроенный в сервопривод двигателя **AX-12A**.

Для использования команды **Moving Velocity**, задающей скорость вращения мотора, необходимо: вставить новую строку **№4** в программу показанную, на рисунке 2.20: нажать на правую кнопку мыши и в появившемся меню нажать позицию **Insert Line**.

Далее в пустой строке **№4** два раза нажимаем левую кнопку мыши и выбираем пункт **?=? (LOAD)** (рисунок 3.28). Появляется поле **RoboPlus Task – Untitled*** (рисунок 3.29), содержащее программу, где после оператора, задающего скорость вращения **400**, в строке №4 находится команда, содержащая два вопросительных знака, разделенных знаком равно.

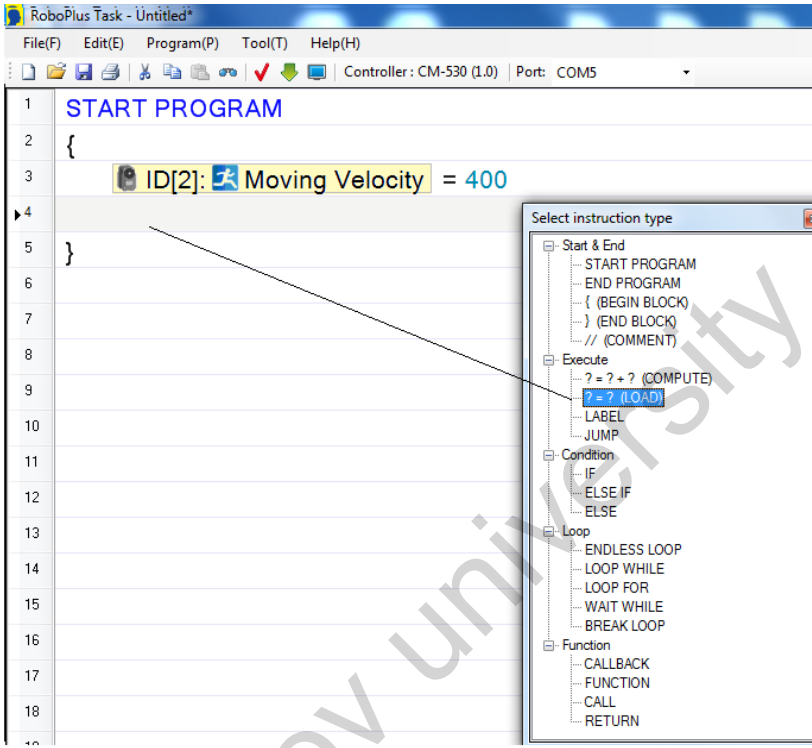


Рисунок 3.28 Процесс вставки команды,
задающей скорость вращения мотора

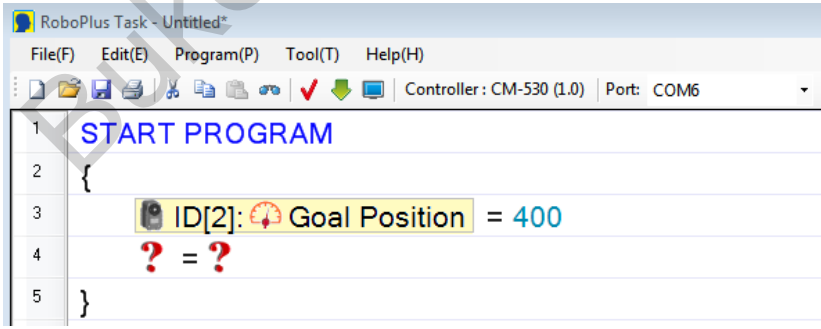


Рисунок 3.29 Вставка **? = ?** на позицию оператора,
задающего скорость вращения мотора

На рисунке 3.29 показано окно **RoboPlus Task – Untitled***, содержащее программу, в которой первая команда в строке **№3** приказывает мотору **№2** вращаться к позиции **400;** в строке **№4** вместо двух знаков вопроса теперь надо ввести команду, задающую скорость вращения мотора **№2.**

В строке **№4** щелкаем два раза на левый знак вопроса и в появившемся окне **Set device or Number** выбираем пункт **DYNAMIXEL** – далее переходим в подменю **DX/RX/AX Series**: далее в верхней правой части окна появляется меню, в котором в пункте **ID:** ставим цифру **2**. Выбираем в правой части окна строку **Moving Velocity** – данная команда позволяет при включении мотора **ID:2** установить необходимую скорость вращения мотора (рисунок 3.30).

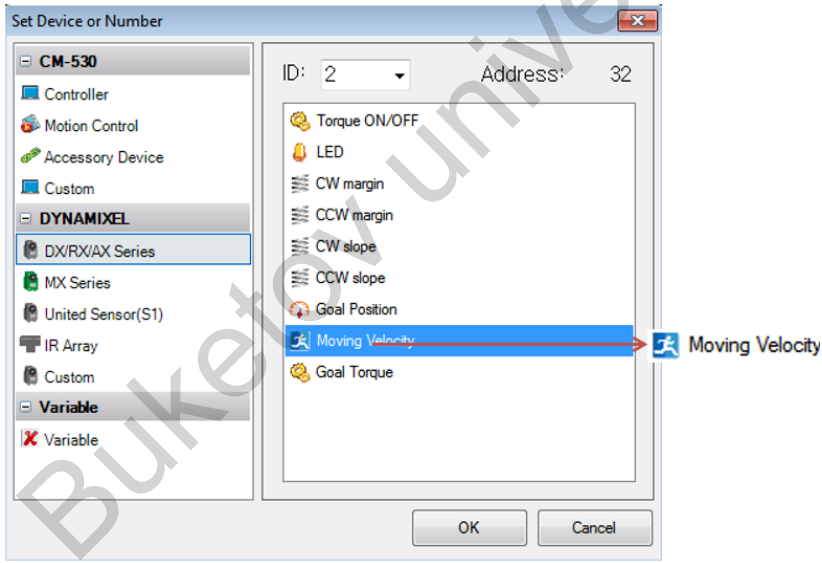


Рисунок 3.30 Выбор команды **Moving Velocity**, задающей нужную скорость вращения

Далее нажимаем **ОК.** В результате на экране появляется окно **RoboPlus Task** с программой, содержащей команду вращения мотора **№2** к позиции **400** (в строке **№3**) и команду в

строке **№4**, приказывающей вращаться мотору **№2,** но скорость мотора еще не указана (рисунок 3.31). Нужная скорость вращения мотора должна быть указана в строке **№4** в правой части команды вместо знака вопроса.

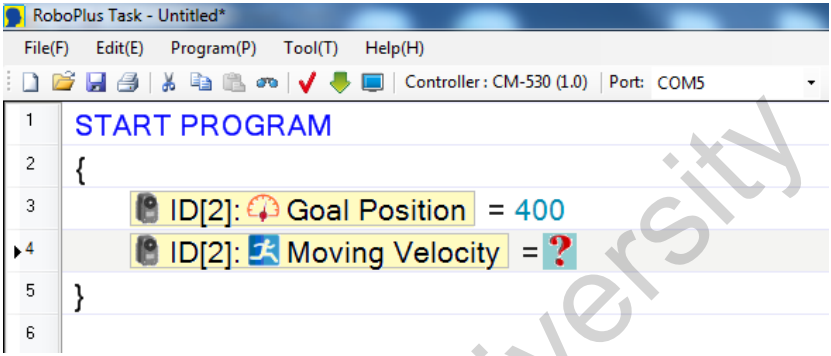


Рисунок 3.31 Показано, что скорость вращения мотора №2 еще не определена

Для задания скорости вращения мотора **№2** два раза щелкаем на правый знак вопроса в строке **№4** - появляется окно **Set device or Number.** Для установки нужной скорости вращения мотора в левой части окна **Set device or Number** заходим в пункт **Constant Value** и заходим в подменю **Number.** (рисунок 3.32). Устанавливаем в правой части окна значение скорости вращения мотора **50**, нажимаем **ОК.**

И на экране появляется окно **RoboPlus Task – Untitled***, содержащее конечный блок команд, приказывающий вращаться мотору **№2** к позиции **400** со скоростью **50** – направление вращения при этом выбирает встроенная автоматика мотора (рисунок 3.33).

Достигнув позиции **400**, мотор **№2** прекратит вращение. Встроенный в мотор редуктор заблокирует ось двигателя в положении позиции **400**, но при этом контроллер двигателя **№2** остается включенным. При выключении контроллера двигателя блокировка двигателя снимается.

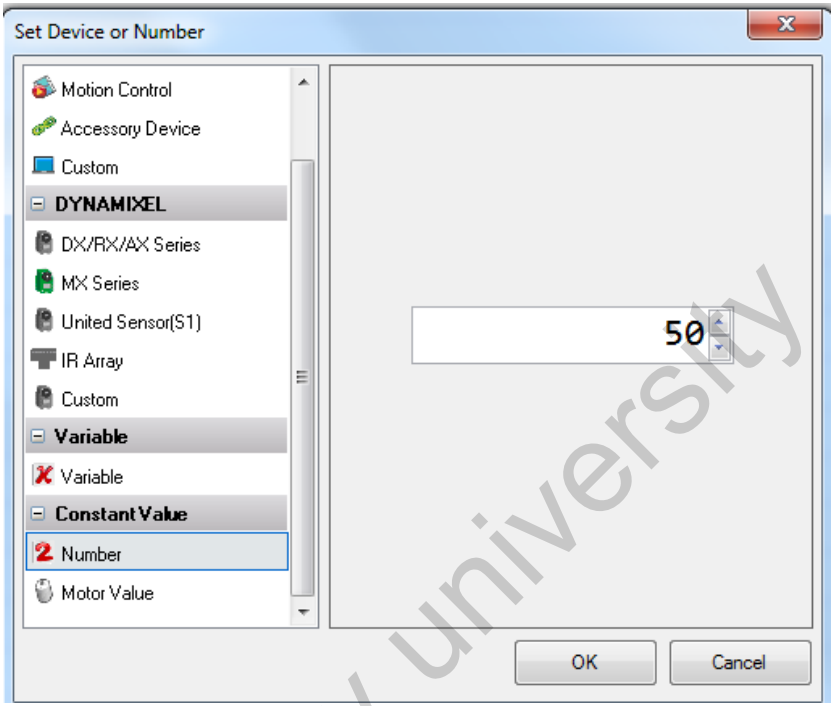


Рисунок 3.32 Окно **Set device or Number** для задания скорости вращения мотора

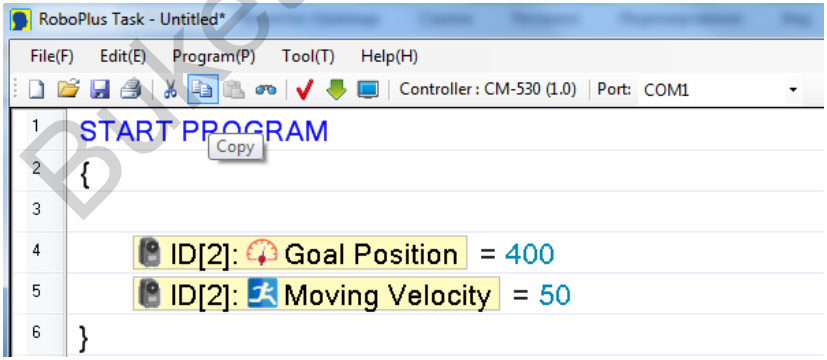


Рисунок 3.33 Показано, что определена позиция, к которой вращается мотор №2, и скорость вращения мотора №2

3.2.3 Задание моторам нужной скорости вращения в заданном направлении

Моторы могут вращаться по часовой стрелке (**CW**) и против часовой стрелки (**CCW**).

