Лоренс Валк

БОЛЬШАЯ КНИГА LEGO MINDSTORMS EV3

Подробное руководство для начинающих
по постройке и программированию роботов

no starch press


mindstorms
EV3

Laurens Valk

THE LEGO® MINDSTORMS® EV3 DISCOVERY BOOK

A beginner's guide to building and programming robots

Лоренс Валк

БОЛЬШАЯ КНИГА LEGO MINDSTORMS EV3

Подробное руководство для начинающих
по постройке и программированию роботов

Москва
2017

УДК 379.8
ББК я92
Л78

The LEGO Mindstorms EV3 Discovery Book
ISBN 978-1-59327-532-7 published by No Starch Press.

Лоренс, Валк.

Л78 Большая книга LEGO MINDSTORMS EV3 / Лоренс Валк ; [пер. с англ. С. В. Черникова]. — Москва : Издательство «Э», 2017. — 408 с. : ил. — (Подарочные издания. Компьютер).

ISBN 978-5-699-94356-2

Большая книга LEGO MINDSTORMS EV3 представляет собой полное руководство для начинающих. Вы начнете с основ и постепенно сможете конструировать все более сложных роботов.

Вы научитесь собирать и программировать колесные транспортные средства, которые смогут перемещаться по комнате и следовать по трассе; обтекаемый гоночный автомобиль на дистанционном управлении; шестиногого шагающего робота-муравья; роботизированную руку, которая может самостоятельно находить, поднимать и перемещать предметы; говорящего и ходящего человекоподобного робота.

Все, что нужно для постройки собственных моделей, вы найдете в наборе LEGO MINDSTORMS EV3 (#31313).

УДК 379.8
ББК я92

ISBN 978-5-699-94356-2

LEGO® MINDSTORMS® EV3 DISCOVERY BOOK

THE LEGO® MINDSTORMS® EV3 DISCOVERY BOOK

a beginner's guide to building and programming robots

laurens **valk**

no starch press

LEGO® MINDSTORMS® EV3 DISCOVERY BOOK

Руководство для начинающих по сборке и программированию роботов

Лоренс Валк

Printed in USA
Fourth printing

20 19 18 17 16 4 5 6 7 8 9 10

ISBN-10: 1-59327-532-3
ISBN-13: 978-1-59327-532-7

Publisher: William Pollock
Production Editor: Serena Yang
Interior Design: Octopod Studios
Developmental Editor: Seph Kramer
Technical Reviewer: Claude Baumann
Copyeditor: Julianne Jigour
Compositors: Riley Hoffman and Alison Law
Proofreader: Paula L. Fleming
Indexer: BIM Indexing & Proofreading Services

For information on distribution, translations, or bulk sales, please contact No Starch Press, Inc. directly:
No Starch Press, Inc.
245 8th Street, San Francisco, CA 94103
phone: 1.415.863.9900; fax: 1.415.863.9950; info@nostarch.com; www.nostarch.com

The Library of Congress has cataloged the first edition as follows:
Valk, Laurens.
The LEGO Mindstorms NXT 2.0 discovery book : a beginner's guide to building and programming robots / Laurens Valk.
p. cm.
Includes index.
ISBN-13: 978-1-59327-211-1
ISBN-10: 1-59327-211-1
1. Robots--Design and construction--Popular works. 2. Robots--Programming--Popular works. 3. LEGO toys. I. Title.
TJ211.15.V353 2010
629.8'92--dc22
2010011157

Об авторе

Лоренс Валк — инженер-робототехник из Нидерландов, окончил Дельфтский технический университет по специальности «Машиностроение» с дипломом бакалавра. Валк состоит в международной группе MINDSTORMS Community Partners (MCP), сообществе фанатов серии MINDSTORMS, помогающих тестировать и разрабатывать новые продукты в этой линейке. Конструирование роботов он начал с системы EV3 за год до ее выхода для широкой публики в 2013 году, и один из его роботов представлен на упаковке EV3 в качестве официальной бонусной модели.

Лоренсу нравится придумывать новые модели и составлять обучающие материалы по конструированию и программированию роботов, чтобы любители роботов по всему миру могли воссоздавать его проекты и узнавать о робототехнике что-то новое. Он принимал участие в написании нескольких книг о роботах LEGO, включая ставшее бестселлером первое издание данной книги, The LEGO MINDSTORMS NXT 2.0 Discovery Book (No Starch Press, 2010). Лоренс Валк ведет блог о роботах на сайте robotsquare.com.

О техническом редакторе

Клод Бауман изучал сложную робототехнику LEGO MINDSTORMS в течение пятнадцати лет на дополнительных занятиях в школе. Он создал ULTIMATE ROBOLAB, среду для кросс-компиляции, благодаря которой стало возможно визуальное программирование блока LEGO RCX. С ее помощью он сумел разработать единственную в мире самотиражирующуюся программу для LEGO RCX — некоторые называют ее вирусом. Недавно он как член группы MINDSTORMS Community Partners (MCP) участвовал в проектировании нового микрокомпьютера EV3 (Intelligent Brick). Бауман был консультантом и оценивал различные школьные проекты по робототехнике; он написал книгу Eureka! Problem Solving with LEGO Robotics (NTS Press, 2013), несколько статей и текстов выступлений для конференций. Особенно его интересует вопрос локализации звука. Клод возглавляет сеть учебных заведений в Люксембурге, он женат, у него трое детей и трое замечательных внуков.

Благодарности

В самую первую очередь я хочу поблагодарить читателей первого издания этой книги. Ваши бесчисленные электронные письма и комментарии, которые приходили мне со всего света, вдохновили меня переработать книгу, кроме того, ваши отзывы подтолкнули меня к раскрытию новых тем.

Эта книга по своей философии и структуре аналогична первому изданию, но, поскольку модель LEGO MINDSTORMS NXT уступила место модели EV3, книгу, по сути, пришлось переписывать заново. Это стало возможным только благодаря помощи многих талантливых людей.

Огромная благодарность Клоду Бауману, который рецензировал эту книгу и проверял технические моменты, а также подсказал, как можно ее улучшить. Еще хочу поблагодарить Марка-Андре Базерги (Marc-André Bazergui), Мартина Богартса (Martijn Boogaarts), Кеннета Мэдсена (Kenneth Madsen) и Ксандера Солдата (Xander Soldaat) за то, что они еще в 2012 году тестировали прототипы роботов, о которых говорится в этой книге.

Кроме того, я благодарю сотрудников издательства No Starch Press, которые обеспечили успех первого издания этой книги и помогали мне в работе над новым изданием. Благодарю моего издателя Уильяма Поллока (William Pollock), моего редактора Сефа Крамера (Seph Kramer), Серену Йанг (Serena Yang), которая следила за сроками, Райли Хофмана (Riley Hoffman) и Элисон Ло (Alison Law), которые поместили голый текст на красочные страницы, а также Ли Полер (Leigh Poehler) за помощь в решении всех моих деловых вопросов в течение последних лет.

Благодарю компанию LEGO Group, которая занимается разработкой такой удивительной и развивающей серии роботов и привлекает к этому процессу членов сообщества. Благодарю команду разработчиков LEGO MINDSTORMS EV3, в частности Камиллу, Дэвида, Флемминга, Хенрика, Ларса Джо, Лассе, Ли, Линду, Мэри, Стивена и Уиллема.

Благодарю сообщество LDraw за разработку инструментов, необходимых для создания описанных в этой книге инструкций по сборке. Отдельная благодарность Филиппу Хурбейну (Philippe Hurbain) за построение в LDraw трехмерных моделей деталей EV3, Майклу Лачману (Michael Lachmann) за создание приложения MLCad, Трэвису Коббсу (Travis Cobbs) за разработку LDView и Кевину Клейгу (Kevin Clague) за разработку LPub 4 и LSynth. Я выражаю признательность Джону Хансену (John Hansen), который создал инструмент захвата изображения экрана для EV3.

И наконец, я благодарю своих друзей и членов семьи за поддержку во время написания этой книги. И особая благодарность Фабьен, которая беспрестанно подбадривала меня, побуждая завершить этот проект. Спасибо тебе, ты — лучшая.

Краткое содержание

Содержание

ЧАСТЬ IV. РОБОТЫ-ЖИВОТНЫЕ И ТРАНСПОРТНЫЕ СРЕДСТВА

ЧАСТЬ V. РАЗРАБОТКА СЛОЖНЫХ ПРОГРАММ

Введение

Вы готовы открыть для себя захватывающий мир робототехники? Раз вы читаете эту книгу, я могу сделать вывод, что в качестве обучающего инструмента вы взяли набор LEGO MINDSTORMS EV3, и мне кажется, это отличный выбор.

Я впервые познакомился с серией MINDSTORMS в 2005 году, когда мне было тринадцать лет. У меня появился набор Robotics Invention System, версия, продававшаяся в то время. Сначала это было просто хобби, но позже я настолько увлекся роботами, что решил получить инженерное образование.

LEGO MINDSTORMS — это отличный способ познакомиться со множеством понятий из сферы робототехники и инженерии, например программированием и применением моторов и датчиков.

Цель этой книги — помочь вам исследовать огромный спектр возможностей конструкторов MINDSTORMS. Я надеюсь, что вам будет так же интересно работать с этим набором, как и мне, и что вы при этом сможете узнать много нового!

Зачем нужна эта книга

Набор LEGO MINDSTORMS EV3 содержит множество деталей и инструкции по сборке пяти роботов. Я считаю, что вам будет очень интересно собирать и программировать этих роботов, но у новичков, которые хотят пойти дальше базовых моделей, могут возникнуть затруднения. В этом наборе есть все инструменты, необходимые для конструирования основных роботов, но в инструкциях приводится лишь небольшая часть тех сведений, которые вам нужны для создания и программирования ваших собственных, новых роботов.

Эта книга составлена в форме практического руководства, которое познакомит вас с ресурсами и возможностями LEGO MINDSTORMS EV3, когда вы будете учиться придумывать, конструировать и программировать свои уникальные модели роботов.

Подходит ли вам эта книга

Эта книга написана для тех, кто никогда прежде не сталкивался ни с конструированием, ни с программированием LEGO MINDSTORMS. По мере прочтения вы будете продвигаться от самых азов до продвинутого уровня программирования, научитесь создавать все более сложные модели роботов. Новичкам лучше начать с главы 1, а затем пошагово выполнить инструкции из главы 2, чтобы собрать и запрограммировать базового робота. Более опытные пользователи MINDSTORMS могут сразу перейти к той главе, в которой описаны наиболее затруднительные для них моменты, и двигаться уже оттуда. Главы для опытных программистов собраны в части V, а модели роботов, описанные в части VI, будут особенно интересны пользователям продвинутого уровня.

Как пользоваться этой книгой?

Хотя этой книгой можно пользоваться как обычным справочником, она задумывалась скорее как учебное пособие. Я собрал воедино задания по конструированию, программированию и робототехнике, чтобы избежать длинных теоретических рассуждений, которые довольно трудно воспринимать. Например, вы изучите основные методы программирования и одновременно с этим узнаете, как научить своего первого робота двигаться. Кроме того, создавая новых роботов, вы получите представление о продвинутых техниках программирования.

Эта книга построена по принципу «обучение через практику», который, по моему мнению, лучше всего подходит для тех, кто хочет научиться конструировать и программировать роботов MINDSTORMS.

Практикумы

Чтобы вам было проще понять информацию из каждой главы, я дополнил книгу множеством так называемых практикумов. Выполняя задания из *практикумов*, вы научитесь расширять программы из примеров или даже придумывать абсолютно новые программы. Например, когда вы узнаете, как запрограммировать робота, чтобы он воспроизводил звуки и отображал текст на экране, вам будет предложено написать программу, благодаря которой робот во время своей речи будет выводить на экране субтитры.

В конце многих глав вы увидите разделы «Сделай сам». Там вы сможете почерпнуть идеи, как доработать или изменить робота, сконструированного в данной главе. Например, у вас будет задание улучшить гоночного робота, добавив шестеренки (зубчатые колеса) между моторами и колесами, чтобы он ездил быстрее. У вас даже будет задание создать нового робота, который превратит конструктор MINDSTORMS EV3 в охранную сигнализацию!

Оценка сложности и времени

Чтобы вам было проще выбрать те практикумы, которые вы захотите выполнить, я присвоил каждому из них определенный уровень сложности. Для выполнения простых заданий () обычно нужно создать или расширить программу теми же способами, которые описаны в примерах. Средняя () сложность указывает на то, что вам нужно подумать еще немного и, возможно, объединить новый материал с тем, что вы изучили ранее. Сложные практикумы () заставят вас проявить свои творческие способности и выйти за рамки примеров из книги.

Указывая степень сложности, я исхожу из того, что вы будете читать эту книгу в порядке расположения глав. То есть

задание, отмеченное как «сложное» в главе 4, на самом деле может быть довольно простым по сравнению с заданием, отмеченным как «сложное» в главе 19. Кроме того, для каждого практикума указано, сколько времени необходимо для его выполнения. С этой точки зрения задания делятся на короткие (), средние () или длинные (). Как правило, короткие практикумы можно выполнить, внеся всего несколько изменений в программу из примера, а длинные практикумы подразумевают создание абсолютно новой программы.

Разделы «Сделай сам» обычно требуют больше времени, поскольку они включают в себя и конструирование, и программирование. Они оцениваются в соответствии с тем, в какой степени в них представлено конструирование () и программирование ().

Поиск решений

В некоторых практикумах вы найдете советы, в каком направлении двигаться, но каждую задачу можно решить несколькими способами. Вовсе не обязательно в точности следовать указаниям, вы спокойно можете найти новое решение, которое не пришло мне в голову. Уровень сложности и время, необходимое для выполнения каждого практикума, тоже приблизительны. Не переживайте, если на определенное задание у вас ушло немного больше времени. Просто помните, что решение задач в первую очередь должно приносить вам удовольствие!

Решение некоторых практикумов можно найти в архиве, доступном по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**. Эти варианты помогут вам на первых этапах, но для выполнения тех заданий, которые не вошли в архив, вам нужно будет применить свои творческие способности.

Что вы найдете в каждой главе

Ниже представлен краткий обзор всех шести частей этой книги. Некоторые термины могут оказаться новыми для вас, но, дочитав книгу до конца, вы будете знать о них все.

Часть I. Приступаем!

Часть I, а в частности глава 1, повествует о деталях, входящих в набор MINDSTORMS EV3. В главе 2 вы соберете своего первого робота и узнаете, что такое микрокомпьютер EV3. В главе 3 вы познакомитесь с программным обеспечением EV3, с помощью которого вы будете программировать роботов. В главе 4 рассказывается, как с помощью этого программного обеспечения заставить робота двигаться. Вы напишете свои первые программы, пользуясь базовыми программными блоками. И наконец, в главе 5 вы узнаете основные способы программирования, например как научить робота повторять одно и то же действие или выполнять несколько задач одновременно.

Часть II. Программирование роботов с датчиками

В этой части вы узнаете все о датчиках, которые представляют собой важную составляющую роботов MINDSTORMS. В главе 6 вы прочитаете, как добавить датчик касания к созданному ранее роботу, и изучите необходимые для работы с датчиками способы программирования. Затем в главе 7 вы познакомитесь с датчиком цвета, в главе 8 — с инфракрасным маяком, а в главе 9 — с двумя видами встроенных датчиков.

Часть III. Способы конструирования роботов

Эта часть посвящена входящим в состав набора MINDSTORMS EV3 строительным деталям LEGO Technic. В главе 10 вы научитесь пользоваться балками, осями и соединительными деталями, а в главе 11 узнаете, для чего роботам нужны зубчатые колеса.

Часть IV. Роботы-животные и транспортные средства

Когда вы освоитесь с моторами и датчиками, нужно будет применить эти знания на практике и собрать двух роботов. В главе 12 вы построите гоночную машину EV3, а в главе 13 — робота-муравья.

Часть V. Разработка сложных программ

Часть V посвящена более сложным способам программирования. Вы узнаете, что такое шины данных (глава 14), как пользоваться значениями датчиков и как проводить вычисления с помощью EV3 (глава 15), как научить робота запоминать данные, содержащие переменные (глава 16). В главе 17 рассказывается, как объединить все эти способы программирования и создать робота, на экране которого можно будет рисовать, как на электронной доске для рисования типа советской игры «Волшебный экран».

Часть VI. Роботы-машины и гуманоиды

Познакомившись с моторами, датчиками и различными способами программирования, вы перейдете к этой части, в которой сделаете двух сложных роботов. В главе 18 рассказывается о том, как собрать и запрограммировать SNATCH3R — автономного робота-манипулятора, способного самостоятельно находить, захватывать и поднимать и перемещать инфракрасный маяк. А в главе 19 вы создадите LAVA R3X — изображенного на обложке этой книги гуманоида, который может ходить и разговаривать. Он спроектирован по примеру знаменитого робота-гуманоида Alpha Rex из предыдущего поколения LEGO MINDSTORMS.

Примеры к книге

По адресу **eksmo.ru/files/Lego_Mindstorms_Primers.zip** вы сможете скачать архив со всеми программами-примерами из этой книги и решениями некоторых представленных здесь практикумов.

Заключение

MINDSTORMS — это источник творческого вдохновения и стимул к познанию для людей всех возрастов. Поэтому берите набор MINDSTORMS EV3, начинайте читать главу 1 и откройте для себя яркий мир LEGO MINDSTORMS. Я надеюсь, что эта книга даст простор вашему воображению!

ЧАСТЬ I

Приступаем!

1

Подготовка к работе с конструктором EV3

Все роботы, о которых пойдет речь в этой книге, могут быть построены с использованием одного набора LEGO MINDSTORMS EV3 (артикул 31313). Если у вас есть набор, показанный на рис. 1.1, вы можете сразу приступить к работе. Если вы приобрели набор LEGO MINDSTORMS образовательный набор EV3 (артикул 45544), см. в приложении В список деталей, которые необходимы для работы над проектами, представленными в этой книге.

В этой главе вы узнаете об интеллектуальном модуле EV3 (программируемом блоке) и других деталях в наборе MINDSTORMS EV3, скачаете и установите программное обеспечение, которое понадобится для программирования роботов.

Рис. 1.1. Набор LEGO MINDSTORMS EV3 (артикул 31313) содержит все компоненты, необходимые для создания роботов, описанных в этой книге

Содержимое коробки

Набор LEGO MINDSTORMS EV3 поставляется с большим количеством строительных деталей Technic, а также электронных компонентов, включающих моторы, датчики, микрокомпьютер EV3, маяк для дистанционного управления роботом и кабели (рис. 1.2). В процессе чтения этой книги вы научитесь использовать каждый из этих компонентов. Кроме того, в конце книги приведен полный список деталей.

Для сборки роботов EV3 используются *моторы* большого или среднего размера, предназначенные для приведения в движение их колес, рук и других движущихся частей. *Датчики* позволяют получать данные из окружающей среды, например анализировать цвет поверхности или приблизительное расстояние до объекта. *Кабели* соединяют моторы и датчики с модулем EV3. *Удаленный инфракрасный маяк*, или просто маяк, можно использовать для дистанционного управления и отслеживания робота.

Модуль EV3

Модуль EV3, *программируемый блок*, или просто *EV3*, представляет собой небольшой компьютер, который управляет моторами и датчиками робота, позволяя ему передвигаться самостоятельно. Вскоре вы соберете робота, который автоматически обходит препятствия на своем пути. Когда датчик сообщает EV3, что рядом находится какой-то объект, EV3 передает моторам соответствующую команду, чтобы робот обошел его.

Ваш робот способен делать это благодаря использованию программы-списка действий, которые он будет выполнять, как правило, по очереди. Вы сами будете разрабатывать программы на компьютере с помощью специальной программы LEGO MINDSTORMS EV3.

После того как вы закончите создание программы, вы установите ее на модуль EV3 с помощью кабеля USB, который входит в комплект, и ваш робот будет готов делать то, на что запрограммирован.

Рис. 1.2. Набор MINDSTORMS EV3 содержит строительные детали Technic, моторы, датчики, модуль EV3, маяк для дистанционного управления роботом и кабели

Для питания модуля EV3 используется либо блок с шестью батарейками типа AA, как показано на рис. 1.3, либо аккумуляторная батарея LEGO EV3 (№ 45501) и зарядное устройство (№ 8887). В случае установки аккумуляторной батареи модуль EV3 немного увеличивается в размерах. Это не мешает при создании роботов из этой книги, но если вы захотите собрать робота TRACK3R, который изображен на упаковке LEGO MINDSTORMS EV3, нужно будет его немного изменить, чтобы освободить чуть больше места для модуля EV3. Для питания инфракрасного маяка используйте две батарейки AAA.

Рис. 1.3. Для обеспечения питания модуля EV3 можно использовать либо шесть батареек типа AA, либо аккумуляторную батарею EV3

Сортировка деталей

Чтобы сэкономить время при поиске деталей LEGO Technic, задумайтесь о сортировке и хранении их в органайзере, например таком, какой показан на рис. 1.4. Так упростится процесс сборки моделей, описанных в книге, и процесс создания собственных роботов. Вы сможете сразу увидеть, что нужные детали закончились, и вам не придется тратить время на поиски деталей, которых у вас нет.

При сортировке деталей рекомендуется ориентироваться на их функциональность. Например, отделить балки от зубчатых колес, осей и т.д. Если в органайзере недостаточно отделений для каждого типа деталей, храните вместе легко различимые детали. Например, лучше хранить короткие серые оси с короткими красными осями, чем смешивать серые оси различной длины.

К наборам EV3 прилагается набор наклеек, по одной для каждой белой панели. Сразу наклейте все, как показано на рис. 1.5; узор на наклейке поможет вам определить, какую панель — большого или малого типа — нужно использовать при сборке.

Трасса для выполнения миссий

На внутренней стороне коробки, в которую упакован набор MINDSTORMS EV3, находится трасса для выполнения миссий (рис. 1.6). Вы можете запрограммировать роботов на взаимодействие с этой трассой, например, на движение по толстой красной линии (см. главу 7). Вы можете перейти

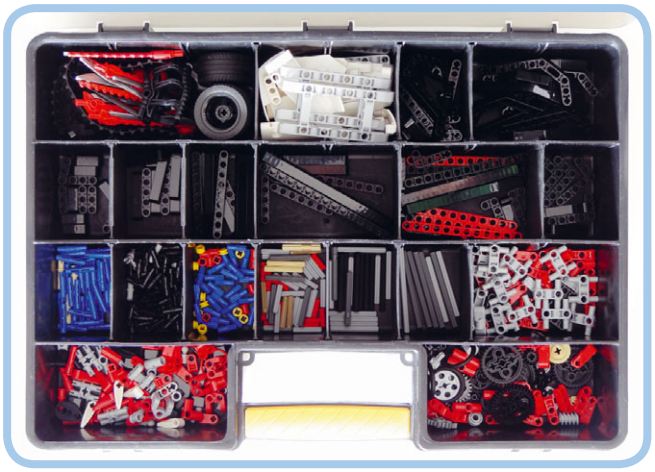


Рис. 1.4. Пример органайзера, в котором разложены строительные детали Technic из одного набора MINDSTORMS EV3. Размеры этого контейнера составляют 45 × 33 × 9 см

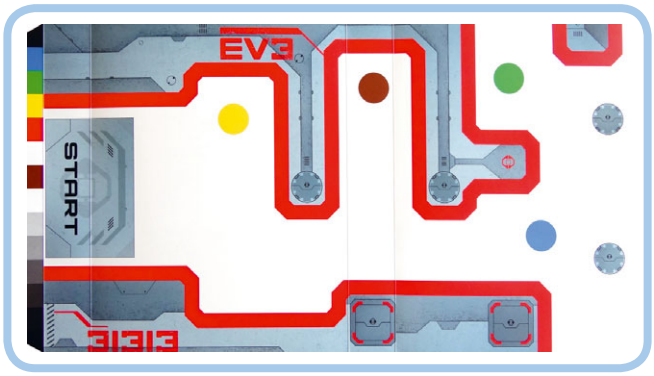


Рис. 1.6. Трасса для выполнения миссий. Чтобы ее подготовить, осторожно разрежьте картонную коробку от набора MINDSTORMS EV3 — обратите внимание на пунктирную линию и изображение ножниц

Рис. 1.5. Чтобы аккуратно прикрепить наклейки на панели, сначала соедините элементы панели с помощью двух черных штифтов. Так будет легче выровнять наклейку. Удалите штифты, когда закончите

по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**, чтобы загрузить и распечатать трассу для передвижения робота.

Управление роботом

Набор MINDSTORMS EV3 позволяет управлять роботом различными способами, как показано на рис. 1.7. Из этой книги вы узнаете, как с помощью редактора EV3 разрабатывать программы, благодаря которым робот сможет автоматически выполнять определенные действия. Вы научитесь управлять роботом дистанционно, посылая команды с помощью удаленного инфракрасного маяка, который поставляется в наборе, или смартфона/планшета со специальным загружаемым программным обеспечением. Эти программы позволят управлять моторами и датчиками вашего робота и даже создать настраиваемый удаленный интерфейс.

Загрузка и установка программного обеспечения EV3

Прежде чем вы начнете разрабатывать программы для роботов, вам необходимо скачать и установить специальное обеспечение для программирования EV3. Чтобы выполнить следующие шаги, требуется подключение к Интернету.

Если компьютер, который вы будете использовать для программирования, не подключен к Интернету, выполните шаги 1 и 2 на компьютере с доступом в Интернет и скачайте установочный файл на Flash-носитель объемом не менее 1 Гб. Затем скопируйте файл с Flash-носителя на свой компьютер и переходите к шагу 3.

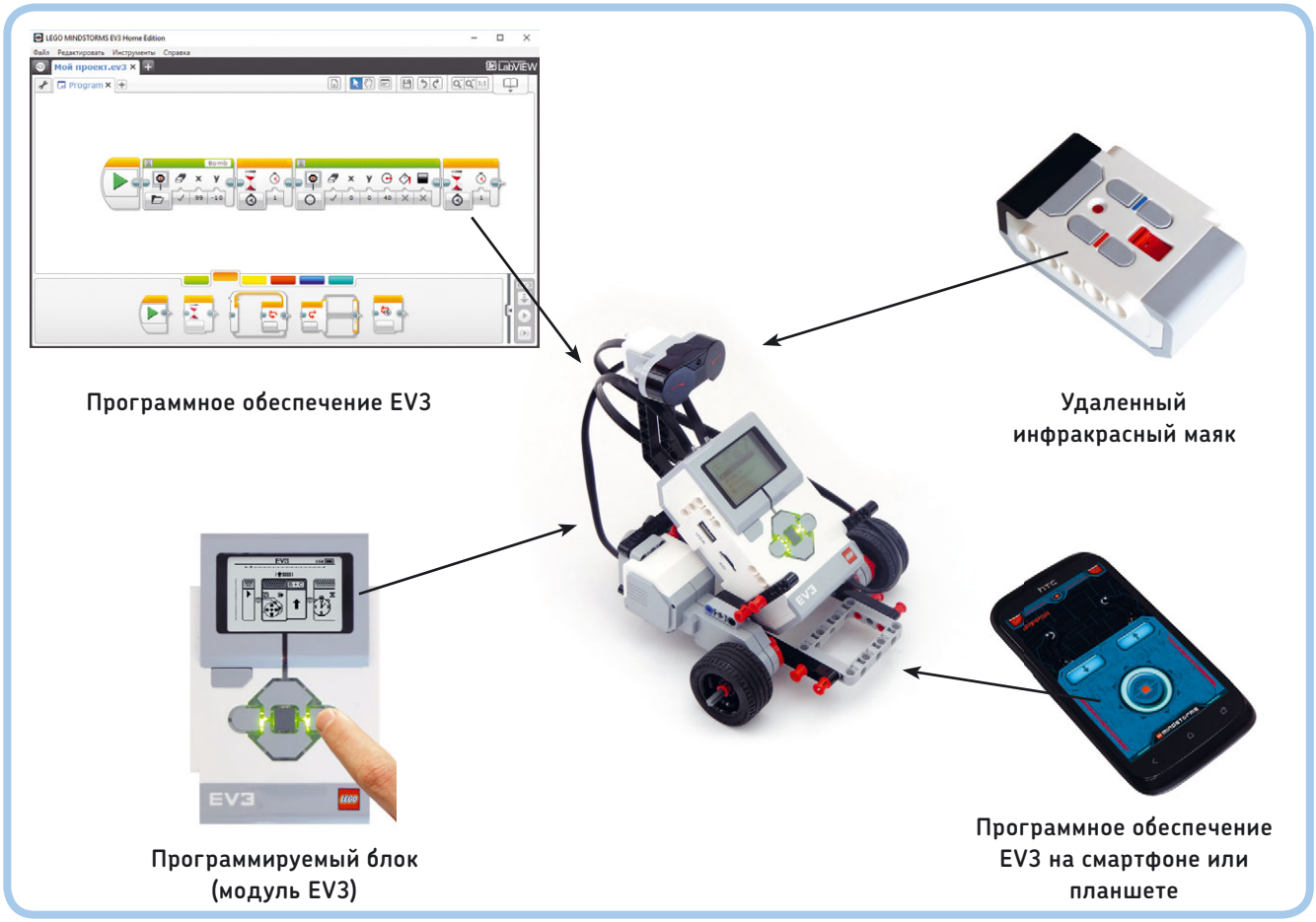


Рис. 1.7. Вы можете разрабатывать программы, позволяющие вашему роботу передвигаться автоматически, а также управлять им с помощью удаленного инфракрасного маяка или мобильного устройства со специальным программным обеспечением

1. Перейдите по ссылке **www.lego.com/ru-ru/mindstorms/**. Выберите желаемый язык, а затем щелкните мышью по ссылке **Загрузки** (Downloads). Прокрутите страницу и нажмите кнопку **Загрузить программное обеспечение (PC/MAC)** (Download Software (PC/MAC)), как показано на рис. 1.8. Обратите внимание, что вы можете выбрать любой язык интерфейса, но в этой книге описана русская версия.
2. На следующей странице выберите операционную систему (рис. 1.9). Если вы используете операционную систему Windows XP/Vista/7/8/10, нажмите кнопку **Windows** и сохраните установочный файл на компьютер. Если вы используете операционную систему macOS версии 10.6 или более поздней, нажмите кнопку **OS X**.

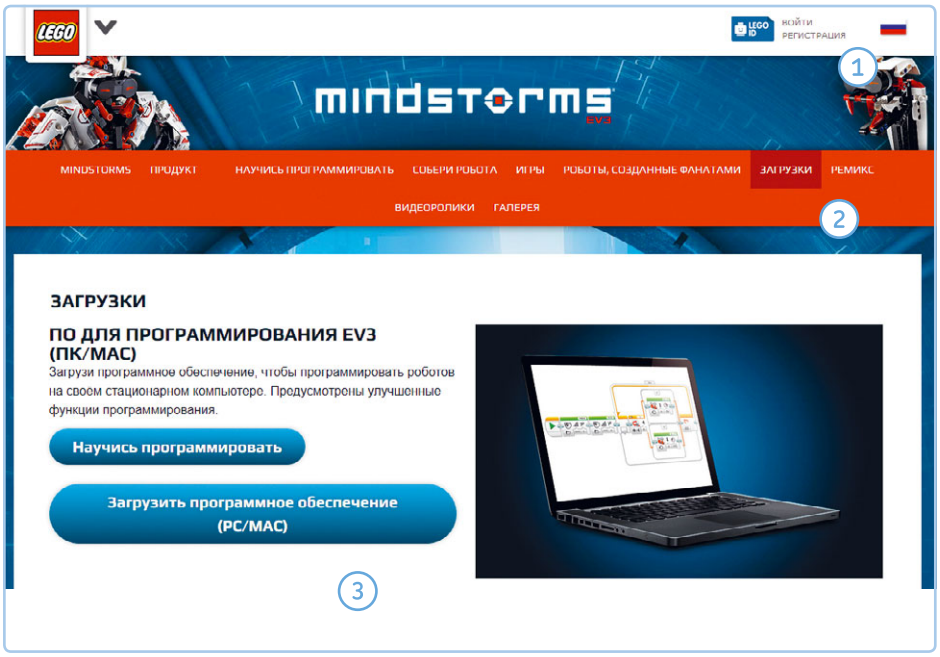


Рис. 1.8. Страница загрузки на веб-сайте LEGO MINDSTORMS EV3. Здесь вы тоже можете загрузить дополнительный контент, например руководство пользователя

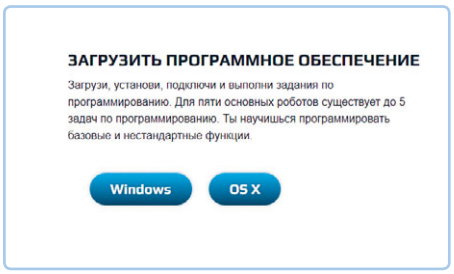


Рис. 1.9. Выберите используемую операционную систему, Windows или macOS

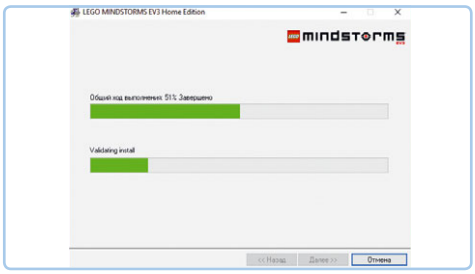


Рис. 1.10. Процесс установки. Запустите программу установки, дважды щелкнув мышью по файлу, который вы скачали

ПРИМЕЧАНИЕ **Если загрузка занимает много времени — файл имеет размер около 600 Мб, — вы можете перейти к главе 2 и начать конструировать! Вернитесь на эту страницу, когда загрузка будет завершена.**

3. В операционной системе Windows дважды щелкните мышью по файлу, который вы только что скачали, и установите программное обеспечение в соответствии с инструкциями, отображенными на экране (рис. 1.10). В операционной системе macOS дважды щелкните мышью по файлу с расширением *.dmg*, который вы скачали, а затем дважды щелкните мышью по появившемуся пакету. Для установки программного обеспечения следуйте инструкциям на экране.
4. После завершения установки и после того, как вы перезагрузили компьютер при появлении соответствующего запроса, вы найдете ярлык **LEGO MINDSTORMS EV3 Home Edition** на рабочем столе. Дважды щелкните по нему мышью, чтобы запустить программу. С этого момента доступ к Интернету не требуется.

ПРИМЕЧАНИЕ **Чтобы обновить программное обеспечение, просто скачайте последнюю версию и установите ее, используя те же шаги. Вам не нужно деинсталлировать текущую версию.**

Заключение

Теперь, когда у вас есть все, что нужно, чтобы собрать и запрограммировать первого робота, пришло время конструирования. Из главы 2 вы узнаете больше о модуле EV3, моторах и удаленном инфракрасном маяке, по мере того как будете собирать своего первого робота.

2

Конструирование первого робота

Из главы 1 вы узнали, что роботы EV3 состоят из моторов, датчиков и программируемого модуля EV3. Чтобы было проще понять, как работает каждый из компонентов, в этой главе мы применим только некоторые из них.

В этой главе вы будете использовать модуль EV3 и два больших мотора для сборки передвижного робота под названием EXPLOR3R, который изображен на рис. 2.1. На него вы установите инфракрасный датчик для дистанционного управления. После того как закончите сборку, вы узнаете, как управлять роботом с помощью кнопок модуля EV3 и удаленно.

Рис. 2.1. EXPLOR3R перемещается на двух передних колесах и опорном заднем колесе

Применение инструкций по сборке

Набор LEGO MINDSTORMS EV3 содержит множество балок и осей, которые имеют разную длину. Чтобы помочь вам найти правильную деталь, их длины указаны в инструкции на разных этапах сборки (рис. 2.2).

Чтобы определить длину балки, просто подсчитайте количество отверстий в ней (на схеме длина балки обозначена квадратиком с числом 11). Чтобы определить длину оси, положите ее рядом с балкой и подсчитайте количество отверстий, которые она покрывает (на рисунке длина оси обозначена кружком с цифрой 3).

При соединении балок или других деталей с помощью штифтов не забудьте выбрать правильный штифт, обращая внимание на его цвет, как показано на рис. 2.3. Это важно, поскольку гладкие штифты вращаются свободно — они нужны для шарнирных соединений, — а штифты с выступами блокируют вращение — они необходимы для создания неподвижных конструкций.

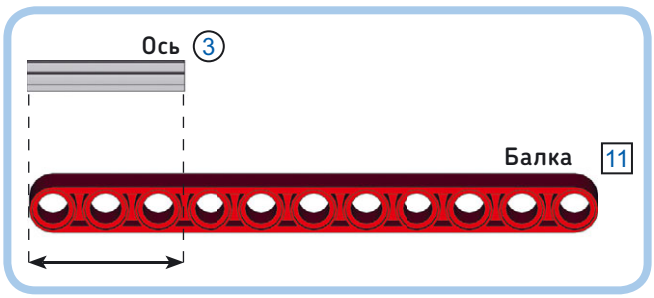


Рис. 2.2. Балки и оси бывают разной длины, так что не забудьте выбрать правильную деталь в процессе сборки. Длину можно определять так, как показано на этом рисунке, или использовать диаграмму на внутренней стороне обложки

Сборка EXPLOR3R

Чтобы начать собирать эту модель, подберите необходимые детали — они показаны на рис. 2.4. Затем соберите робота по рисункам на следующих страницах.