Для задания направления вращения моторов необходимо в окне **Set device or Number** зайти в пункт **Constant Value** и далее подпункт **Motor Value**. В правой части окна указываем в строке **Direction** направление вращения по часовой (**CW**) или против часовой (**CCW**) стрелки. В строке **Power** указываем нужную скорость вращения, в нашем случае это **50** и нажимаем ОК **(**рисунок 3.34).

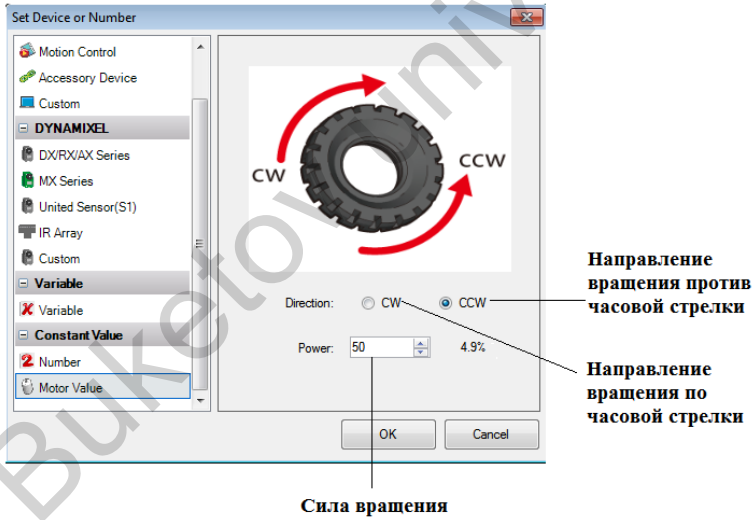


Рисунок 3.34 Окно для указания направления вращения моторов

В результате на экране появится следующая программа, которая будет приказывать мотору **2** двигаться до позиции **0**, против часовой стрелки со скоростью **50** (рисунок 3.35).

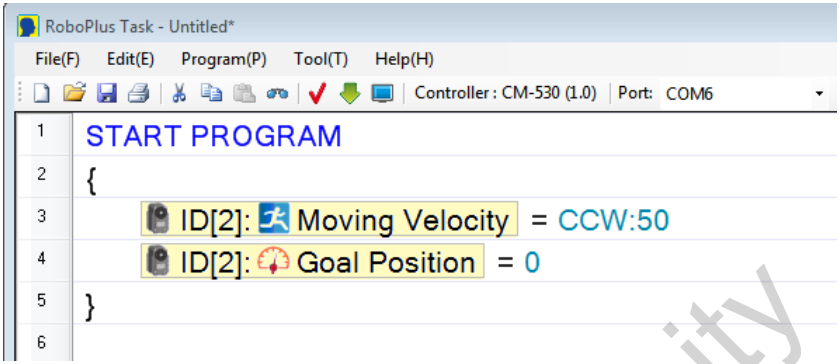
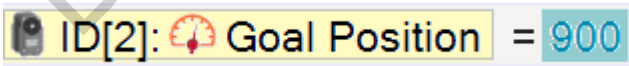


Рисунок 3.35 Программа вращения оси двигателя против часовой стрелки

Для выполнения этой команды контроллер CM-530 должен быть подключен к компьютеру и должен быть подключен к питанию. Контроллер CM-530 должен находиться в режиме **PROGRAM** (мигает синяя лампочка на контроллере в столбике PC LINK). После нажатия команды **Download Program (D)** программа записывается в процессор CM-530 и на экране компьютера появляется окно предупреждения, в котором надо нажать кнопку OK. Для выполнения программы мотором №2 необходимо перевести контроллер CM-530 в режим **PLAY** и нажать кнопку **START**.

Для того чтобы заставить двигаться мотор далее против часовой стрелки, например до **900** позиции, дописываем два оператора, и получаем программу, указанную на рисунке 3.36.

При выполнении программы, записанной на рисунке 3.36, сначала мотор №2 дойдет до позиции **400** со скоростью 50 против часовой стрелки, а затем мотор №2 начнет вращение



от позиции **400** до позиции **900** уже со скоростью **150**

в том же направлении (против часовой стрелки).

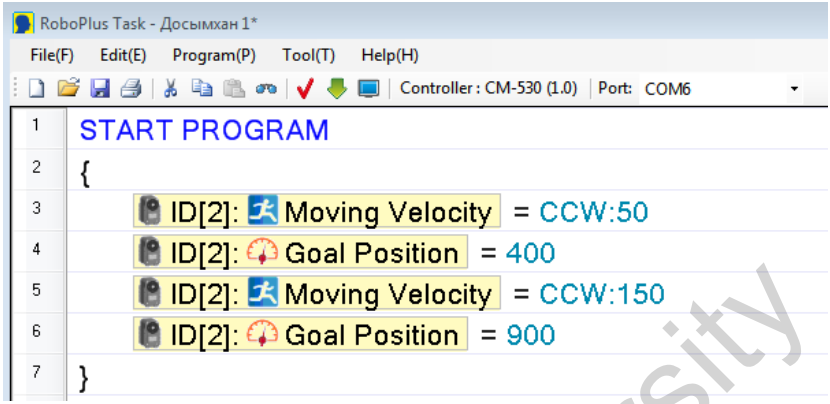
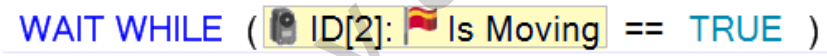


Рисунок 3.36 Два этапа движения оси двигателя до позиции 900

3.2.4 Команда WAIT WHILE (остановка текущего действия)



Данная команда называется **Цикл с предусловием.** Предусловие записано внутри скобок. Данный цикл будет выполняться до тех пор, пока не сработает условие, записанное внутри скобки для данного номера мотора.

Is MOVING == TRUE означает, что мотор двигается, пока прописанное внутри цикла условие не выполнится – как только условие, прописанное внутри цикла выполняется, то есть становится верным, **выполнение цикла прекращается.** Конечная позиция движения мотора прописана где-то ранее по тексту программы, например, на рисунке 3.36 указано, что мотор номер **№2** должен двигаться к позиции **400**. Как только мотор достиг позиции **400,** срабатывает условие, написанное в операторе **WAIT WHILE** для мотора **№2**, и данный мотор перестает двигаться, потому что условие, записанное в операторе **WAIT WHILE**, становиться верным.

1. Команда **WAIT WHILE** в тексте программ устанавливается после команды, приказывающий моторам произвести какое-то действие.

2. Команда **WAIT WHILE** фактически означает для данного мотора прекращение действия, которое было прописано в программе ранее.

Для вызова команды **WAIT WHILE** необходимо щелкнуть два раза левой кнопкой мыши на пустой строке и в появившемся меню (рисунок 3.37) выбрать пункт **WAIT WHILE**.

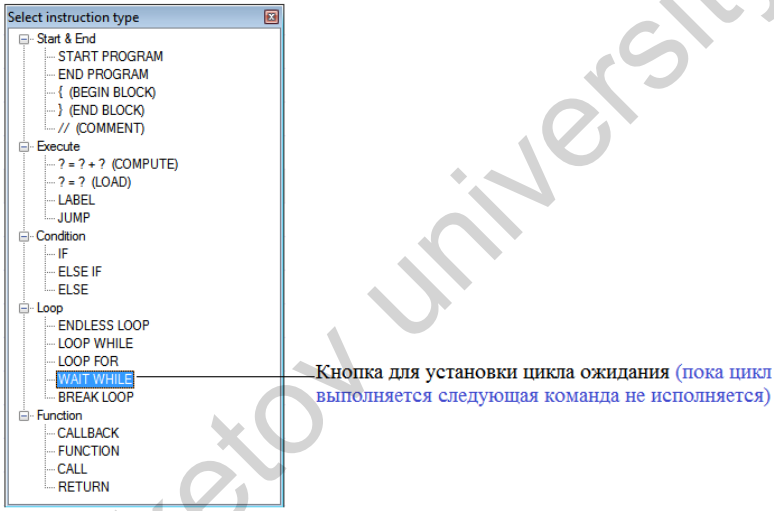


Рисунок 3.37 Установка оператора WAIT WHILE

В результате в окне RoboPlus Task в строке №5 появится оператор **WAIT WHILE** (рисунок 3.38).

В операторе **WAIT WHILE** в скобках щелкаем по правому знаку вопроса дважды. Появляется окно **Set device or Number** - пункт **DYNAMIXEL** – далее переходим в подменю **Constant Value** – выбираем запись **TRUE/FALSE** - появляется окно **Set device or Number**, в правой части которого выбираем **TRUE** (рисунок 3.39).

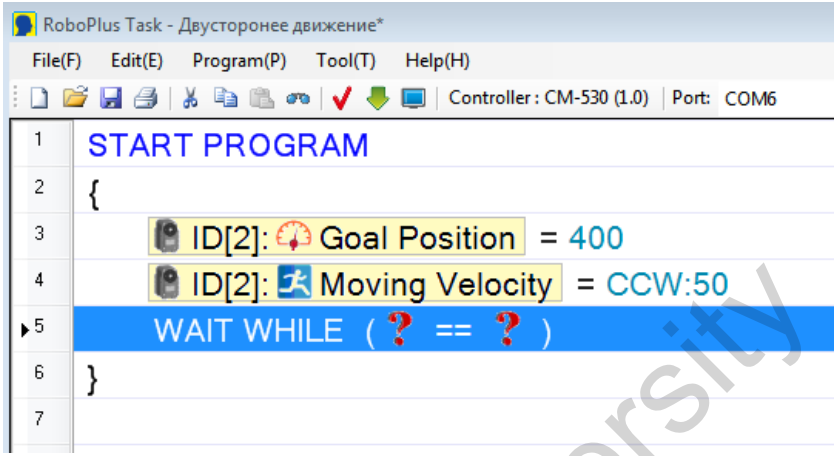


Рисунок 3.38 Результат установки оператора WAIT WHILE

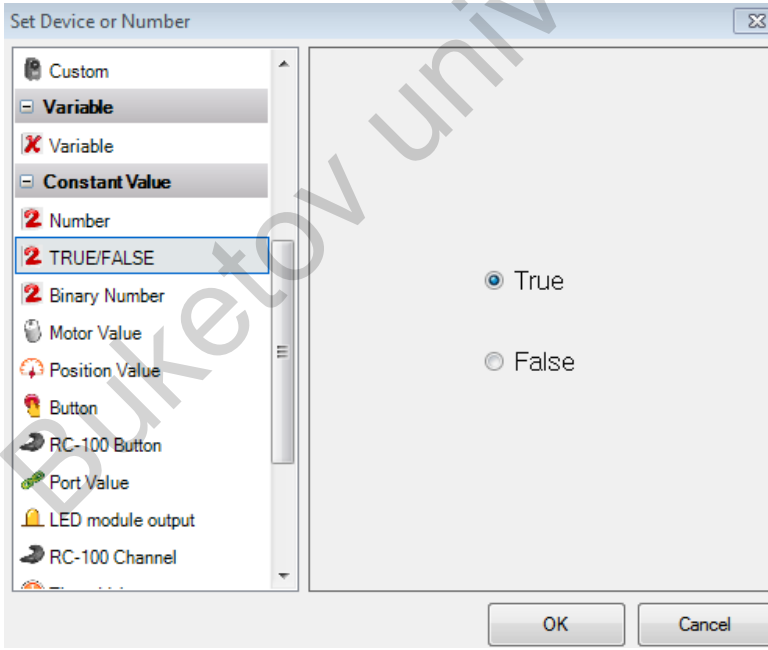


Рисунок 3.39 Выбор логического значения TRUE
для оператора WAIT WHILE

В результате (рисунок 3.40) на экране в окне R**oboPlus Task** в строке **№5** появится запись:

WAIT WHILE (? == TRUE),

которая показывает, что выполнение действий, указанных для данного мотора (номер мотора пока не указан) операторами, сработавшими ранее оператора **WAIT WHILE**, прекратится (в данной программе эти операторы находятся в строках №3 и №4), когда условие, показанное в скобках оператора **WAIT WHILE**, станет верным (TRUE показывает, что условие стало верным).