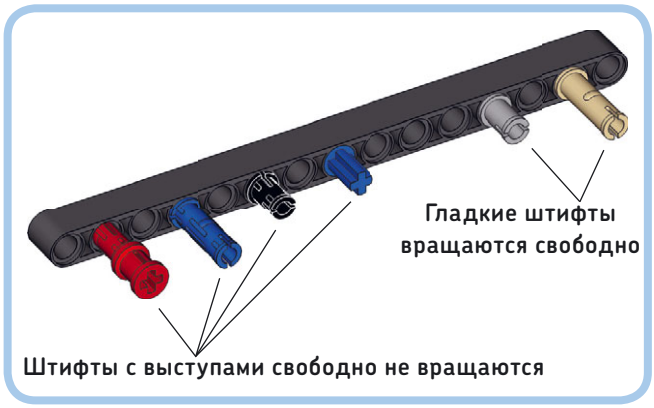


Рис. 2.3. Набор MINDSTORMS EV3 содержит штифты с выступами и гладкие штифты. В процессе сборки по инструкциям, приведенным в этой книге, выбирайте штифты правильного типа, основываясь на их цвете

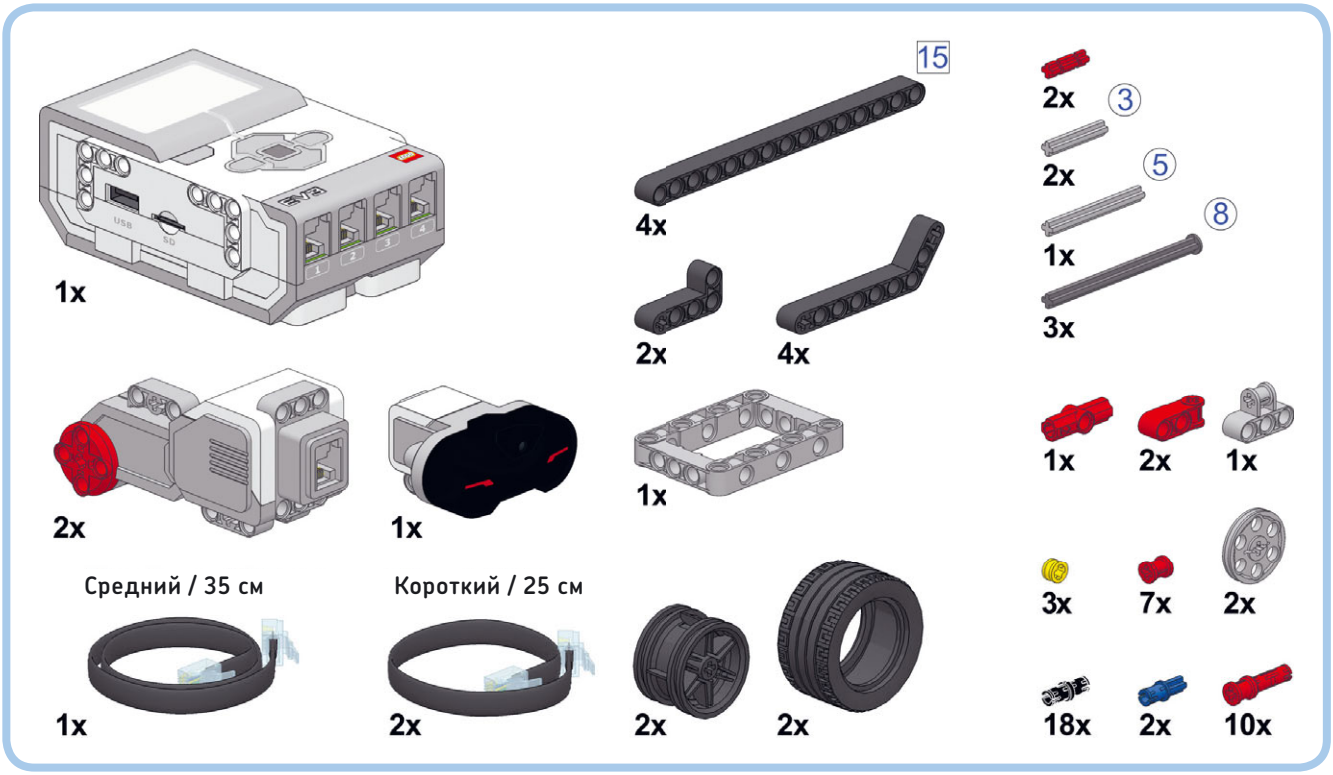
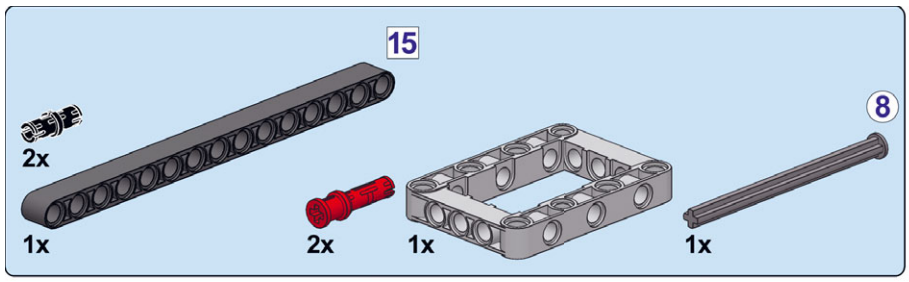
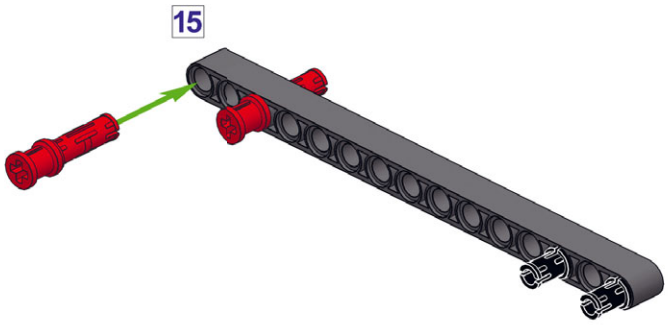