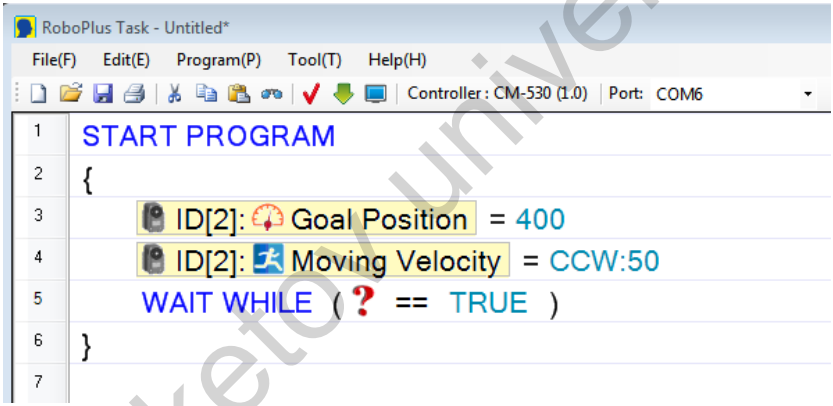


Рисунок 3.40 Показано, что не указан номер мотора в строке №5

Теперь необходимо в левой части условия оператора **WAIT WHILE** вместо знака вопроса указать номер мотора, работу которого контролирует условие, прописанное в скобках оператора **WAIT WHILE**, и указать само это условие ? == TRUE.

Щелкаем по левому знаку вопроса дважды, появляется окно **Set Device or Number**. Выбираем строку **DYNAMIXEL**- далее подпункт **DX/RX/AX Series** – потом в верхней части правого

поля выбираем номер мотора **ID:2** и в правой нижней части поля выбираем команду Is Moving (рисунок 3.41). **Нажимаем ОК**.

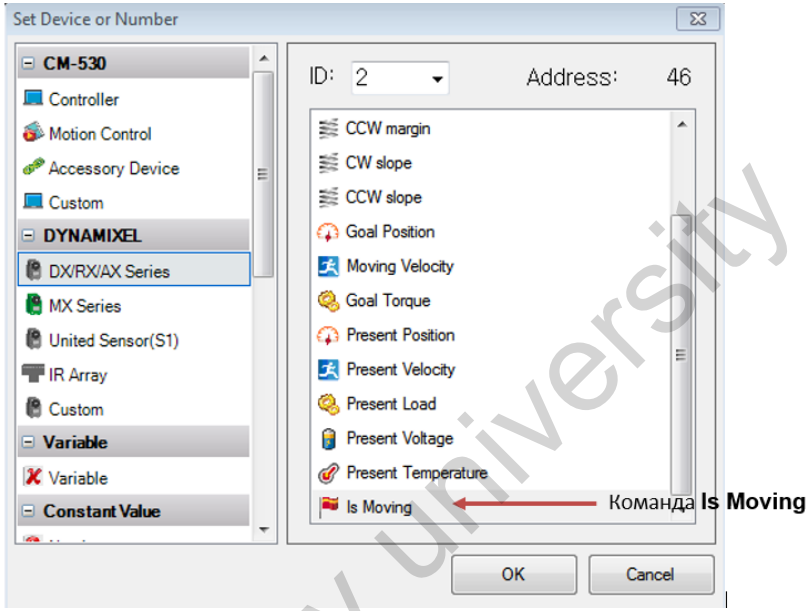
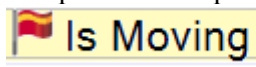


Рисунок 3.41 Выбор команды **Is Moving**

В результате в окне **RoboPlus Task** в строке **№5** появляется запись (рисунок 3.42):

в которой в логическом условии внутри скобок указано, что условие контролирует работу мотора **№2** 

и указано, что работа мотора **№2** прекращается, когда логическое условие 

, записанное в скобках оператора **WAIT WHILE,** станет верным.

На рисунке 3.42 представлен конечный вид программы с оператором WAIT WHILE, отменяющим действие других

операторов для данного мотора при срабатывании логического условия.

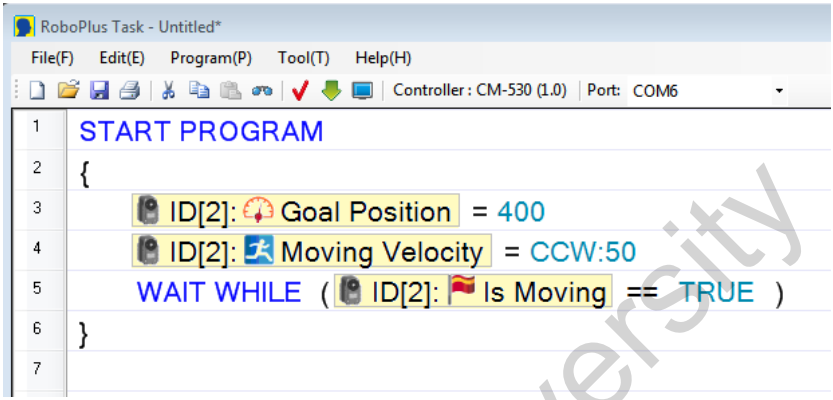


Рисунок 3.42 Конечный вид программы

На рисунке 3.42 изложена алгоритм программы, которая приказывает мотору **№2** двигаться к позиции **400** против часовой стрелки со скоростью **50.** Когда мотор достигнет позиции **400**, оператор **WAIT WHILE**, расположенный в строке №5, прекратит действие команд в строках **№3** и **№4** и мотор номер **№2** прекратит движение.

3.2.5 Пример использования WAIT WHILE

Рассмотрим пример использования данного оператора. На рисунке 3.43 команды в строках №3 и №4 приказывают двигаться мотору №2 с текущей позиции до позиции **400** со скоростью **50** против часовой стрелки. Когда мотор достигнет позиции **400**, команда **WAIT WHILE** в строке №5 укажет, что действие команд в строках №3 и №4 закончилось.

Далее выполняются команды в строках **№6** и **№7**, которые приказывают мотору **№2** двигаться до позиции **900** со скоростью **150** по часовой стрелки.

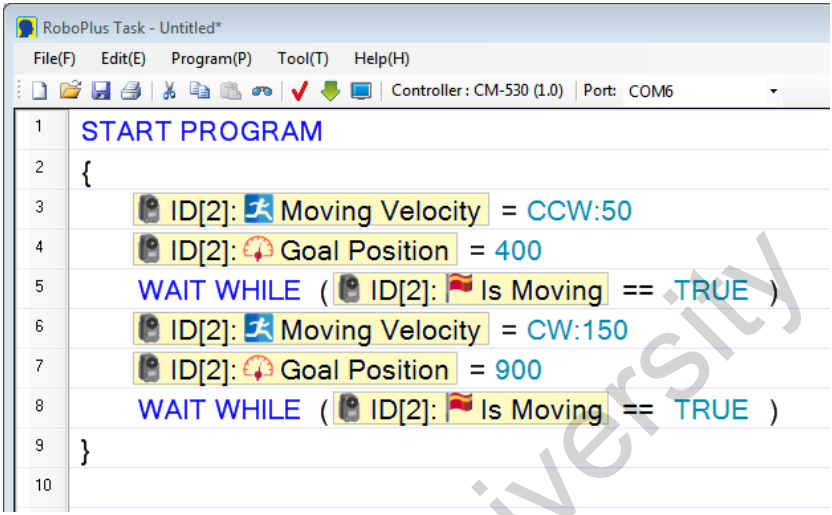


Рисунок 3.43 Вращение мотора против и по часовой стрелке

Когда мотор **№2** достигнет позиции **900**, сработает расположенный в строке **№8** оператор **WAIT WHILE** и выполнение команд, расположенных в строках **№6** и **№7** прекратится.

Применение команды **WAIT WHILE** очищает собственную память моторов. Команды записываются как в память процессора CM 530 так и в мини-память моторов, к которым обращается процессор CM 530. В результате применение команды **WAIT WHILE** при повторном выполнении данной программы движения мотора номер №2 начнется с 900 позиции и пойдет на 400 позицию.

3.2.6 Применение Инфракрасного датчика

Инфракрасные датчики подключаются к **GPIO** портам (смотри рисунок 3.44), расстояние обнаружения объекта не превышает **15 см.**

Рисунок 3.44 Инфракрасный датчик

Рассмотрим **управление позицией мотора уровнем сигнала от инфракрасного датчика** (рисунок 3.45).

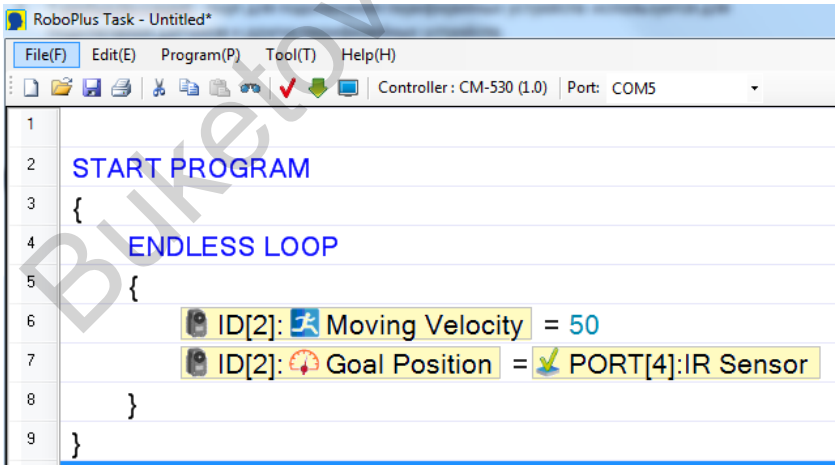
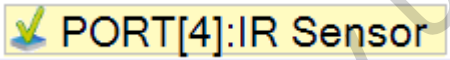


Рисунок 3.45 Управление позиции мотора сигналом от инфракрасного датчика

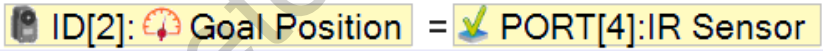
В программе на рисунке 3.45 мотор со скоростью 5**0** производит вращение к позиции (номера позиций могут меняться от **0** до **1023**), задаваемой инфракрасным датчиком, подключенным к **GPIO порту №4** контроллера **CM-530:** чем выше уровень теплового излучения регистрируемого инфракрасным датчиком, тем больше величина аналогового сигнала, поступаемого с инфракрасного датчика на **GPIO порт №4** контроллера (величина сигнала также может меняться от **0** до **1023**), тем позиция, на которую переместился мотор **№2**, имеет больший номер.

Для работы с инфракрасным датчиком необходимо в окне Set device or Number (рисунок 3.46) зайти в пункт Accessory Device (дополнительные устройства) и в правой части окна выбрать запись IR Sensor (инфракрасный датчик).

Также в верхней части окна Set device or Number надо указать номер GPIO порта контроллера CM-530, к которому подключен инфракрасный датчик (в данном случае это порт №4).



данная запись показывает, что к GPIO порту №4 контроллера CM-530 подключен инфракрасный датчик.



Если, например величина сигнала подаваемого на GPIO порт № 4 равна 500, то мотор перемещается к позиции 500, если величина сигнала уменьшается до 0, то соответственно мотор перемещается к позиции 0.