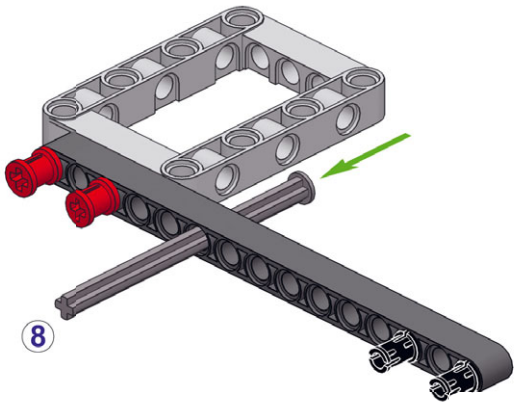
Рис. 2.4. Список деталей для EXPLOR3R

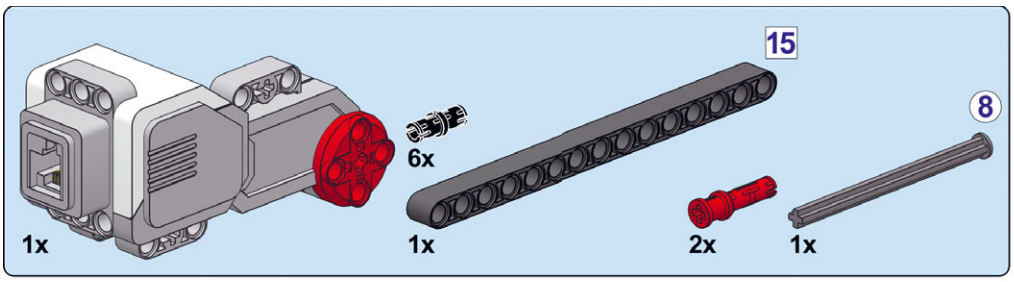


1

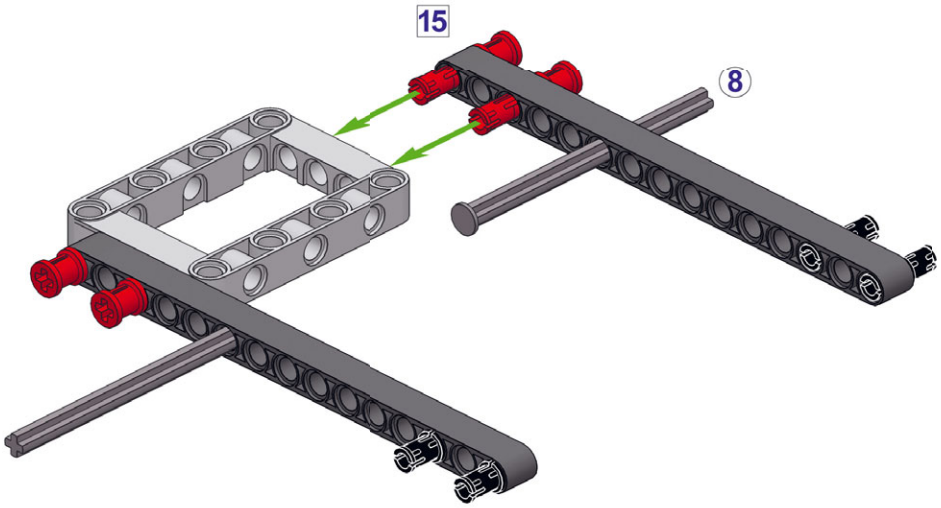


2

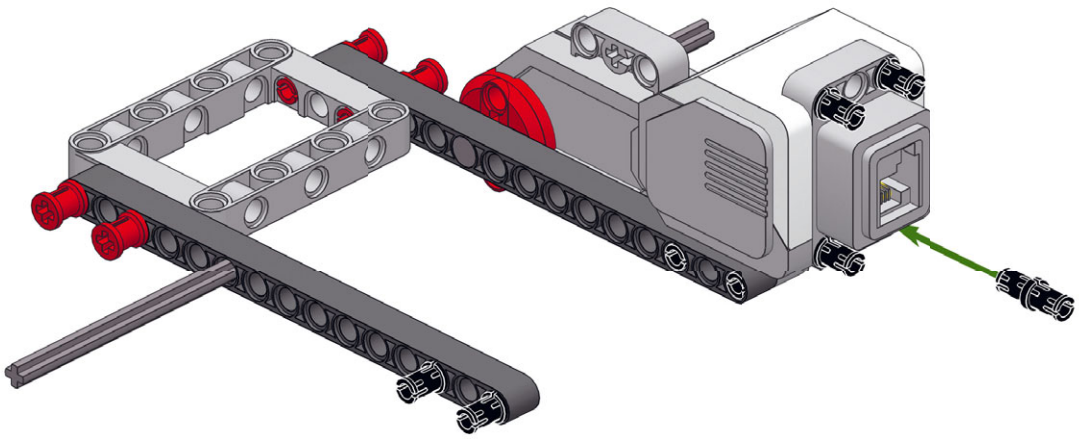


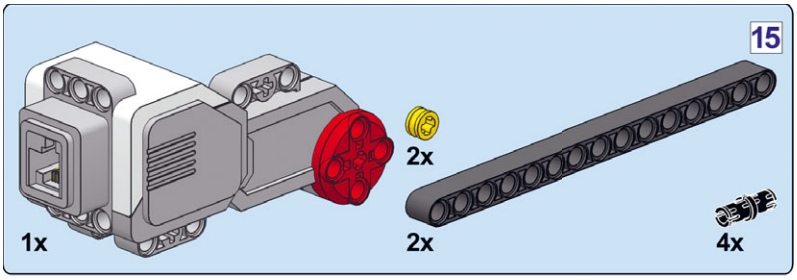


3

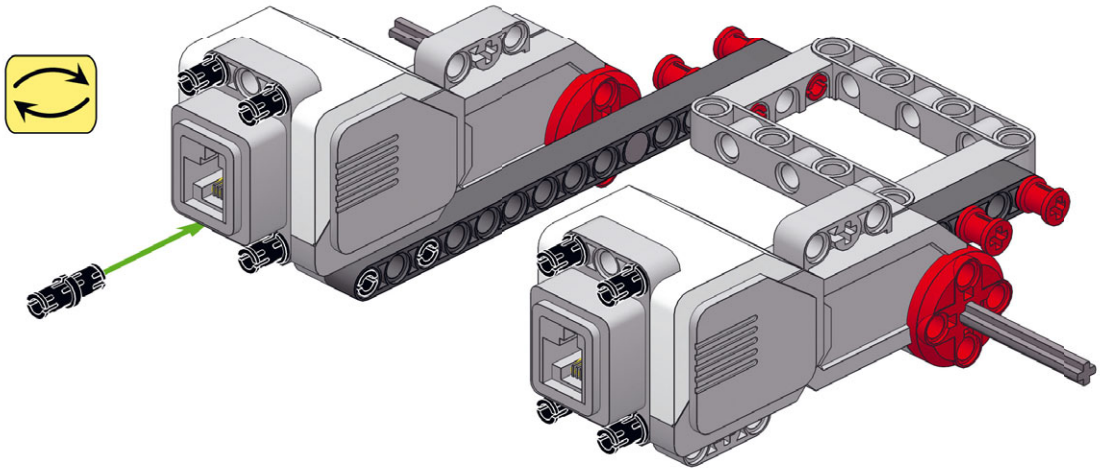


4

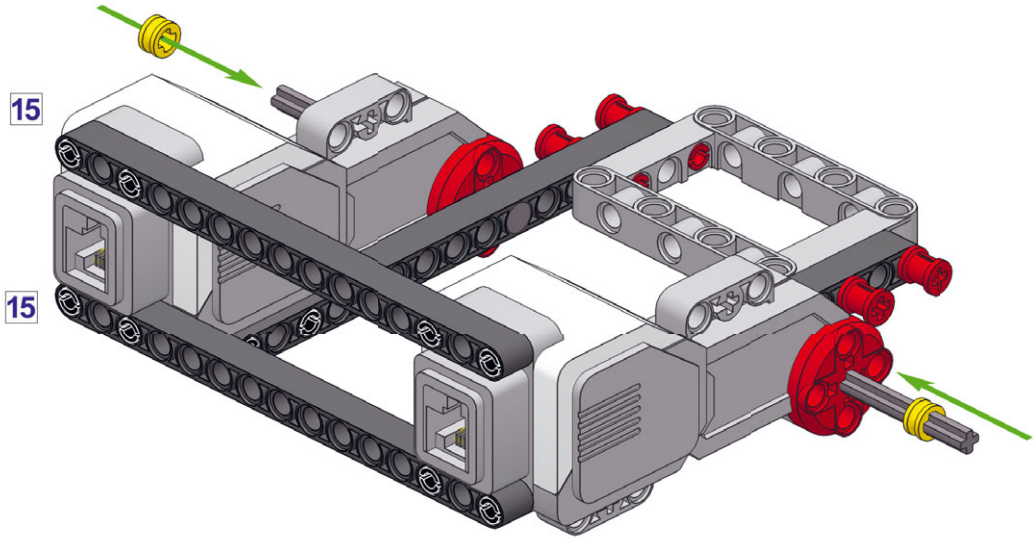


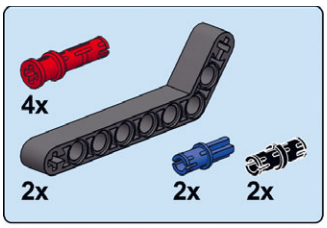


5

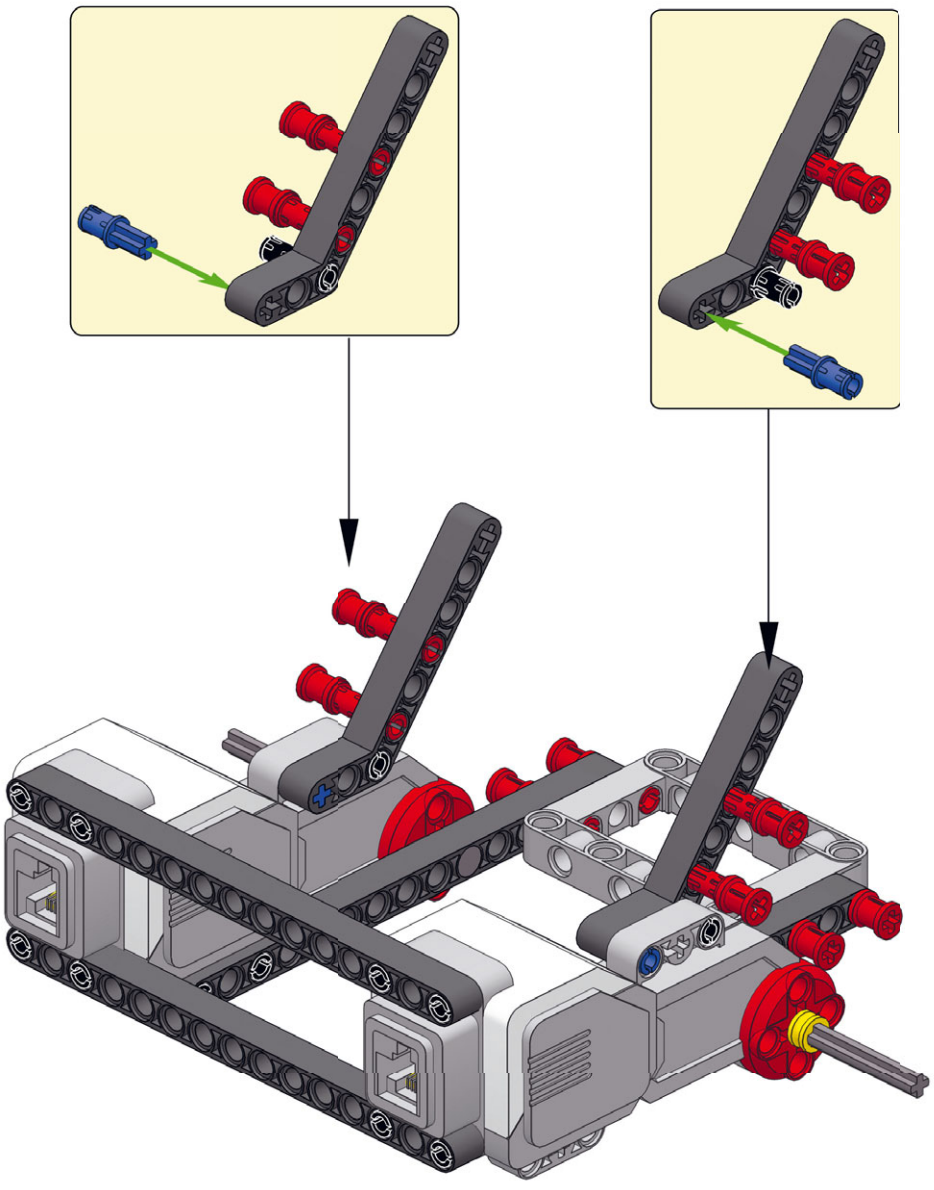


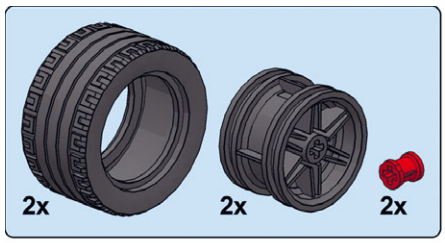
6



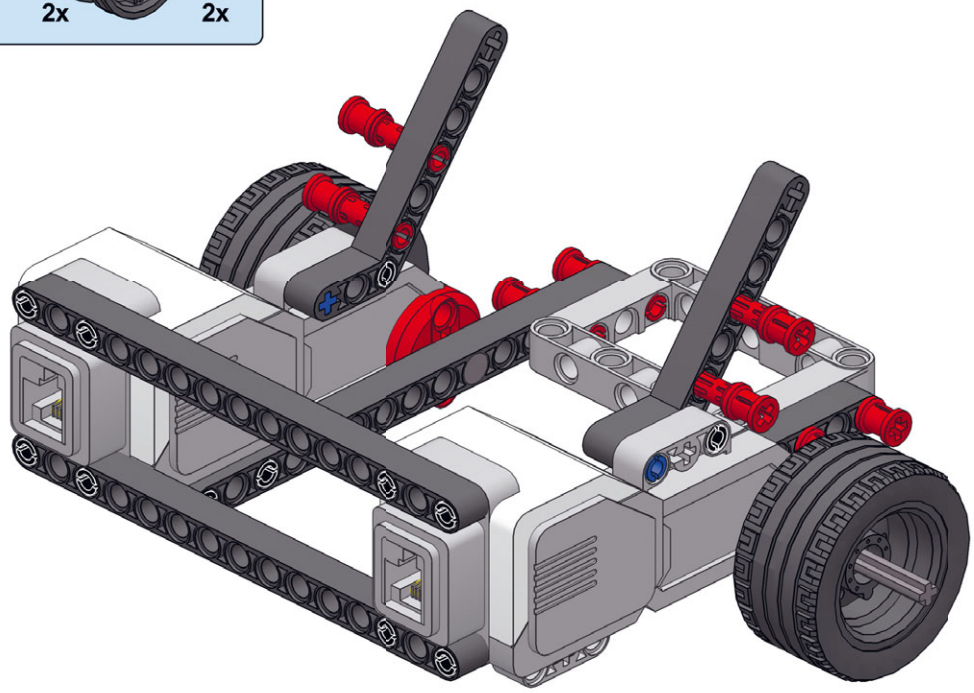


7

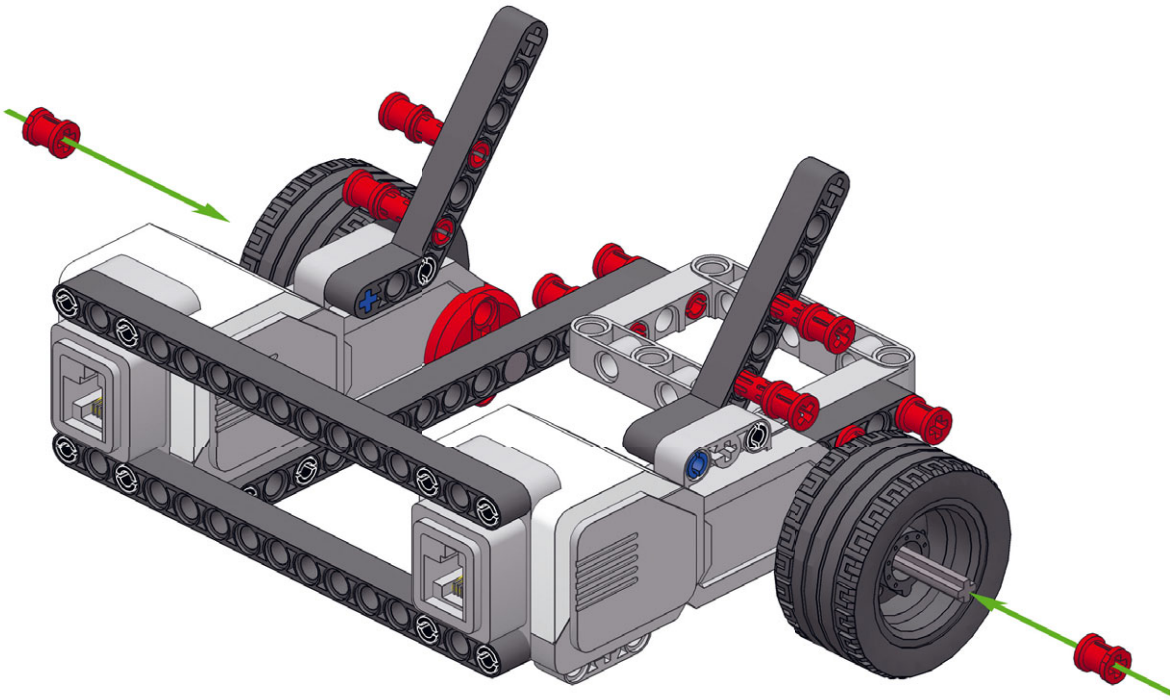


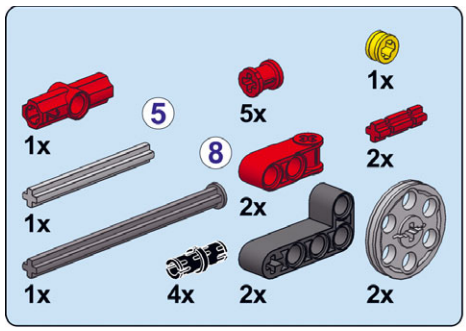


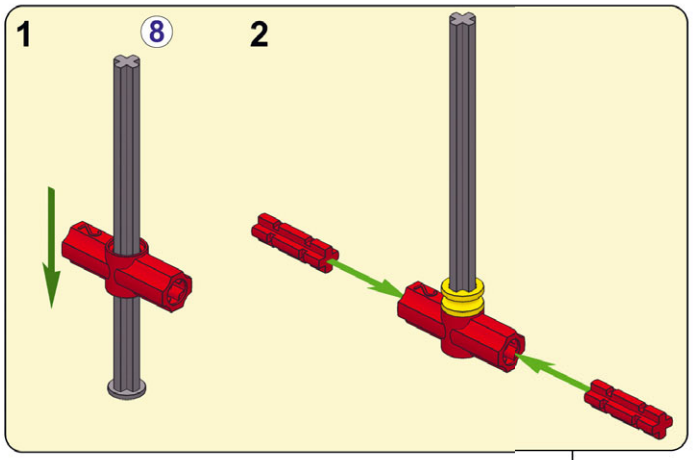
8

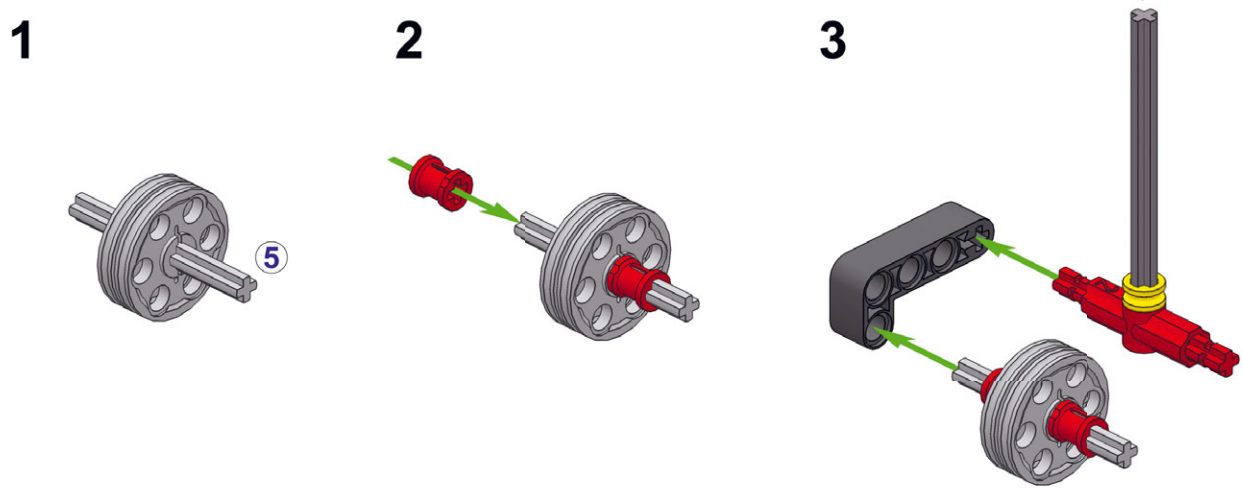


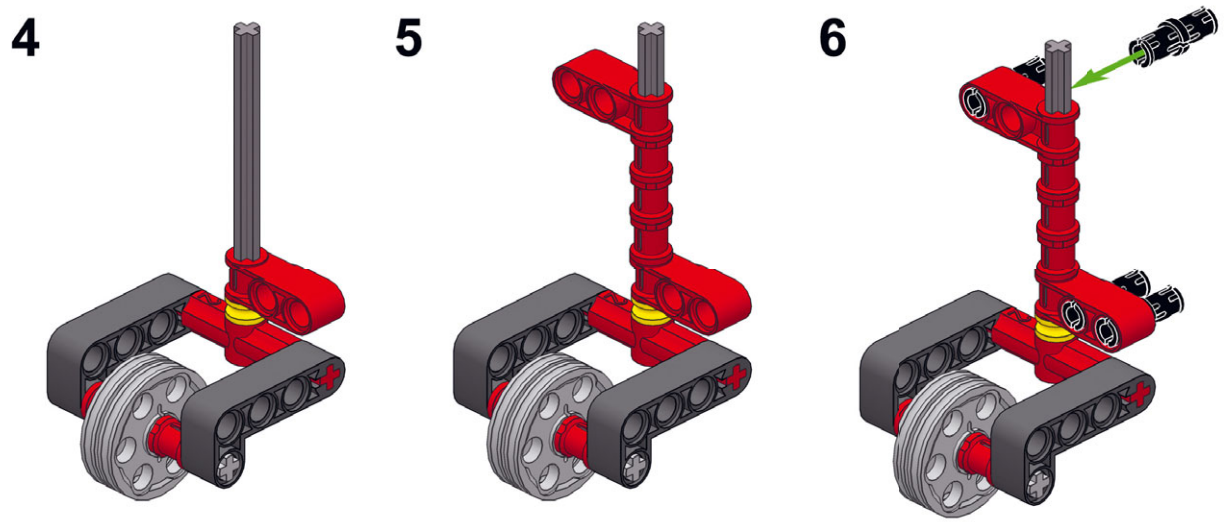
9



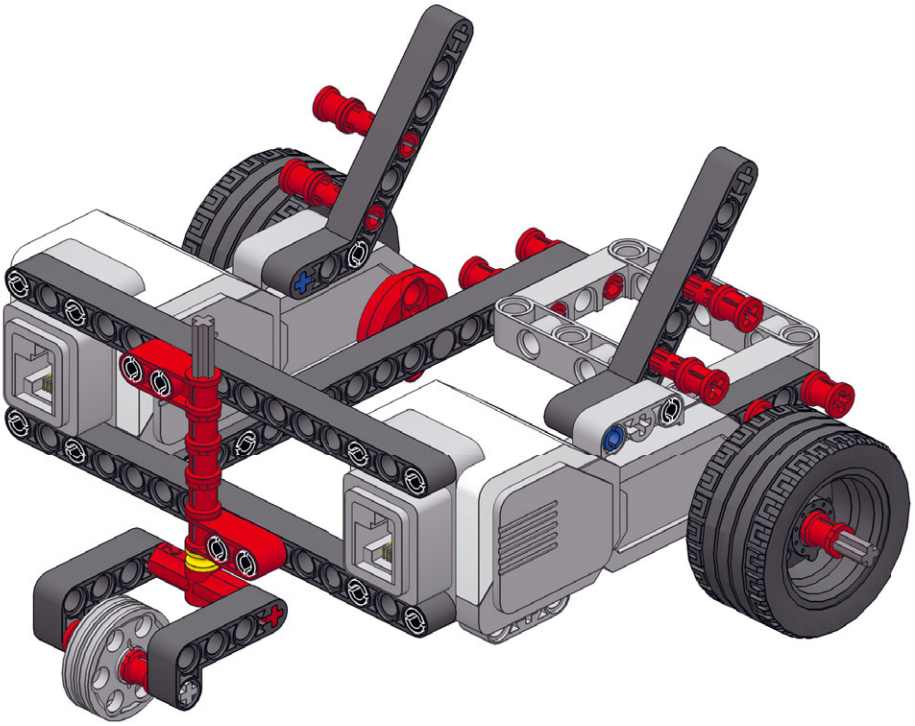

5
1x
5x
1x
8
2x
2x
1x
4x
2x
2x


1
8
2

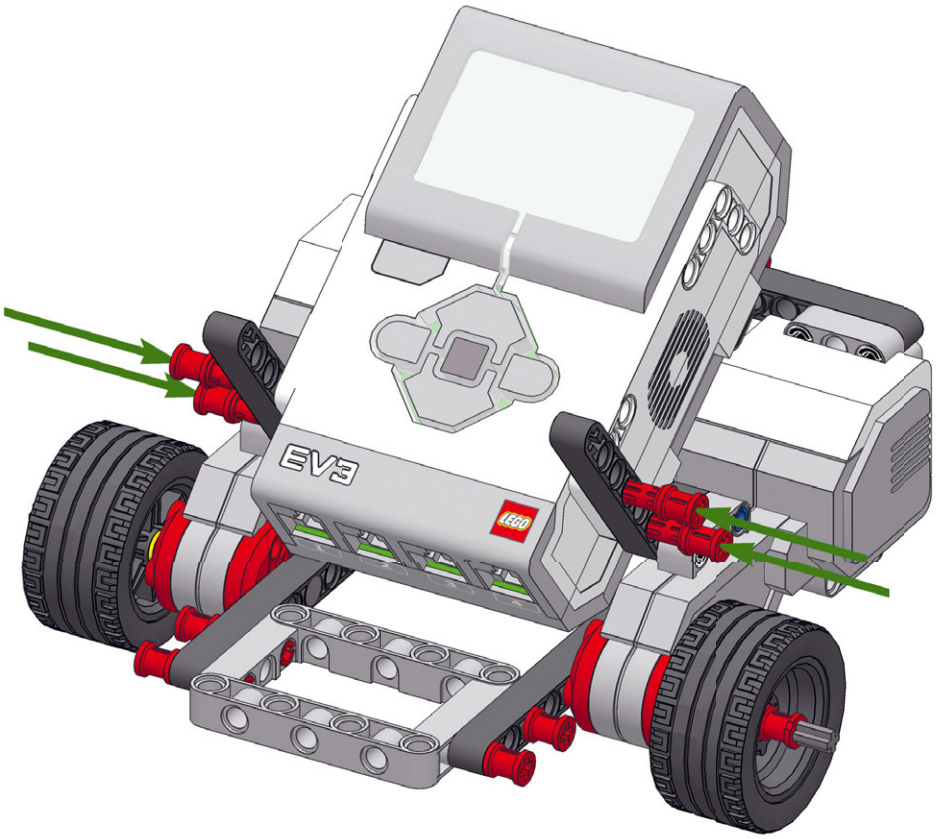

1
5
2
3

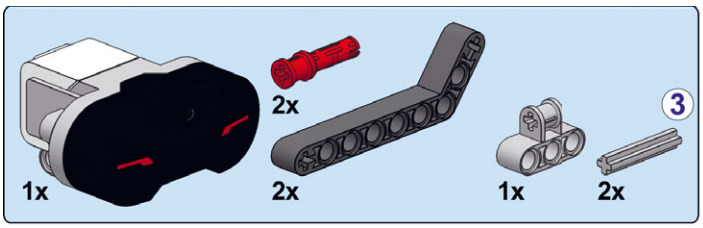

4
5
6

10

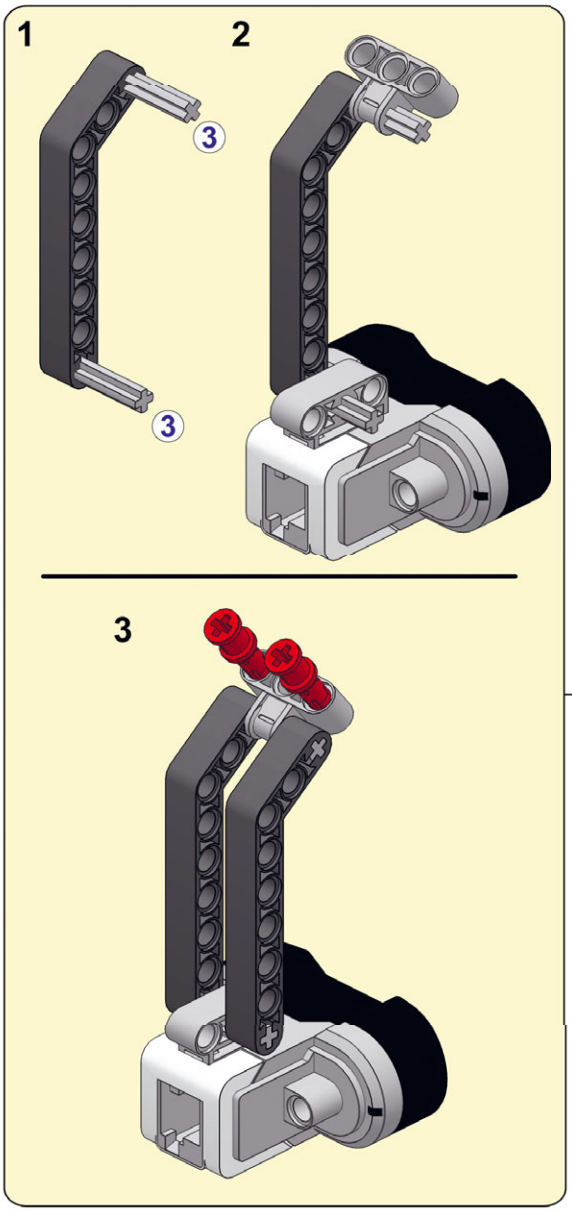


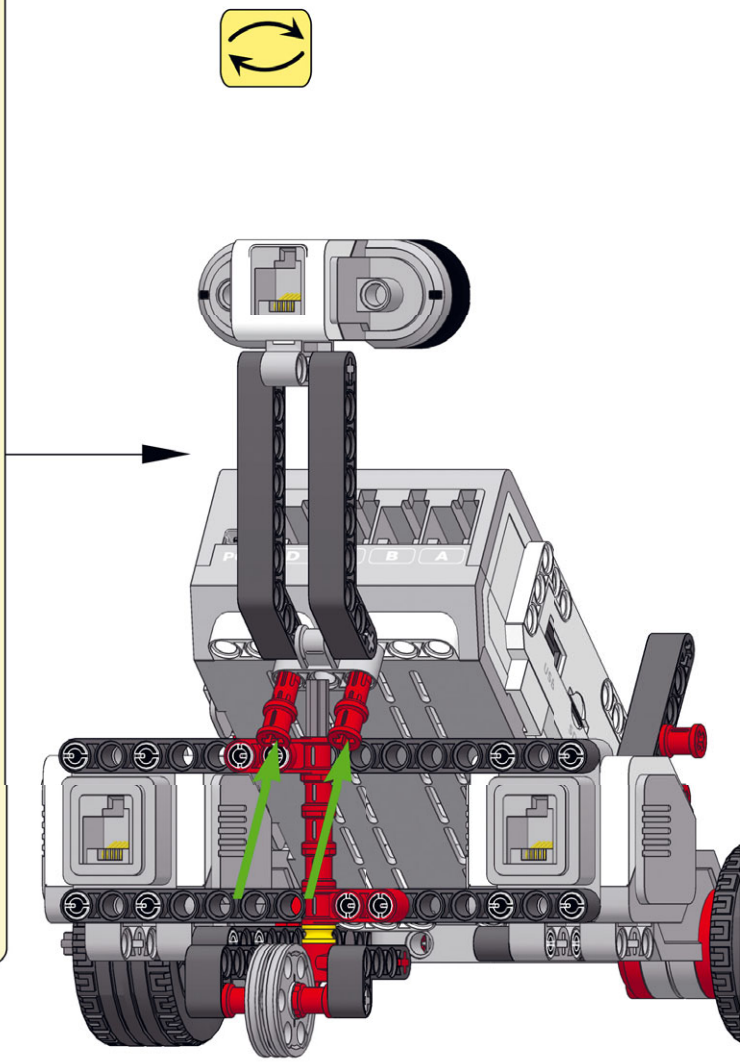
11



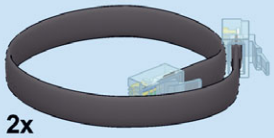


12



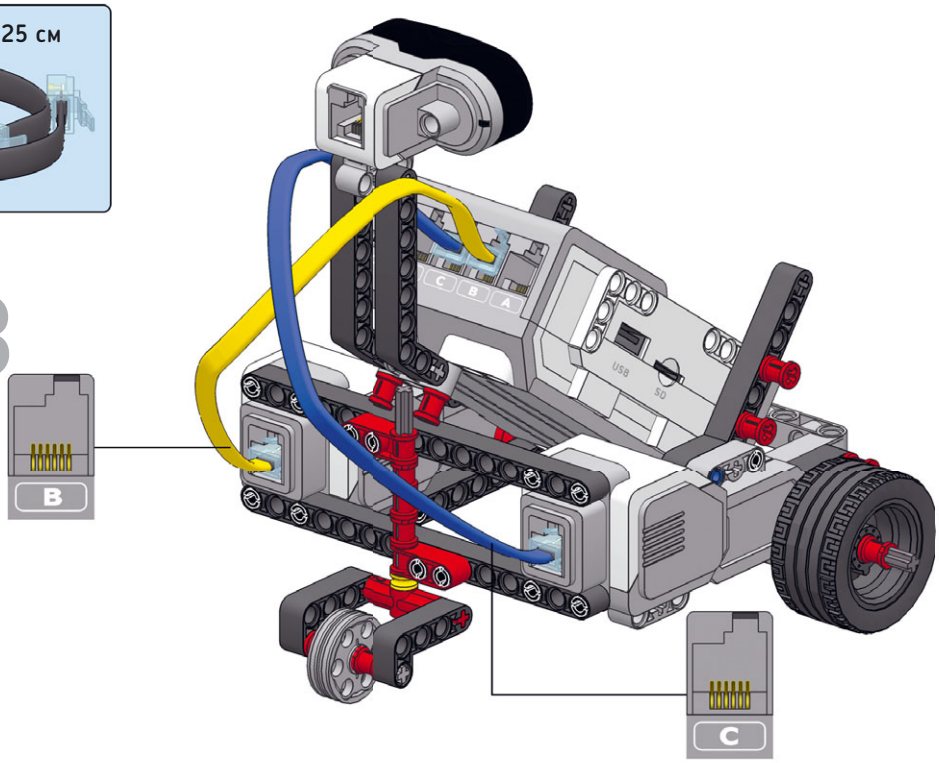


Короткий / 25 см

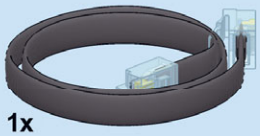


2x

13

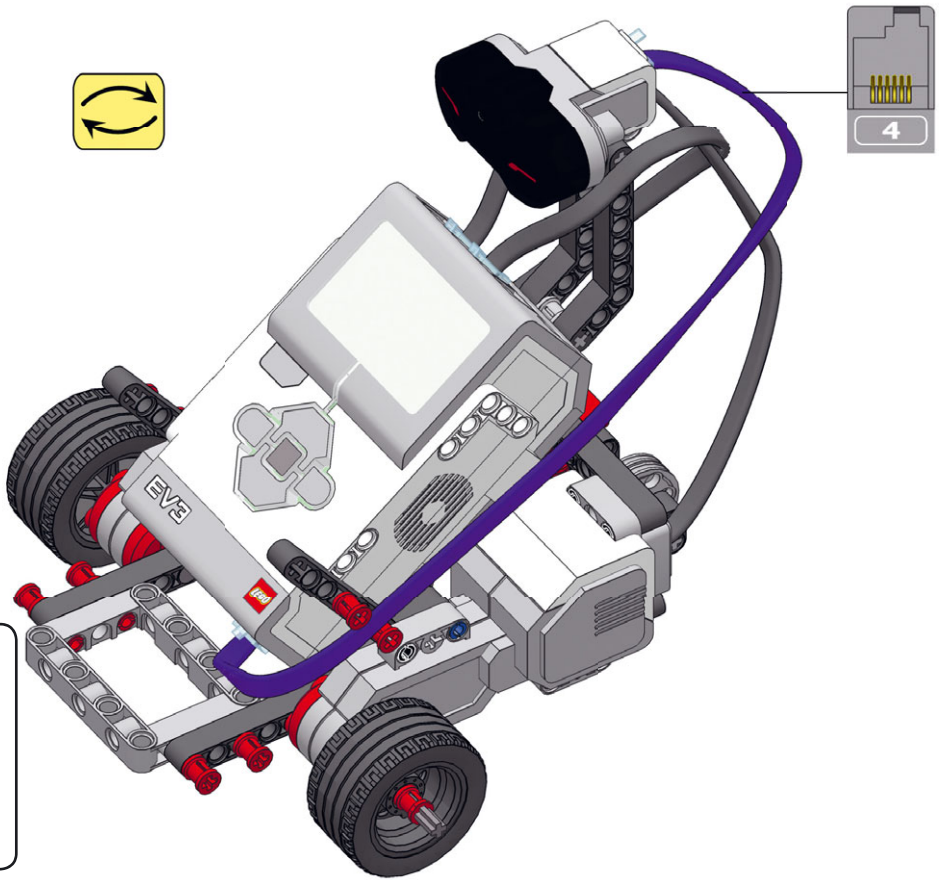


Средний / 35 см



1x

14



Подключите кабели, как показано на рисунке. Кабели на самом деле не окрашены, они выделены в инструкции, чтобы вам было проще понять, как подключить каждый кабель.

Входные/ выходные порты и кабели

Поздравляем — вы закончили сборку EXPLOR3R!

Теперь давайте поговорим о проводах, которые вы только что подсоединили к модулю EV3. Вы подключили два больших мотора к портам вывода EV3, помеченным буквами **B** и **C**. Оба больших и средний моторы всегда должны подключаться к *выходным* портам — **A**, **B**, **C** или **D**, как показано на рис. 2.5. Датчики должны подключаться к *входным* портам — 1, 2, 3 или 4. Датчики будут подробно описаны во второй части книги.

Набор MINDSTORMS EV3 содержит кабели трех типов: четыре коротких кабеля (25 см), два кабеля средней длины (35 см) и один длинный кабель (50 см). Всегда прокладывайте кабели вокруг вашего робота так, чтобы они не мешали никаким вращающимся деталям, и следите, чтобы они не касались поверхности, по которой робот передвигается.

Модуль EV3 оборудован двумя USB-разъемами. Тот, который расположен в верхней части блока и помечен буквами **PC** (рис. 2.5), используется для передачи программ с компьютера на робота. Другой USB-порт, находящийся сбоку, предназначен для подключения внешних устройств, таких как модуль Wi-Fi, к EV3. Слот для карты памяти MicroSD, расположенный рядом с боковым USB-портом, позволяет расширять объем встроенной памяти, ограниченный 4 Мб. Объема памяти встроенного хранилища будет достаточно для всех задач в этой книге.

Управление модулем EV3

Прежде чем перейти к программированию, описанному в главе 3, попробуйте с помощью кнопок на модуле EV3 (рис. 2.6) перемещаться по меню и запускать сохраненные программы.

*Рис. 2.5. Моторы подключаются к выходным портам, а датчики — к входным. Порт USB, который помечен буквами **PC**, используется для передачи инструкций с компьютера на модуль EV3*

Рис. 2.6. Помимо экрана, модуль EV3 оборудован кнопками с подсветкой

*Рис. 2.7. Включение модуля EV3 с помощью центральной кнопки открывает меню с четырьмя вкладками. Вкладка **Выполнить последние** (Run Recent), находящаяся справа, содержит список программ, которые использовались последними*

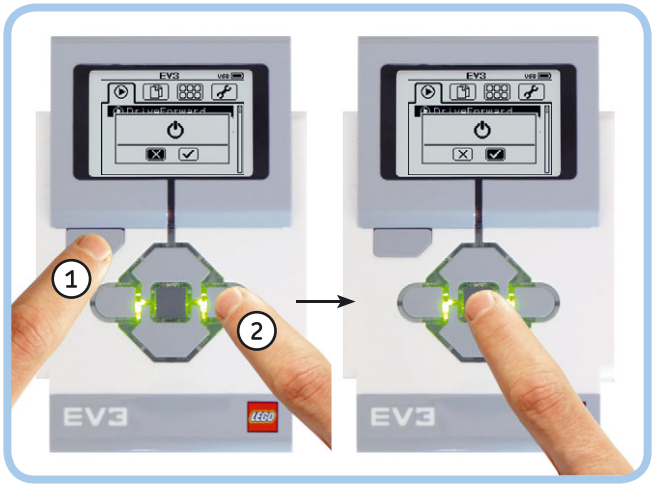


Рис. 2.8. Выключение модуля EV3

Включение/выключение модуля EV3

Чтобы включить модуль EV3, нажмите центральную кнопку, как показано на рис. 2.7. Индикатор состояния модуля EV3 окрасится в красный цвет. После завершения запуска, примерно через 30 секунд, цвет сменится на зеленый, и вы увидите на экране меню с четырьмя вкладками. Вы научитесь пользоваться ими в последующих главах. Каждая вкладка содержит определенный набор файлов или функций. Перечислим их слева направо.

Выполнить последние (Run Recent): эта вкладка содержит программы, которые запускались последними.

Навигация по файлам (File Navigation): эта вкладка отображает папки для каждого программного проекта, который вы передали на модуль EV3. В каждой папке вы найдете программы и связанные с ними файлы, например звуковые.

Приложения модуля (Brick Apps): эта вкладка содержит приложения для получения информации с датчиков и управления моторами вручную или дистанционно.

Настройки (Settings): эта вкладка позволяет установить пользовательские настройки, такие как включение режима Bluetooth и уровень громкости звука.

Чтобы выключить модуль EV3, вернитесь к меню **Выполнить последние** (Run Recent), и нажмите кнопку «Назад». Когда вы увидите значок выключения питания, выберите вариант ✓, чтобы выключить модуль EV3, либо вариант × для отмены (рис. 2.8). Если вы выключаете модуль EV3, чтобы заменить батарейки, обязательно подождите, пока погаснет красный световой индикатор

состояния, иначе вы потеряете все программы, переданные на модуль EV3 с того момента, как вы его включили.

Выбор и запуск программ

Вы можете перейти на любую из четырех вкладок с помощью кнопок «Влево» и «Вправо». Нажатие кнопки «Назад» возвращает вас к вкладке **Выполнить последние** (Run Recent). Используя кнопки «Вверх» и «Вниз», вы можете выделить нужный элемент на текущей вкладке. Чтобы выбрать нужный элемент, нажмите центральную кнопку.

Роботы EV3 начинают выполнять действия, когда вы выбрали и запустили программу, переданную на модуль EV3. Вы еще этого не сделали, но можете запустить пробную программу, которая уже записана в памяти модуля EV3 и называется Demo. Чтобы протестировать нашего робота EXPLOR3R, запустите эту программу, перейдя на вкладку **Навигация по файлам** (File Navigation) и выбрав программу Demo, как показано на рис. 2.9.

Если вы собрали конструкцию правильно, ваш робот должен издать несколько звуков, проехать вперед, а затем повернуть налево два раза, а также отобразить пару глаз на экране. Зеленый индикатор состояния во время работы программы будет мигать. Чтобы прервать выполнение программы, нажмите кнопку «Назад». Теперь, так как вы уже запустили программу, она появится на вкладке **Выполнить последние** (Run Recent).

ПРИМЕЧАНИЕ **Демонстрационная программа сделана с использованием программного обеспечения модуля. Программы, которые вы создадите на своем компьютере, вы будете запускать точно так же.**

Дистанционное управление роботом

После того как сборка робота завершена, очень важно проверить его технические функции, прежде чем начать программирование. Это нужно для того, чтобы выявить такие проблемы, как отсутствие подключения кабелей или зубчатых колес.

Вы можете вручную управлять моторами робота с помощью приложений Motor Control и IR Control, как показано на рис. 2.10. Приложение Motor Control позволяет запускать моторы с помощью кнопок на модуле EV3. Приложение IR Control позволяет управлять роботом с помощью удаленного инфракрасного маяка. Выберите пункт **IR Control** на вкладке **Приложения модуля** (Brick Apps) и используйте инфракрасный маяк, чтобы робот начал двигаться (см. рис. 2.10). Помимо того что это способ управлять вашим роботом дистанционно, это к тому же очень весело!

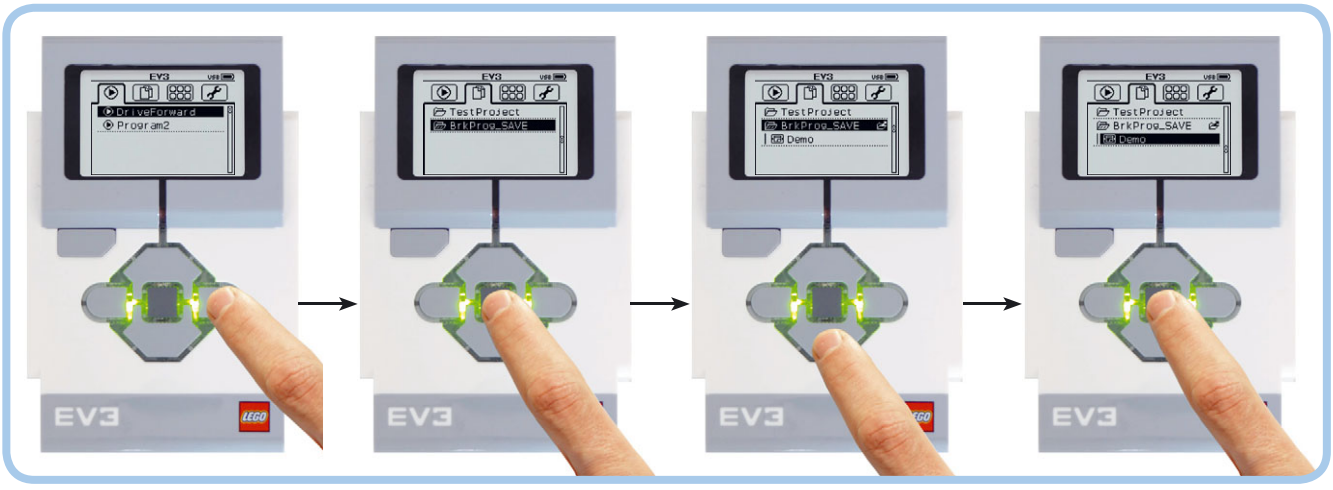


*Рис. 2.9. Для запуска программы Demo перейдите на вкладку **Навигация по файлам** (File Navigation), выберите папку **BrkProg_Save** и откройте ее с помощью центральной кнопки. Далее выберите пункт **Demo** с помощью кнопки «Вниз» и нажмите центральную кнопку. Собственные программы вы тоже найдете на вкладке **Навигация по файлам** (File Navigation). На рисунке вы можете увидеть мой проект под названием TestProject*

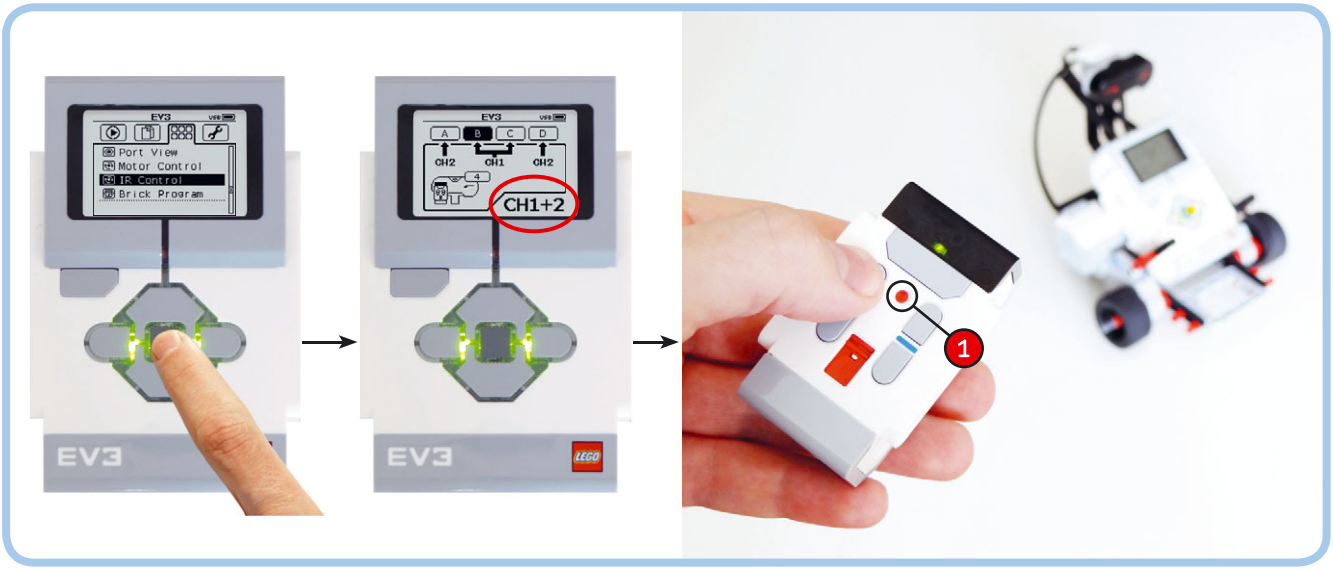


Рис. 2.10. Для активации режима дистанционного управления перейдите на вкладку ***Приложения модуля*** *(Brick Apps) и выберите пункт* ***IR Control****. Если в правом нижнем углу экрана символы CH1+2 не отображаются, нажмите центральную кнопку еще раз. В этой конфигурации вы управляете моторами, подключенными к портам B и C, при помощи инфракрасного маяка, настроенного на канал 1. Канал 1 определяет красный переключатель, находящийся в верхней части*

ПРИМЕЧАНИЕ **Инфракрасный датчик играет роль приемника инфракрасного сигнала удаленного маяка. Вы не можете использовать удаленный маяк без датчика. Приложение IR Control требует подключения этого датчика к входному порту 4, как это было указано при сборке EXPLOR3R.**

Заключение

В этой главе вы научились обращаться с двумя основными компонентами робота: модулем EV3 и моторами. Когда вы запустили программу Demo, модуль EV3 включил моторы, которые привели робота в движение. В главах 3 и 4 вы узнаете, как функционируют эти программы, а также как разработать собственные программы с помощью программного обеспечения EV3. Во второй части книги мы еще вернемся к инфракрасному датчику и удаленному инфракрасному маяку.

3

Создание и модификация программ

Следующее, что понадобится вам после сборки робота, это программа. Например, такая, которая заставит двигаться EXPLOR3R вперед, а затем повернуть налево или направо.

В этой главе вы узнаете, как создавать и редактировать программы на основе программного обеспечения EV3. Конечно, вы можете создавать программы для ваших роботов без компьютера, используя возможности модуля EV3. Но такие программы обладают довольно ограниченным функционалом и не позволят вам получить доступ ко многим возможностям модуля EV3.

Обзор возможностей программирования на модуле EV3 вы найдете в приложении Б.

Рис. 3.1. Робот подключен к компьютеру с помощью USB-кабеля, входящего в комплект. Используйте USB-порт в верхней части модуля EV3, как показано на рисунке

Первая небольшая программа

Сейчас вы создадите небольшую программу и передадите ее на модуль EV3 робота. Для этого выполните следующие действия.

1. Подключите робота к компьютеру с помощью USB-кабеля, который входит в комплект (рис. 3.1), и включите модуль EV3. Вам нужно будет подключать робота к компьютеру каждый раз, когда потребуется загрузить в него программу.
2. Запустите программу EV3, дважды щелкнув мышью по ярлыку LEGO MINDSTORMS EV3 Home Edition на рабочем столе. После загрузки программы вы должны увидеть начальный экран — *лобби*, где вы будете создавать новые проекты и открывать существующие.
3. Создайте проект новой программы, щелкнув мышью по кнопке + в левой верхней части экрана, как показано на рис. 3.2.

ПРИМЕЧАНИЕ **Если вы видите всплывающее окно, которое содержит сообщение с запросом на обновление версии встроенного программного обеспечения модуля EV3, выполните действия, описанные в разделе «Обновление встроенного программного обеспечения EV3» приложения А.**

4. Щелкните мышью по блоку **Рулевое управление** (Move Steering) и установите его так, как показано на рис. 3.3. Помните, что программа по сути — это список инструкций, которые должен выполнять робот. Этот блок представляет собой инструкцию, программирующую робота двигаться вперед.
5. Теперь нажмите кнопку **Загрузить и запустить** (Download and Run) (рис. 3.4). Компьютер загрузит созданную программу в модуль EV3 вашего робота, который после этого начнет двигаться вперед. Чтобы загрузить и запустить эту программу еще раз, нажмите кнопку **Загрузить и запустить** (Download and Run) еще раз.

Если ваш робот переместился вперед на небольшое расстояние, вы создали свою первую программу. Поздравляю!

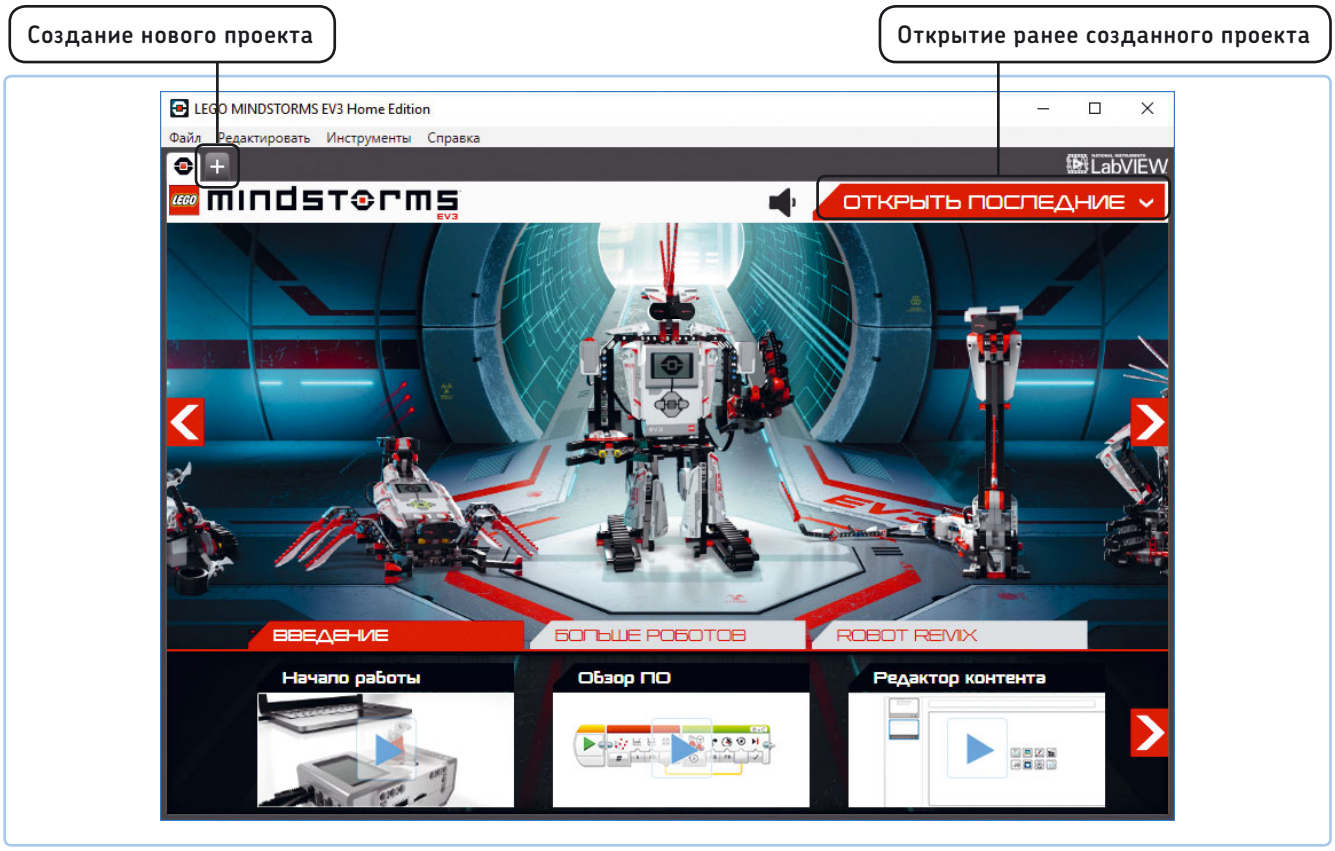


Рис. 3.2. Программное лобби EV3. Нажмите на символ +, чтобы начать новый программный проект

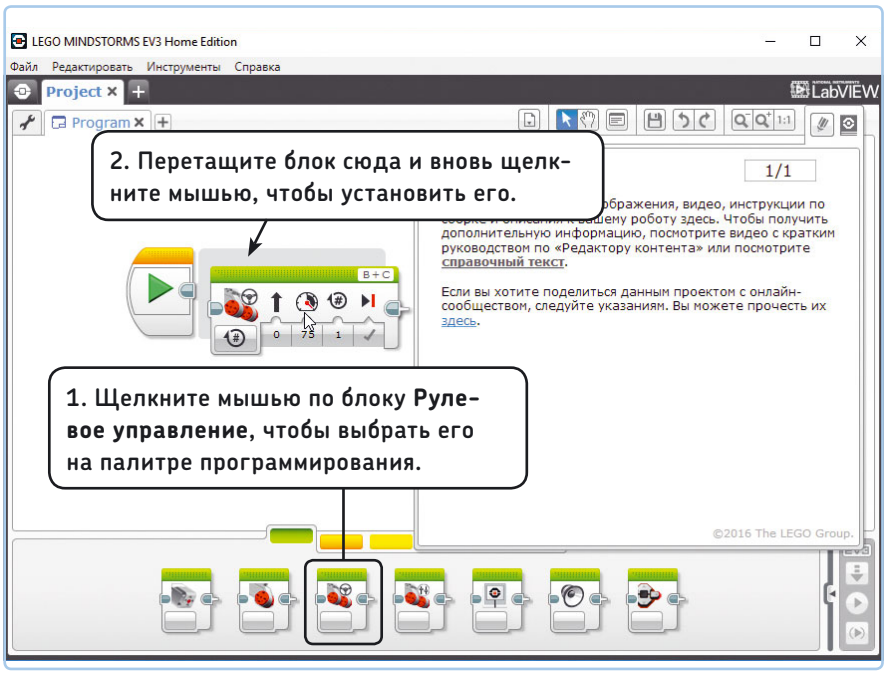


*Рис. 3.3. Установка программного блока в проекте. При перетаскивании и отпускании мыши новый блок автоматически стыкуется с оранжевым блоком **Начало** (Start), который по умолчанию уже находится в новом проекте*

Рис. 3.4. Загрузка программы в память модуля EV3 робота и ее запуск. Если буквы EV3 на палитре подсвечены красным, значит, робот успешно подключен к компьютеру

ПРИМЕЧАНИЕ **Если вы не можете загрузить программу в модуль EV3 и символ EV3 на экране компьютера окрашен в серый (EV3), а не красный (EV3) цвет, проблема может заключаться в USB-соединении. Попробуйте отключить USB-кабель и подключить его снова. Если это не поможет, выключите и снова включите модуль EV3. В приложении А вы найдете дополнительную информацию о решении проблем.**

Разработка простых программ

Прекрасно, ваш робот двигается, но как это происходит? В следующих разделах мы изучим различные части программного обеспечения EV3, чтобы вы разобрались, как создавать и изменять базовые программы, прежде чем переходить к разработке более сложных программ.

После запуска программы ваш экран должен выглядеть так, как показано на рис. 3.5. Далее я опишу каждый отмеченный компонент интерфейса.

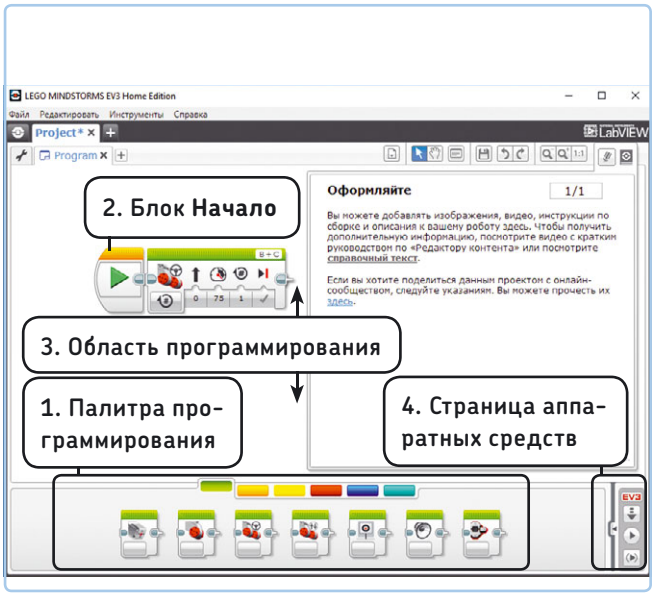


Рис. 3.5. Окно программного обеспечения EV3 состоит из нескольких компонентов. Вы использовали каждый из отмеченных компонентов, когда создавали свою первую программу

1. Палитра программирования

Все программы для модуля EV3 состоят из *программных блоков*. Каждый блок побуждает робота совершать какое-либо действие, например двигаться вперед или издавать звук. Вы найдете блоки на *палитре программирования* (рис. 3.6).

Рис. 3.6. Палитра программирования

Существует несколько категорий блоков, которые находятся на разных вкладках и маркированы цветом. Вы научитесь использовать *блоки действий* (зеленые), *блоки управления операторами* (оранжевые) и собственные *блоки (контейнеры)*, которые называются *моими* (светло-голубые) в главах 4 и 5. Взаимодействию с датчиками с помощью блоков управления операторами вы научитесь в части II этой книги.

Блоки датчиков (желтые) и *блоки операций с данными* (красные) обсуждаются в части V книги. По мере изучения книги вам будут встречаться некоторые из *блоков дополнений* (темно-синие).

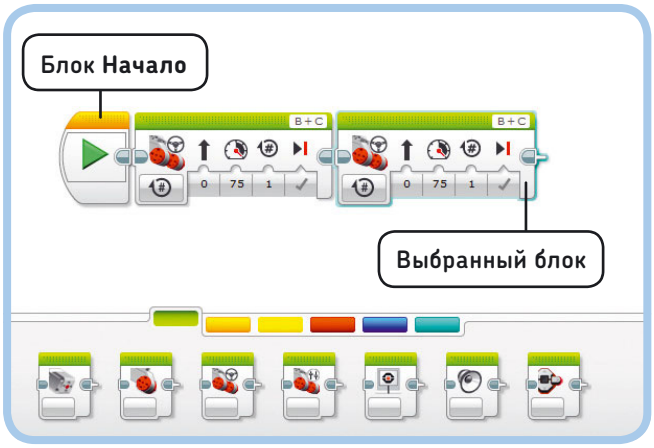


Рис. 3.7. После того как вы выбрали блок из палитры программирования, поместите его в область программирования. Если это первый блок в программе, поместите его сразу после блока ***Начало***

2. Блок Начало

Все программы всегда начинаются с блока **Начало** (Start). Когда вы выбираете первый блок из палитры программирования, вы присоединяете его к блоку **Начало** (Start), как показано на рис. 3.7. Робот выполняет команды блоков в созданной вами программе один за другим, слева направо, начиная с блока, расположенного следом за блоком **Начало** (Start).

Если вы случайно удалили блок **Начало** (Start), перетащите новую копию с оранжевой вкладки палитры программирования.

3. Область программирования

Вы будете создавать свои программы в *области программирования*. После того как вы поместили в эту область блок, вы можете передвинуть его, удерживая левую кнопку мыши. (Нажав и удерживая левую кнопку мыши на блоке, перетащите его.) Если выделить несколько блоков, вы можете сразу переместить все выбранные блоки с помощью мыши. Чтобы удалить блок из области программирования, выберите его, а затем нажмите клавишу **Del** на клавиатуре.

Рис. 3.8. Программные блоки обычно располагаются по прямой линии, но вы можете разместить их и каким-либо другим образом, подключая их друг к другу с помощью соединителей (вверху). Вы можете удалить соединитель, щелкнув мышью по его штекеру (внизу). Щелчок мышью по разъему первого блока удаляет соединитель и состыковывает блоки непосредственно. Щелчок мышью по штекеру соединителя у разъема второго блока удаляет соединитель, при этом блоки остаются на своих местах. Щелчок мышью по разъему второго блока не приводит к какому-либо результату

Как правило, вы будете размещать программные блоки по прямой, как показано на рис. 3.7, но иногда имеет смысл расположить их в ином порядке, чтобы избежать путаницы в области программирования. В этом случае вы должны соединять блоки с помощью специальных *соединительных шин*, или просто *соединителей*, как показано на рис. 3.8.

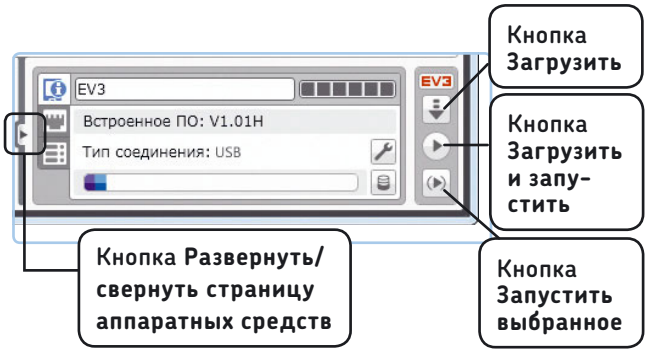


Рис. 3.9. Страница аппаратных средств

Блок, который не пристыкован к другому блоку и не подключен к нему с помощью соединителя, будет окрашен в серый цвет и не будет воздействовать на робота.

4. Страница аппаратных средств

Используйте страницу аппаратных средств для переноса программ в память модуля EV3, просмотра состояния EV3 и любых подключенных устройств, а также для настройки соединения между EV3 и вашим компьютером. Щелкните мышью по переключателю в виде треугольника, чтобы развернуть страницу аппаратных средств, как показано на рис. 3.9.

В этой книге я познакомлю вас со многими функциями, представленными на этой странице.

Загрузка и запуск программы

Чтобы перенести программу в память модуля EV3, убедитесь, что EV3 подключен к компьютеру, а затем нажмите кнопку **Загрузить и запустить** (Download and Run) на странице аппаратных средств. Робот должен издать звук, демонстрируя, что программа успешно передана, после чего она запустится автоматически. Программа прекратит работу после того, как закончится выполнение всех ее блоков.

После того как программа отправлена в память модуля EV3, у вас есть возможность запускать эту программу, даже если вы отключите USB-кабель. Ваши программы остаются в памяти модуля EV3, даже когда вы выключаете EV3, что позволяет запускать их каждый раз, когда захотите поиграть с роботом.

Запуск программы вручную

Когда программа прекращает работу или вы останавливаете ее нажатием кнопки «Назад» на модуле EV3, вы можете перезапустить ее вручную с помощью специальных кнопок EV3, о которых говорилось в главе 2. Все программы, которые вы загрузили в память модуля, вы найдете на вкладке **Навигация по файлам** (File Navigation) модуля EV3. Программы, которые вы запускали последними, отображаются на вкладке **Выполнить последние** (Run Recent).

ПРИМЕЧАНИЕ **Программы могут не отображаться на вкладке Выполнить последние (Run Recent), но вы все равно сможете найти их на вкладке Навигация по файлам (File Navigation).**

Загрузка программы без последующего запуска

Не всегда нужно запускать программу автоматически, сразу после того, как вы загрузили ее в память модуля. Например, если ваш робот находится на столе и вы программируете его на движение, он может просто упасть. Чтобы передать программу в EV3 без автоматического запуска, нажмите кнопку **Загрузить** (Download) на странице аппаратных средств. После того как программа завершит загрузку, о чем вы будете осведомлены соответствующим звуковым сигналом, отключите USB-кабель, а затем запустите программу с помощью элементов управления на модуле EV3.

Запуск выбранных блоков

Нажмите кнопку **Запустить выбранное** (Run Selected) для запуска только тех блоков, которые вы выделили в области программирования. Это может быть полезно для тестирования фрагмента большой программы. Чтобы выбрать несколько блоков, заключите их в выделение, перетаскивая мышь с нажатой кнопкой, либо, нажав и удерживая клавишу **Shift**, щелкайте мышью по очереди по блокам, которые вы хотите выбрать. Для отмены выбора щелкните мышью в любой свободной от блоков позиции области программирования.

ПРИМЕЧАНИЕ **Кроме кабеля USB, можно передавать программы в память модуля EV3 беспроводным способом через Bluetooth или Wi-Fi. В приложении А рассказывается, как использовать страницу аппаратных средств для установки беспроводных соединений такого типа.**

Проекты и программы

Когда робот будет собран, вы наверняка захотите создать для него несколько программ. Каждая программа определяет разное поведение робота, но все же имеет смысл держать соответствующие программы вместе *в проекте*. Далее вы узнаете, как управлять *файлами проектов* и программами, содержащимися в них, используя элементы управления, отмеченные на рис. 3.10.

5. Организация файлов

Ранее, когда вы разработали свою первую программу, вы фактически создали новый файл проекта с одной пустой программой в нем, как показано на рис. 3.11. Чтобы добавить еще одну новую программу в текущий проект, щелкните мышью по вкладке **+**, которая называется **Добавить программу** (Add program), как показано на рисунке.

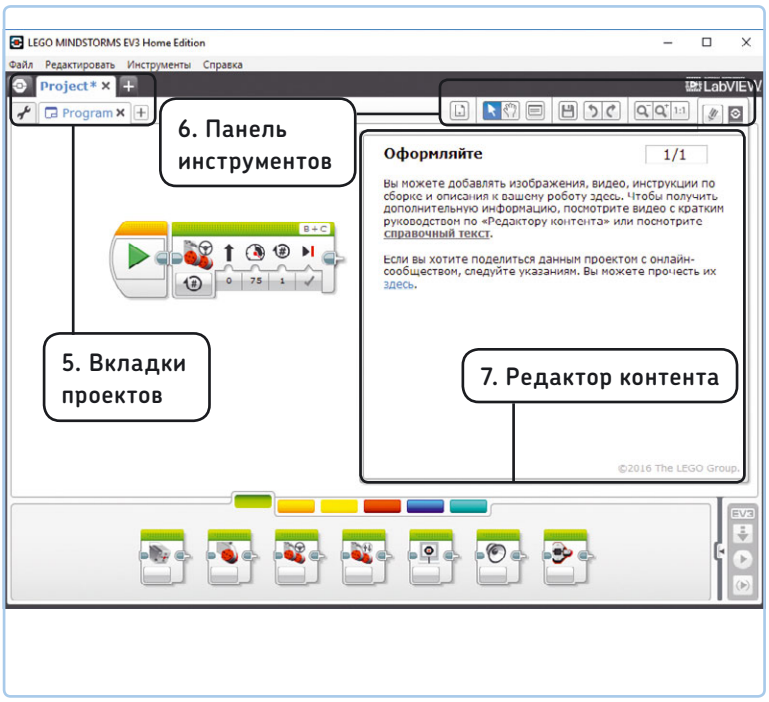


Рис. 3.10. Элементы для управления проектами и программами в них

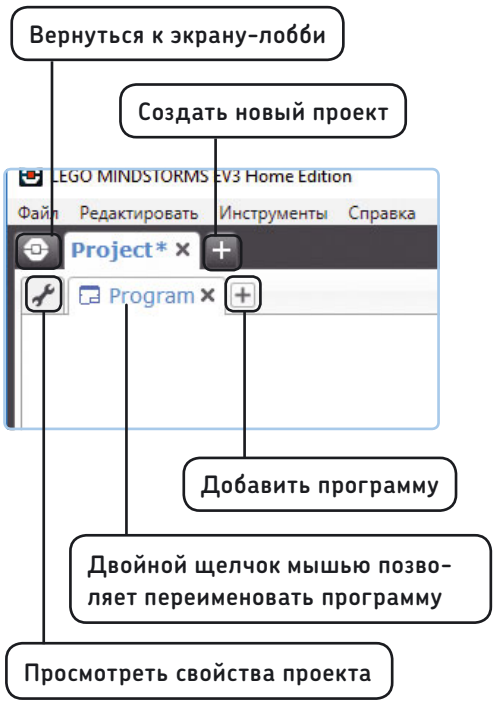


Рис. 3.11. Создание новых проектов и программ

Сохранение проектов и программ

Во время разработки программ нужно часто сохранять ваши проекты, чтобы не потерять выполненную работу. Чтобы сохранить сразу все программы в проекте, нажмите кнопку **Сохранить проект** (Save) на панели инструментов или сочетание клавиш **Ctrl+S**. Если вы сохраняете проект в первый раз, вам будет предложено выбрать имя для него. В этом случае введите MyFirstProject и нажмите кнопку **Сохранить** (Save).

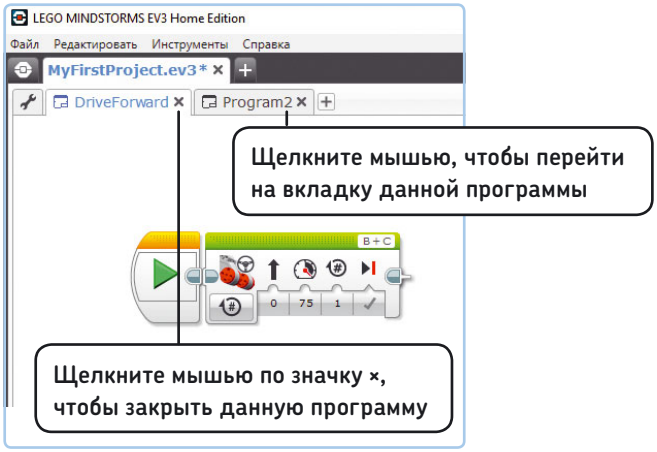


*Рис. 3.12. Вкладки нескольких программ в проекте. Символ * в имени MyFirstProject.ev3 * указывает на несохраненные изменения в вашем проекте*

Чтобы открыть проект, сохраненный ранее, выберите команду меню **Файл** ▸ **Открыть проект** (File ▸ Open Project) или используйте раскрывающийся список **Открыть последний** (Open Recent) на начальном экране-лобби (см. рис. 3.2) и найдите файл проекта (как правило, они хранятся в каталоге *\Users\ваше_имя\Documents\LEGO Creations\MINDSTORMS EV3 Projects*).

Чтобы закрыть программу или проект, щелкните мышью по значку × на соответствующей вкладке, как показано на рис. 3.12. Для переключения на другую открытую программу в текущем проекте просто перейдите на соответствующую вкладку. Чтобы вновь открыть закрытую программу, воспользуйтесь кнопкой **Список программ** (Program List) на панели инструментов (см. раздел 6. «Панель инструментов» далее в этой главе).

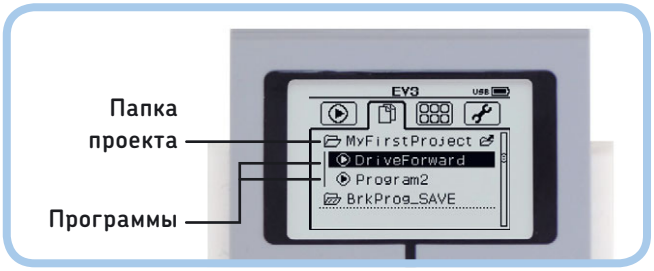


*Рис. 3.13. Программы хранятся в папках проектов на вкладке **Навигация по файлам**. Имя папки совпадает с именем проекта*

Переименование проектов и программ

Чтобы изменить название программы, дважды щелкните мышью по ее вкладке и введите новое имя. Например, я переименовал первую программу на рис. 3.12, присвоив ей название *DriveForward*. Чтобы изменить название *проекта*, вы должны создать новый проект, основанный на текущем, выбрав команду меню **Файл** ▸ **Сохранить проект как** (File ▸ Save Project As) и указав новое имя.

Выбирайте описательные имена для ваших проектов и программ, чтобы легко найти их на экране модуля EV3.

Поиск проектов и программ на модуле EV3

При нажатии кнопки **Загрузить** (Download) или **Загрузить и запустить** (Download and Run) завершенный проект передается в память модуля EV3, как показано на рис. 3.13. На экране модуля EV3, на вкладке **Навигация по файлам** (File Navigation), вы увидите отдельную папку для каждого проекта, содержащую все его программы, а также любые файлы, используемые в проекте, такие как изображения или звуки. Запуск программы осуществляется ее выбором и нажатием центральной кнопки.

Если вы вставили MicroSD-карту в модуль EV3, вы увидите папку с именем *SD_Card* на вкладке **Навигация по файлам** (File Navigation), в которой расположен каталог *MyFirstProject*.

ПРИМЕЧАНИЕ **Создание единого проекта с несколькими программами для каждого вашего робота — это неплохо. Но загрузка проекта в EV3 может занять длительное время, если ваш проект содержит много программ и особенно если включает в себя звуковые файлы.**

Изменение свойств проекта

Щелчком мышью по значку в виде гаечного ключа слева от вкладок программ в окне EV3 открывается страница **Свойства проекта** (Project Properties) (рис. 3.14). Здесь вы можете добавить информацию о вашем проекте (описание, фотографию или даже видеозапись) и поделиться проектом с другими людьми.

На странице **Свойства проекта** (Project Properties) перечислены все файлы, используемые в вашем проекте,

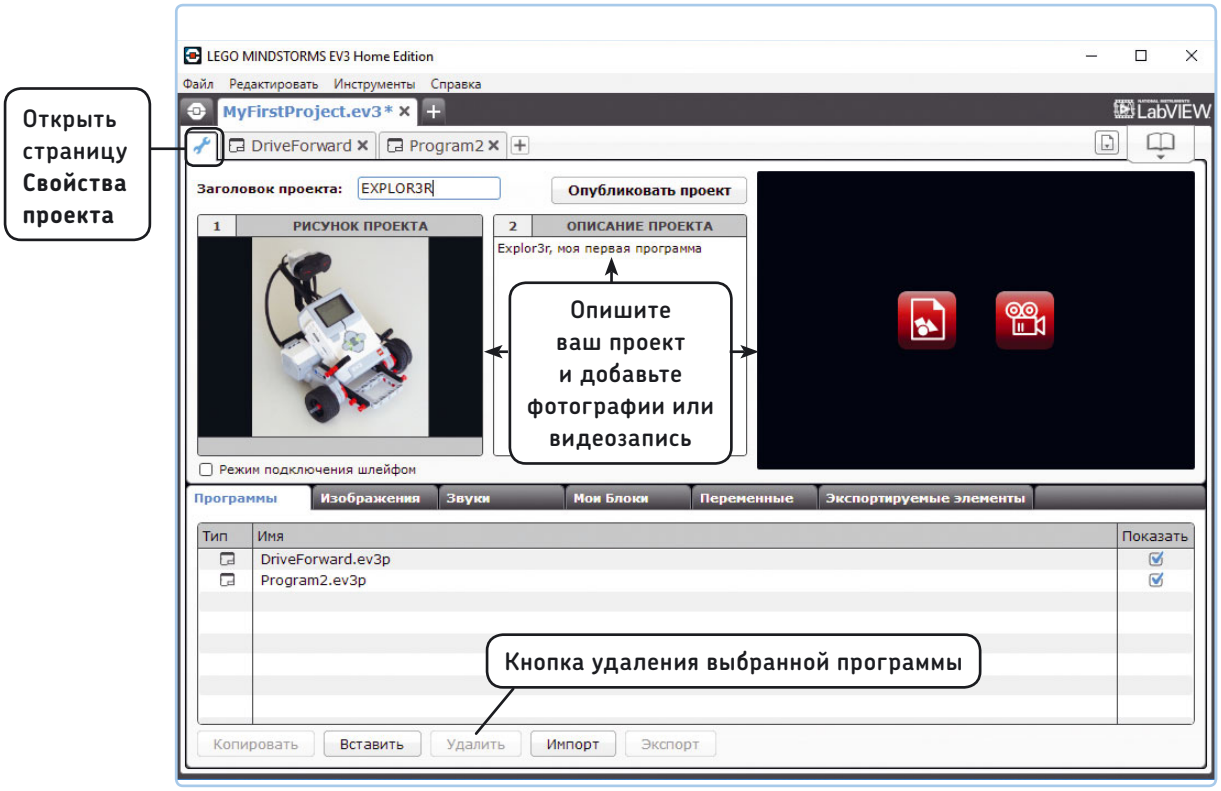


*Рис. 3.14. Страница **Свойства проекта**. Вы можете дополнить свой проект фотографиями и описанием или поделиться им с другими людьми. Дважды щелкните по имени программы в списке, чтобы открыть ее, или выберите ее и нажмите кнопку **Удалить** (Delete), чтобы удалить из проекта*

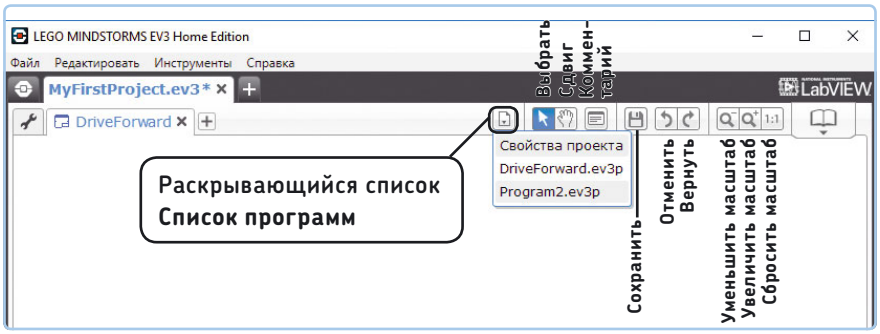


Рис. 3.15. Панель инструментов

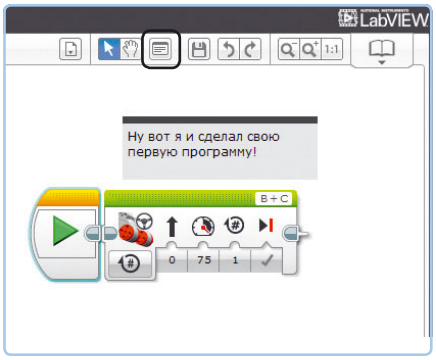


Рис. 3.16. Комментарии в области программирования

в том числе программы и звуки. Чтобы вновь открыть закрытую программу в проекте, дважды щелкните мышью по ее имени. Чтобы удалить программу из проекта, выберите ее и нажмите кнопку **Удалить** (Delete).

6. Панель инструментов

С помощью панели инструментов (рис. 3.15) вы можете открывать и сохранять программы из вашего проекта, отменять и возвращать отмененные изменения в вашей программе и использовать средства навигации по программе.

Использование инструментов выбора, сдвига и масштабирования

Если кнопка инструмента **Выбрать** (Select) на панели инструментов окрашена в синий цвет, как показано на рис. 3.15, вы можете использовать мышь, чтобы добавлять, передвигать и менять конфигурацию программных блоков в области программирования. Перемещаться по области программирования можно с помощью клавиш ←, ↑, ↓ и → на клавиатуре. Бóльшую часть времени вы будете использовать именно инструмент **Выбрать** (Select).

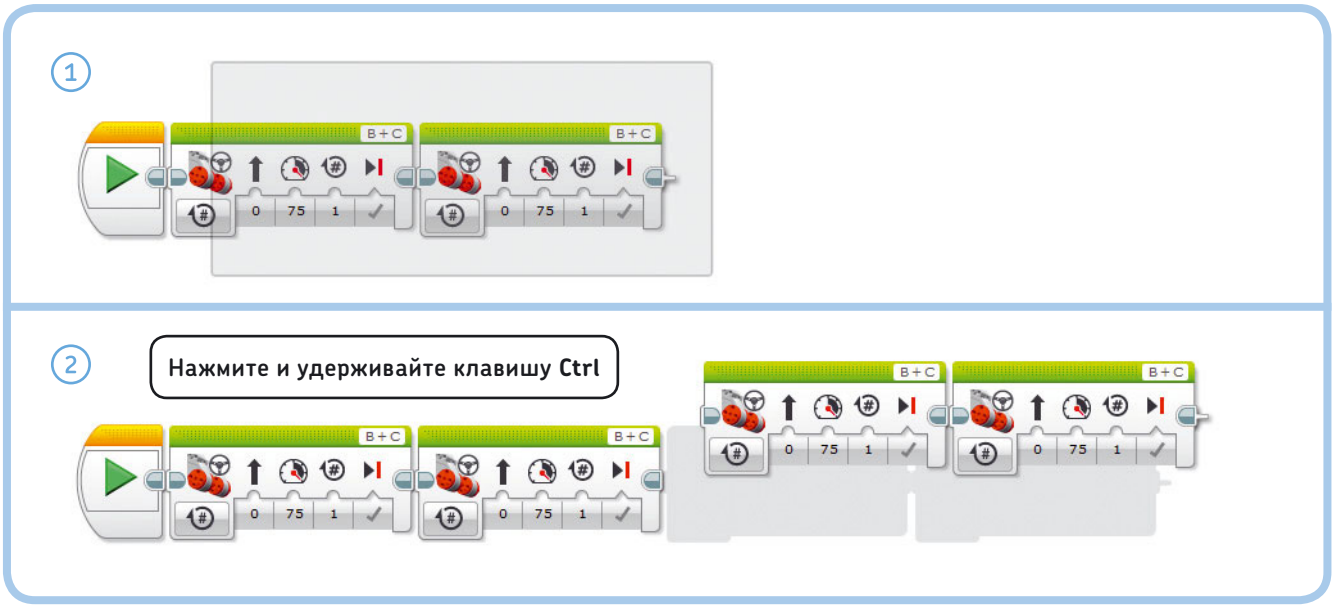


*Рис. 3.17. Дублирование последовательности блоков. 1. Удерживайте нажатой левую кнопку мыши, заключите в выделение последовательность блоков, которую вы хотите дублировать. 2. Удерживая нажатой клавишу **Ctrl**, перетащите блоки мышью в новую позицию. В операционной системе macOS используйте клавишу ⌘*

При выборе инструмента **Сдвиг** (Pan) по области программирования вы можете перемещаться с помощью мыши. Это особенно полезно, если вы разрабатываете сложные программы, которые не отображаются целиком в окне программы. Для перехода к определенной части программы выберите инструмент **Сдвиг** (Pan), а затем, нажав и удерживая кнопку мыши в области программирования, перетащите мышь.

Удерживание в нажатом состоянии клавиши **Alt** при использовании инструмента **Выбрать** (Select) позволяет достичь того же эффекта.

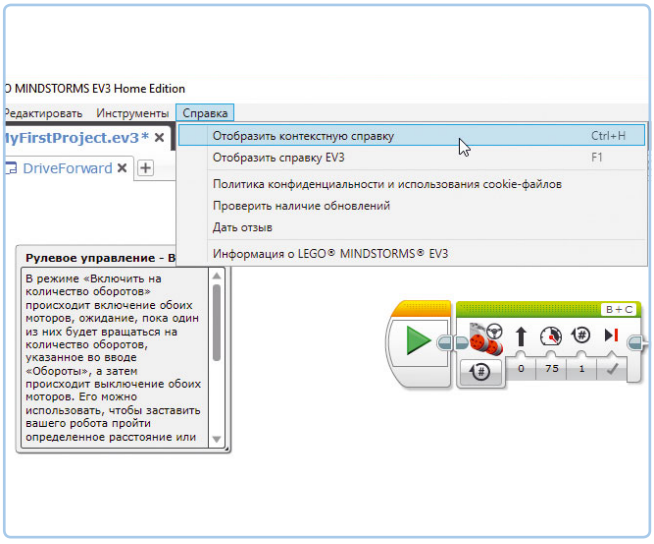


*Рис. 3.18. Окно контекстной справки отображает информацию о выбранном блоке (блок **Рулевое управление** в данном случае)*

Для получения обзорного представления о большой программе нажмите кнопку **Уменьшить масштаб** (Zoom out), чтобы на экране отобразилось больше блоков. Нажмите кнопку **Увеличить масштаб** (Zoom in) или **Сбросить масштаб** (Zoom reset), чтобы вернуться к обычному режиму просмотра.

Создание комментариев

Кнопка **Комментарий** (Comment) используется для размещения комментариев в области программирования. Они не влияют на вашу программу, но помогают запомнить, за что отвечает каждая ее часть. При нажатии кнопки **Комментарий** (Comment) на панели инструментов в области программирования появляется соответствующее окно. Изменять размер и перемещать окно можно с помощью мыши. Чтобы ввести комментарий, дважды щелкните в окне, как показано на рис. 3.16. Чтобы удалить комментарий, щелкните мышью по соответствующему окну и нажмите кнопку **Delete** на клавиатуре.

Создание дубликатов блоков

Когда вы разрабатываете программу, иногда требуется продублировать последовательность блоков, а не перемещать их заново из палитры один за другим. Чтобы скопировать группу выделенных блоков, перетащите ее в новое расположение, удерживая клавишу **Ctrl**, как показано на рис. 3.17. То же самое вы можете сделать, выбрав по очереди команды меню **Редактировать** ▸ **Копировать** (Edit ▸ Copy) и **Редактировать** ▸ **Вставить** (Edit ▸ Paste), но в этом случае вы не сможете выбрать позицию в области программирования, куда будут помещены копии блоков.

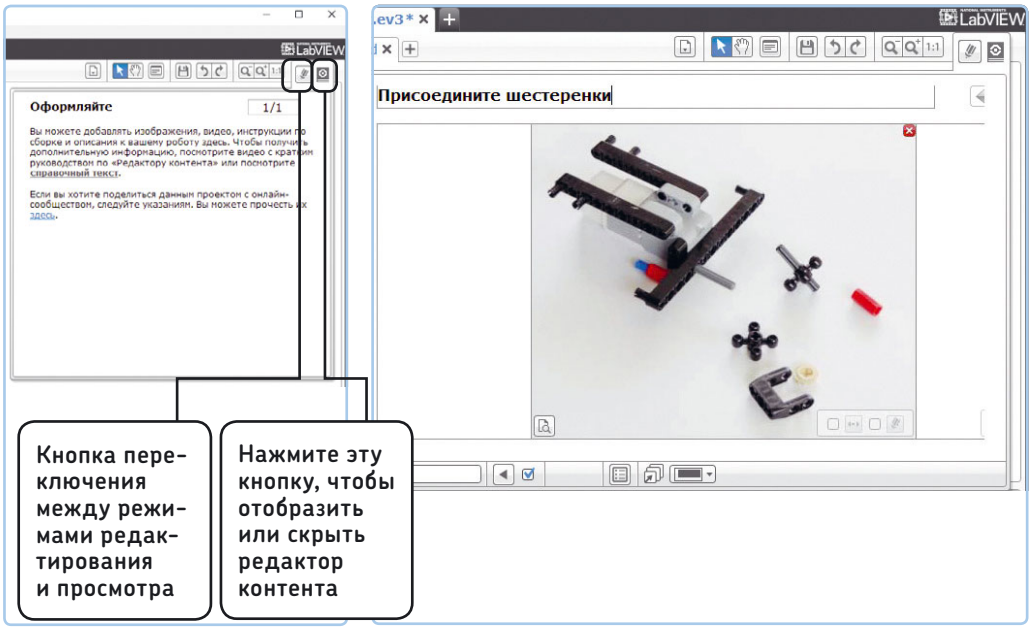


Рис. 3.19. Используйте редактор контента для документирования работы над проектом (слева). Например, вы можете добавить изображение, которое показывает последовательность сборки робота (справа)

Рис. 3.20. Помимо пяти фирменных проектов EV3, вам доступны 12 бонусных роботов, таких как этот гоночный грузовик

Справочная документация

Чтобы познакомиться со справочной информацией обо всех программных блоках — помимо вводной информации, содержащейся в этой книге, — выберите команду меню **Справка ▸ Отобразить справку EV3** (Help ▸ Show EV3 Help), как показано на рис. 3.18. Для получения дополнительных сведений, к примеру о блоке **Рулевое управление** (Move Steering), откройте справочную документацию и выберите раздел **Программные блоки ▸ Блоки действий ▸ Рулевое управление** (Programming Blocks ▸ Action Blocks ▸ Move Steering).

При выборе команды меню **Справка ▸ Отобразить контекстную справку** (Help ▸ Show Context Help) появившееся окно отобразит информацию о выбранном блоке или кнопке, как показано на рис. 3.18. Щелкните мышью по ссылке **Дополнительная информация** (More Information) для просмотра соответствующей страницы в справочной системе во Всемирной паутине.

На начальном экране-лобби вы найдете *дополнительное руководство*, в котором модуль EV3 и другое оборудование описаны более подробно.

7. Редактор контента

Редактор контента открывается с помощью кнопки в правой части панели инструментов (рис. 3.19). В окне редактора контента вы можете добавить дополнительную информацию о вашем проекте, например описание того, как работает программа или как собрать робота. Это может быть полезно как для собственного использования, так и чтобы другие люди легко могли собрать такого же робота, как ваш. Вы можете добавить текст и изображения, чтобы представить свой проект другим людям, в интуитивно понятном интерфейсе редактора.

Сборка фирменных роботов EV3 и дополнительных моделей

После того как вы получили некоторый опыт в создании и программировании роботов EV3, вы можете собрать пять роботов, таких как EV3RSTORM, доступных на начальном экране-лобби программы. Вы также можете попробовать собрать 12 бонусных моделей, которые представлены на вкладке **Больше роботов** (More Robots) на начальном экране-лобби программы. Я построил модель робота RAC3 TRUCK, показанную на рис. 3.20.

Заключение

В этой главе вы изучили основы работы с программным обеспечением EV3. Теперь вы знаете, как создавать, редактировать и сохранять проекты и программы, а также как передавать программы в память модуля EV3.

В главе 4 мы возьмемся за программирование всерьез!

4

Работа с блоками действий

В главе 3 вы изучили основы — как создать программу и передать ее в память модуля робота EXPLOR3R. В этой главе вы узнаете, как использовать программные блоки для разработки программ и как заставить робота EXPLOR3R двигаться. Еще вы сделаете так, чтобы ваш робот проигрывал звуки и отображал текст/изображения на экране модуля EV3. И научитесь управлять подсветкой модуля EV3. После того как вы немного поработаете с примерами программ из этой главы, вы сможете решить некоторые задачи программирования самостоятельно!

Принцип работы программных блоков

Программы EV3 составляются из программных блоков, каждый побуждает робота совершать какое-либо определенное действие, например двигаться вперед в течение одной секунды. Программы запускают блоки один за другим, начиная с первого блока, находящегося слева. После того как первый блок завершает работу, программа переходит ко второму блоку и так далее. Когда последний блок завершил работу, программа тоже завершает работу.

Каждый блок имеет один или несколько *режимов* и, в свою очередь, несколько *параметров* для каждого режима. Вы можете изменить работу блока путем изменения его режима и параметров. Например, на рис. 4.1 оба блока — это **Рулевое управление** (Move Steering), но для них установлены разные режимы. Поскольку блоки используют различные режимы и параметры, они программируют робота выполнять два разных действия.

Существует много типов программных блоков, каждый имеет собственное название и уникальный значок, чтобы вы могли отличить их друг от друга. Разные типы блоков предназначены для различных целей. Аналогичные блоки сгруппированы по цвету на палитре программирования. В этой главе я сосредоточусь на использовании блоков действий (зеленые блоки на палитре программирования).

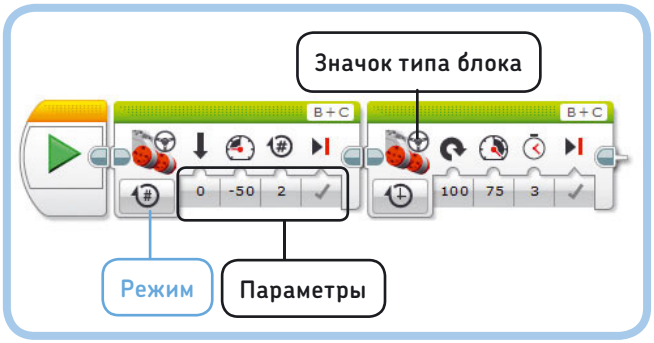


Рис. 4.1. Чтобы настроить блок действий, измените его режим и параметры. Например, первый блок дает команду роботу двигаться назад, а второй программирует его повернуть направо. Вы научитесь создавать эту программу в следующем разделе

Блок Рулевое управление

Блок **Рулевое управление** (Move Steering) управляет работой моторов робота. Используя его в вашей программе, вы можете заставить робота EXPLOR3R двигаться вперед или назад, влево или вправо. Мы использовали блок **Рулевое управление** (Move Steering) в главе 3, чтобы робот EXPLOR3R двигался вперед в течение короткого времени.

Блок Рулевое управление в действии

Для начала давайте создадим небольшую программу, чтобы опробовать блок **Рулевое управление** (Move Steering) в действии. Эта программа будет двигать робота EXPLOR3R назад, пока колеса не сделают два полных оборота, а затем быстро вращать его вправо в течение трех секунд. Поскольку это два разных действия, вы будете использовать два блока **Рулевое управление** (Move Steering).

1. Создайте новый проект под названием EXPLOR3R-4. Вы будете использовать этот проект для всех программ,

которые вы делаете в этой главе. Измените название первой программы на Move.

2. Перетащите из палитры программирования два блока **Рулевое управление** (Move Steering) в область программирования, как показано на рис. 4.2.

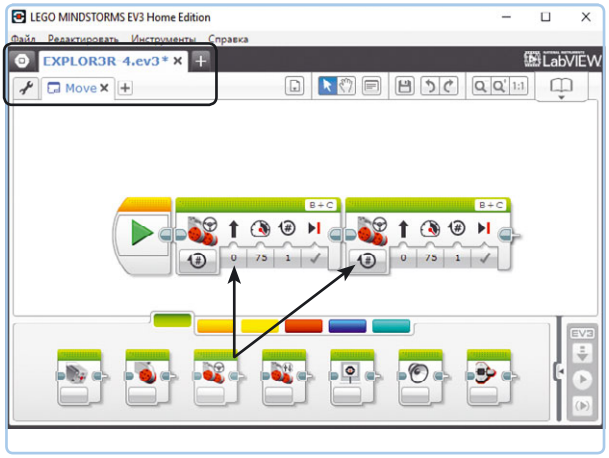


*Рис. 4.2. Создание программы Move в проекте EXPLOR3R-4. Выберите блок **Рулевое управление** на палитре программирования и установите его рядом с блоком **Начало**. Поместите второй блок рядом с первым*

3. Блоки, которые вы только что разместили, по умолчанию настроены так, чтобы робот перемещался вперед в течение некоторого времени, но нам нужно настроить первый блок **Рулевое управление** (Move Steering) так, чтобы робот получил команду двигаться назад на два оборота колес. Для достижения этой цели измените настройки первого блока, как показано на рис. 4.3.

4. Теперь необходимо изменить настройки на втором блоке. Этот блок будет передавать EXPLOR3R команду быстро вращаться вправо в течение трех секунд. Во-первых,

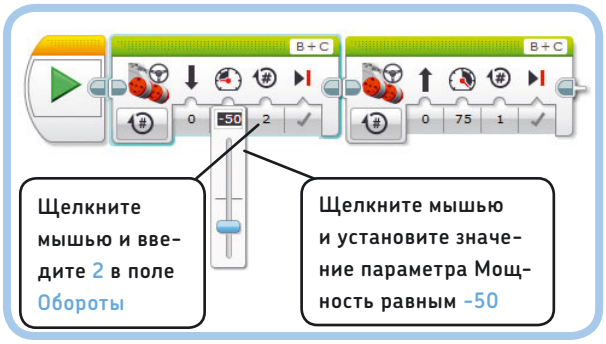


*Рис. 4.3. Настройте первый блок, присвоив параметру **Мощность** значение **–50**. Для этого переместите ползунковый регулятор вниз до положения **–50** или введите вручную значение **–50**. Отрицательные значения инструктируют робота двигаться в обратном направлении. Затем присвойте параметру **Обороты** значение **2**, чтобы робот прекратил движение, когда колеса совершат два полных оборота*

выберите пункт **Включить на количество секунд** (On for Seconds), как показано на рис. 4.4.

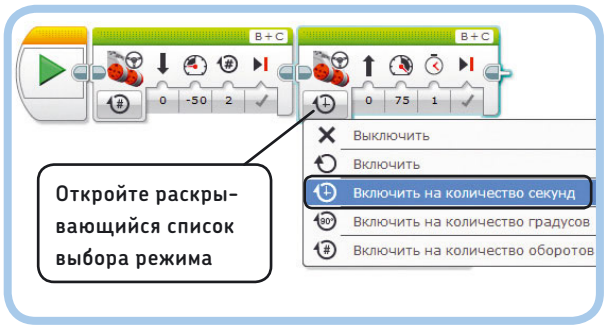


*Рис. 4.4. Нажмите кнопку выбора режима второго блока и выберите пункт **Включить на количество секунд***

5. Чтобы робот быстро вращался вправо в течение трех секунд, измените настройки второго блока так, как показано на рис. 4.5.

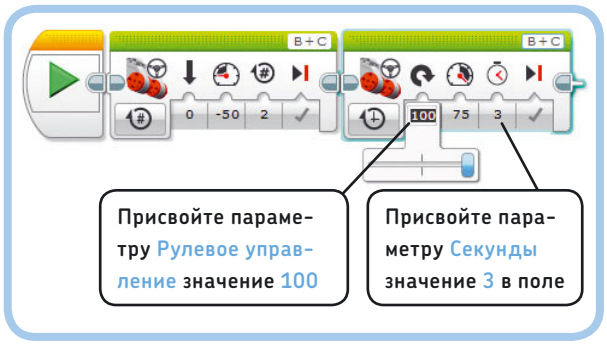


*Рис. 4.5. Настройте второй блок, переместив ползунковый регулятор **Рулевое управление** вправо до упора, а затем введите число **3** в поле **Секунды***

6. После того как вы настроили оба блока **Рулевое управление** (Move Steering), вы можете загрузить программу в модуль EV3 вашего робота и запустить ее. Программа успешно работает, если EXPLOR3R движется назад до тех пор, пока оба колеса не сделают два оборота, а затем быстро поворачивается в течение трех секунд.

ПРИМЕЧАНИЕ Если ваш робот поворачивает налево, а не направо, вполне возможно, что вы неправильно подключили моторы к портам модуля EV3. Изучите шаги на с. 41, чтобы проверить подключение кабелей на роботе.

Режимы и параметры

Теперь мы внимательно рассмотрим настройки программных блоков, чтобы лучше понимать, как функционирует наша программа-пример. Действие каждого блока определяется его режимом и параметрами. Рис. 4.6 демонстрирует несколько вариантов настройки блока **Рулевое управление** (Move

Steering), помимо тех параметров, которые мы уже использовали. Блок **Рулевое управление** (Move Steering) поддерживает несколько *режимов* работы. Выбрать режим можно в раскрывающемся списке. В каждом режиме блок работает несколько иначе. Например, первый блок в нашей программе находится в режиме **Включить на количество оборотов** (On for Rotations), который позволяет указать количество оборотов мотора, в течение которых робот будет двигаться. Второй блок находится в режиме **Включить на количество секунд** (On for Seconds), что позволяет определить время работы моторов в секундах.

Блок **Рулевое управление** (Move Steering) поддерживает следующие режимы.

- **Включить** (On): моторы включены.
- **Выключить** (Off): моторы отключены.
- **Включить на количество оборотов** (On for Rotations): моторы включаются на определенное число полных оборотов; затем останавливаются.
- **Включить на количество секунд** (On for Seconds): моторы включаются на определенное количество секунд; затем останавливаются.
- **Включить на количество градусов** (On for Degrees): моторы включаются на определенное количество градусов вращения; затем останавливаются.