Так как кодовый блок программы находится внутри бесконечного цикла ENDLESS LOOP, то происходит постоянный опрос инфракрасного датчика - сигнал с датчика постоянно поступает на **GPIO порт №4** и в зависимости от поступающего сигнала происходит вращение мотора **№2** к заданным инфракрасным датчиком позициям.

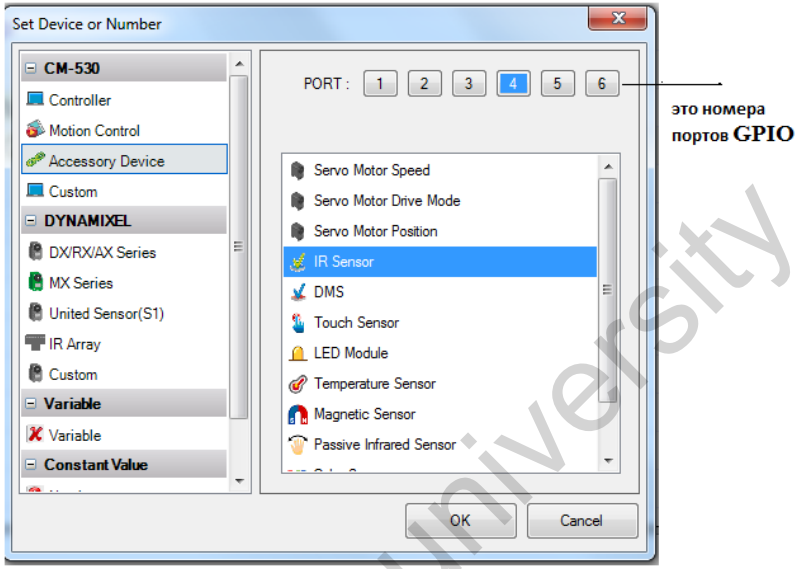


Рисунок 3.46 Окно Set device or Number с выбранной слева записью Accessory Device, выбранные справа ИК датчик и справа сверху номер GPIO порта

Включение мотора при срабатывании инфракрасного датчика (использование логического оператора IF)

Часто возникает задача включить мотор (или произвести какое-то аналогичное действие), если на датчике сигнал превысил определенное значение: превышение сигнала больше некоторого означает, что что-то изменилось в среде, окружающей датчик. Например, возросла температура среды, что привело к увеличению теплового потока на датчик инфракрасного излучения и в результате уровень сигнала с датчика ИК возрос. Рассмотрим программу, показанную на рисунке 3.47.

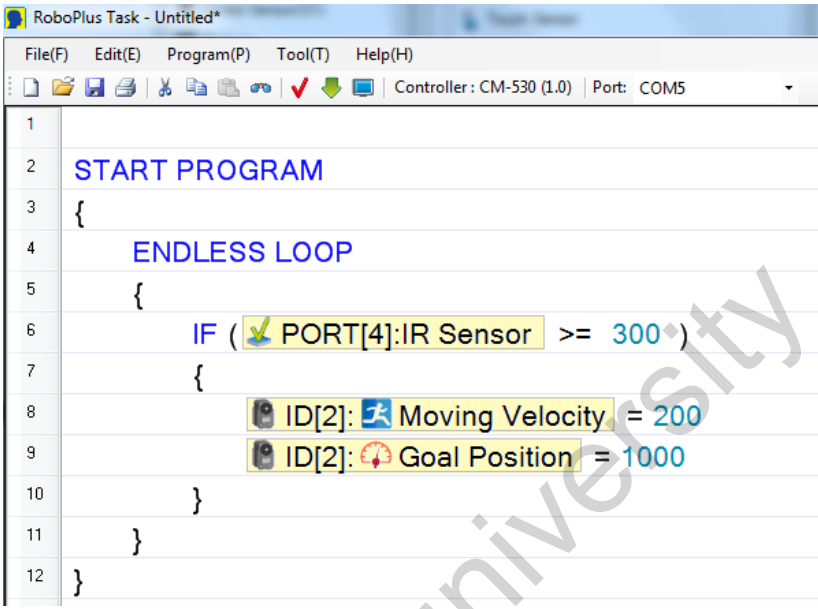


Рисунок 3.47 Перемещение мотора на заданную позицию при превышении определенного уровня инфракрасного излучения, поступаемого на ИК-датчик

В данной программе внутрь бесконечного цикла ENDLESS LOOP вставлен логический оператор IF (рисунок 3.48).

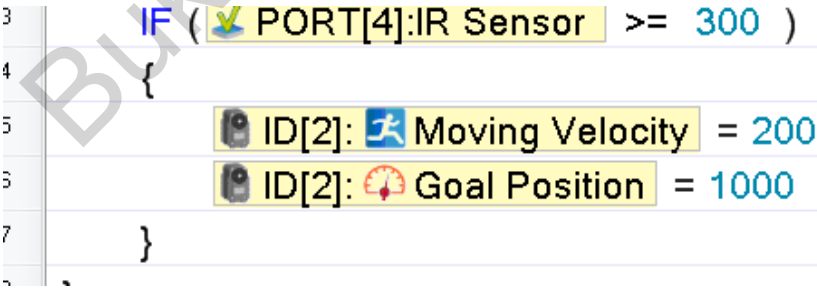
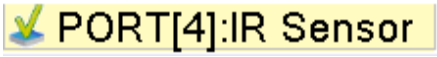


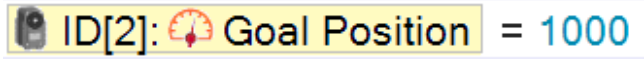
Рисунок 3.48 Показано использование инфракрасного датчика в комплексе с логическим оператором IF

Инфракрасный датчик подключен к порту №4



.

Если сигнал с ИК датчика меньше **300**, мотор **№2** не вращается, находясь на какой-то случайной позиции; если сигнал, регистрируемый ИК датчиком, больше **300** (это максимум, который может обнаружить датчик ИК излучения при температуре объекта 36.6C°), то мотор **№2** начинает вращаться до позиции **1000**

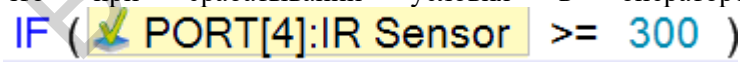


со скоростью **200**. Если после достижения позиции **1000** сигнал с инфракрасного датчика станет меньше **1000**, позиция двигателя №2 не изменится.

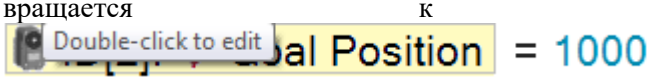
Программу, указанную на рисунке 3.47, можно на практике применить, например, в случае перегрева помещения – при возрастании температуры в помещении возрос уровень сигнала от ИК датчика и вследствие этого включился мотор и открыл люк вентиляции (и далее этот люк вентиляции остается постоянно открытым).

Использование оператора IF…ELSE в комплексе с инфракрасным датчиком

В отличии от оператора **IF** оператор **IF…ELSE** позволяет производить два действия. Оператор **IF** производит **только одно действие**, например перемещение мотора к заданной позиции при срабатывании условия. В предыдущем примере показано, что при срабатывании условия в операторе **IF**



мотор вращается к позиции



и там остается, даже когда датчик прекращает фиксировать сигнал – работает принцип **УШЕЛ И НЕ ВЕРНУЛСЯ**.

Оператор **IF…ELSE позволяет производить второе действие**, если условие, прописанное в операторе **IF**, не выполняется.

Рассмотрим случай, когда нахождение мотора в одной из двух возможных позиций контролируется уровнем сигнала инфракрасного датчика (рисунок 3.49).

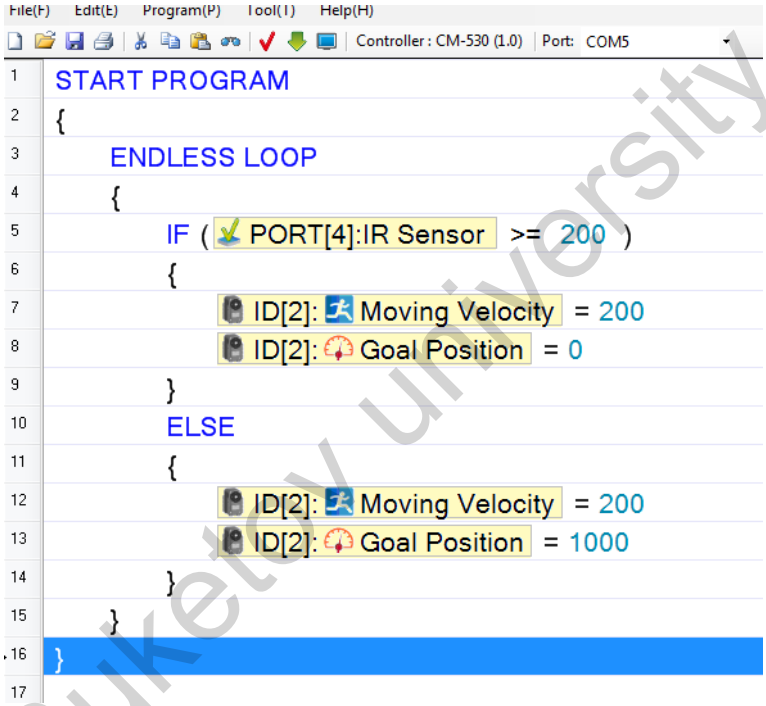
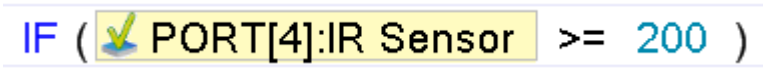


Рисунок 3.49 Нахождение мотора в одной из двух возможных позиций

На рисунке 3.49 показана программа, при работе которой возможно два устойчивых состояния мотора – если инфракрасный датчик фиксирует тепловое излучение



выше некоторого уровня **200**, то мотор **№2** переходит в позицию ноль 

.

Наличие теплового излучения задается условием, что сигнал, поступающий на **GPIO порт 4** контроллера от ИК датчика, больше или равен **200.**

Если тепловое излучение перестает регистрироваться ИК датчиком (при этом уровень сигнала, поступающего на **GPIO порт 4** контроллера с ИК датчика, становиться меньше **200**), то мотор **№2** возвращается в основное состояние, соответствующее позиции **1000**:

Вышеизложенная программа с использованием логического оператора **IF…ELSE** на практике применяется в системах автоматического открывания дверей при обнаружении перед дверью человека. Когда человек подходит к автоматической двери, его тепловое излучение фиксируется инфракрасным датчиком, срабатывает условие логического оператора **IF** и мотор начинает движение к заранее заданной позиции, открывая при этом автоматическую дверь. Когда человек уходит из зоны действия ИК датчика уровень теплового излучения становится ниже порогового значения, заданного в условии оператора **IF** (при этом условие оператора **IF** перестает выполняться) и срабатывает логический оператор **ELSE,** мотор начинает возвращаться к основной позиции, закрывая при этом дверь.

3.2.7 Звуковое сопровождение

Контроллер CM 530 **способен издавать звуки заданной тональности в течение заданного времени** (рисунок 3.50).