Режимы **Включить** (On) и **Выключить** (Off) вы научитесь использовать в разделе «Режимы “Включить” и “Выключить” блоков действий» далее в этой главе.

Порты

В раскрывающемся списке **Порты** (Ports), расположенном в правом верхнем углу блока, вы можете выбрать те порты выхода модуля EV3, к которым вы подключили моторы, чтобы программа правильно определяла, какие моторы следует включать. Моторы EXPLOR3R подключены к портам B и C, поэтому в нашей программе-образце мы оставили значение **B + C** без изменений.

Рулевое управление

Как вы могли заметить при работе с программой Move, вы можете создать управляемого робота. Для регулировки рулевого управления вашего робота щелкните мышью по кнопке параметра **Рулевое управление** (Steering) и переместите ползунковый регулятор влево или вправо, в зависимости от того, куда по вашему желанию должен направляться робот.

Разве может транспортное средство повернуть без руля? Нет. Вот и этот блок позволяет контролировать рулевое управление робота путем управления двумя колесами независимо друг от друга. Чтобы робот двигался прямо, оба колеса должны вращаться в одном направлении с одинаковой скоростью. Чтобы сделать поворот, одно колесо должно вращаться быстрее, чем другое, или же колеса должны вращаться в противоположных направлениях, чтобы робот совершил поворот на месте. На рис. 4.7 показано, как различные комбинации значений параметров **Мощность** (Power) и **Рулевое управление** (Steering) программируют EXPLOR3R сделать поворот.

Мощность

Параметр **Мощность** (Power) управляет скоростью вращения моторов. Значение параметра **Мощность** (Power) 0 означает, что колеса не двигаются вообще, а значение 100 устанавливает максимальную скорость работы моторов. Отрицательные значения, такие как –100 или –30, означают, что робот будет двигаться задним ходом. Пример такого движения вы видели в программе Move.

Вращения, секунды или градусы

В зависимости от выбранного режима третий параметр блока **Рулевое управление** (Move Steering) позволяет определить, как долго моторы должны работать. Так, присвоение значения **3** параметру **Секунды** (Seconds) в режиме **Включить на количество секунд** (On for Seconds) означает, что EXPLOR3R будет двигаться в течение трех секунд.

Режим **Включить на количество градусов** (On for Degrees) позволяет назначить, на сколько градусов должны повернуться колеса.

Значение 360 градусов равно одному полному обороту колеса, а 180 градусов — вращение колеса на половину оборота.

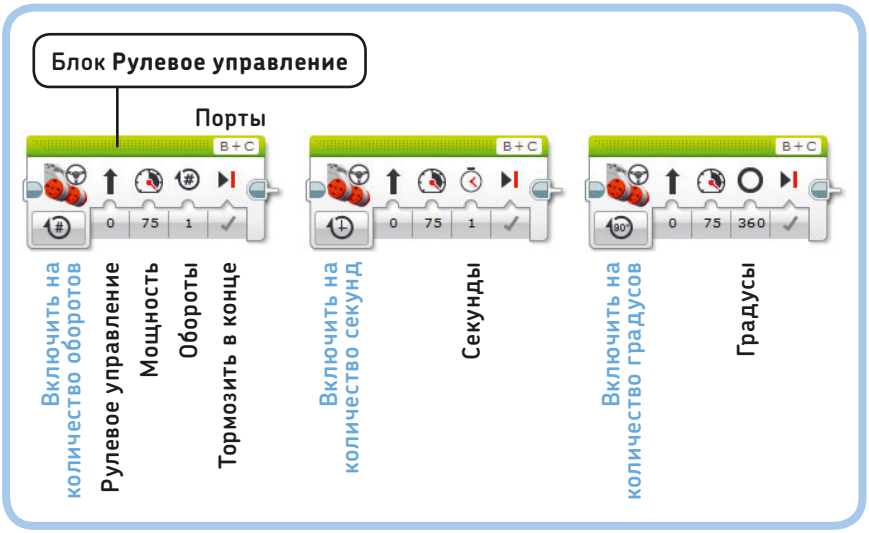


*Рис. 4.6. Режимы (выделены синим цветом) и соответствующие параметры (выделены черным цветом) блока **Рулевое управление**. Чтобы выбрать другой режим, нажмите кнопку выбора режима и выберите подходящий пункт в раскрывающемся списке. Большинство параметров будут одинаковыми независимо от того, какой режим вы выбираете*

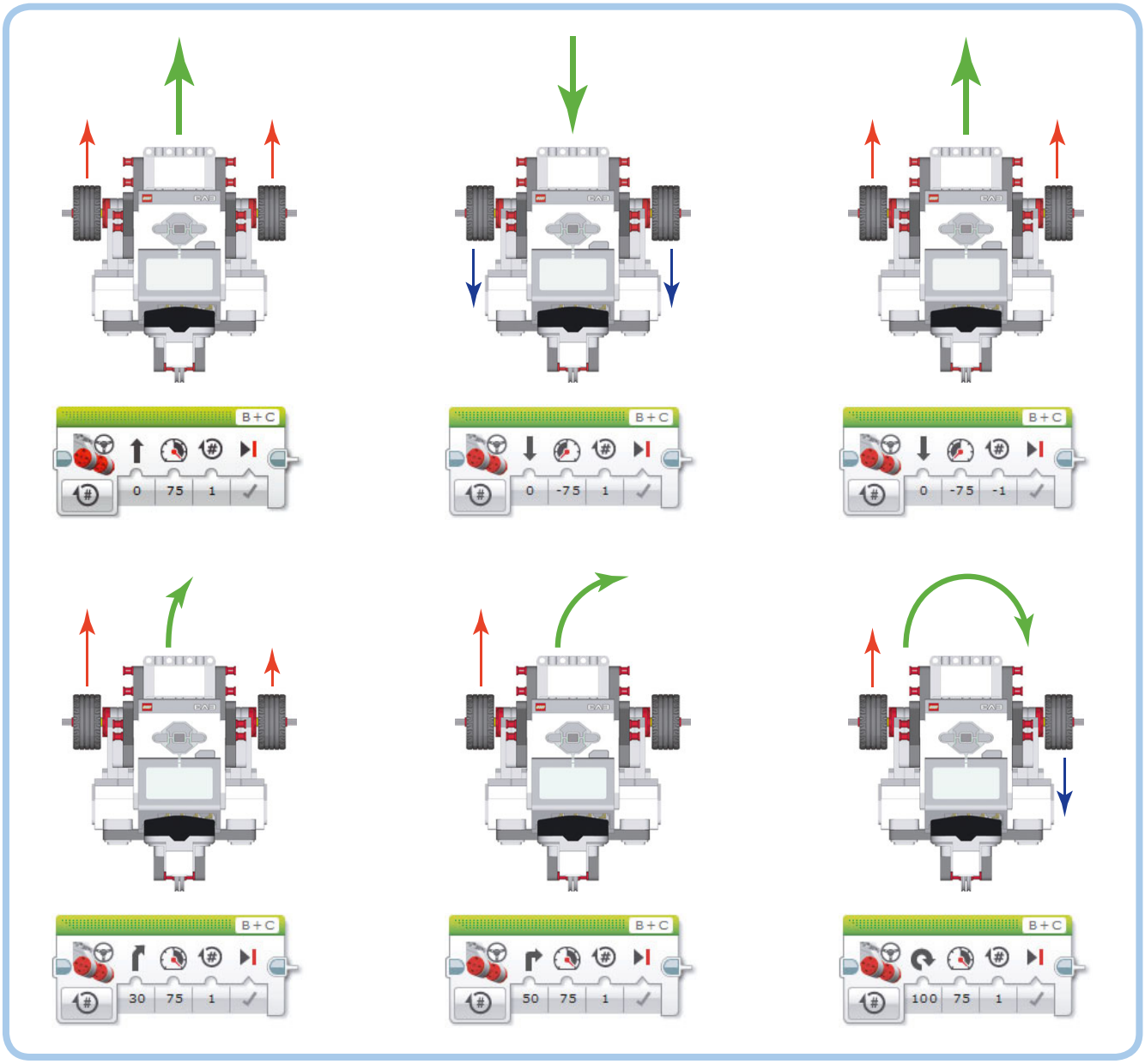


*Рис. 4.7. Чтобы ваш робот поворачивал, отрегулируйте параметр **Рулевое управление**; программа будет управлять скоростью и направлением обоих моторов, чтобы робот совершил поворот. Красные стрелки показывают, что колесо движется вперед, а синие — что оно движется назад. Зеленая стрелка показывает направление, в котором в результате движется сам робот*

ПРИМЕЧАНИЕ **Присвоение отрицательного значения параметру Мощность (Power), например –75, и положительного значения параметру Обороты (Rotation), например 1 или 360 градусов, приводит к тому, что робот движется в *обратном направлении*. То же самое справедливо и для положительных значений параметра Мощность (Power) (к примеру, 75) и отрицательных значений параметра Обороты (Rotation) (например, –1). Тем не менее, если вы присвоите отрицательные значения обоим параметрам, и Мощность (Power) и Обороты (Rotation), например –75 и –1 соответственно, робот будет двигаться *вперед!* Обратитесь к рис. 4.7, если не понимаете, как комбинация значений параметра Мощность (Power) и Рулевое управление (Steering) будет влиять на вашего робота.**

Когда блок переведен в режим **Включить на количество градусов** (On for Degrees), вы можете задать колесам количество полных оборотов, которые они должны совершить, как вы видели в первом блоке программы-образца.

Торможение в конце

Параметр **Тормозить в конце** (Brake at End) управляет остановкой моторов после того, как они совершат заданное количество вращений (в том числе в градусах или секундах). Выбор значения **Тормозить** (Brake) (**Истина** (True)) (выберите пункт ✓) немедленно останавливает работу моторов, а **Двигаться накатом** (Overrun) (**Ложь** (False)) (✗) — останавливает моторы плавно.

Уточнение угла поворота

При настройке параметров блока **Рулевое управление** (Move Steering) вы можете подумать, что для поворота робота на 90 градусов нужно параметру **Градусы** (Degrees) присвоить значение 90, но это не так. На самом деле, параметр **Градусы**

ПРАКТИКУМ № 1: УСКОРЕНИЕ!

Сложность: **Время:**

Теперь, когда вы узнали некоторые важные сведения о блоке **Рулевое управление** (Move Steering), вы готовы к экспериментам с ним. Цель этого практикума — создание программы, которая сначала инструктирует робота двигаться медленно, а затем ускориться.

Разместите десять блоков **Рулевое управление** (Move Steering) в области программирования и настройте первые два, как показано на рис. 4.8. Настройте третий таким же образом, но присвойте параметру **Мощность** (Power) значение **30**. Увеличивайте это значение на 10 в каждом следующем блоке, пока не достигнете максимальной скорости мотора.

Блоки находятся в режиме **Включить на количество секунд** (On for Seconds). После того как вы проверили программу, смените режим всех десяти блоков на **Включить на количество оборотов** (On for Rotations), присвойте параметру **Обороты** (Rotation) значение **1** и запустите программу снова. Выполнение какой программы занимает больше времени? Можете ли вы объяснить, чем обусловлена такая разница?

Рис. 4.8. Первые блоки программы для практикума № 1. Создавайте новую программу для каждого практикума и сохраняйте ее, когда закончили, чтобы вы могли продолжать работать над программой позднее!

ПРАКТИКУМ № 2: УТОЧНЕНИЕ ПОВОРОТОВ!

Сложность: **Время:**

Можете ли вы сделать так, чтобы робот совершал поворот на месте на 90 градусов? Создайте новую программу с одним блоком **Рулевое управление** (Move Steering), настроенным на режим **Включить на количество градусов** (On for Degrees), как показано на рис. 4.9. Убедитесь, что ползунковый регулятор **Рулевое управление** (Steering) смещен до упора вправо, как это было сделано в программе Move. На сколько градусов должны повернуться *колеса* робота, чтобы он сделал точный поворот на 90 градусов?

Начните с присвоения значения 275 параметру **Градусы** (Degrees). Если этого недостаточно, попробуйте значение 280, 285 и так далее, запуская программу каждый раз, чтобы увидеть, совершает ли робот нужный поворот.

После того как вы определили правильное значение для совершения поворота на 90 градусов, выясните, какое значение вы должны задать, чтобы робот сделал поворот на 180 градусов.

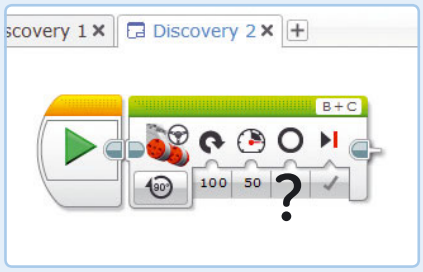


Рис. 4.9. Программа для практикума № 2. Какое значение нужно задать, чтобы робот повернул на 90 градусов? Какое значение вы должны использовать для поворота на 180 градусов?

ПРАКТИКУМ № 3: ПОКАТАЕМСЯ!

Сложность: **Время:**

Создайте программу с тремя блоками **Рулевое управление** (Move Steering), чтобы EXPLOR3R двигался вперед в течение трех секунд при 50-процентной мощности, повернулся на 180 градусов, а затем вернулся в исходное положение. При настройке блока, который позволяет роботу разворачиваться (второй блок), используйте значение **Градусы** (Degrees), которое вы определили в практикуме № 2.

ПРАКТИКУМ № 4: РОБОТ-ПИСАТЕЛЬ!

Сложность: **Время:**

Используйте блоки **Рулевое управление** (Move Steering), чтобы разработать программу, которая управляет движением EXPLOR3R, как будто он пишет первую букву вашего имени. Сколько блоков вам нужно использовать для «написания» этой буквы?

СОВЕТ **Для создания плавных поворотов используйте ползунковый регулятор Рулевое управление (Steering).**

(Degrees) определяет, на сколько градусов поворачивается мотор и, как следствие, колеса. Фактический угол поворота в градусах для моторов разных роботов различен. Практикум № 2 положит начало вашим экспериментам по определению значения **Градусы** (Degrees) для вашего робота.

Блок Звук

Очень интересно создавать программы, которые заставляют EXPLOR3R двигаться. Этот процесс становится еще более увлекательным при использовании блока **Звук** (Sound), в результате чего робот начнет издавать звуки. Робот может воспроизводить два типа звуков: простой тональный сигнал или заранее записанный звук, например аплодисменты или слово «Привет». Когда вы используете блок **Звук** (Sound) в ваших программах, робот будет казаться более интерактивным и реалистичным, потому что он сможет «общаться».

Параметры блока Звук

Несмотря на то что каждый программный блок позволяет роботу делать что-то уникальное, все блоки используются одним и тем же образом. Другими словами, вы можете просто выбрать блок **Звук** (Sound) из палитры программирования и поместить его в область программирования, так же как и блок **Рулевое управление** (Move Steering) ранее. После того как блок установлен, выберите режим, а затем настройте параметры, чтобы указать, какой звук вы хотите услышать.

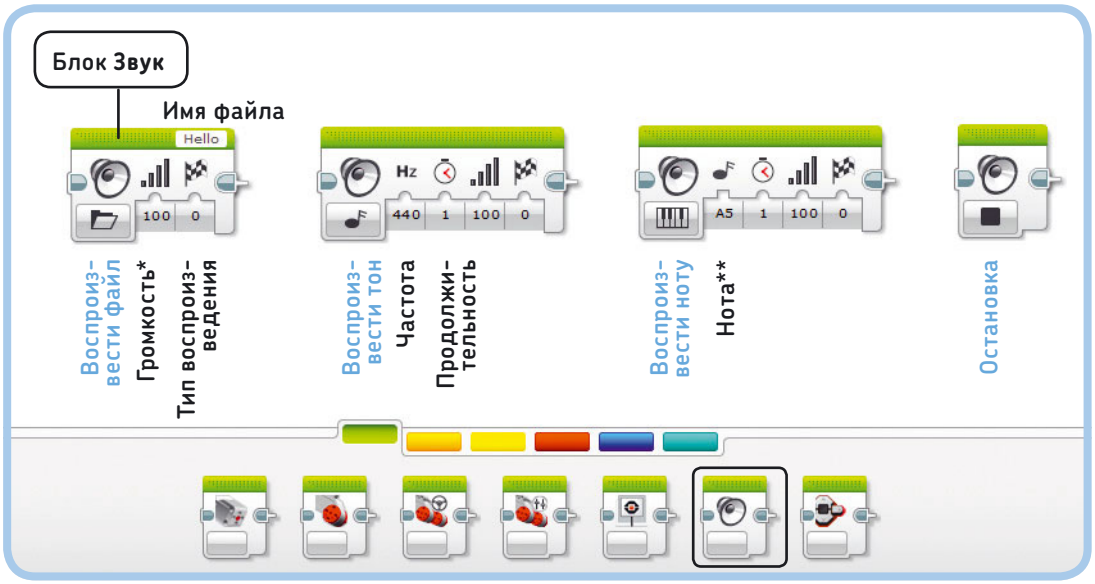
Но прежде, чем мы создадим программу с блоком **Звук** (Sound), давайте кратко рассмотрим различные режимы и параметры этого блока, которые показаны на рис. 4.10.

Блок **Звук** (Sound) поддерживает четыре режима.

- **Воспроизвести файл** (Play File): Воспроизводится предварительно записанный звуковой файл, например слово «привет».
- **Воспроизвести тон** (Play Tone): Воспроизводится тон на определенной частоте (высоте) в течение определенного промежутка времени.
- **Воспроизвести ноту** (Play Note): Звучит нота пианино в течение определенного промежутка времени.
- **Остановка** (Stop): Прекращается любой звук, который воспроизводится в данный момент.

Имя файла

В режиме **Воспроизвести файл** (Play File) вы можете указать звук, щелкнув мышью по полю **Имя файла** (File Name) и выбрав файл из появившегося меню. Вы можете выбрать звук из имеющихся категорий, таких как **Животные** (Animals), **Цвета** (Colors), **Связь** (Communication) и **Числа** (Numbers). Вы также можете записывать и добавлять собственные звуки, используя



* В русской версии программы слово Volume ошибочно переведено как «Том». Правильно — Громкость. *Здесь и далее прим. пер.*

** В русской версии программы слово Note ошибочно переведено как «Примечание». Правильно — Нота.

*Рис. 4.10. Четыре режима (выделены синим цветом) блока **Звук** и соответствующие параметры (выделены черным цветом)*

вкладку **Инструменты** ▸ **Редактор звука** (Tools ▸ Sound Editor). (См. раздел **Инструменты** ▸ **Редактор звука** (Tools ▸ Sound Editor) справочной системы.)

Громкость

В качестве значения параметра **Громкость** (Volume) укажите число от 0 (тихо) до 100 (громко), чтобы установить уровень громкости звука, который вы хотите воспроизвести.

Тип воспроизведения

Используйте параметр **Тип воспроизведения** (Play Type), чтобы управлять поведением робота в процессе воспроизведения звука. Выберите пункт **Ожидать завершения (0)** (Wait for Completion (0)), чтобы приостановить программу, пока воспроизводится звук. Выберите вариант **Воспроизвести один раз (1)** (Play Once (1)), чтобы позволить программе перейти к выполнению следующего блока, пока звук продолжает воспроизводиться. При выборе варианта **Повторить (2)** (Repeat (2)) звук будет повторяться до тех пор, пока оставшиеся программные блоки не закончат свою работу. Для большинства программ вы будете использовать вариант **Ожидать завершения (0)** (Wait for Completion (0)).

Нота или частота

В зависимости от установленного режима вы можете присвоить значение параметру **Нота*** (Note) с помощью фортепианной клавиатуры или параметру **Частота** (Tone), указав его в герцах (Гц). Тон 440 Гц (значение параметра **Частота** (Tone) по умолчанию) хорошо воспринимается человеческим ухом, поэтому прекрасно подходит для тестирования программ. Например, вы можете воспроизвести звук, обозначающий, что конкретный программный блок закончил работу.

Продолжительность

В поле **Продолжительность** (Duration) задайте, сколько секунд будет воспроизводиться нота или тон.

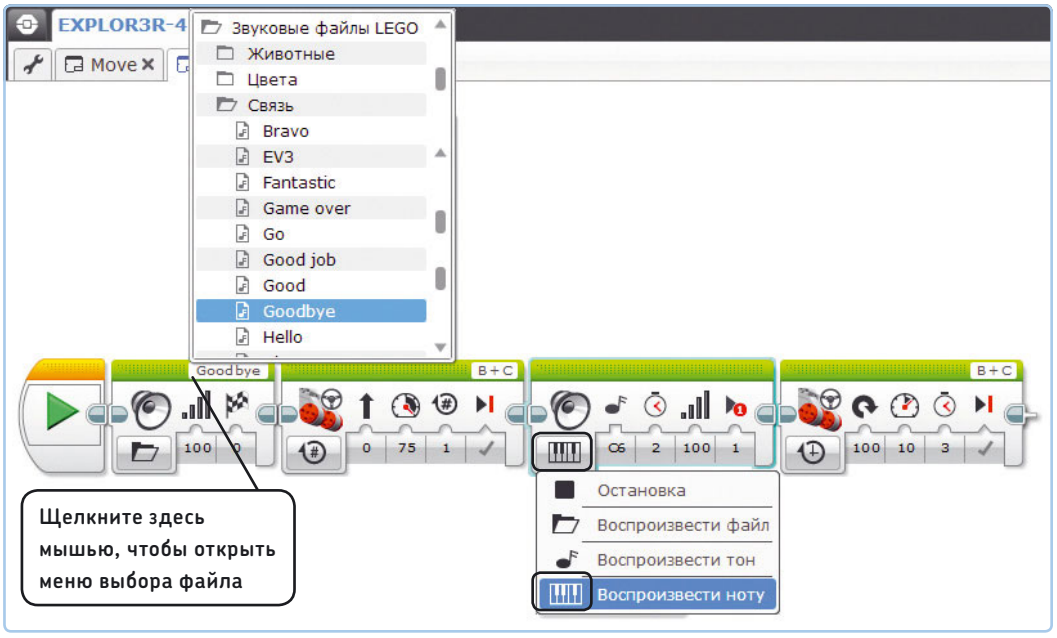
Использование блока Звук

Теперь давайте создадим программу под названием SoundCheck. Она научит робота двигаться и воспроизводить звуки, чтобы вы могли опробовать блок **Звук** (Sound) в действии. Для начала создайте новую программу, как изображено на рис. 4.11. Для создания этой программы и других примеров в книге выполните следующие действия для каждого блока в программе.

1. Найдите нужный блок на палитре программирования, ориентируясь по его цвету и значку/названию, а затем поместите его в область программирования. Например, блок **Звук** (Sound) окрашен в зеленый цвет, а его значок представлен в виде динамика.

2. Выберите подходящий режим, как показано на рис. 4.11. Значок в виде клавиш пианино на третьем блоке указывает на то, что вы должны выбрать пункт **Воспроизвести ноту** (Play Note).

3. И наконец, настройте на блоке остальные параметры. Присвойте соответствующие значения параметрам **Рулевое управление** (Steering) и **Мощность** (Power) в блоке **Рулевое управление** (Move Steering). Ваш блок должен выглядеть точно так же, как на иллюстрации.



* В русской версии программы слово Note ошибочно переведено как «Примечание». Правильно — Тон.

** Пункты **Forwards** и **Backward** соответственно в группе **Информация** (Information).

*Рис. 4.11. Программа SoundCheck. Чтобы выбрать режим работы блока, откройте раскрывающийся список, показанный внизу, и выберите пункт **Воспроизвести ноту** со значком в виде клавиш пианино*

ПРАКТИКУМ № 5: В КАКУЮ СТОРОНУ, ГОВОРИТЕ?

Сложность: **Время:**

Разработайте программу, подобную SoundCheck, которая будет объявлять направление движения робота. Во время движения вперед он должен сказать: «Вперед», а когда движется назад — «Назад»**. Как вы настроите параметр **Тип воспроизведения** (Play Type) на блоках **Звук** (Sound)?

ПРАКТИКУМ № 6: СТАНЬ ДИДЖЕЕМ!

Сложность: **Время:**

Создавая программу с последовательностями блоков **Звук** (Sound), настроенных для воспроизведения нот, вы можете создавать музыкальные композиции. Сможете ли вы сыграть какую-нибудь известную мелодию с помощью робота EV3 или создать собственную легко запоминающуюся музыку?

После того как вы закончите создание программы, загрузите ее в модуль EV3 своего робота и запустите.

Программа SoundCheck

Теперь, после запуска программы, давайте рассмотрим, как она работает. Первый блок **Звук** (Sound) инструктирует робота EXPLOR3R сказать: «Пока»*. Параметру **Тип воспроизведения** (Play Type) присвоено значение **Ожидать завершения (0)** (Wait for Completion (0)), поэтому робот стоит, пока не закончит произносить слово «Пока». Затем блок **Рулевое управление** (Move Steering) инструктирует робота продвинуться вперед на один оборот, а другой блок **Звук** (Sound) инструктирует модуль EV3 воспроизвести ноту. Этот блок не использует значение **Ожидать завершения (0)** (Wait for Completion (0)) параметра **Тип воспроизведения** (Play Type), поэтому во время воспроизведения звука второй блок **Рулевое управление** (Move Steering) руководит роботом, направляя его вправо в течение трех секунд. После этого робот прекращает движение.

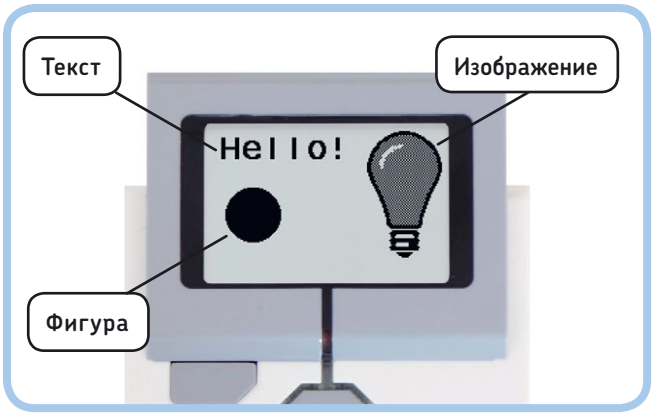


*Рис. 4.12. Используйте блок **Экран** для вывода изображений, текста и фигур на экран модуля EV3*

Блок Экран

В дополнение к перемещению робота и воспроизведению звуков программа EV3 позволяет управлять экраном модуля EV3. Например, вы можете создать программу, которая выводит на экран модуля EV3 графику, показанную на рис. 4.12. Размер экрана составляет 178 пикселов в ширину и 128 в высоту. Пикселы — это маленькие точки, из которых состоит изображение на экране.

Экспериментировать с блоком **Экран** (Display) весьма забавно, но, что более важно, это эффективный способ протестировать свои программы. Например, вы можете отобразить информацию, полученную с датчика, на экране, чтобы посмотреть, работает ли датчик должным образом. К датчикам мы вернемся в части V этой книги.

Вы можете использовать блок **Экран** (Display) для вывода на экран модуля EV3 изображения (например, лампочки), текста (например, «Hello!») или фигуры (такой как сплошной кружок). Один блок **Экран** (Display) не может поместить сразу несколько изображений или текстовых строк на экране. Поэтому вам нужно будет использовать последовательность блоков, чтобы создать изображение, как на рис. 4.12.

Параметры блока Экран

После того как блок **Экран** (Display) вывел какое-либо изображение на экран модуля EV3, программа переходит к следующему блоку — к примеру, **Рулевое управление** (Move Steering). На экране модуля EV3 изображение будет выводиться до тех пор, пока другой блок **Экран** (Display) не начнет отображать что-то иное. Таким образом, в этом примере изображение будет оставаться на экране, пока робот движется.

Когда программа завершает работу, модуль EV3 немедленно возвращается в меню, а это значит, что, если последним в вашем проекте используется блок **Экран** (Display), у вас не будет времени, чтобы увидеть изображение на экране, так как программа прекратит работу. Чтобы финальное изображение было выведено, добавьте другой блок, например **Рулевое управление** (Move Steering), чтобы программа не завершала работу мгновенно. На рис. 4.13 показаны четыре основных режима работы блока **Экран** (Display).

* Пункт **Goodbye** в группе **Связь** (Communication).

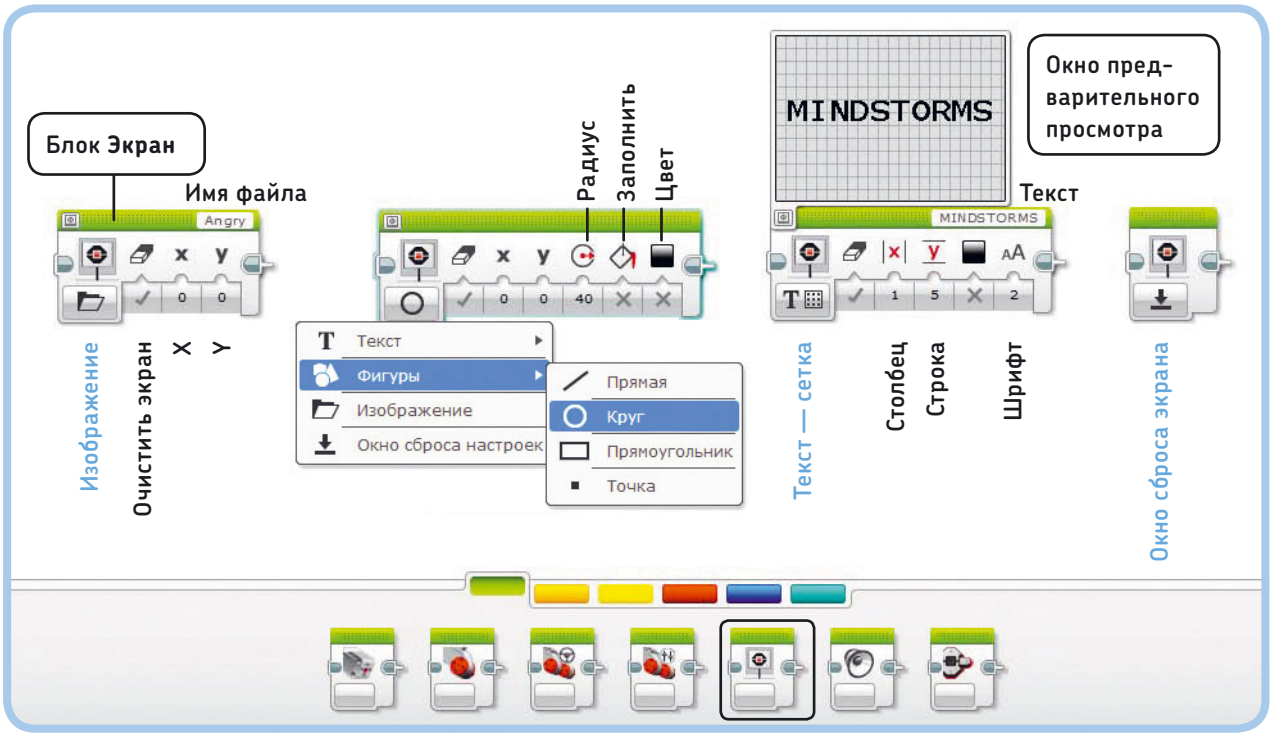


*Рис. 4.13. Четыре режима (выделены синим цветом) блока **Экран** и их параметры (выделены черным цветом). Вы можете предварительно просмотреть изображение, нажав кнопку в левом верхнем углу блока, чтобы увидеть результат настройки. Когда вы будете удовлетворены, закройте окно предварительного просмотра*

- **Текст** (Text): на экране отображается строка текста.
- **Фигуры** (Shapes): на экране отображается прямая линия, круг, прямоугольник или точка (на выбор).
- **Изображение** (Image): на экране отображается выбранное изображение, например «смайлик».
- **Окно сброса настроек** (Reset Screen): изображение исчезает с экрана, и отображается логотип MINDSTORMS, который вы обычно видите, когда запускаете программу без блока **Экран** (Display).

Подрежимы

Некоторые режимы содержат подрежимы. Когда вы выбираете, например, режим **Фигуры** (Shapes) в блоке **Экран** (Display), вам нужно выбрать один из четырех подрежимов: **Прямая** (Line), **Круг** (Circle), **Прямоугольник** (Rectangle) или **Точка** (Point), как показано на рис. 4.13. Выбор режима **Фигуры** ▸ **Круг** (Shapes ▸ Circle) приводит к отображению блоком **Экран** (Display) круга на экране модуля EV3. А используя параметры блока **Экран** (Display), вы можете менять положение круга, его радиус, заливку и цвет.

Имя файла

В режиме **Изображение** (Image) вы можете щелкнуть мышью по полю **Имя файла** (File Name), чтобы выбрать изображение из таких категорий, как **Глаза** (Eyes), **Выражения** (Expressions), **Предметы** (Objects) и **LEGO**. Кроме того, вы можете создавать или загружать свои изображения, выбрав команду меню **Инструменты** ▸ **Редактор изображения** (Tools ▸ Image Editor). (См. раздел **Инструменты** ▸ **Редактор изображения** (Tools ▸ Image Editor) справочной системы.)

Очистка экрана

Параметр **Очистить экран** (Clear Screen) позволяет указать, следует ли сбрасывать изображенное на экране перед тем, как на нем будет показано что-то еще (для этого нужно установить значение **Истина** (True)), или добавить что-то новое к тому, что уже присутствует на экране (для этого нужно присвоить значение **Ложь** (False)). Вам потребуется последовательность блоков **Экран** (Display), чтобы отобразить более одного объекта на экране. Первый блок должен очистить экран перед тем, как будет показано что-то новое, а другие блоки будут просто добавлять на экран какой-либо объект. Для достижения этой цели присвойте параметру **Очистить экран** (Clear Screen) первого блока значение **Истина** (True), а на последующих блоках — значение **Ложь** (False). Вы увидите, как это работает, в программе DisplayTest.

Радиус и заполнение

Некоторые параметры являются специфическими для различных режимов работы блока **Экран** (Display). Например, параметр **Радиус** (Radius) определяет размер круга, а **Заполнить** (Fill) позволяет создать заполненный круг (значение **Истина** (True)) или только контур (значение **Ложь** (False)).

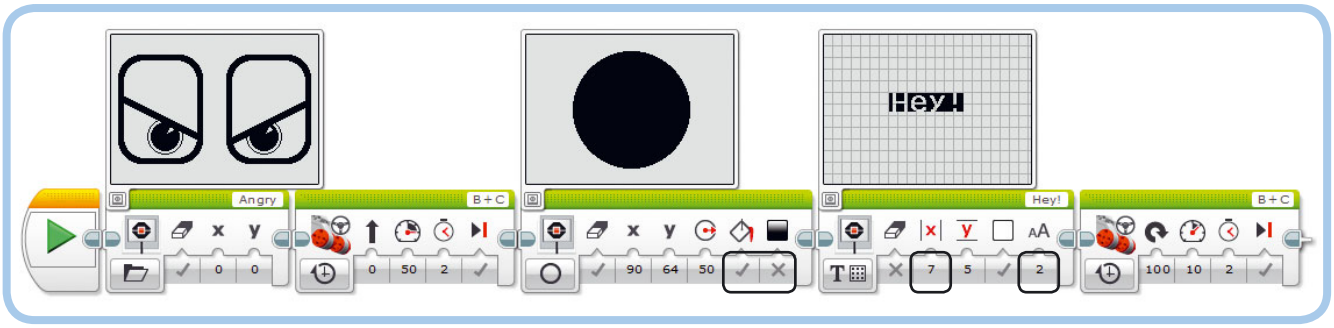


Рис. 4.14. Программа DisplayTest делает так, что движение вашего робота сопровождается отображением графики на экране модуля EV3. Окна предварительного просмотра показаны на иллюстрации для удобства, при необходимости вы можете скрыть их

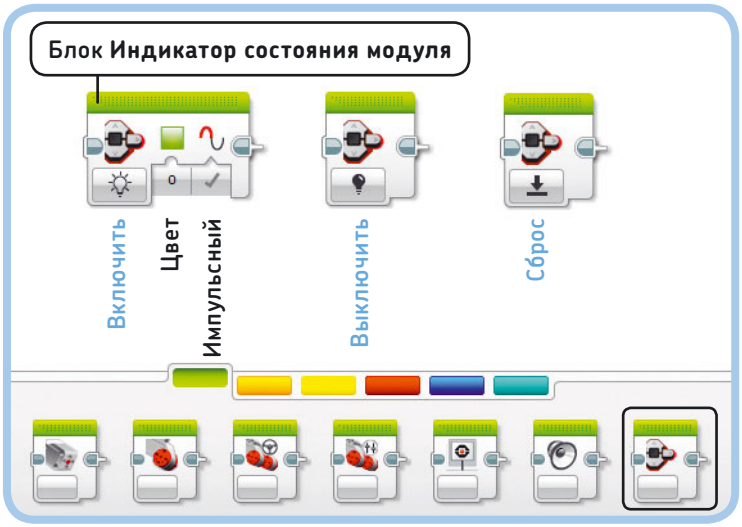


*Рис. 4.15. Три режима блока **Индикатор состояния модуля** и его параметры.*

Цвет

Обычно параметру **Цвет** (Color) присваивается значение **Ложь** (False), что означает черный цвет, но, если вы ранее окрасили область экрана, скажем, создали черный круг, вы можете добавить поверх него белый текст, присвоив параметру **Цвет** (Color) значение **Истина** (True).