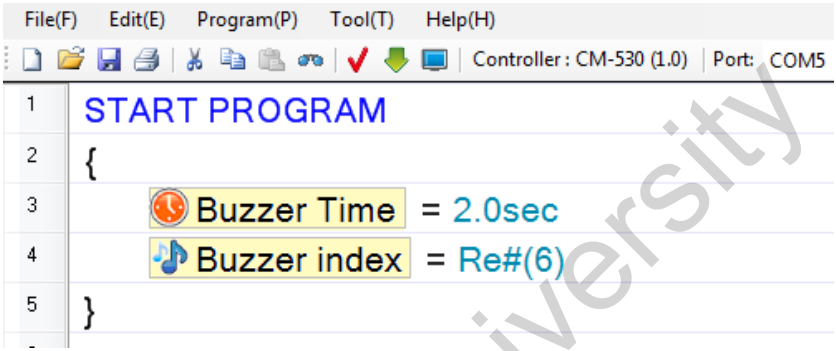
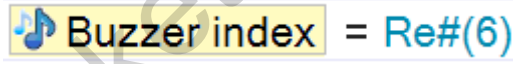


Рисунок 3.50 Программа генерации музыкальной ноты

На рисунке 3.50 представлена программа, при исполнении которой контроллер CM-530 в течение **2-х** секунд

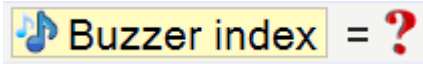
издает ноту **РЕ**



шестой октавы. Контроллер **CM-530** позволяет запрограммировать извлечение любой ноты в течение любого заданного времени.

Для задания необходимого звука необходимо в окне **Set device or Number (**рисунок 3.51) зайти в пункт **Variable** и в правой части окна выбрать запись **Buzzer index** и нажать **ОК**. В поле RoboPlus Task вместо выражения

появится строка



.

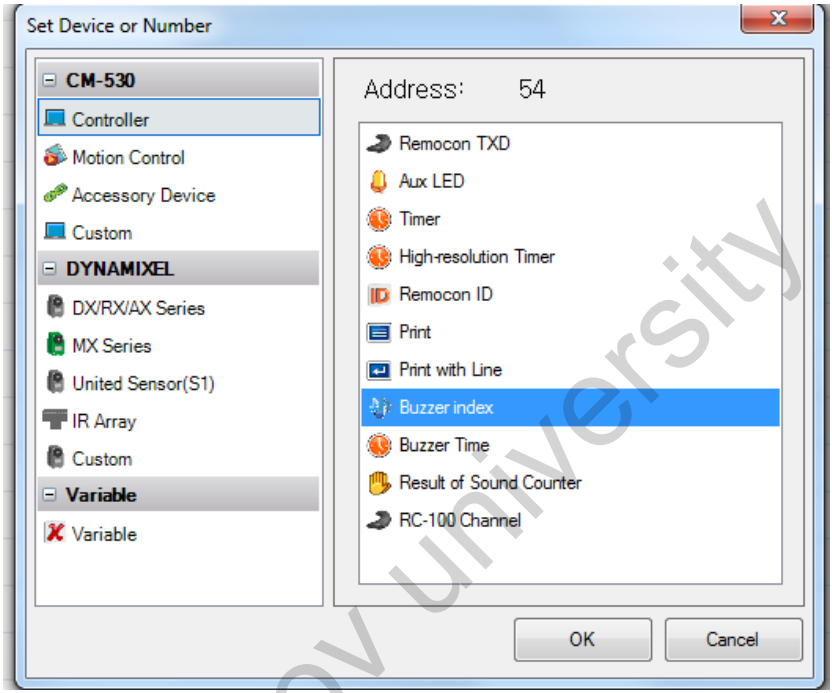
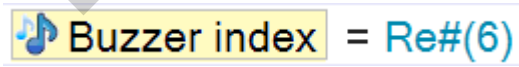


Рисунок 3.51 Выбор пункта музыкального сопровождения

Далее щелкаем два раза по правому знаку вопроса и в левой нижней части окна **Set device or Number** выбираем пункт **Musical Scale** – в правой части окна устанавливаем необходимую ноту (рисунок 3.52). Нажимаем **ОК** и в поле RoboPlus Task появляется команда 

, которая указывает, что звучит нота **Ре шестой октавы.**

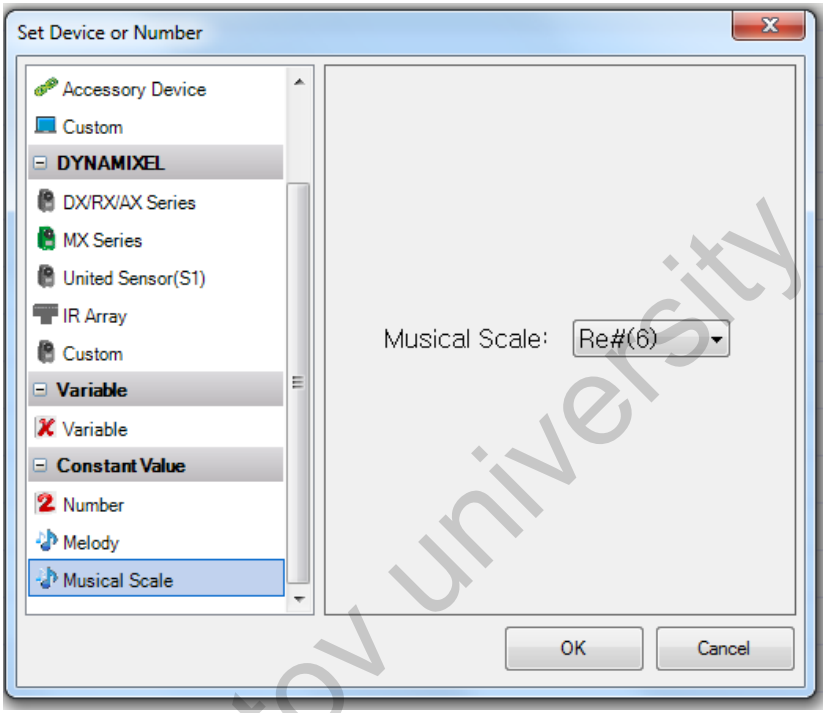


Рисунок 3.52 Выбор ноты (нота РЕ шестой октавы)

Для задания нужного времени звучания ноты необходимо разместить в рабочем поле окна выражение ? = ? и два раза щелкнуть мышью по левому знаку вопроса: в окне **Set device or Number** зайти в пункт **Variable** и правой части окна выбрать запись **Buzzer Timer** (время звучания) (рисунок 3.53).

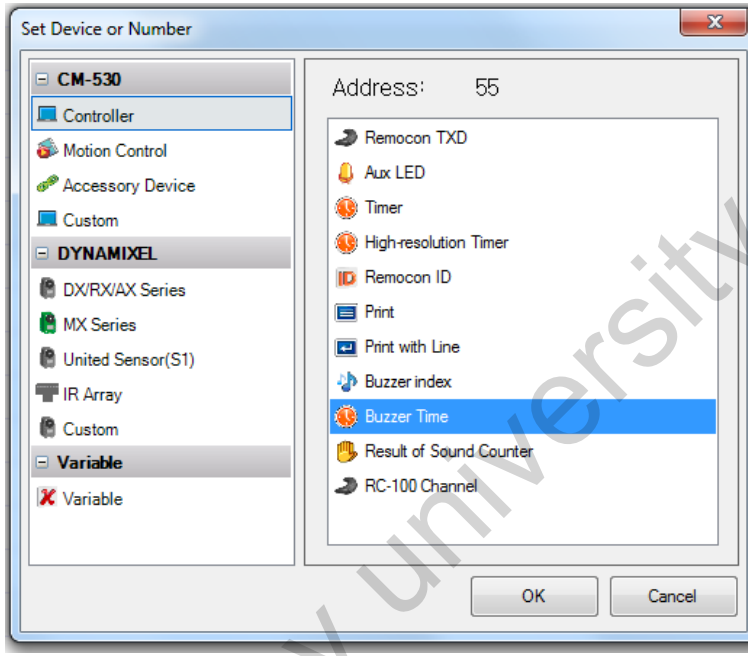
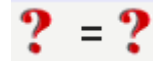
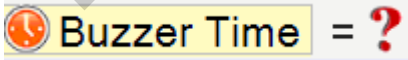


Рисунок 3.53 Выбор пункта Buzzer Timer

В результате в поле **RoboPlus Task** появляется команда 

, которая указывает, что время звучания ноты **2 секунды**.

На экране вместо выражения  появится запись 

в поле **RoboPlus Task**. Щелкаем по правому знаку вопроса и в окне **Set device or Number** в левой нижней части окна выбираем пункт **Buzzer Timer Value** (рисунок 3.54).

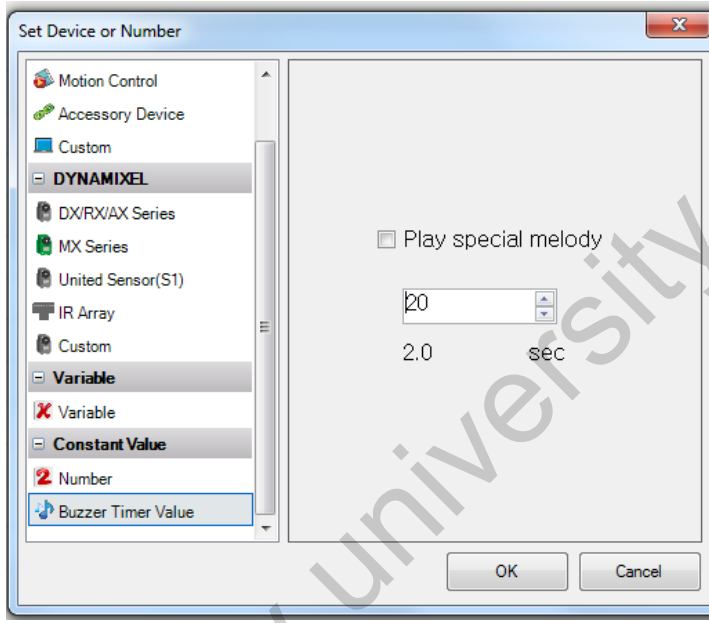


Рисунок 3.54 Установка времени звучания ноты

3.2.8 Программа ручного управления роботом Bioloid

Применяется для определения позиций конечных точек движения.

Включаем программу **RoboPlus** появляется диалоговое окно **RoboPlus** (рисунок 3.55).

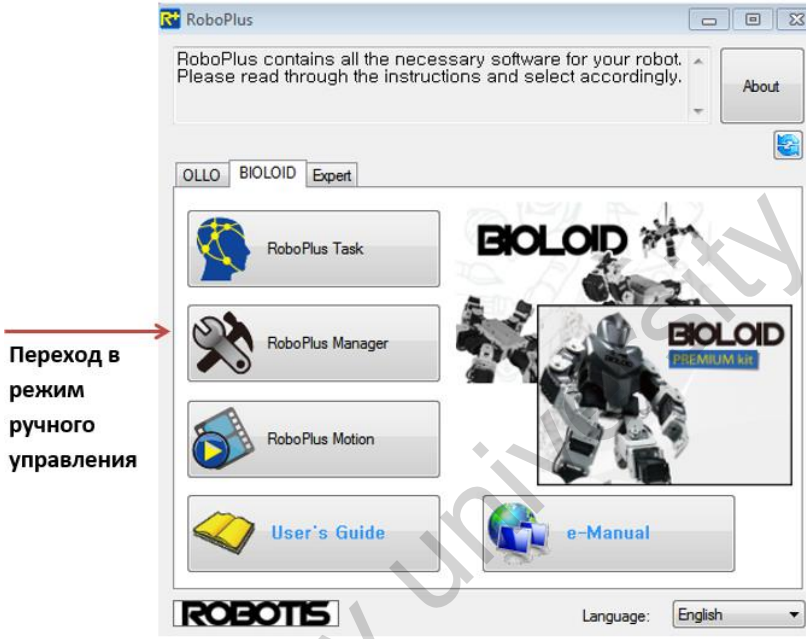


Рисунок 3.55 Выбор программы ручного управления

Для перехода в режим ручного управления нажимаем кнопку **RoboPlus Manager**, находящуюся на левой части окна. Она выглядит следующим образом:

В результате данного действия появляется окно **RoboPlus Manager** (рисунок 3.56).