Текст и размер шрифта

В режиме **Текст** (Text) введите строку текста, который вы хотите отобразить, например MINDSTORMS, в поле, показанное на рис. 4.13. Текстовая строка может содержать только латинские буквы, цифры и некоторые специальные символы. Вы можете выбрать размер шрифта, присвоив параметру **Шрифт** (Font size) значение **0** (обычный), **1** (жирный) или **2** (большой).

X, Y, столбец и строка

Если вы выводите изображение, текст в режиме **Пикселы** (Pixels) или фигуру, вы можете выбрать расположение объекта, используя параметры **X** (положение относительно левого края экрана) и **Y** (положение по отношению к верхней части). Пользуясь предварительным просмотром изображения (см. рис. 4.13), вы можете легко настроить положение объекта на экране по осям *x* и *y*.

ПРИМЕЧАНИЕ **Помните, что вы можете найти более подробную информацию о каждом блоке в справочной документации. Чтобы посмотреть ее для блока Экран (Display), выберите команду меню Отобразить справку EV3 (Help ▸ Show EV3 Help) и перейдите в раздел Программные блоки ▸ Блоки действий ▸ Экран (Programming Blocks ▸ Action Blocks ▸ Display).**

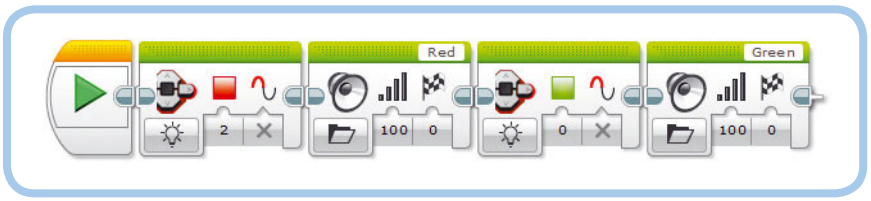
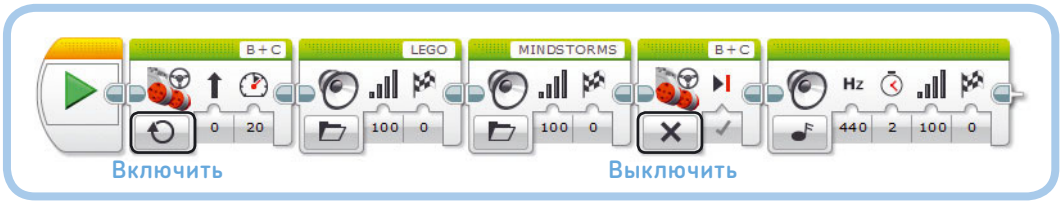


Рис. 4.16. Программа ButtonLight

Положение текста в режиме **Сетка** (Grid) регулируется параметрами **Столбец** (Column) (значения в диапазоне 0–20) и **Строка** (Row) (значения в диапазоне 0–11) вместо параметров **X** и **Y**, упрощая, таким образом, выравнивание нескольких строк текста.



* Пункты **Hello**, **Good**, **Morning** и **Goodbye** в группе **Связь** (Communication).

*Рис. 4.17. Программа OnOff. Правильно настройте режимы в блоках **Рулевое управление***

ПРАКТИКУМ № 7: СУБТИТРЫ!

Сложность: **Время:**

Создайте программу, используя четыре блока **Звук** (Sound), с помощью которых робот будет здороваться, желать доброго утра и прощаться*. Используйте блоки **Экран** (Display), чтобы показать фразы, которые произносит робот, в виде субтитров на экране модуля EV3, а также для очистки экрана каждый раз, когда робот начинает произносить новую фразу. Как вы разместите блоки **Экран** (Display), до или после блоков **Звук** (Sound)?

ПРАКТИКУМ № 8: ВОСЬМЕРКА ДЛЯ EXPLOR3R!

Сложность: **Время:**

Разработайте программу, позволяющую EXPLOR3R перемещаться по траектории в виде восьмерки, как показано на рис. 4.15. В процессе движения робот должен демонстрировать на экране разные выражения глаз. Для этого выбирайте различные изображения из категории **Глаза** (Eyes).

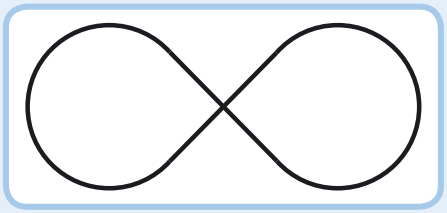


Рис. 4.18. Траектория пути для практикума № 8. Попробуйте сделать траекторию движения робота в виде шаблона, который выглядит так, как показано здесь. На данном этапе робот не должен точно следовать по линии, с этим мы разберемся в главе 7

Использование блока Экран

Давайте проверим функциональность блока **Экран** (Display), разработав программу, которая выводит изображения на экран модуля EV3 во время движения робота. Создайте программу под названием DisplayTest и разместите три блока **Экран** (Display) и два блока **Рулевое управление** (Move Steering) в области программирования, как показано на рис. 4.14. Затем настройте параметры каждого блока, как показано на этом же рисунке. После того как настройка будет завершена, перенесите программу в память модуля EV3 вашего робота и запустите ее.

Программа DisplayTest

В программе DisplayTest блоки **Экран** (Display) выводят на экран несколько изображений, а блоки **Рулевое управление** (Move Steering) задают перемещение робота EXPLOR3R. Первый блок **Экран** (Display) очищает экран, прежде чем вывести изображение (сердитые глаза). Затем робот начинает двигаться, а изображение остается на экране, пока выполняется первый блок **Рулевое управление** (Move Steering). Следующий блок **Экран** (Display) (заполненный круг) тоже очищает экран, удаляя изображение с сердитыми глазами перед тем, как вывести на экран изображение круга. Программа переходит к другому блоку **Экран** (Display), который выводит белый текст поверх круга. Этот блок не очищает экран, поэтому вы будете видеть на нем и круг, и текст. Наконец, блок **Рулевое управление** (Move Steering) поворачивает робота вправо, и программа завершается.

Индикатор состояния модуля

Индикатор состояния, расположенный вокруг кнопок на корпусе модуля EV3, обычно окрашен в зеленый цвет. Во время выполнения программы он мигает. С помощью блока **Индикатор состояния модуля** (Brick Status Light) вы можете изменить поведение подсветки. Доступны три режима, все они показаны на рис. 4.16.

- **Включить** (On): позволяет включить подсветку и выбрать ее цвет: зеленый (**0**), оранжевый (**1**) или красный (**2**). Параметр **Импульсный** (Pulse) позволяет выбрать, будет ли индикатор мигать (**Истина** (True)) или светиться непрерывно (**Ложь** (False)).
- **Выключить** (Off): отключает подсветку.
- **Сброс** (Reset): отображает мигающую зеленую подсветку, которую вы обычно видите при выполнении программы.

Давайте создадим программу ButtonLight, чтобы проверить работу этого блока, как показано на рис. 4.17. Индикатор должен светиться красным цветом, когда робот говорит: «Красный», и зеленым, когда он произносит: «Зеленый»*.

Режимы Включить и Выключить блоков действий

Теперь, когда у вас есть навык программирования с использованием некоторых блоков действий, вы готовы рассмотреть режимы **Включить** (On) и **Выключить** (Off), которые доступны

* Пункты **Red** и **Green** соответственно в группе **Цвета** (Colors).

ПРАКТИКУМ № 9: СВЕТОФОР!

Сложность: **Время:**

Измените программу ButtonLight так, чтобы превратить вашего робота в светофор. Создайте программу, с помощью которой робот говорил бы «Стоп», «Приготовьтесь» и «Поехали»*, демонстрируя красную, оранжевую и зеленую подсветку соответственно.

ПРАКТИКУМ № 10: САМОХОДНОЕ РАДИО!

Сложность: **Время:**

Измените программу, которую вы создали в практикуме № 6 на с. 64, так, чтобы робот ехал вперед во время воспроизведения мелодии. Используйте один блок **Рулевое управление** (Move Steering) в режиме **Включить** (On) в начале программы и еще один в режиме **Выключить** (Off) в конце. Что произойдет, если добавить дополнительные блоки **Рулевое управление** (Move Steering) в режиме **Включить** (On) (с различными настройками рулевого управления) в вашу программу?

в блоке **Рулевое управление** (Move Steering) и некоторых других блоках. Блок **Рулевое управление** (Move Steering) в режиме **Включить** (On) включит моторы, а затем программа сразу же перейдет к следующему блоку. Пока программа выполняется, моторы продолжают работать до тех пор, пока программа не достигнет блока, который отправит команду остановиться или сделать что-то другое. Режим **Выключить** (Off) выключает моторы. Чтобы разобраться в работе этих функций, создайте программу OnOff, показанную на рис. 4.18.

После запуска программы робот начнет двигаться и одновременно произнесет фразу «LEGO MINDSTORMS». Когда фраза закончится, движение прекратится и воспроизведется звуковой тон. Моторы должны перестать работать, в точности когда робот заканчивает говорить фразу «LEGO MINDSTORMS», несмотря на то что заранее не указано, как долго они должны работать. Это позволяет сделать так, чтобы робот двигался, пока вы не остановите его при помощи команды **Выключить** (Off), независимо от того, что он делает.

Если вы создаете программу с только одним блоком **Рулевое управление** (Move Steering) в режиме **Включить** (On), вы можете предположить, что робот будет двигаться вперед бесконечно. На самом деле, этот блок включает моторы, а затем программа завершается, поскольку заканчивает выполнение всех своих блоков; ну а когда программа завершает работу, моторы останавливаются.

* Пункты **Stop**, **Activate** и **Go** соответственно в группах **Связь** (Communication) и **Информация** (Information).

Блоки Независимое управление моторами, Большой мотор и Средний мотор

Помимо блоков **Рулевое управление** (Move Steering), существует еще три блока, которые отвечают за работу моторов. Первый из них — это блок **Независимое управление моторами** (Move Tank), который можно использовать для управления транспортным средством с двумя большими моторами или двумя колесами, таким как EXPLOR3R, который вы собрали раньше. Но вместо настройки параметров **Рулевое управление** (Steering) и **Мощность** (Power) для всего робота теперь вы можете выбрать мощность работы каждого мотора независимо друг от друга. Используя различные комбинации мощности для левого и правого колес, вы можете управлять EXPLOR3R так, чтобы он поворачивал в нужные вам стороны (см. рис. 4.7).

Программа Tank демонстрирует, как работает блок **Независимое управление моторами** (Move Tank) (рис. 4.19). Этот блок по функциональности аналогичен блоку **Рулевое управление** (Move Steering), но параметры устанавливаются по-другому. Используйте тот блок,

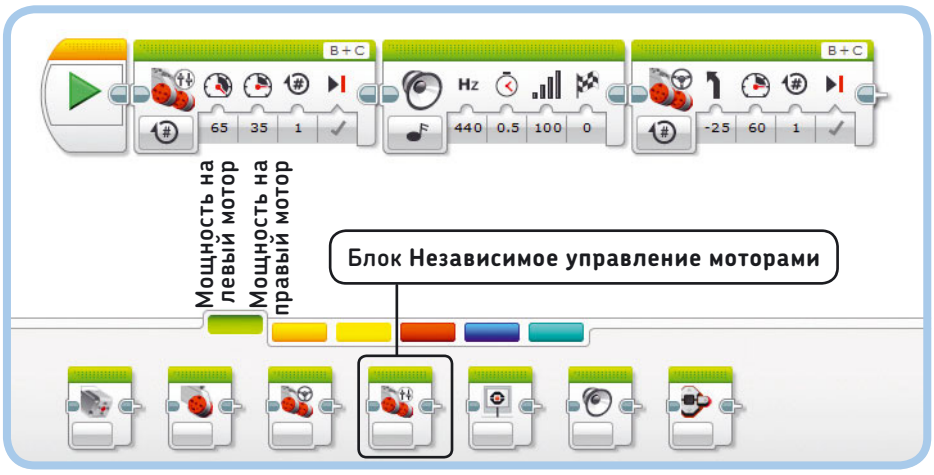


*Рис. 4.19. Программа Tank: робот делает плавный разворот вправо согласно инструкции блока **Независимое управление моторами**. Затем он останавливается, воспроизводит звук, и снова двигается, теперь на основе команд блока **Рулевое управление***

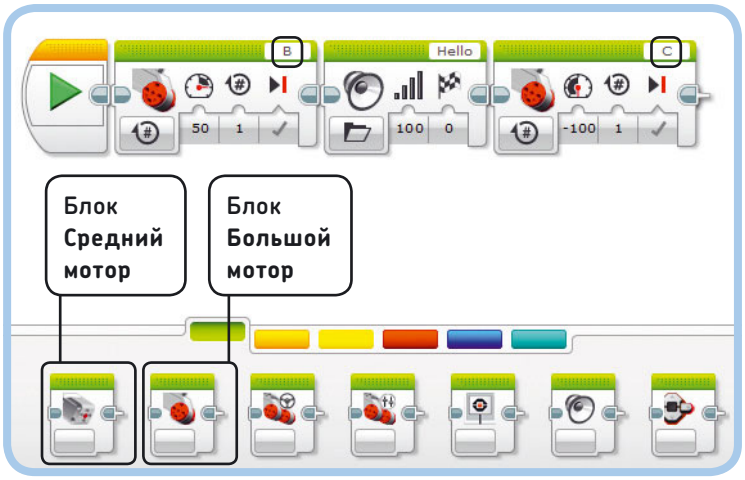


Рис. 4.20. Программа LargeMotor: Левый мотор вращается вперед, робот говорит «Привет!», правый мотор вращается в обратном направлении

ПРАКТИКУМ № 11: ВРЕМЯ КРУЖИТЬСЯ!

Сложность: **Время:**

Можете ли вы разработать программу, которая перемещала бы EXPLOR3R по кругу диаметром около метра? Для достижения этой цели вам потребуется только один блок **Рулевое управление** (Move Steering). Какое значение нужно присвоить параметру **Рулевое управление** (Move Steering) и как долго моторы должны работать? Как параметр **Рулевое управление** (Move Steering) влияет на диаметр окружности? Влияет ли на диаметр окружности изменение параметра **Мощность** (Power)? Когда вы сделаете это задание, попытайтесь достичь того же эффекта с блоком **Независимое управление моторами** (Move Tank).

ПРАКТИКУМ № 12: НАВИГАТОР!

Сложность: **Время:**

Разработайте программу на основе блоков **Рулевое управление** (Move Steering), которая задаст траекторию движения EXPLOR3R, изображенную на рис. 4.21. Во время движения робот должен отображать стрелки на экране модуля EV3, показывающие, куда он движется. Закончив, он должен отобразить знак «Стоп». В дополнение к отображению робот должен озвучивать направление, в котором он движется. Как вы настроите параметр **Тип воспроизведения** (Play Type) блоков **Звук** (Sound)?

СОВЕТ **Все знаки, указывающие направление, которые изображены на рис. 4.21, вы можете найти в разделе Информация (Information) раскрывающегося списка Имя файла (File Name) блока Экран (Display).**

который вы предпочитаете, при разработке своих программ для транспортных средств такого типа.

Если один из моторов запрограммирован так, чтобы вращаться быстрее другого (как в программе Tank), работа блока заканчивается, как только самый быстрый мотор сделает указанное число оборотов или градусов. То же самое справедливо и для блока **Рулевое управление** (Move Steering).

Для управления большим мотором вы можете использовать блок **Большой мотор** (Large Motor). Этот блок прекрасно подойдет для механизмов, оборудованных только одним мотором, например клешни, захватывающей объект. Режимы и параметры этого блока аналогичны блоку **Независимое управление моторами** (Move Tank), но применяются только к выбранному мотору.

Программа на рис. 4.20 содержит два блока **Большой мотор** (Large Motor), отвечающие по отдельности за вращение левого (B) и правого (C) моторов соответственно.

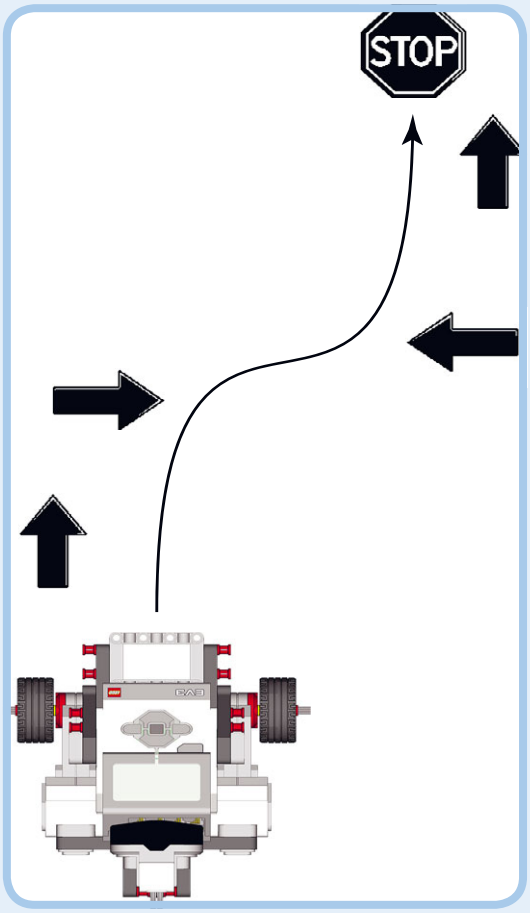


Рис. 4.21. Траектория движения и знаки, указывающие направление движения, для практикума № 12

ПРАКТИКУМ № 13: ТАНЦУЮЩИЙ РОБОТ

Сложность: **Время:**

С помощью блоков **Звук** (Sound), сделайте так, чтобы EXPLOR3R воспроизводил музыкальные фрагменты и тоны непрерывно, при этом двигаясь зигзагами (используйте блоки **Рулевое управление** (Move Steering)). После каждого поворота робот должен воспроизводить другой звук.

СОВЕТ **В некоторых блоках Звук (Sound) попробуйте параметру Тип воспроизведения (Play Type) присвоить значение Повторить (2) (Repeat (2)).**

СДЕЛАЙ САМ № 1: РОБОТ-УБОРЩИК!

Сборка: **Программирование:**

Добавьте нужные детали LEGO к вашему роботу, чтобы он мог держать тряпку для пыли на полу перед собой. Затем разработайте программу, используя блоки **Рулевое управление** (Move Steering), чтобы робот двигался и протирал пыль. Вместо использования программы вы можете удаленно управлять вашей уборочной машиной, как описано в главе 2. Уборка никогда не была такой веселой!

Блок **Средний мотор** (Medium Motor) аналогичен блоку **Большой мотор** (Large Motor), за исключением того, что используется для управления одним средним мотором, входящим в комплект набора LEGO MINDSTORMS EV3. Вы опробуете его в действии в главе 12.

СДЕЛАЙ САМ № 2: ИСКУССТВО С EXPLOR3R!

Сборка: **Программирование:**

В этом практикуме перед вами стоит задача дополнить конструкцию робота EXPLOR3R. Используя детали LEGO, создайте для вашего робота такую дополнительную конструкцию, чтобы он мог удерживать ручку. Перемещаясь по большому листу бумаги, робот будет ручкой рисовать линии и фигуры. Для начала вы можете нарисовать с помощью робота восьмерку, показанную в практикуме № 8 на с. 70.

Фиксация ручки на роботе, безусловно, подходит для рисования простых фигур, но в этом случае вы ограничены в том, что вы можете нарисовать или написать, потому что ручка не отрывается от бумаги. Используйте средний мотор из набора MINDSTORMS EV3, чтобы отрывать ручку от бумаги; с помощью кабеля подключите этот мотор к выходному порту A. Управлять этим мотором лучше всего с помощью блока **Средний мотор** (Medium Motor). Можете ли вы сделать так, чтобы робот написал ваше имя?

Когда закончите, сфотографируйте свою конструкцию и добавьте фотографии в редактор контента. Как это сделать, рассказано в главе 3.

Дальнейшее изучение

Вы изучили основы программирования с помощью набора LEGO MINDSTORMS EV3. Примите мои поздравления! Теперь вы умеете программировать роботов, чтобы они двигались, воспроизводили звуки, мигали световыми индикаторами и отображали текст и изображения на экране модуля EV3. В главе 5 вы узнаете больше об использовании программных блоков, в том числе как приостанавливать выполнение программы и повторно выполнять инструкции последовательности блоков.

Но, прежде чем двигаться дальше, попытайтесь решить некоторые из практических задач, которые представлены ниже, чтобы закрепить свои навыки программирования.

ПРИМЕЧАНИЕ **Не забудьте сохранить свои программы после того, как выполните задачи. Позднее вы сможете использовать их в качестве основы для разработки более сложных программ.**

5

Ожидание, повторение, контейнеры и многозадачность

В предыдущей главе вы изучили, как запрограммировать робота на выполнение различных действий, например на передвижение. В этой главе вы узнаете, как приостановить выполнение программы с помощью блока **Ожидание** (Wait), как повторить последовательность действий с помощью блока **Цикл** (Loop), как запустить несколько блоков одновременно и даже как создать собственный блок.

Блок Ожидание

До сих пор вы использовали блоки, которые программируют робота на движение, воспроизведение звуков или вывод информации на экран. Теперь речь пойдет о блоке, который ничего не делает, кроме приостановки выполнения программы в течение заданного промежутка времени. Этот блок носит название **Ожидание** (Wait), он показан на рис. 5.1.

Настройки ожидания

Работа с блоком **Ожидание** (Wait) выполняется так же, как и с любым другим программным блоком: поместите его в область программирования, а затем настройте режим его работы и параметры. В этой главе вы будете использовать только один режим — **Время** (Time).

В режиме **Время** (Time) блок **Ожидание** (Wait) приостанавливает выполнение программы на определенное количество времени, например пять секунд. После того как время истекло, программа переходит к следующему программному блоку. Количество секунд, которое вы указываете как значение параметра **Секунды** (Seconds), может быть либо целым числом, например 14, либо десятичным, например 1,5. Чтобы приостановить выполнение программы на 50 миллисекунд (0,05 секунды), вы должны ввести 0,05.

Использование блока Ожидание

Зачем нужно использовать блок, который ничего, кроме ожидания, не делает? Давайте взглянем на блок **Ожидание** (Wait)

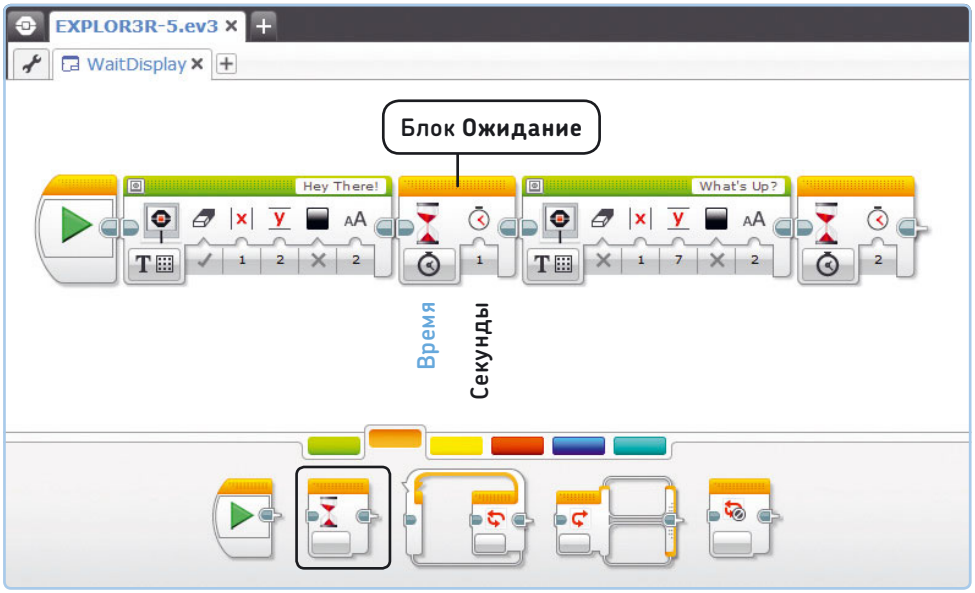


*Рис. 5.1. Программа WaitDisplay содержит два блока **Ожидание**, работающих в режиме **Время** и позволяющих приостановить выполнение программы после отображения текста на экране модуля EV3*

в действии. Создайте новый проект, сохраните его под именем EXPLOR3R-5, а в нем создайте программу WaitDisplay в соответствии с инструкциями, приведенными на рис. 5.1.

Программа WaitDisplay

При запуске программы на экране модуля EV3 должен появиться текст «Hey There!» и следом, через секунду, «What's Up?». Второй блок **Ожидание** (Wait) дает вам время, чтобы успеть прочитать текст, который отображается на экране модуля EV3. Без него программа завершит работу сразу после того, как текст будет выведен на экран, поэтому прочитать сообщение будет невозможно. Блоки **Ожидание** (Wait) могут пригодиться при программировании робота с датчиками, в чем вы убедитесь в следующей главе.

ПРАКТИКУМ № 14: ОСТАВЬТЕ СООБЩЕНИЕ!

Сложность: **Время:**

Дополните программу WaitDisplay так, чтобы робот отображал сообщение, в котором говорится, куда вы отправились, когда вышли из дома. Добавьте блоки **Ожидание** (Wait) между строк текста, чтобы ваше сообщение было удобно читать. Не забудьте проинструктировать членов семьи, как получить доступ к сообщению!

ПРАКТИКУМ № 15: ТАЙМЕР ДЛЯ НАСТОЛЬНЫХ ИГР!

Сложность: **Время:**

Создайте программу, отображающую на экране модуля EV3 таймер, который можно использовать во время настольных игр. Экранный таймер должен показывать, сколько времени осталось для каждого хода (рис. 5.2). Когда время истекло, робот должен сказать «Game Over!», сообщая таким образом, что ход окончен и настала очередь следующего игрока. В качестве альтернативы вы можете использовать блоки **Звук** (Sound), чтобы ваш робот говорил, сколько осталось времени.

СОВЕТ **Используйте несколько блоков Экран (Display) в режиме Изображение (Image) с блоками Ожидание (Wait) между ними. Выбирайте пункты Timer 0, Timer 1 и так далее в списке изображений.**

Рис. 5.2. Таймер для настольных игр из практикума № 15

Блок Цикл

Представьте, что вы идете по траектории квадратной формы, как показано на рис. 5.3. При этом вы повторяете определенный шаблон несколько раз: идете прямо, затем поворачиваете направо, идете прямо, поворачиваете направо и так далее.

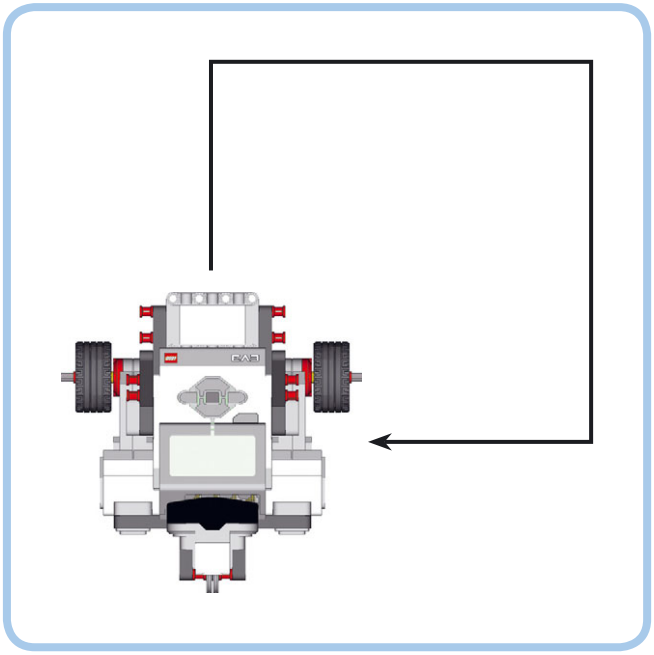


Рис. 5.3. The EXPLOR3R движется по траектории квадратной формы

Для создания такого шаблона движения для вашего робота вы можете использовать один блок **Рулевое управление** (Move Steering), чтобы робот шел прямо, а второй, чтобы он поворачивал направо. Чтобы робот прошел полный квадрат и вернулся в исходное положение, вы должны использовать каждый блок четыре раза, в общей сложности восемь блоков.

Вместо того чтобы использовать восемь блоков **Рулевое управление** (Move Steering), чтобы создать такую программу, гораздо проще применить блок **Цикл** (Loop), который позволяет повторять последовательности блоков, размещенных внутри него. Блоки **Цикл** (Loop) особенно пригодятся в случае, если вы хотите повторить определенные действия много раз.

Использование циклов

Блок **Цикл** (Loop) (рис. 5.4) многократно запускает блоки, которые вы поместили в него. В зависимости от режима, который вы выбираете, он запускает эти блоки либо заданное количество раз, либо в течение определенного количества секунд, либо до тех пор, пока вы вручную не прервете

выполнение программы на модуле EV3. В последующих главах вы научитесь использовать многие из этих режимов.

В верхней части каждого блока **Цикл** (Loop) вы можете указать имя цикла для описания функций блоков, которые вы поместили в него. Вы можете вручную изменить размер блока, если это необходимо, как показано на рис. 5.4. Существует также функция **Параметр цикла** (Loop Index), о которой вы узнаете в главе 14, а пока можете не обращать на нее внимания.

*Рис. 5.4. Блок **Цикл** в режиме **Подсчет**. В данной конфигурации программа будет запускать любые блоки, размещенные в пределах этого цикла, четыре раза. В других режимах блоки, расположенные внутри цикла, могут повторяться в течение указанного количества секунд или неограниченного времени*

Чтобы поместить выбранные блоки в цикл, просто перетащите один блок или их последовательность в блок **Цикл** (Loop), как показано на рис. 5.5.

Блок Цикл в действии

Чтобы попробовать блок **Цикл** (Loop) в действии, создайте программу OneSquare, показанную на рис. 5.6. При ее запуске робот должен воспроизвести звуковой тон, проехать по квадратной траектории, воспроизвести другой звук, а затем остановиться. Если ваш робот не выполняет поворот на 90 градусов, попробуйте изменить значение в градусах во втором блоке **Рулевое управление** (Move Steering) так, как вы делали это в практикуме № 2 на с. 61

Вложенные циклы

Программа OneSquare (рис. 5.6) задает роботу EXPLOR3R квадратную траекторию движения.

Вы можете использовать еще один блок **Цикл** (Loop), зациклив движение по траектории, чтобы робот несколько раз повторил заданный маршрут. В режиме **Неограниченный** (Unlimited) робот будет продолжать движение по квадрату бесконечно. Попробуйте реализовать такое поведение с помощью программы InfiniteSquare, показанной на рис. 5.7. Чтобы это сделать, дополните программу, которую вы только что создали, добавив второй блок **Цикл** (Loop) из палитры программирования и установив режим **Неограниченный** (Unlimited). Перетащите цикл Square, созданный ранее, и второй блок **Звук** (Sound) в новый блок **Цикл** (Loop). Теперь робот будет двигаться по квадратной траектории, говоря «Пока» после завершения каждого квадрата, пока вы не завершите программу нажатием кнопки «Назад» на модуле EV3.

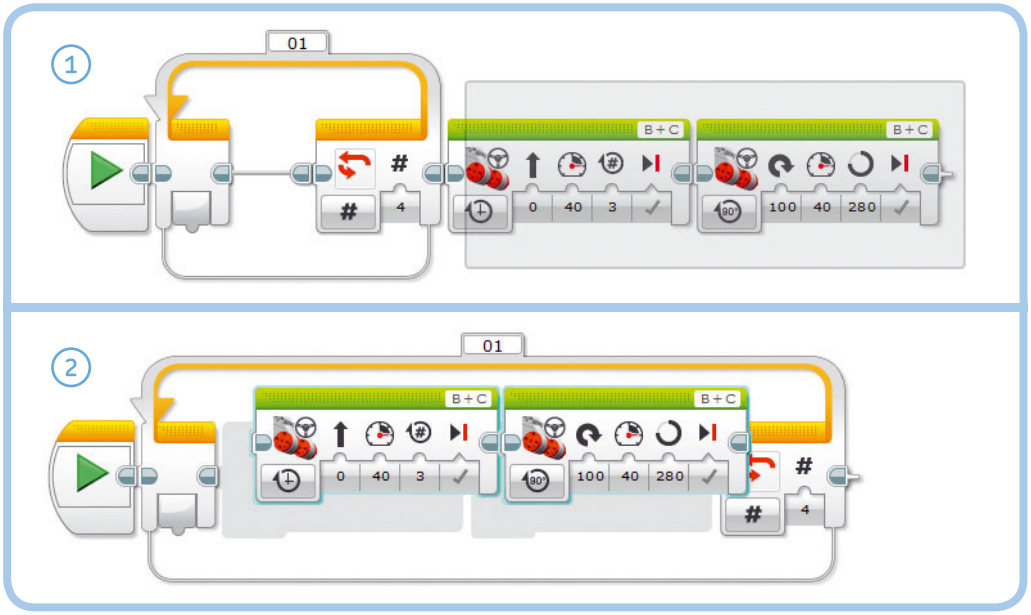


*Рис. 5.5. Чтобы разместить блоки внутри цикла, сначала поместите все необходимые блоки в область программирования (1). Затем выделите блоки, которые вы хотите поместить в цикл, и перетащите их в блок **Цикл** (Loop) (2). При этом блок **Цикл** (Loop) автоматически изменит размер, чтобы создать пространство для перетаскиваемых блоков. При перемещении блока **Цикл** (Loop) по области программирования его содержимое остается внутри него*

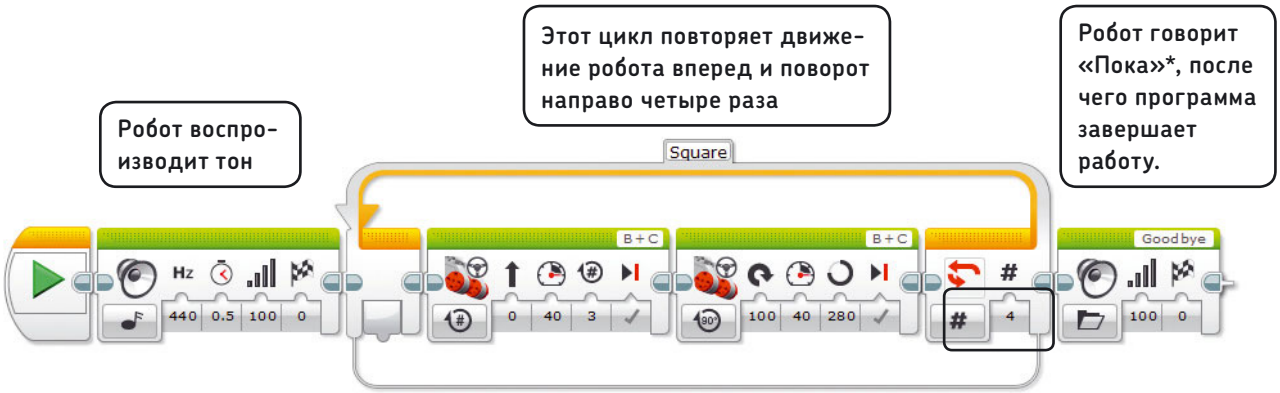


*Рис. 5.6. Программа OneSquare содержит цикл, который выполняется четыре раза. После того как два блока внутри цикла были запущены по четыре раза (результат — движение по квадратной траектории), программа переходит к следующему блоку, в данном случае **Звук** (Sound). В поле имени цикла вы можете ввести слово Square**, как показано на рисунке, чтобы описать предназначение этого цикла*

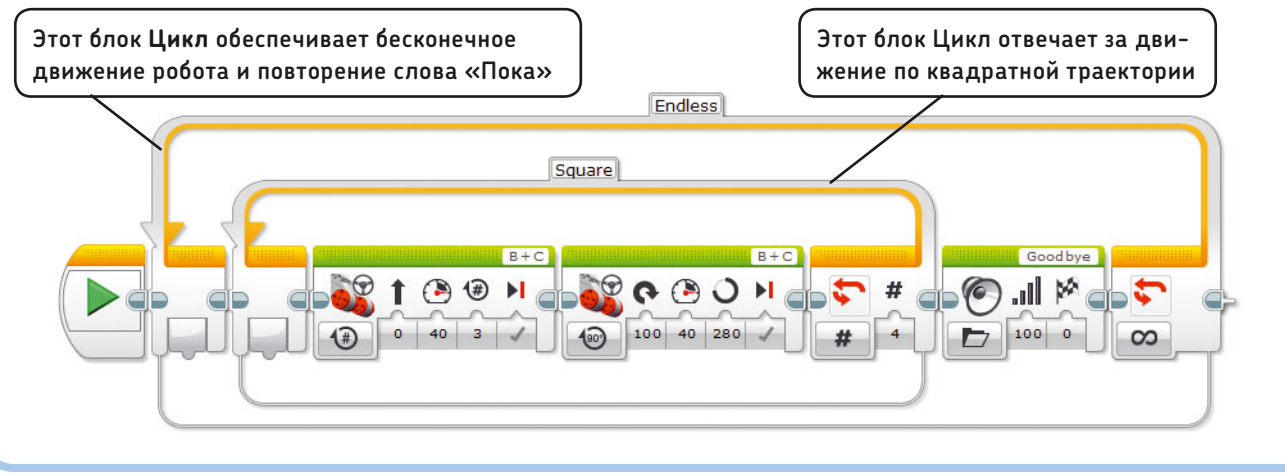


*Рис. 5.7. Программа InfiniteSquare показывает, как один блок **Цикл** вкладывается в другой. Вложенный (внутренний) цикл обеспечивает движение робота EXPLOR3R по квадратной траектории, а внешний отвечает за бесконечное повторение роботом этого перемещения и воспроизведение слова «Пока»*

ПРАКТИКУМ № 16: ОХРАНА КОМНАТЫ

Сложность: **Время:**

Разработайте программу, которая позволит роботу EXPLOR3R постоянно двигаться вперед и назад перед дверью вашей спальни, как будто охраняя ее (рис. 5.8). Используйте один блок **Цикл** (Loop) в режиме **Неограниченный** (Unlimited) и два блока **Рулевое управление** (Move Steering) — один для движения вперед и второй для разворота.