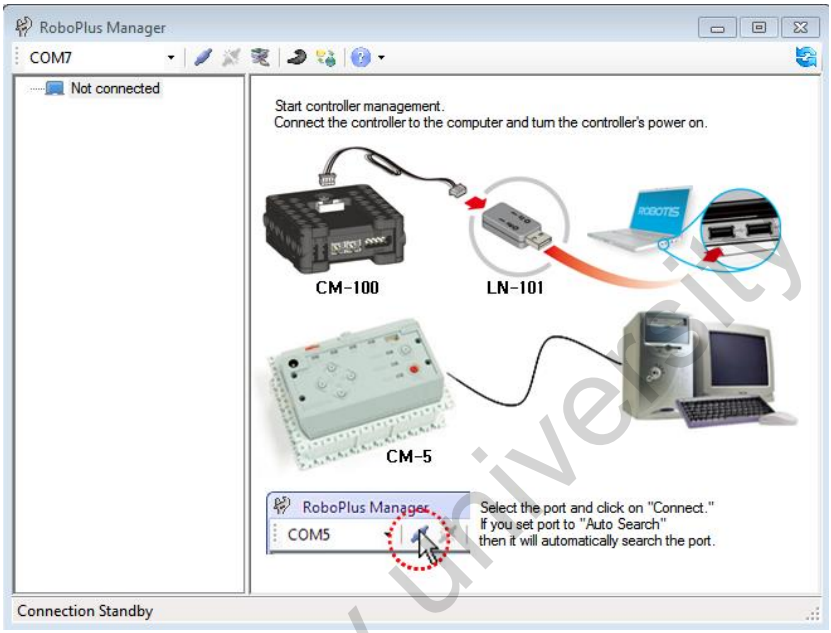
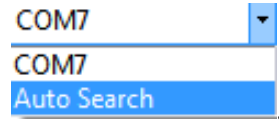


Рисунок 3.56 RoboPlus Manager

Для того, чтобы включить робота, необходимо в верхнем левом углу окна **RoboPlus Manager** выбирать окно:



И далее кликаем по **AutoSearch** (АвтоПоиск). В результате на экране появляется окно (рисунок 3.57).

На данном рисунке 3.57 в левой части окна **RoboPlus Manager** представлен список всех моторов, с которыми контроллер CM-530 имеет логическое соединение.

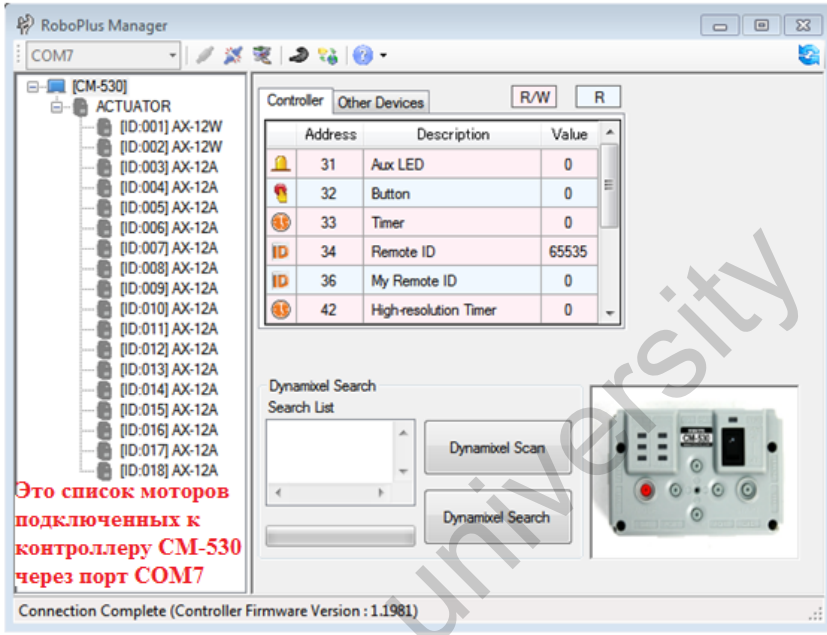


Рисунок 3.57 Список моторов подключенных к контроллеру CM 530

В левой части окна **Robo Plus Manager** указано, что все 18 моторов робота BIOLOID подключены. Номера моторов в списке указаны через идентификаторы типа **ID( )** – например, на рисунке 3.58 в списке из 18 моторов выделен мотор № 7 как (**ID:007).**

Наличие таблицы подключенных моторов позволяет очень эффективно определять неисправные моторы. Если какой-то мотор не работает, он отсутствует в данной таблице.

Рассмотрим подробнее работу с данной таблицей. Выделяем мотор номер 4. В правой части окна **RoboPlus Manager** появляется характеристики данного мотора (рисунок 3.59). Например, в строке №3 показан номер мотора, а строка номер 25 позволяет включить или выключить свечение встроенного в мотор светодиода. В 30 строке данной таблицы

включается возможность установки мотора вручную на заданную позицию (рисунок 3.60).

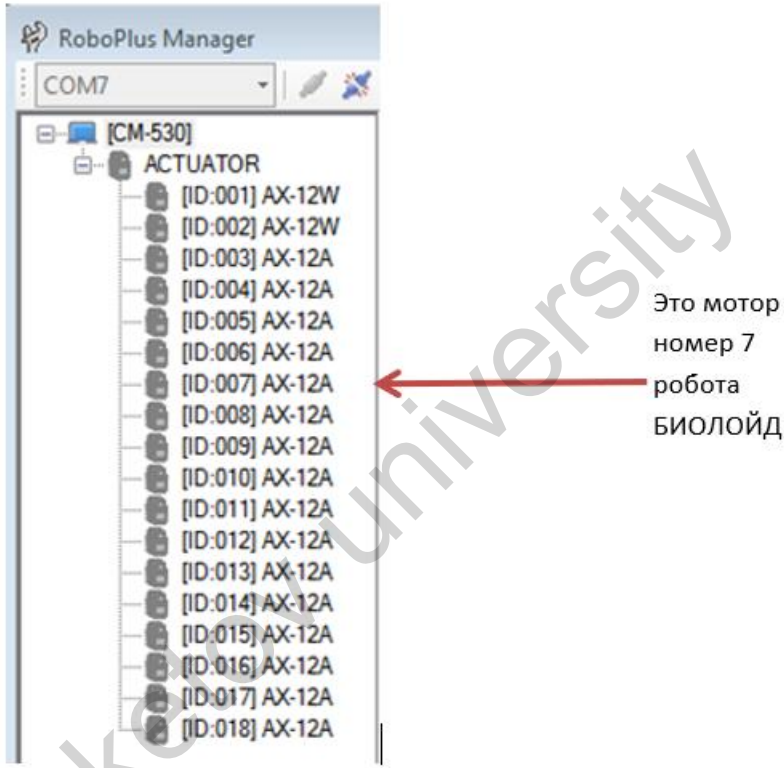


Рисунок 3.58 В списке моторов показан мотор №7 (ID:007)

В правой части окна расположен регулятор положения сервомотора. Вращая на регуляторе кружок, можно привести мотор **№4** в нужную позицию. Например, на рисунке указана позиция **512,** что соответствует левой руке робота **BIOLOID,** опущенную вертикально вниз, позиция **0** означает левая рука поднята максимально верх. Позиция **230** - левая рука поднята на **90** градусов.

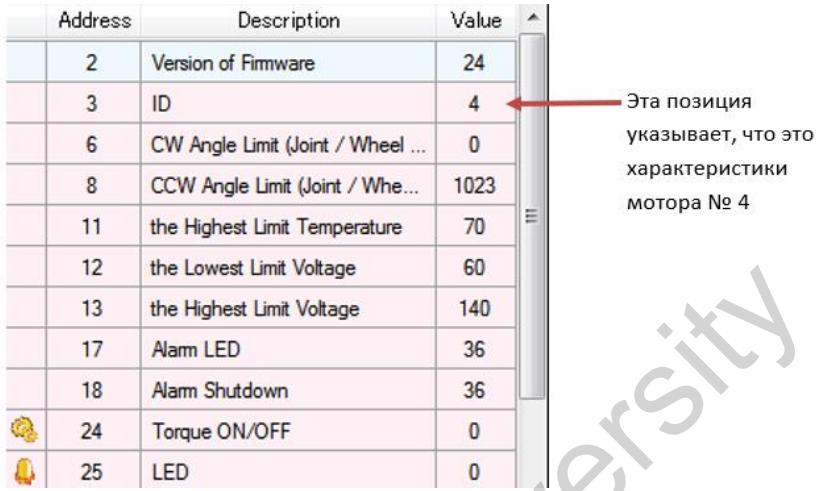


Рисунок 3.59 Указаны характеристики мотора №4

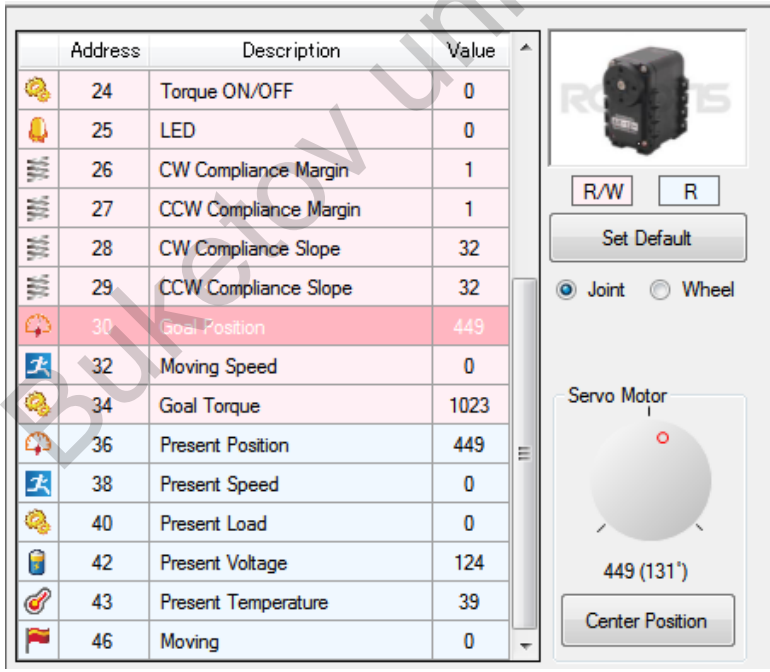


Рисунок 3.60 Ручное установление позиции мотора

3.3 Примеры программ управления человекообразным роботом Bioloid

Как уже говорилось раннее, в состав рассматриваемого человекообразного робота входит 18 моторов. На рисунках 3.61 и 3.62 показаны вид спереди и вид сзади соответственно.