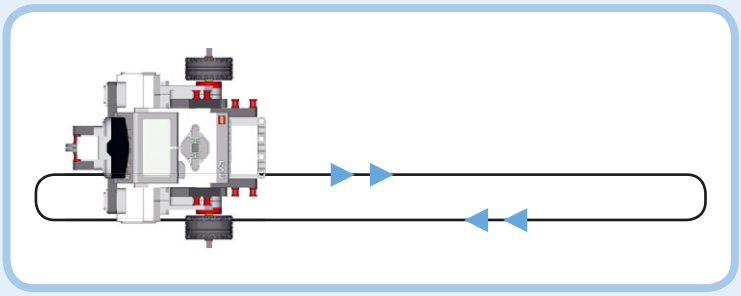


Рис. 5.8. Траектория движения робота EXPLOR3R в практикуме № 16

* Пункт **Goodbye** в группе **Связь** (Communication).
** Квадрат *(англ.)*.

ПРАКТИКУМ № 17: ТРЕУГОЛЬНИК!

Сложность: **Время:**

Вы разработали программу, которая обеспечивает вашему роботу движение по квадратной траектории. Как вы могли бы изменить программу OneSquare, чтобы траектория движения робота стала треугольной? А как насчет шестиугольника? Используйте дополнительные блоки **Цикл** (Loop), чтобы повторить каждую траекторию пять раз.

СОВЕТ **В практикуме № 2 на с. 61 вы определили количество градусов, на которое должны повернуться колеса, чтобы робот сделал поворот на 180 градусов. Можете ли вы использовать это значение для расчета количества градусов, необходимых для поворота на 120 градусов, чтобы робот мог ездить по треугольной траектории?**

Создание контейнеров «Мой блок»

В дополнение к использованию предустановленных блоков вы можете создавать свои, называемые *моими блоками*. В этой книге мы назовем их *контейнерами «Мой блок»*, чтобы избежать путаницы, к тому же *мои блоки* представляют собой, по сути, контейнеры, внутри которых расположены предустановленные блоки. Каждый контейнер «Мой блок» позволяет объединить несколько стандартных программных блоков в один. Контейнеры «Мой блок» особенно удобны, если вы хотите использовать определенную последовательность блоков в вашей программе более одного раза. Например, вы можете создать контейнер с инструкциями, чтобы робот говорил: «Привет! Доброе утро!» и изменял цвет индикатора состояния модуля на красный всякий раз, когда вы используете этот контейнер. В обычной ситуации, чтобы робот все это делал, требуется пять программных блоков. Если вам нужно, чтобы эту комбинацию действий робот совершал несколько раз, вместо того чтобы каждый раз помещать все пять блоков, гораздо проще создать один контейнер «Мой блок», сочетающий в себе эти блоки, который можно повторно использовать. Кроме того, использование контейнера «Мой блок» поможет вам в организации вашей программы, поскольку на экране будет меньше блоков.

Контейнеры «Мой блок» на практике

Чтобы продемонстрировать функциональность контейнера «Мой блок», давайте напишем программу, с помощью которой EXPLOR3R будет говорить: «Привет! Доброе утро!»* — двигаться вперед, а затем говорить то же самое еще раз. Поскольку робот будет приветствовать вас два раза, вы создадите контейнер «Мой блок» с именем **Talk**, чтобы упростить повторение этого действия, как показано на рис. 5.9–5.11. После того как вы создадите свой контейнер «Мой блок», вы сможете помещать его в программу всякий раз, когда требуется, чтобы робот EXPLOR3R говорил: «Привет! Доброе утро!»

1. Создайте новую программу под названием MyBlockDemo, в области программирования которой разместите и настроите пять блоков для создания приветствия робота (рис. 5.9). Затем выделите эти пять блоков (контур выделенных блоков окрашен в синий цвет) и выберите команду меню **Инструменты** ▸ **Конструктор Моего Блока** (Tools ▸ My Block Builder).
2. Появится окно **Конструктор Моего Блока** (My Block Builder), показанное на рис. 5.10. Присвойте контейнеру «Мой блок» имя, например **Talk**, в поле **Имя** (Name). Используйте поле **Описание** (Description), чтобы описать ваш контейнер. Это поможет вспомнить его предназначение, когда вы захотите использовать его повторно. И, наконец, выберите значок, например в виде динамика, чтобы создать ассоциацию, что этот контейнер используется для воспроизведения звуков. Нажмите кнопку **Завершить** (Finish).
3. После создания контейнер «Мой блок» появится в области программирования, заменив собой блоки, которые там находились изначально, как показано на рис. 5.11. Поменяйте блоки местами, если возникнет такая необходимость.

Использование контейнеров «Мой блок» в программах

Теперь, когда контейнер «Мой блок» готов, вы можете найти его на светло-голубой вкладке палитры программирования, как показано на рис. 5.12. Добавьте блок **Рулевое управление** (Move Steering) и еще одну копию контейнера «Мой блок», чтобы завершить программу MyBlockDemo. При запуске программы робот должен произнести «Привет! Доброе утро!», проехать вперед, а затем поприветствовать вас снова. Во время воспроизведения звука индикатор состояния должен быть красным, а во время движения робота — зеленым.

Обратите внимание, как использование контейнеров «Мой блок» упрощает программу для понимания или объяснения другу. По этой причине иногда полезно заключить блоки вашей программы в несколько контейнеров «Мой блок», даже если вы используете их всего один раз.

Изменение контейнеров «Мой блок»

После того как вы создали контейнер «Мой блок», вы можете поменять блоки, которые в нем находятся. Для этого дважды

* Пункты **Hello**, **Good**, **Morning** в группе **Связь** (Communication).

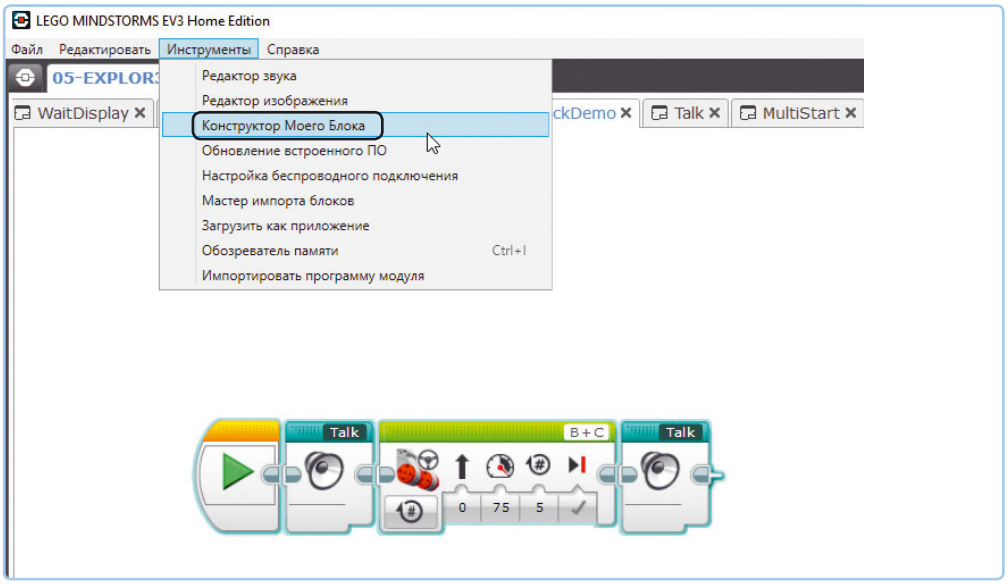


*Рис. 5.9. Поместите блоки в область программирования и настройте их, как показано на рисунке. Затем заключите в рамку выделения все блоки, кроме оранжевого блока **Начало**, и выберите команду меню **Инструменты** ▸ **Конструктор Моего Блока***

щелкните мышью по контейнеру «Мой блок» в области программирования, чтобы показать его содержимое, а затем измените его, как если бы это была обычная программа. Когда вы закончите, щелкните мышью по значку × на вкладке с именем контейнера, чтобы сохранить изменения и вернуться в программу, в которой используется данный контейнер «Мой блок».

Чтобы переименовать контейнер «Мой блок», дважды щелкните мышью по его имени на вкладке и измените текст.

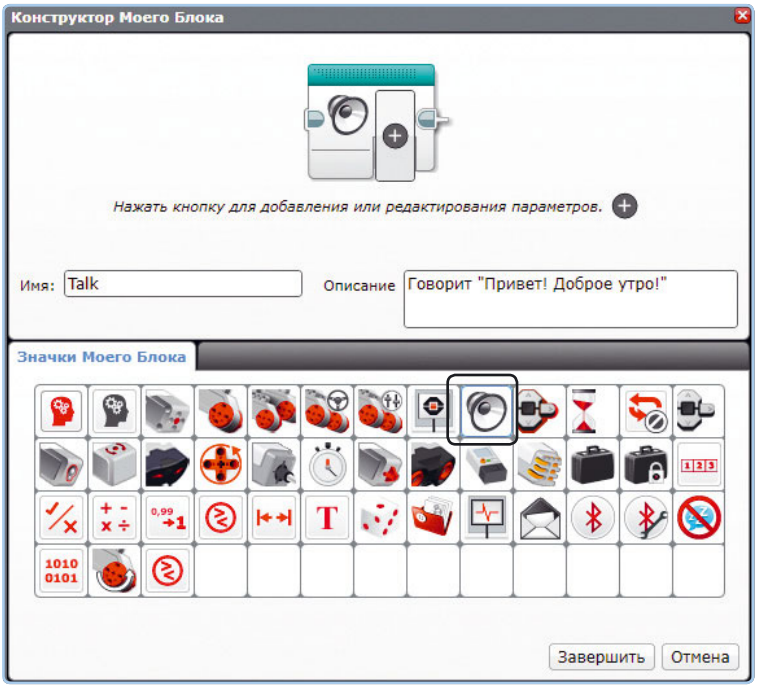


*Рис. 5.10. Окно **Конструктор Моего Блока**. Укажите имя, описание и выберите значок для вашего контейнера «Мой блок». Затем нажмите кнопку **Завершить**, чтобы завершить создание контейнера*

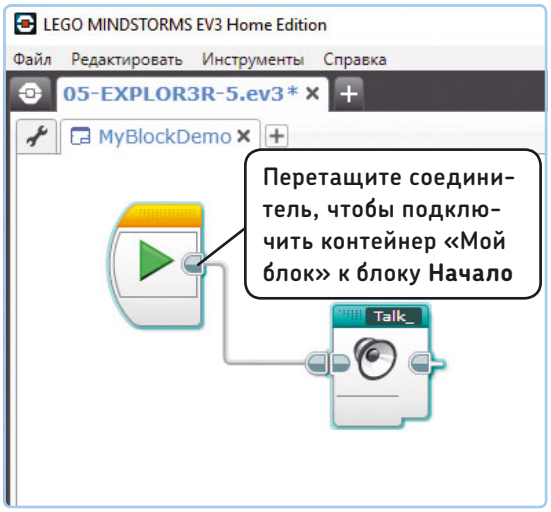


*Рис. 5.11. Законченный контейнер «Мой блок» в области программирования. Если созданный контейнер не примыкает к блоку **Начало**, нажав и удерживая кнопку мыши на правом выходном разъеме блока **Начало**, перетащите соединитель к входному разъему контейнера*

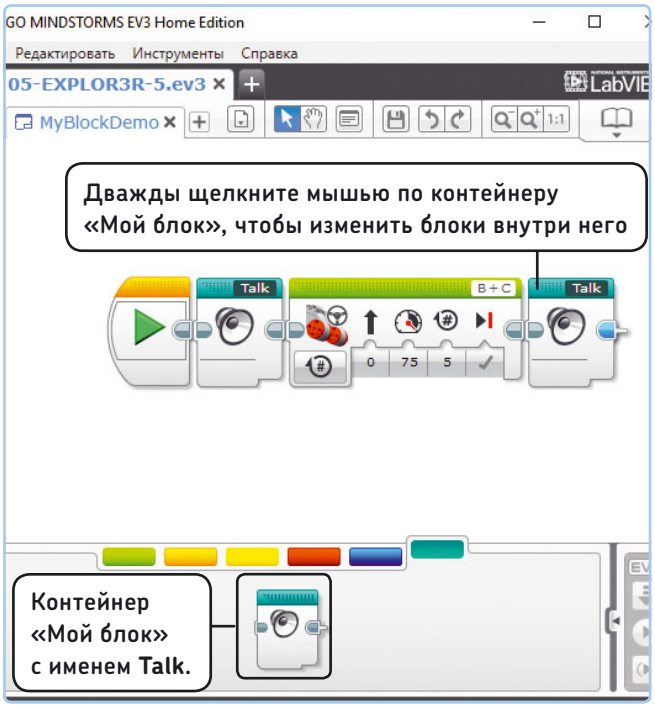


Рис. 5.12. Готовая программа MyBlockDemo

Таким же образом вы переименовываете обычные программы, как показано на рис. 3.11 в главе 3.

Управление контейнерами «Мой блок» в проектах

Вы можете использовать созданные контейнеры «Мой блок» в любой программе в рамках того же проекта. Например, контейнер **Talk** можно использовать в программах в файле проекта EXPLOR3R-5. Но может возникнуть ситуация, когда потребуется использовать контейнеры «Мой блок» в других проектах.

Чтобы скопировать в другой проект контейнер «Мой блок», который вы только что сделали (как показано на рис. 5.13), перейдите на страницу **Свойства проекта** (Project Properties) (1), выберите пункт **Talk.EV3p** на вкладке **Мои блоки** (My Blocks) (**2**), а затем нажмите кнопку **Копировать** (Copy) (**3**). Теперь, не закрывая текущий проект, откройте созданный ранее проект: EXPLOR3R-4 (**4**). Перейдите на вкладку **Мои блоки** (My Blocks) этого проекта на странице **Свойства проекта** (Project Properties) (**5**) и нажмите кнопку **Вставить** (Paste) (**6**). Теперь вы можете использовать контейнер **Talk** в проекте EXPLOR3R-4.

Вместо копирования вы можете нажать кнопку **Экспорт** (Export), которая позволяет сохранить контейнер «Мой блок» в файл. Сохраненный таким образом файл вы можете передать другу по электронной почте, чтобы он добавил этот файл в свой проект, используя кнопку **Импорт** (Import). Чтобы удалить контейнер «Мой блок» из проекта, выберите его и нажмите кнопку **Удалить** (Delete).

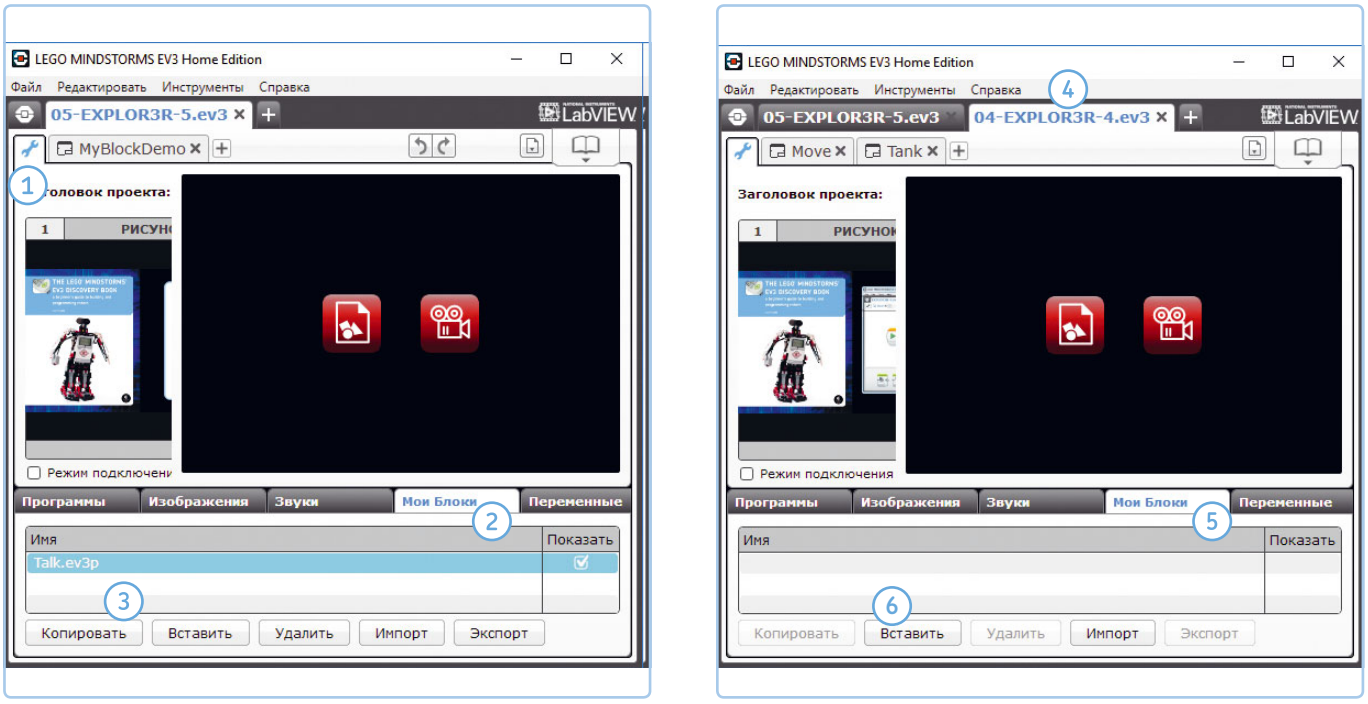


Рис. 5.13. Копирование контейнера «Мой блок» из проекта EXPLOR3R-5 в проект EXPLOR3R-4

ПРАКТИКУМ № 18: МОЙ КВАДРАТ!

Сложность: **Время:**

Откройте программу OneSquare, показанную на рис. 5.6, и поместите блок **Цикл** (Loop) и его содержимое в контейнер «Мой блок», присвоив ему имя **MySquare**. Теперь вы можете использовать этот контейнер всякий раз, когда захотите, чтобы ваш робот ездил по квадратной траектории.

ПРАКТИКУМ № 19: МОЯ МЕЛОДИЯ!

Сложность: **Время:**

Помните мелодию, которую вы сделали с помощью блоков **Звук** (Sound) в практикуме № 6 на с. 64? Преобразуйте эту последовательность блоков **Звук** (Sound) в контейнер «Мой блок», и вы сможете в любое время использовать вашу любимую мелодию в своих программах.

ПРИМЕЧАНИЕ **Вы можете использовать тот же метод для обмена между проектами другими файлами, такими как отдельные программы, пользовательские звуки и изображения.**

Многозадачность

Все блоки, которые вы использовали до сих пор, выполняются по одному в том порядке, в котором они расположены в области программирования. Тем не менее модуль EV3 поддерживает многозадачность, позволяя выполнять инструкции нескольких блоков одновременно. Для этого применяются либо несколько блоков **Начало** (Start) или разделенные соединители, называемые *шинами последовательности*. Вы увидите, что эти методы очень похожи.

Использование нескольких блоков Начало

Самый простой способ заставить работать параллельно (одновременно) две последовательности блоков заключается в добавлении второго блока **Начало** (Start), как показано на рис. 5.14. После нажатия кнопки **Загрузить и запустить** (Download and Run) обе последовательности начинают работать одновременно. Программа заканчивается, когда обе последовательности блоков завершают работу. Чтобы проверить работу одной последовательности без участия второй, нажмите кнопку в виде зеленой стрелки на соответствующем блоке **Начало** (Start).

Когда вы запустите эту программу, робот начнет двигаться и воспроизводить звук.

Параллельное соединение блоков

Еще один способ реализовать многозадачность заключается в разделении соединителей (шин последовательности), как

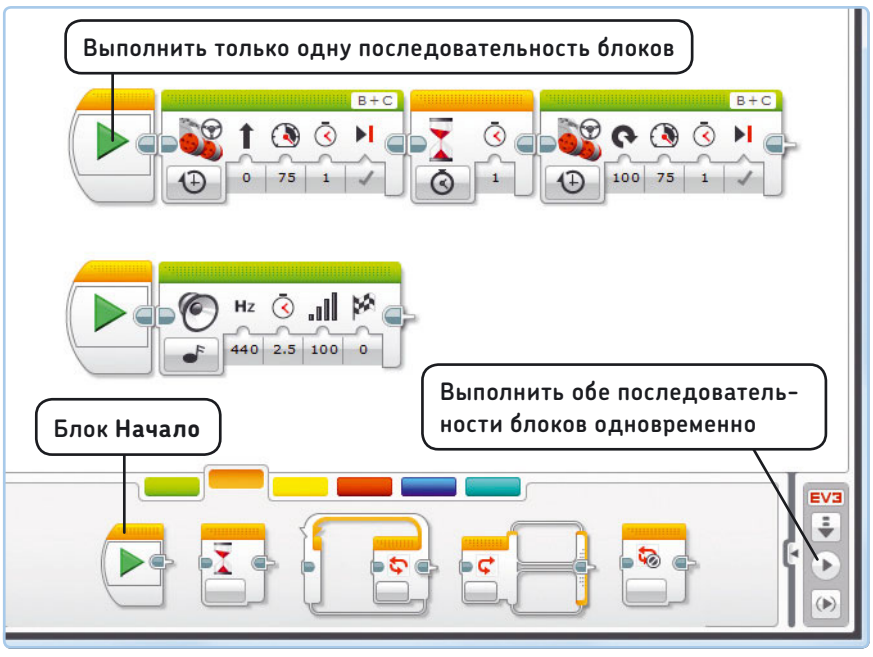


*Рис. 5.14. Реализация многозадачности с помощью двух блоков **Начало** в программе MultiStart. Блок **Начало** доступен на оранжевой вкладке вместе с другими блоками управления операторами*

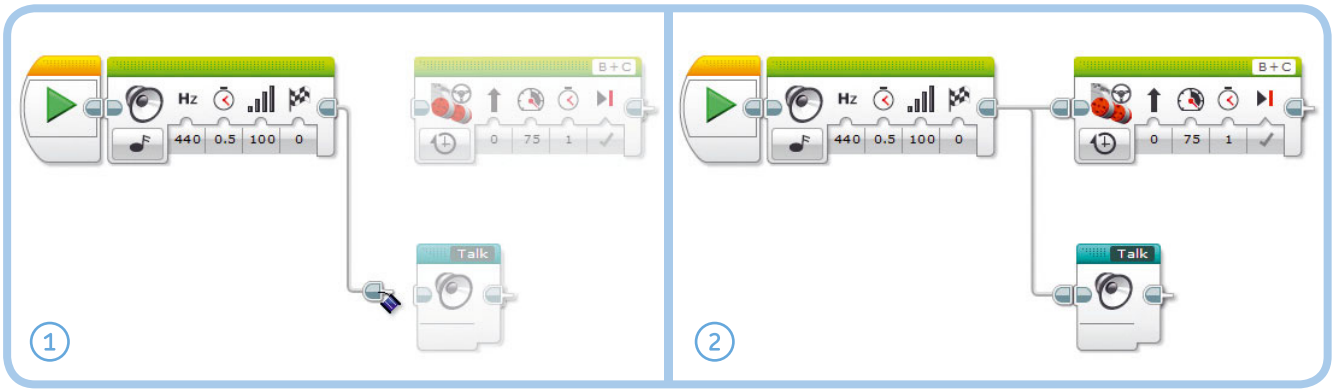


*Рис. 5.15. Реализация многозадачности с помощью соединителей (шин последовательности) в программе MultiSequence. Чтобы достичь этого результата, сначала поместите все необходимые блоки в область программирования, а затем подключите соединителями каждую последовательность блоков. Это возможно, даже если блоки **Звук** и **Рулевое управление** состыкованы. Блоки будут разделены автоматически, когда вы подключите контейнер **Talk***

показано на рис. 5.15. Такой вариант может пригодиться, если вам нужны две параллельные последовательности блоков, которые включаются не с начала программы. В программе MultiSequence показано, как робот воспроизводит мелодию, а затем два действия происходят одновременно: робот едет вперед и произносит «Привет! Доброе утро!» с помощью контейнера **Talk**, который вы создали ранее.

Избегание конфликтов ресурсов

Точно так же, как вы не можете одновременно идти и вперед, и назад, вы не можете запрограммировать робота в одной последовательности двигаться в одном направлении, а в другой — в противоположном направлении. В этом случае возникает *конфликт ресурсов*, когда две последовательности блоков пытаются управлять одним мотором или датчиком одновременно.

К сожалению, программное обеспечение EV3 не сообщит вам, возникает ли конфликт ресурсов в вашей программе; вероятнее всего, программа продолжит работать, но результат будет непредсказуемым. Например, робот не сможет двигаться в заданном направлении. Чтобы избежать этой проблемы, не используйте один и тот же мотор или датчик более чем в одной последовательности.

По возможности старайтесь избегать многозадачности, потому что конфликты ресурсов могут возникнуть неожиданно. Иногда вы можете выполнить ту же задачу, используя только одну последовательность блоков. Например, вместо перемещения и воспроизведения звуков с помощью двух параллельных последовательностей, вы можете выполнить оба действия из одной последовательности, как вы увидите в практикуме № 21.

ПРАКТИКУМ № 20: ДА ЗДРАВСТВУЕТ МНОГОЗАДАЧНОСТЬ!

Сложность: **Время:**

Запрограммируйте робота на бесконечное движение по квадратной траектории и одновременное повторение слов «LEGO, MINDSTORMS, EV3».

Дальнейшее изучение

Завершив изучение первой части этой книги, вы получили прочную основу из нескольких важнейших методов программирования. В этой главе вы узнали, как использовать блоки **Ожидание** (Wait) и **Цикл** (Loop), как создавать и редактировать контейнеры «Мой блок» и как сделать вашего робота многозадачным.

В следующей части книги вы будете создавать роботов, которые могут взаимодействовать с окружающей средой с помощью датчиков. Но прежде чем заняться этим, немного потренируйтесь с навыками, которые вы приобрели в этой главе. Для этого решите следующие практические задачи.

ПРАКТИКУМ № 21: ОДНОЗАДАЧНОСТЬ!

Сложность: **Время:**

Программа MultiStart — простой пример, который иллюстрирует принцип многозадачности. Но не всегда необходимо использовать несколько последовательностей для одновременного движения и воспроизведения звуков. Можете ли вы создать новую программу, которая делает то же самое, что и программа MultiStart, только с одной последовательностью блоков?

Чтобы усложнить задачу, сделайте то же самое для программы MultiSequence.

СОВЕТ **Как это сделать, вы узнали в главе 4. А за что отвечает параметр Тип воспроизведения (Play Type) на блоке Звук (Sound)?**

СДЕЛАЙ САМ № 3: ГОСПОДИН EXPLOR3R!

Сборка: **Программирование:**

Добавьте средний мотор, чтобы создать машущую руку для EXPLOR3R. Используйте дополнительные детали LEGO для дальнейшего украшения робота и превращения его в господина EXPLOR3R. Запрограммируйте его постоянно махать рукой и одновременно произносить приветствие, например «Доброе утро!».

ПРАКТИКУМ № 22: СЛОЖНЫЕ ФИГУРЫ!

Сложность: **Время:**

Разработайте программу, согласно которой робот EXPLOR3R будет двигаться по шаблону, показанному на рис. 5.16, воспроизводя различные звуки.

СОВЕТ **Если вы посмотрите внимательно, то увидите, что предлагаемый путь можно разделить на четыре одинаковые части, поэтому можно настроить последовательность блоков Рулевое управление (Move Steering) только для одной из этих частей. Затем вы можете поместить эти блоки в цикл, настроенный на четырехразовое повторение.**

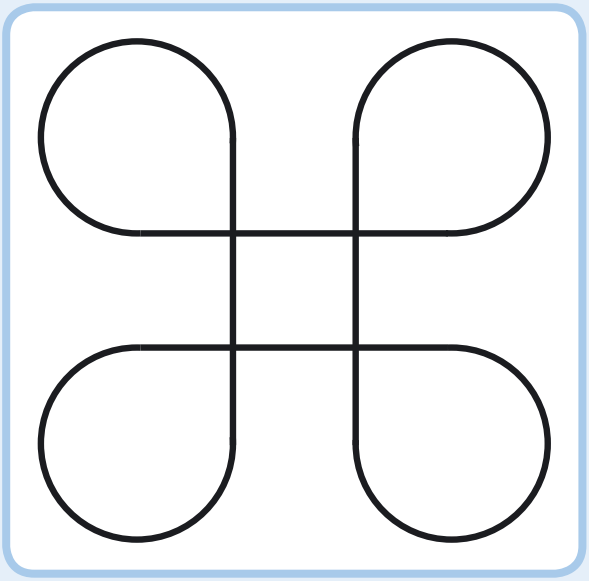


Рис. 5.16. Траектория движения для практикума № 22

ЧАСТЬ II

Программирование роботов с датчиками

6

Предназначение датчиков

Набор LEGO MINDSTORMS EV3 включает в себя три типа датчиков: *касания*, *цвета* и *инфракрасный*. Вы можете использовать эти датчики, чтобы ваш робот реагировал на окружающую его среду. Например, вы можете запрограммировать робота, чтобы он воспроизводил звук, когда видит вас, избегал препятствий во время перемещения или двигался по цветным линиям. В этой части книги вы научитесь создавать роботов, которые используют эти датчики.

Чтобы узнать, как работать с датчиками, мы дополним робота EXPLOR3R, добавив бампер, который определяет препятствия с помощью датчика касания, как показано на рис. 6.1. После того как вы научитесь создавать программы, которые используют датчик касания, в последующих главах вы узнаете, как использовать и другие датчики.

Рис. 6.1. Робот EXPLOR3R использует бампер с датчиком касания для определения объектов, в сторону которых он движется

Что такое датчики?

Роботы LEGO MINDSTORMS не способны видеть или чувствовать, как люди, но, если снабдить их *датчиками*, они смогут собирать и передавать информацию об окружающей среде. При разработке программ, которые интерпретируют информацию, получаемую с датчиков, вы можете сделать так, чтобы ваш робот казался разумным, запрограммировав его на определенные реакции. Например, можно создать программу, благодаря которой робот будет говорить «синий», когда датчик определяет синюю бумагу.

Общее представление о датчиках в наборе MINDSTORMS EV3

Ваш набор MINDSTORMS EV3 содержит три датчика, которые можно прикрепить к роботу (рис. 6.2), а также некоторые встроенные датчики. *Датчик касания* определяет, нажата ли красная кнопка на датчике или отпущена. *Датчик цвета* определяет цвет поверхности и яркость источника света, о чем вы подробнее узнаете в главе 7. *Инфракрасный датчик* (рассматривается в главе 8) измеряет приблизительное расстояние до соседнего объекта, а также принимает сигналы от удаленного инфракрасного маяка.

Кроме того, каждый мотор в комплекте EV3 имеет встроенный *датчик вращения мотора* для определения позиции мотора и скорости, а модуль EV3 может определить, какие из его кнопок нажаты (см. главу 9).

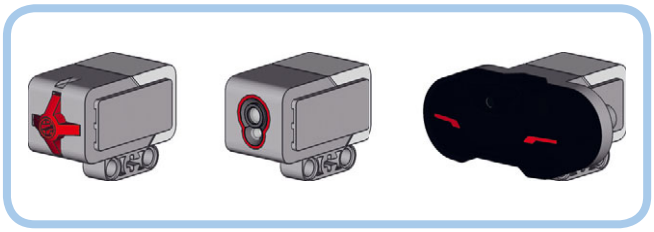


Рис. 6.2. Набор EV3 комплектуется датчиком касания (слева), датчиком цвета (средний) и инфракрасным датчиком (справа)

Обзор датчика касания

Датчик касания позволяет роботу «чувствовать», определяя, нажата или отпущена в настоящий момент красная кнопка на датчике, как показано на рис. 6.3. Модуль EV3 получает эту информацию от датчика и может использовать ее в программах. Например, вы можете запрограммировать вашего робота говорить «Привет!» всякий раз, когда вы нажимаете кнопку датчика касания.

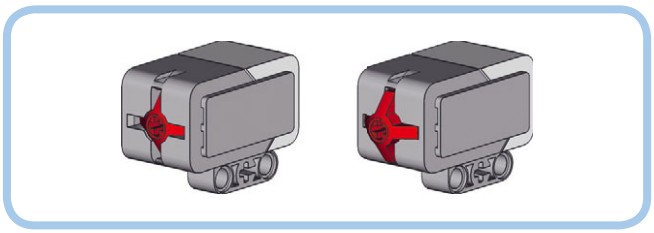
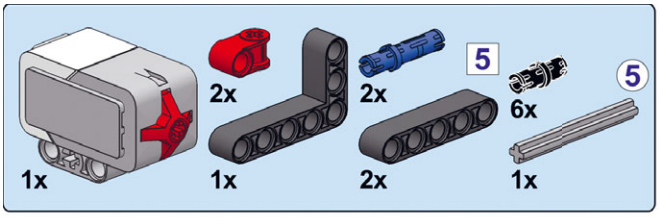


Рис. 6.3. Датчик касания определяет, нажата ли красная кнопка (слева) или отпущена (справа)

Несмотря на свою простоту, датчик касания полезен для многих приложений. Например, роботы могут использовать его для определения препятствий перед ними. С помощью датчика касания вы можете узнать, что какой-то механизм вашего робота достиг заданного положения. В главе 18, к примеру, вы будете использовать датчик касания для определения, поднята ли рука робота во время его движения.

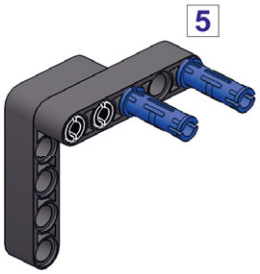
Сборка бампера с датчиком касания

Если вы соберете бампер и прикрепите к нему датчик касания, то каждый раз, когда EXPLOR3R наедет на объект, датчик будет нажат. Программа в модуле EV3 может использовать эту информацию, чтобы направить робота в другую сторону. Соберите бампер и прикрепите его к роботу, как показано на следующих страницах. Обязательно подключите датчик касания к входному порту 1 с помощью короткого кабеля.