Рисунок 3.61 Робот Bioloid – вид спереди

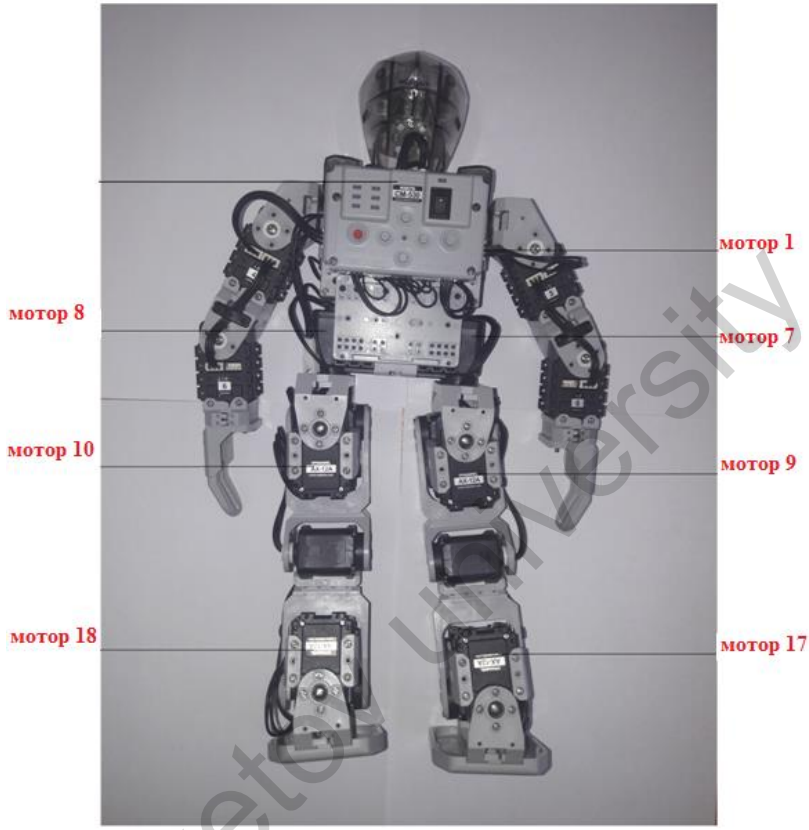


Рисунок 3.62 Робот Bioloid – вид сзади

3.3.1 Программа поднятия и опускания правой руки вверх (используются моторы №3, №1)

На представленном рисунке 3.63 показан программный код, при исполнении которого робот поднимает правую руку вверх до позиции **512,** а потом опускает эту руку вниз до позиции **0.**

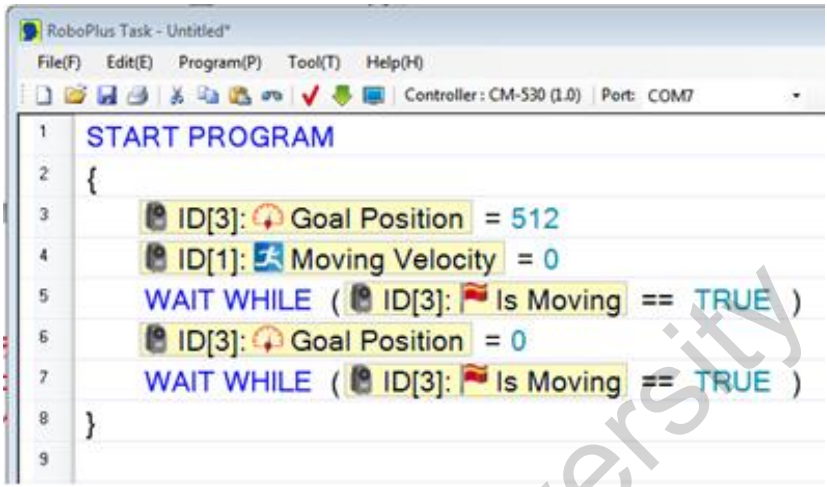
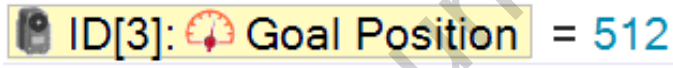
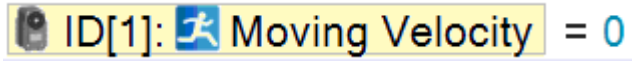


Рисунок 3.63 Робот поднимает и опускает правую руку

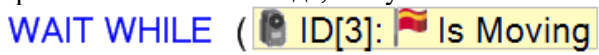
В строке **№3** указана команда 

, которая приказывает мотору **№3** двигаться к позиции **512** (позиции привязаны к действиям конкретного мотора). На рисунках 3.61, 3.62 показано расположение моторов робота Bioloid.

Для фиксации плоскости движения правой руки при движении мотора **№3** к позиции **512** (чтобы рука не проскальзывала в плечевом суставе, который сделан на моторе **№ 1**) необходимо активировать мотор **№1**. При активации мотора **№1** становятся невозможными движения, не прописанные в программе (при активации мотора встроенный в мотор редуктор блокирует любые проскальзывания). Для предотвращения проскальзывании в моторе **№1** в программе в строке **№4** действует команда 

.

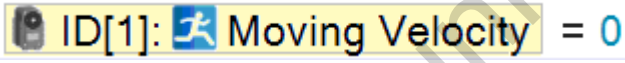
Данная команда устанавливает скорость движения мотора **№1**, равную нулю, что включает редуктор, блокирующий проскальзывание мотора **№1.**

По достижении позиции **512** мотора **№3** необходимо указать контроллеру робота Bioloid, что действие команды, расположенной в строке **№3,** закончилось – для этого применяется команда, указанная в строке **№5** 

(рисунок 3.63). После данной команды движение вверх полностью прекращается. Правая рука при этом находится в максимально вертикальной позиции (мотор **№3** остановился на позиции **512**).

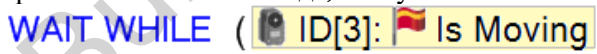
В строке **№6** указана команда 

, которая приказывает мотору **№3** двигаться к позиций **0,** мотор **№3** при этом двигается из позиции **512** к позиции **0**.

Так как действие команды по активации мотора **№1** не было отменено 

(строка программы **№ 4**), мотор **№1** по-прежнему активирован и редуктор данного мотора не позволяет проскальзывать.

Данная команда устанавливает скорость движения мотора **№1**, равную нулю, что включает редуктор, блокирующий проскальзывание.

По достижении позиции **0** мотора **№3** необходимо опять указать контроллеру робота Bioloid, что действие команды, расположенной в строке **№6,** закончилось – для этого применяется команда, указанная в строке 7 

После данной команды движение вверх полностью прекращается. Правая рука при этом максимально опущена (мотор **№3** остановился на позиции 0).

Оператор 

необходимо применять всегда, когда необходимо указать, что какое-то движение конкретного мотора закончилось.

3.3.2 Программа поочередного взмаха обоими руками

Используются моторы **№1, №2, №3, №4**: сначала правая рука поднимается вертикально вверх, а затем опускается в начальное положение: используются моторы **№ 1, №3**. Данное действие происходит 2 раза.

Затем левая рука поднимается вертикально вверх – затем левая рука опускается в начальное положение (используются моторы **№2, №4).** Это действие происходит 2 раза (рисунок 3.64).

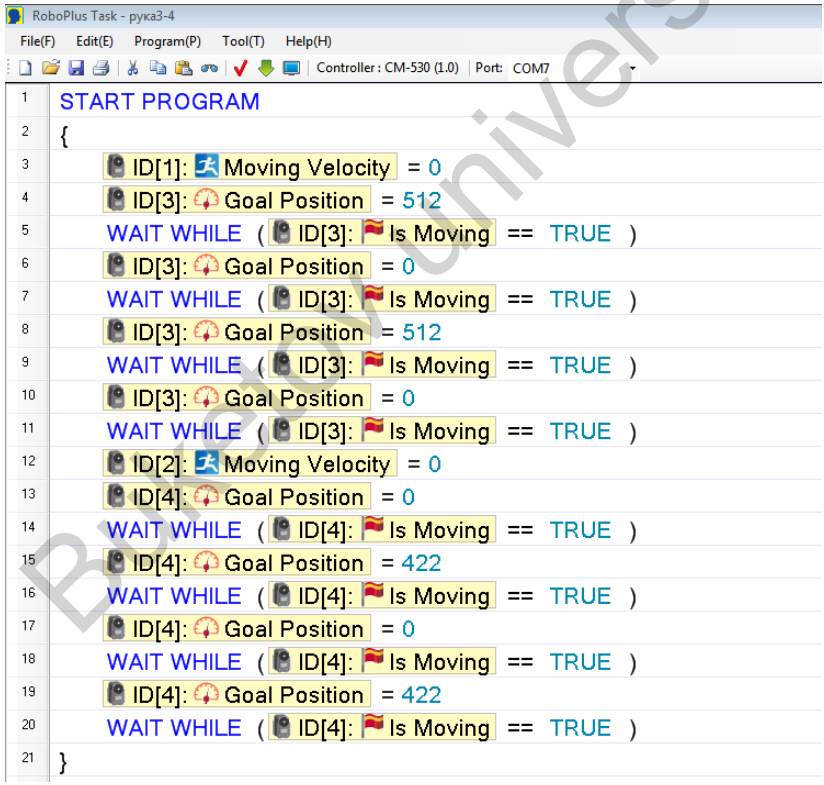
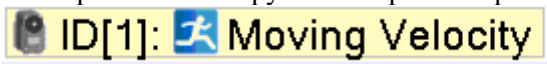


Рисунок 3.64 Поочередный взмах правой и левой руками робота

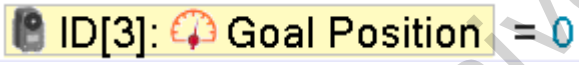
В строке номер **№3** активируем мотор **№1** при помощи команды 

. Данная активация необходима для того, чтобы редуктор мотора **№1** заблокировал вал вращения мотора **№1**, для предотвращения прокрутки вала в данном моторе.

Далее в строке **№4** указываем что мотор **№3** должен начать движение к позиции **512**. Мотор будет двигаться до тех пор, пока не достигнет позиции **512** – выключение мотора **№3** произойдет при помощи оператора 

.

В результате мотор **№ 3** поднял правую руку в вертикальное положение.

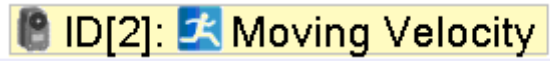
Далее в строке **№6** стоит оператор 

, приказывающий двигаться мотору **№3** в позицию **0** (мотор движется из позиций **512** к позиции **0**). По достижении позиций **0** срабатывает оператор 

, находящийся в строке **№7**. В результате мотор **№3** опустил правую руку в начальное положение.

В строках **№8–11** повторяются действия, выполняемые в строках **№3–7**, в результате чего правая рука поднимается вертикально верх, а затем опускается вниз.

В результате действий, указанных в строках №3–11, правая рука 2 раза поднялась вертикально вверх и 2 раза опустилась вниз (использовались моторы №3, №1).

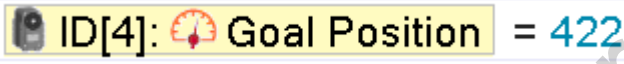
В строке номер **№12** активируем мотор **№2** при помощи команды 

. Данная активация необходима для того, чтобы редуктор мотора **№2** застабилизировал ось вращения мотора **№2** и не происходило прокрутки в данном моторе.

Далее в строке **№13** указываем, что мотор **№4** должен начать движение к позиции **0** 

. Мотор будет двигаться до тех пор, пока не достигнет позиции **0** – выключение мотора **№4** произойдет при помощи оператора 

, расположенного в **№14** строке. В результате мотор **№4** поднял левую руку в вертикальное положение.

Далее в строке **№15** стоит оператор 

, который приказывает двигаться мотору в позицию **422** (мотор движется из позиций 0 к позиций **422)**. По достижении позиций **422** срабатывает оператор 

, находящийся в строке **№16**. В результате мотор **№4** опустил левую руку в начальное положение.

В строках **№17–20** повторяются действия, выполняемые в строках **№13–16**, в результате чего левая рука поднимается вертикально вверх, а затем опускается вниз. В результате действии указанных в строках **№12–20** левая рука 2 раза поднялась вертикально верх и 2 раза опустилась вниз (использовались моторы **№4** и **№2**).

В результате действий операторов данной программы моторы №1, №3 и моторы №2, №4 по очереди поднимают вертикально вверх правую и левую руки.

3.3.3 Одновременное поднятие рук моторами №3–4.

На рисунке 3.65 показана программа, реализующая алгоритм одновременного поднятия левой и правой рук.

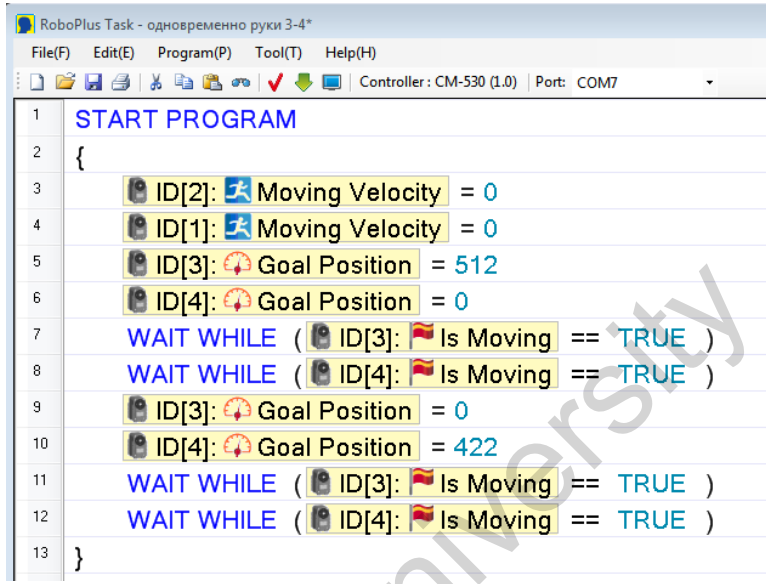
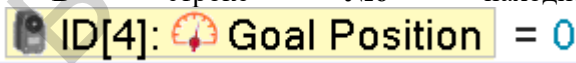


Рисунок 3.65 Одновременное поднятие левой и правой руки

В данной программе одновременно срабатывают моторы **№1, №2, №3, №4**, в результате чего одновременно левая и правая рука поднялись и одновременно опустились. В строке **№5** находится команда

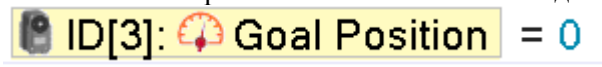
, приказывающая мотору **№3** поднять правую руку вверх вертикально (до позиции 512).

В строке **№6** находится команда



, приказывающая левую руку поднять вертикально вверх (до позиции 0). В строках **№7, №8** находятся операторы:

Эти операторы приказывают прекратить действия моторам **№3** и **№4** при выполнении условий, записанных в строках **№5** и **№6**.

В строке **№9** находится команда 

, приказывающая мотору **№3** опустить правую руку вниз вертикально (до позиции **0**).

В строке **№10** находится команда 

, приказывающая левую руку опустить вертикально вниз (до позиции 422).

В строках **№11, №12** находятся операторы

Они приказывают прекратить действия моторам **№3** и **№4** при выполнении условий, записанных в строках **№9** и **№10.**

3.3.4 Программа «Гимнастика для рук»

В программе «Гимнастика для рук» используются моторы с 1 по 6 (рисунок 3.66).

Команды, расположенные в строках **№3,4,5,6,7,8** приказывают моторам **№4, 3, 5, 6** и **№ 2, 1** произвести следующие действия.

Команды в строках **№3 и №4** приводят в действия моторы **№4 и №3**, в результате чего правая и левая рука поднимаются на **90** градусов (скорость движения моторов **-100**)

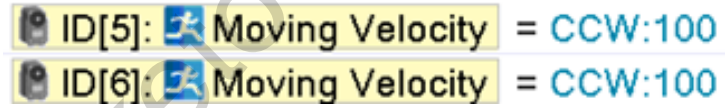
Рисунок 3.66 Программа гимнастика для рук

Направления движения моторов указаны в строках **№ 9, №10**. Моторы **№3, №4** должны дойти до позиций **216**,

а потом остановится. Приказ об остановке моторов **№4 и № 3** размещен в строках **№11, №12**

Далее моторы **№5 и №6** поднимают ладони робота вертикально вверх при помощи команд, находящихся в строке **№13,№ 14**

Позиция **800** для мотора **№5** – это вертикально поднятая ладонь правой руки робота. Позиция **193** для мотора **№6** – это вертикально поднятая ладонь левой руки робота. Моторы **№5 и №6** движутся со скоростью **100**, что указанно в строках №5 и №6



Потом следует команда для моторов **№5 и № 6,** мотором в строке **№15 и №16** прекратить движение по достижении позиций, указанных в строках **№13, №14**

Далее робот, держа руки горизонтально, опускает ладони из положения вертикально вверх в положение вертикально вниз. Данные команды для моторов **№5 и №6** указаны в строках **№17, №18**

По достижению этих позиций моторам **№5 и №6** поступает приказ прекратить данное действие

(строки **№19, №20**).

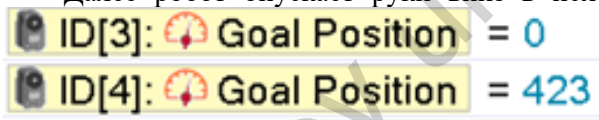
Далее робот поднимает ладони на **90** градусов (работают моторы **№5 и №6**

в строке **№21 и №22**).

В строках **№23, №24** находятся команды по остановке движения моторов

.

Далее робот опускает руки вниз в исходное положение

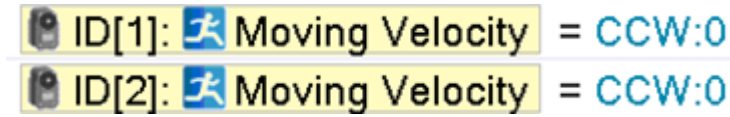


.

Данные команды, находящиеся в строках **№25, №26**, приказывают моторам **№3 и №4** двигаться к позиции **0 и 423**, что соответствует опущенным вниз рукам робота. По достижении этих позиции срабатывают команды остановки движения моторов **№3 и №4**, расположенных в строках **№27, №28**

.

Для того чтобы руки робота не проскальзывали в плечевых суставах в строках **№7 и №8** активируются моторы **№1 и №2**:



.

Контрольные вопросы

1. Дайте определение термину «Человекообразный робот».
2. Какого главное предназначение данных роботов?
3. Через какой контроллер осуществляется всё управление робота Bioloid?
4. Опишите принцип работы микроконтреллера?
5. Какие имеются способы подключения электропитания?
6. Расшифруйте аббревитуру GPIO?
7. Назовите программное обеспечение, которое является единым для всей продукции Robotis и применяется для семейства OLLO, Bioloid, Expert kit?
8. Команда для задания скорости вращения моторов?
9. Команда для остановки текущего действия? Приведите примеры использования.
10. Опишите процедуру установки звукового сопровождения. Приведите пример программы генерации музыкальной ноты.

Список использованной литературы

1 Lego Education: Базовый набор Lego Mindstorms Education EV3 [Электронный ресурс]. Режим доступа: https://www.marwin.kz/education/lego-bazovyy-nabor-lego-mindstorms-education-ev3.html

2 Корягин А.В., Смольянинова Н.М. Физические эксперименты и опыты с Lego Mindstorms EV3: научное издание - М.: ДМК Пресс, 2020. - 181 с

3 Mindstorms EV3 [Электронный ресурс]. Режим доступа: https://softdroids.com/119-mindstorms-ev3.html

4 Курс программирования Lego EV-3 [Электронный ресурс]. Режим доступа: http://itrobo.ru/robototehnika/kurs-programmirovanija-lego-ev3.html

5 Белиовский Н.А., Белиовская Л.Г. Использование Lego-роботов в инженерных проектах школьников. Отраслевой подход. - М.: ДМК Пресс, 2016. - 88 с.

6 Добриборщ Д., Артемов К., Чепинский А. Основы робототехники на Lego Mindstorms Education EV3. Сп -Б Лань, 2018 г. – 108 с.

7 Овсяницкая Л.Ю., Овсяницкий Д.Н. Курс программирования робота EV3 в среде Lego Mindstorms EV3. М.: из-во Перо, 2016. - 300 с.

8 Ультразвуковые датчики [Электронный ресурс]. Режим доступа: https://mirrobo.ru/micro/ultrazvukovye-datchiki

9 Валуев А.А. Конструируем роботов на LegoMindstorms Education EV3. М.: Лаборатория знаний, 2017г. - 79 с.

10 Робот-конструктор Makeblock Ultimate Robot Kit V2.0 (10-в-1) [Электронный ресурс]. Режим доступа: https://attech.kz/p55068217-robot-konstruktor-makeblock.html.

11 Arduino Uno [Электронный ресурс]. Режим доступа: http://arduino.ru/Hardware/ArduinoBoardUno

12 Датчики "Arduino": описание, характеристики, подключение, отзывы [Электронный ресурс]. Режим доступа: https://fb.ru/article/429048/datchiki-arduino-opisanie-harakteristiki-podklyuchenie-otzyivyi

13 Момот М.В. Мобильные роботы на базе Arduino. СПб.: БХВ, 2017. - 288 с.

14 Виды и примеры Arduino модулей [Электронный ресурс]. Режим доступа: https://arduinoplus.ru/arduino-moduli
15 Самые популярные датчики для Arduino [Электронный ресурс]. Режим доступа: http://electrik.info/microcontroller/1447-samye-populyarnye-datchiki-dlya arduino.html
16 Суперайс [Электронный ресурс]. Режим доступа: https://supereyes.ru/ articles/other/obzor kontrollerov_ arduino
17 Высокопроизводительные 8-разрядные RISC микроконтроллеры семейства AVR [Электронный ресурс]. Режим доступа: http://www.gaw.ru/html.cgi/txt/ic/Atmel/micros/avr/start.htm
18 Устройство микроконтроллера ATmega328 - описание, характеристики [Электронный ресурс]. Режим доступа: https://robolive.ru/mikrokontroller-atmega328-opisanie-xarakteristiki
19 Atmega [Электронный ресурс]. Режим доступа: http://avr.ru/docs/d-sheet/atmega
20 Программаторы для AVR микроконтроллеров (USB, COM, LPT) [Электронный ресурс]. Режим доступа: https://ph0en1x.net/73-avr-microcontroller-programmers-usbasp-isp-com-lpt.html
21 Внутрисхемное программирование PIC-контроллеров [Электронный ресурс]. Режим доступа: http://www.5v.ru/icsp.htm
22 JTAG программатор [Электронный ресурс]. Режим доступа: https://www.phyton.ru/programmers/jtag-programmator
23 Makeblock Orion [Электронный ресурс]. Режим доступа: http://learn.makeblock.com /en/makeblock-orion
24 Петин В.А., Биняковский А.А. Практическая энциклопедия Arduino. / OZON. М.: ДМК Пресс, 2017г. - 152 с.
25 Как выбрать датчик для Arduino [Электронный ресурс]. Режим доступа: https://amperka.ru/page/kak-vybrat-datchik-dlya-arduino
26 All On Robots [Электронный ресурс]. Режим доступа: https://www.allonrobots.com/bioloid/
27 ROBOTBAZA [Электронный ресурс]. Режим доступа: https://robotbaza.ru/page/roboplus
28 BIOLOID Beginner [Электронный ресурс]. Режим доступа: https://emanual.robotis.com/docs/en/edu/bioloid/beginner/

Содержание

Учебное издание

Сельдюгаев Олег Борисович,
Казимова Динара Ашубасаровна,
Самойлова Ирина Алексеевна

РОБОТООРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Учебник

Отпечатано с авторского оригинала

Подписано в печать 05.08.2023 г.
Формат 60×84 1/16. Бумага ксероксная. Гарнитура Times.
Усл. печ. л. 14,81 п.л. Тираж 50 экз. Заказ № 80.

Карагандинский университет имени академика Е.А. Букетова
100024, г. Караганда, ул. Университетская, 28.

Отпечатано в Издательстве Карагандинского
университета имени академика Е.А. Букетова
Тел. (7212) 35-63-16. E-mail: izd_kargu@mail.ru