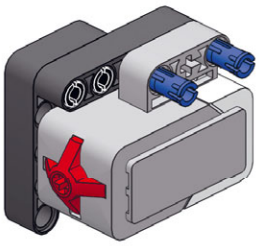


1

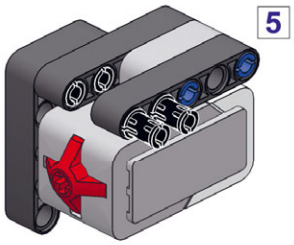
2



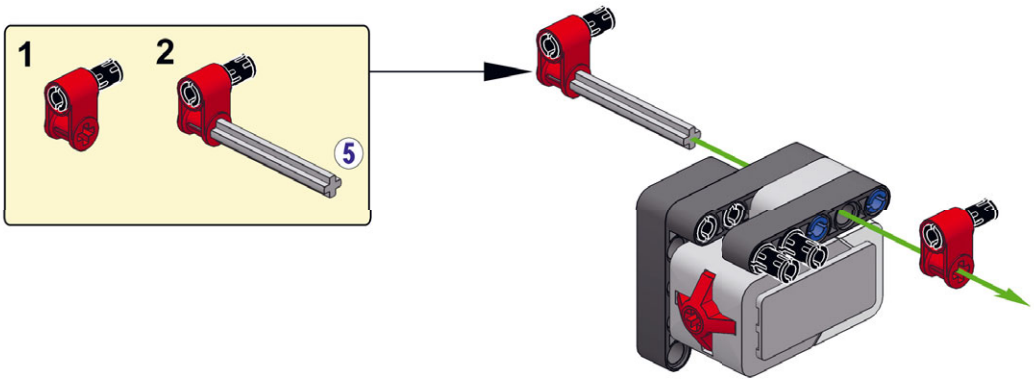
3

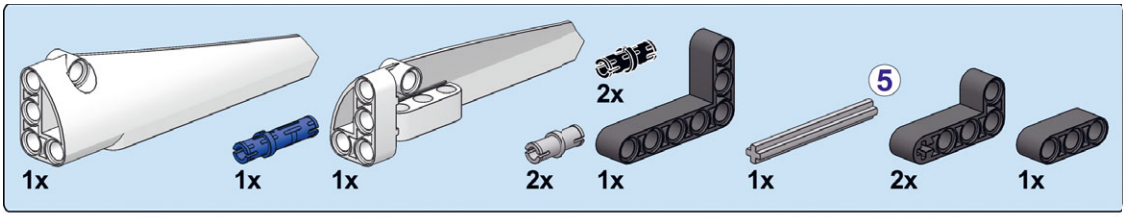


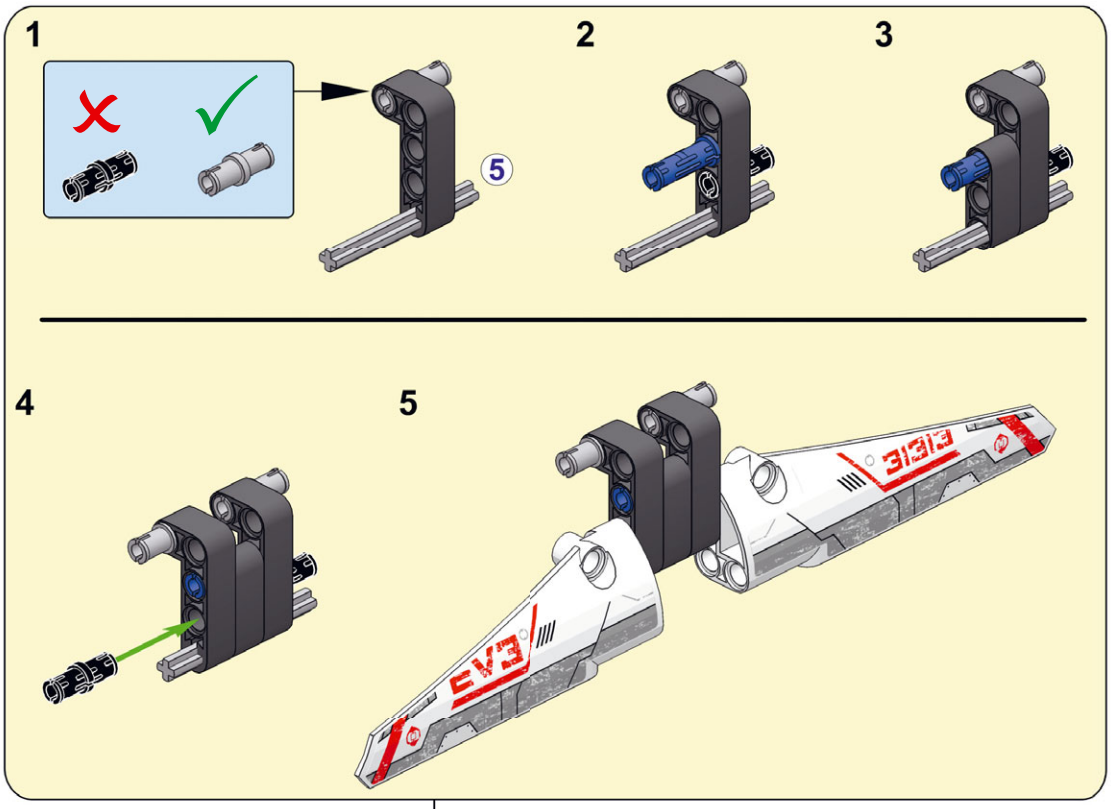
4

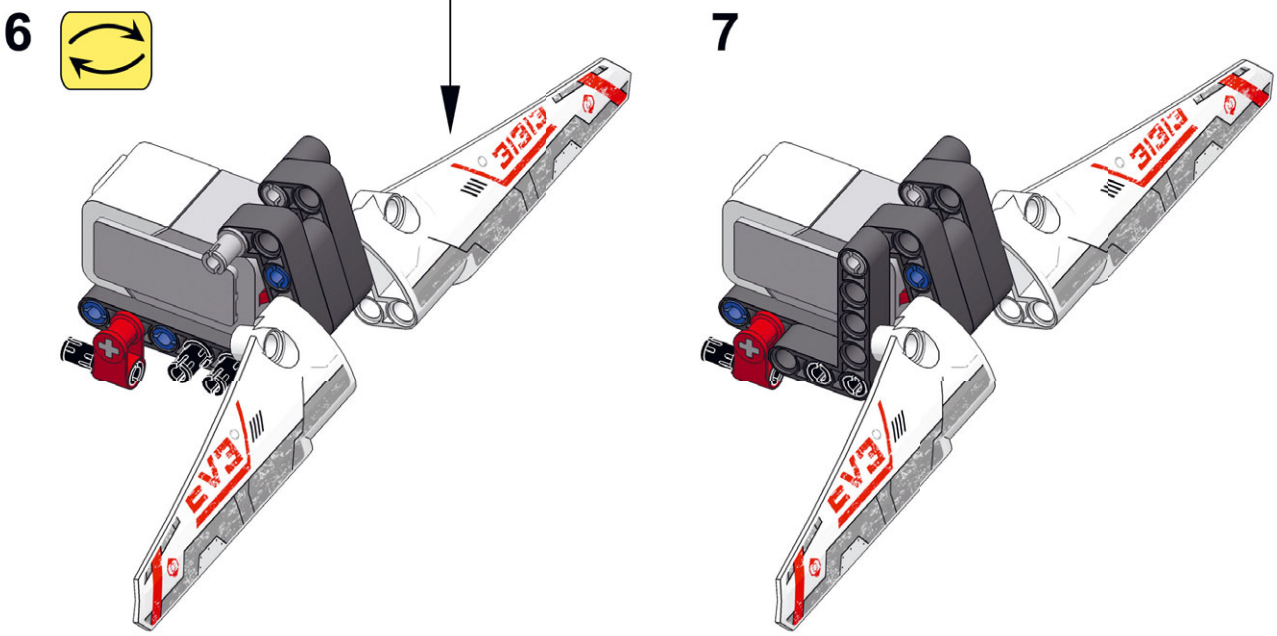


5

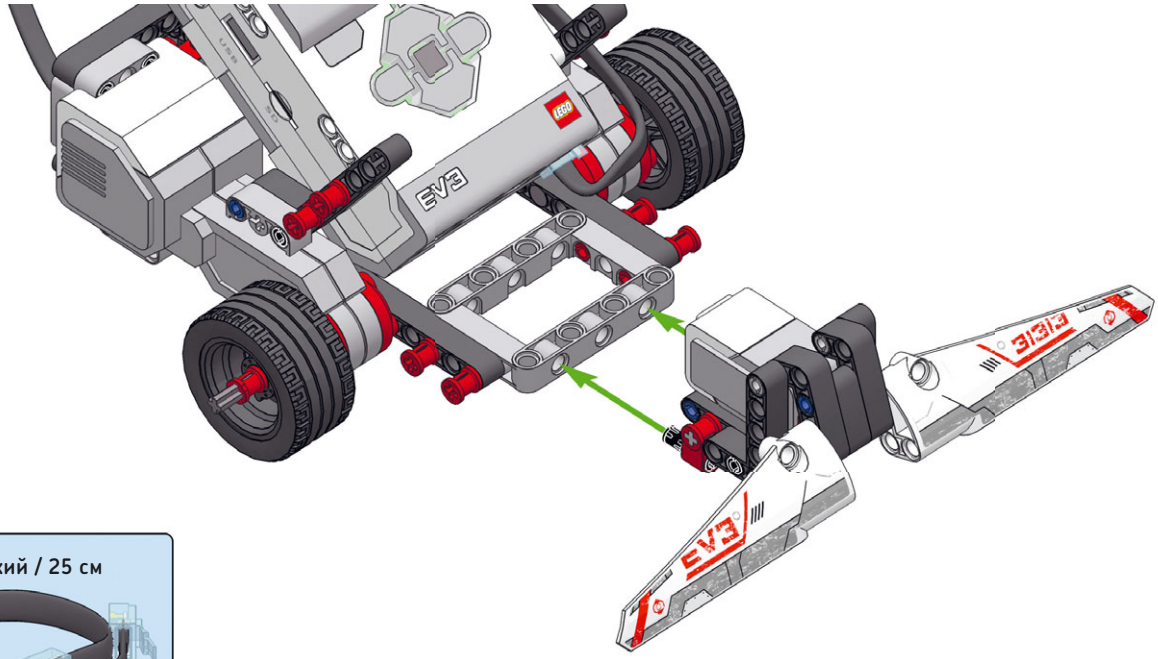



1x
1x
1x
2x
2x
1x
5
1x
2x
1x

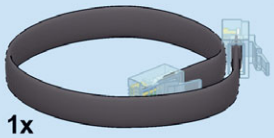

1
5
2
3
4
5


6
7

8

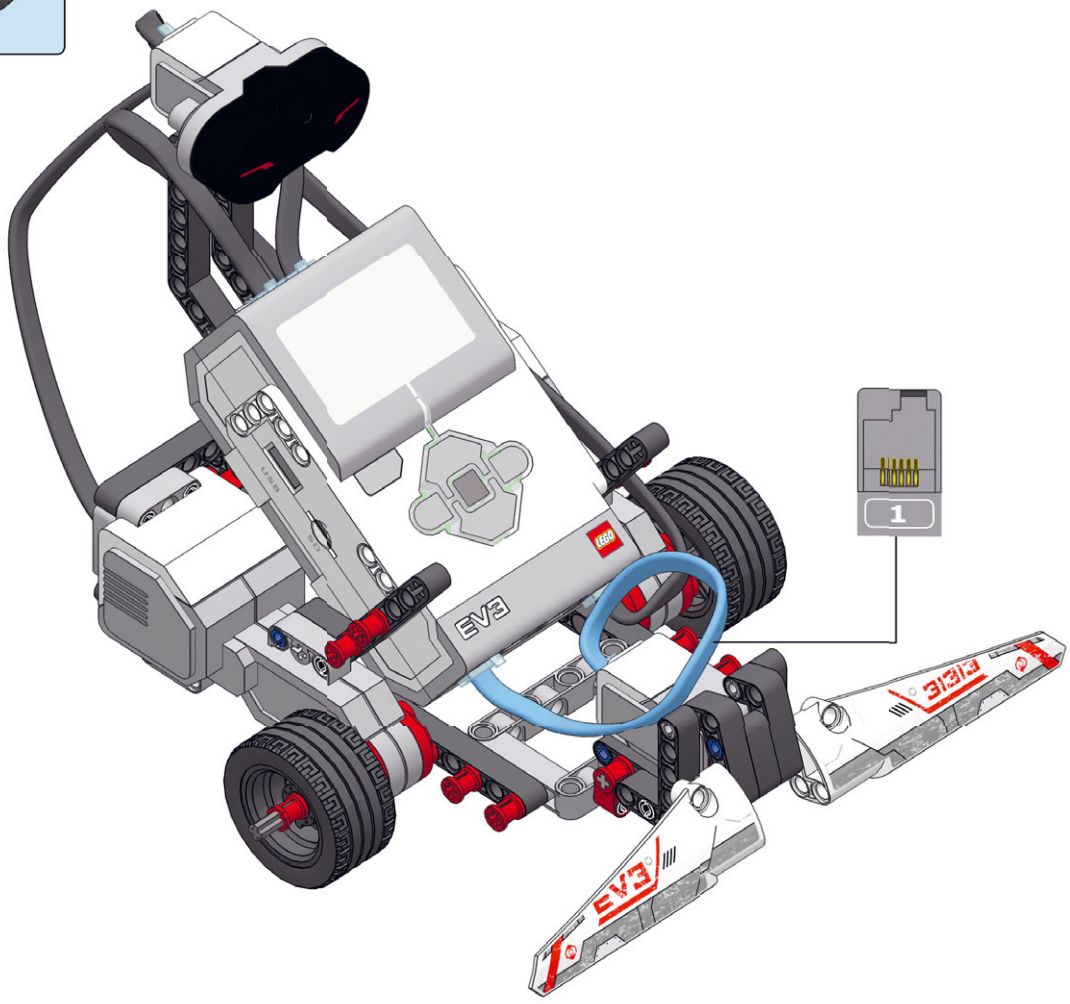


Короткий / 25 см



1x

9



Просмотр значений, полученных датчиками

Вы можете просмотреть данные, собранные каждым датчиком, запустив приложение **Port View** на вкладке **Приложения модуля** (Brick Apps) модуля EV3, как показано на рис. 6.4. Для датчика касания значение 1 означает, что кнопка нажата, а 0 указывает на то, что она отпущена.

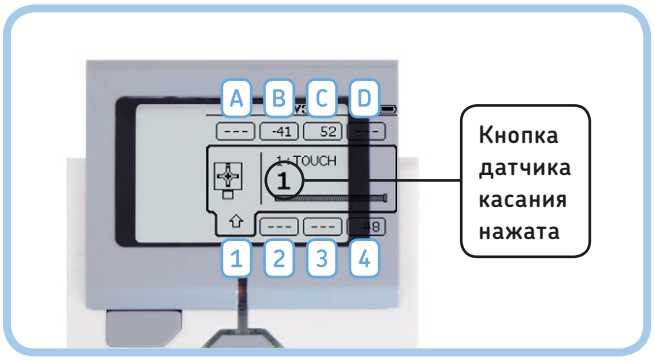


*Рис. 6.4. Приложение **Port View** на вкладке **Приложения модуля**. Модуль EV3 автоматически определяет, какие датчики подключены к EV3, и отображает на экране информацию, получаемую от них. Используйте кнопки «Влево», «Вправо», «Вверх» и «Вниз», чтобы увидеть более подробную информацию о каждом датчике*

Вы можете использовать кнопки на корпусе модуля EV3, чтобы перейти к данным с других датчиков. В правом нижнем углу экрана (порт 4) вы можете увидеть значение расстояния, которое измеряет инфракрасный датчик (48% в данном примере). Два значения в верхней части экрана (–41 и 52) указывают позиции моторов робота, подсоединенных к портам B и C.

Некоторые датчики могут выполнять замеры не одного типа, а больше. Чтобы увидеть другие замеры, к примеру инфракрасного датчика, перейдите к порту 4, нажмите центральную кнопку и выберите режим датчика. По мере чтения книги вы узнаете больше о значении каждого параметра.

Если ваш робот подключен к компьютеру, вы можете просмотреть данные, получаемые от датчиков, на странице

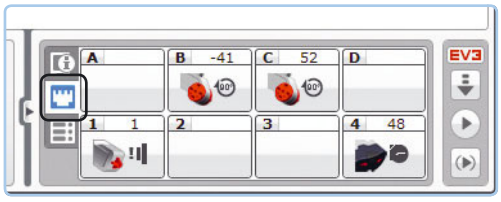


Рис. 6.5. Вы можете просмотреть данные, получаемые от датчиков, в программе EV3. Если значения не обновляются непрерывно, загрузите программу в модуль EV3 робота, чтобы обновить подключение. Щелкните мышью по одному из датчиков, чтобы просмотреть соответствующие данные измерений

аппаратных средств в программном обеспечении EV3, как показано на рис. 6.5. Используйте тот метод, который вам более удобен.

Использование датчиков в программах

Теперь давайте посмотрим, как использовать данные, получаемые от датчиков, в программах. Давайте попробуем использовать датчик касания в программе, с помощью которой робот воспроизводит звук, когда кнопка датчика касания нажата.

Использовать датчики в вашей программе позволяют несколько программных блоков, в том числе **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch). В этой главе вы узнаете, как каждый из них работает с датчиком касания. Те же самые принципы применимы и к другим датчикам из набора MINDSTORMS EV3.

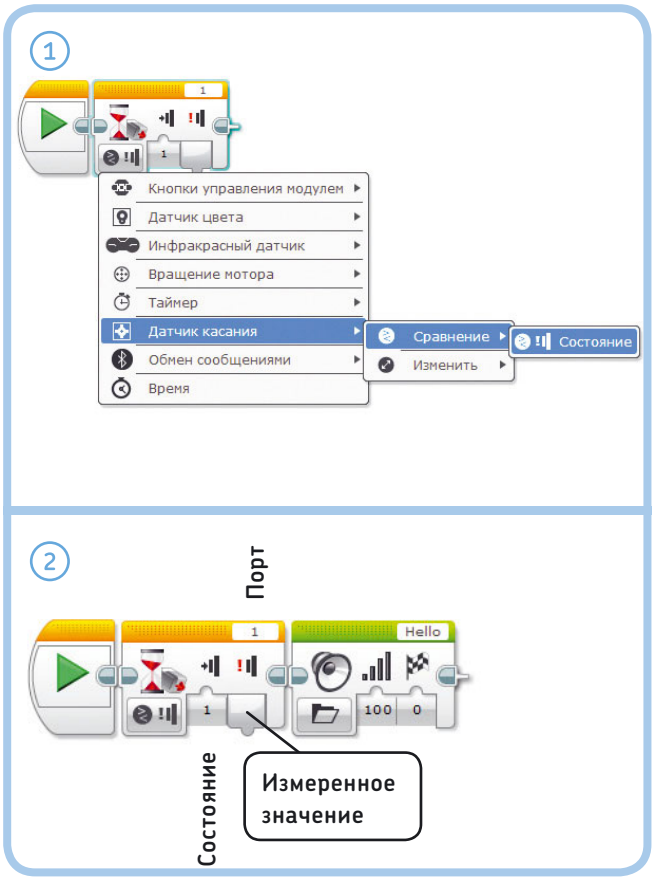


Рис. 6.6. Благодаря программе WaitForTouch робот воспроизводит звук, когда нажата кнопка датчика касания

Датчики и блок Ожидание

Ранее вы использовали блок **Ожидание** (Wait) для приостановки выполнения программы на некоторое время (скажем, пять секунд). Но вы также можете использовать этот блок для приостановки выполнения программы до тех пор, пока не сработает датчик. Например, вы можете настроить блок **Ожидание** (Wait) на приостановку программы до того момента, пока кнопка датчика касания не будет нажата, выбрав режим **Датчик касания** (Touch Sensor), как показано на рис. 6.6.

После выбора этого режима вы должны выбрать подрежим: **Сравнение** (Compare) и **Изменить** (Change). В режиме **Сравнение** (Compare) следует указать параметр **Состояние** (State): должна ли программа ждать, пока кнопка датчика не будет отпущена (0), нажата (1) или щелкнута (2). Если вы выбрали щелчок, программа будет ждать, пока кнопка датчика не будет нажата и сразу же отпущена.

В режиме **Изменить** (Change) программа отслеживает изменения состояния датчика: если кнопка датчика нажата, когда блок начинает работать, программа ожидает, пока кнопка не будет отпущена. И наоборот, если изначально кнопка датчика отпущена, программа приостанавливается до тех пор, пока кнопка датчика не будет нажата.

Параметр **Порт** (Port) позволяет указать, к какому входному порту подключен ваш датчик (в данном случае порт 1). И, наконец, выход **Измеренное значение** (Measured Value) позволяет использовать последний замер датчика в дальнейшем в вашей программе (мы вернемся к этому вопросу в части V).

Использование датчиков с блоком Ожидание

Создайте новый проект под названием EXPLOR3R-Touch с программой WaitForTouch, как показано на рис. 6.6. Для блока **Ожидание** (Wait) выберите режим **Датчик касания** ▸ **Сравнение** ▸ **Состояние** (Touch Sensor ▸ Compare ▸ State). При запуске программы поначалу ничего не будет происходить, но при нажатии кнопки датчика касания (нажатия на бампер) робот должен сказать: «Привет!»

Теперь нажмите кнопку датчика касания и запустите программу снова. Звук воспроизводится сразу, потому что блоку **Ожидание** (Wait) нечего ждать — кнопка датчика уже нажата.

Обход препятствий с помощью датчика касания

Теперь, когда вы познакомились с датчиком касания и блоком **Ожидание** (Wait), вы можете сделать свои программы более интересными. Следующая программа, TouchAvoid, задает движение по комнате и разворот, когда робот EXPLOR3R столкнется с чем-либо, например стеной или стулом. Вы можете увидеть схему программы на рис. 6.7.

Каждое действие в этой схеме вы можете выполнить с помощью одного программного блока. Вы будете использовать блок **Рулевое управление** (Move Steering) в режиме **Включить** (On) для включения моторов, а затем блок **Ожидание** (Wait), чтобы подождать, пока кнопка датчика касания не будет нажата. Обратите внимание, что, когда

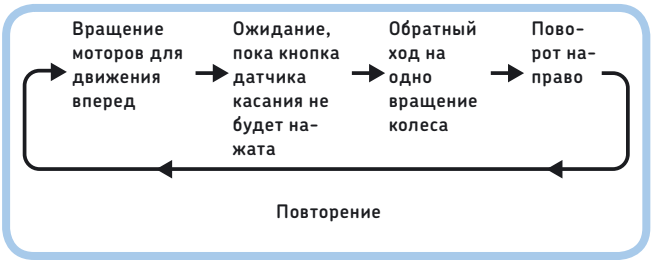


Рис. 6.7. Схема выполнения программы TouchAvoid. После поворота робота направо программа возвращается к началу и повторяется

ПРАКТИКУМ № 23: ПРИВЕТ И ПОКА!

Сложность: **Время:**

Можете ли вы создать программу, с помощью которой робот будет произносить «Привет!», когда вы нажимаете на бампер, а затем «Пока!», когда вы отпускаете бампер?

СОВЕТ **Добавьте еще одну пару блоков Ожидание (Wait) и Звук (Sound) в программу WaitForTouch (рис. 6.6). Первый блок Ожидание (Wait) должен ждать нажатия, а второй — отпускания кнопки датчика. Где вы разместите новые блоки?**

ПРАКТИКУМ № 24: ИЗБЕГАЙТЕ ПРЕПЯТСТВИЙ И ПЛОХОГО НАСТРОЕНИЯ!

Сложность: **Время:**

Дополните программу TouchAvoid, сделав так, чтобы на экране модуля EV3 отображалось счастливое лицо во время движения робота вперед и грустное лицо, когда он едет назад и поворачивает.

СОВЕТ **Поместите два блока Экран (Display) в блок Цикл (Loop).**

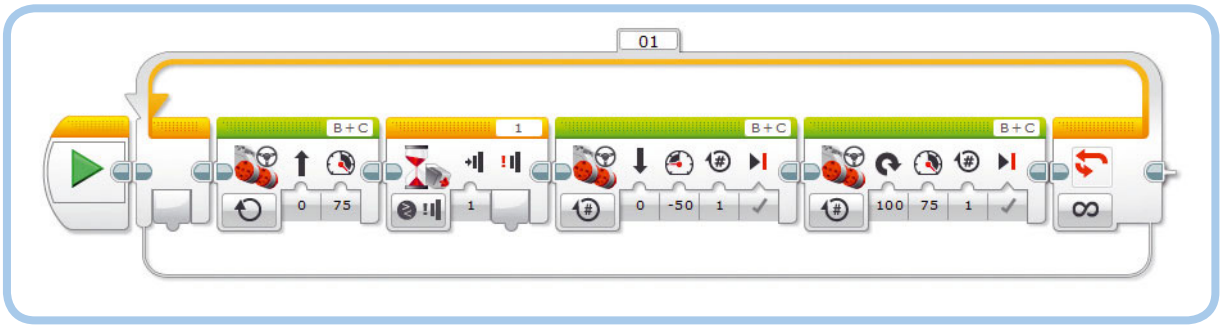


*Рис. 6.8. Программа TouchAvoid. Блок **Ожидание** настроен на режим **Датчик касания** ▸ **Сравнение** ▸ **Состояние***

ПРАКТИКУМ № 25: ДАВАЙТЕ ПОНАЖИМАЕМ!

Сложность: **Время:**

Можете ли вы сделать так, чтобы робот EXPLOR3R двигался в обратном направлении до тех пор, пока бампер не будет нажат, и можете ли вы сделать, чтобы робот остановился, когда вы отпустите бампер? Такое поведение должно продолжаться до тех пор, пока вы не завершите программу вручную. Проверьте свою программу, удерживая бампер нажатым. Вам может показаться, что это вы толкаете робота назад, но на самом деле всю тяжелую работу делает он сам.

СОВЕТ **Вам понадобится блок Цикл (Loop), два блока Ожидание (Wait), а также два блока Рулевое управление (Move Steering), один в режиме Включить (On), а другой в режиме Выключить (Off).**

Использование режима Изменить

До сих пор мы использовали блок **Ожидание** (Wait) в режиме **Сравнение** (Compare), чтобы сделать паузу в выполнении программы до тех пор, пока кнопка датчика касания не займет выбранное вами положение (нажата или отпущена). Теперь мы создадим программу с блоком **Ожидание** (Wait) в режиме **Изменить** (Change), которая будет приостанавливать программу до момента изменения состояния кнопки датчика (или из состояния отпущена к нажата, или от нажата к отпущена).

Соберите и запустите программу WaitForChange, как показано на рис. 6.9.

Если бампер отпущен, когда запускается программа, робот должен ехать, пока не достигнет какого-либо объекта, а затем остановиться. Если при запуске программы бампер уже нажат, робот должен пытаться идти вперед, пока датчик не будет отпущен, а затем остановиться.

Для большинства программ рекомендуется использовать режим **Сравнение** (Compare), так как с ним поведение программа ожидает нажатия, робот продолжает двигаться вперед.

После того как датчик срабатывает, вы используете блок **Рулевое управление** (Move Steering), чтобы отъехать назад, а затем другой такой же блок, чтобы развернуться; оба блока должны находиться в режиме **Включить на количество оборотов** (On for Rotations). После того как робот развернется, программа возвращается к началу, поэтому вы должны поместить описанные четыре блока внутрь блока **Цикл** (Loop), работающего в режиме **Неограниченный** (Unlimited). Теперь создайте программу, как показано на рис. 6.8.

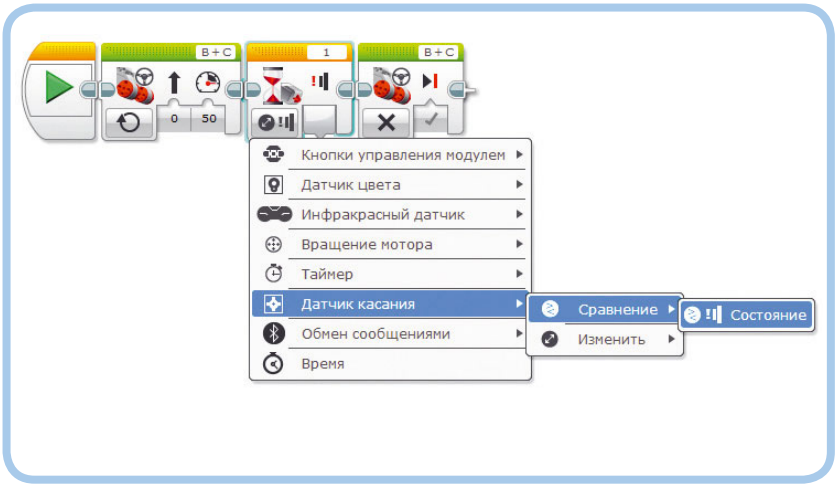


Рис. 6.9. Программа WaitForChange

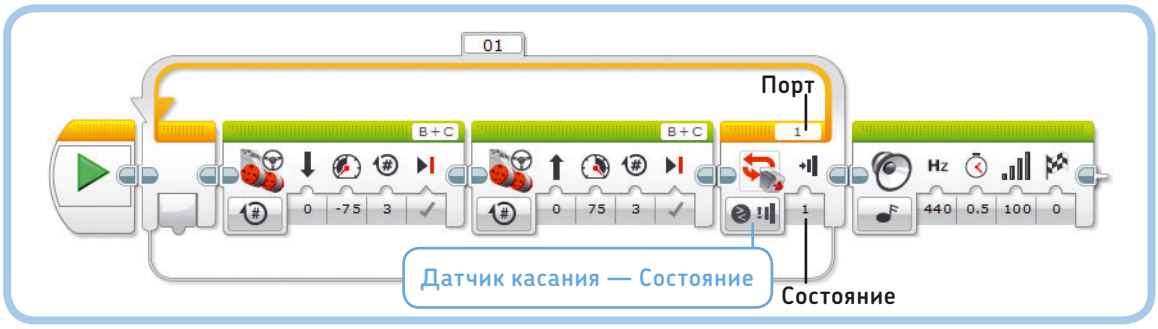


Рис. 6.10. Программа LoopUntilTouch. Чтобы настроить блок ***Цикл****, нажмите кнопку выбора режима и выберите пункт* ***Датчик касания*** ▸ ***Состояние***

вашего робота легче предугадать. Вне зависимости от исходного состояния датчика робот всегда будет ждать изменения своего поведения, пока датчик касания не достигнет заданного вами состояния.

Датчики и блок Цикл

Как вы узнали в главе 5, вы можете настроить блок **Цикл** (Loop) на повторение определенное число раз, или в течение некоторого количества секунд, или на бесконечное зацикливание. Вы также можете запрограммировать блок **Цикл** (Loop) на остановку цикла на основе входного сигнала от датчика. Например, вы можете сделать так, чтобы ваш робот двигался вперед и назад, пока кнопка датчика касания не будет нажата. Чтобы таким образом настроить блок **Цикл** (Loop), выберите режим **Датчик касания** ▸ **Состояние** (Touch Sensor ▸ State), как показано на рис. 6.10. Как и прежде, присвойте значение **1** параметру **Порт** (Port).

Создайте программу LoopUntilTouch и запустите ее, чтобы посмотреть, как она работает. Следует заметить, что программа проверяет показания датчика только один раз за полностью пройденную последовательность блоков в цикле. Для завершения цикла нужно, чтобы кнопка датчика была нажата только после того, как робот продвинется вперед. Если кнопка датчика не будет нажата в этой фазе цикла, робот повторит движение вперед и назад, прежде чем снова проверить состояние датчика касания.

Это обычный режим работы для блока **Цикл** (Loop), но иногда бывает нужно завершить цикл, даже если кнопка датчика не нажата точно в нужное время. Для достижения этой цели присвойте параметру **Состояние** (State) значение **Щелчок** (Bumped) (**2**) и запустите программу еще раз. В этой конфигурации цикл не проверяет, нажата ли кнопка датчика касания в *конце* цикла; скорее, он проверяет, наталкивается ли на что-то бампер (кнопка датчика нажата и отпущена), в любое время в *течение* цикла. Если вы установите такие настройки, то блоки должны прекратить повторяться после того, как они завершают текущий запуск программы. Во время выполнения цикла EV3 непрерывно контролирует состояние датчика касания, поэтому вам не придется беспокоиться об этом.

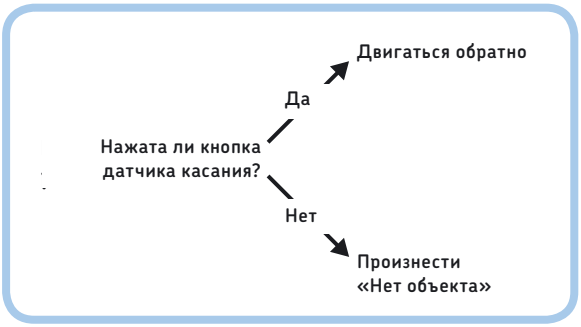


Рис. 6.11. Робот может действовать в зависимости от результата, измеренного датчиком

ПРАКТИКУМ № 26: ВЕСЕЛЫЕ МЕЛОДИИ!

Сложность: **Время:**

Используйте блок **Цикл** (Loop), чтобы робот проигрывал мелодию до тех пор, пока бампер не будет нажат. В этот момент робот должен вскрикнуть и быстро развернуться.

СОВЕТ **Вы можете использовать контейнер «Мой блок», который сделали в практикуме № 19 на с. 78 для воспроизведения мелодии. Если вам еще предстоит создать свою мелодию, просто выберите звуковой файл из списка в блоке Звук (Sound).**

*Рис. 6.12. Блок **Переключатель** проверяет, является ли условие истинным или ложным, и запускает соответствующие блоки. Вы указываете состояние, нужный режим и параметры на блоке **Переключатель***

Датчики и блок Переключатель

Вы можете использовать блок **Переключатель** (Switch), чтобы робот действовал в зависимости от результата, измеренного датчиком. Например, вы можете сделать так, чтобы робот двигался в обратном направлении при нажатии кнопки датчика касания или произносил «Нет объекта*», когда кнопка датчика в отпущенном состоянии (рис. 6.11).

Блок **Переключатель** (Switch) проверяет, является ли данное *условие* (например, «кнопка датчика касания нажата») истинным или ложным, как показано на рис. 6.12.

Блок **Переключатель** (Switch) в этом примере содержит две последовательности блоков; он запускает ту или иную последовательность в зависимости от того, истинно условие или нет. Если условие истинно, то запускается верхняя часть блока **Переключатель** и робот движется в обратном направлении; если условие ложно, выполняется нижняя часть блока и робот произносит «Нет объекта».

* Как и во всех остальных случаях, используется английский аналог выражения — No Object.

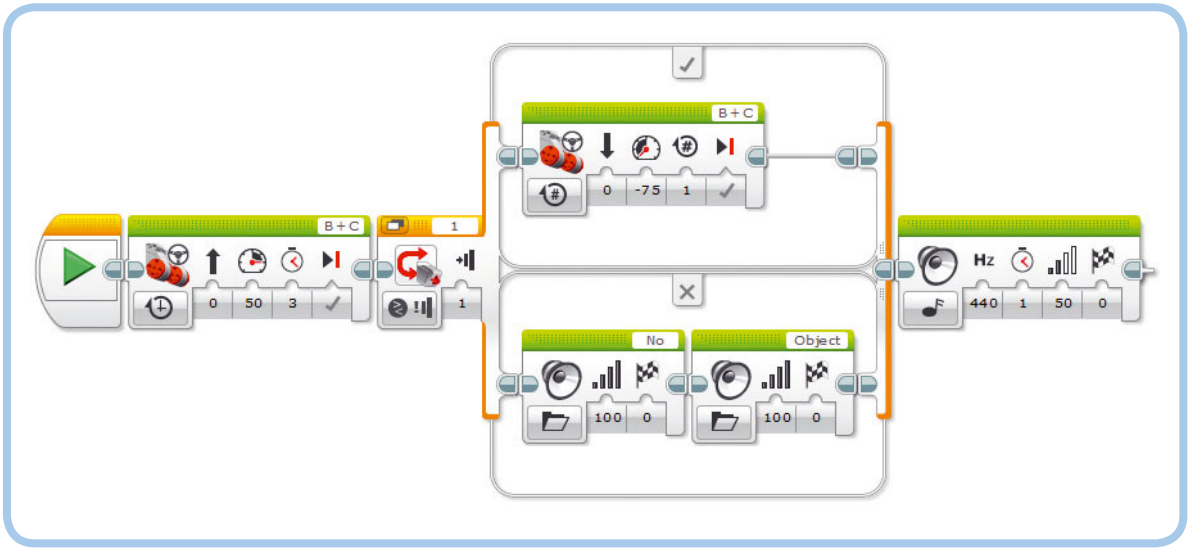


Рис. 6.13. Программа TouchSwitch позволяет роботу решать, что делать, основываясь на информации, полученной от датчика

Настройка блока Переключатель

Вы определяете состояние блока **Переключатель** (Switch) путем выбора его режима и настройки параметров. Когда выполнение программы доходит до блока **Переключатель** (Switch), робот проверяет, является ли условие истинным. Затем он решает, какую последовательность программных блоков следует запустить.

Для каждого датчика используется собственный режим. В данном случае нужно выбрать режим для датчика касания, а именно **Датчик касания** ▸ **Сравнение** ▸ **Состояние** (Touch Sensor ▸ Compare ▸ State) (единственный доступный вариант). После того как вы выбрали этот режим, вы можете с помощью параметра **Состояние** (Status) указать, должна ли кнопка датчика быть нажата (**1**) или отпущена (**0**) в случае истинности условия. Как и ранее, присвойте параметру **Порт** (Port) значение **1**, чтобы указать, к какому порту модуля EV3 подключен датчик касания.

Использование датчиков с блоком Переключатель

Программа TouchSwitch, которую вы создадите, обеспечивает движение робота вперед в течение 3 секунд. Затем, если кнопка датчика касания нажата, робот немного отъезжает назад. Если кнопка датчика не нажата, робот говорит: «Нет объекта». Наконец, независимо от решения блока **Переключатель** (Switch) робот воспроизводит мелодию. Теперь создайте программу, как показано на рис. 6.13.

Попробуйте запустить эту программу несколько раз и определить, когда необходимо нажать кнопку датчика касания, чтобы робот двигался в обратном направлении. Ваши эксперименты должны показать, что робот выполняет замер, когда блок **Переключатель** (Switch) работает, и что он использует это единственное измерение для определения, является ли условие истинным. В этой программе чтение данных с датчика производится только после того, как робот завершит движение вперед. Мелодия играет независимо от того, откатывается ли робот назад или произносит фразу «Нет объекта».

Добавление блоков в переключатель

Внутри блока **Переключатель** (Switch) вы можете поместить неограниченное количество блоков. Если хотя бы в одной части размещено несколько блоков, они просто будут запускаться один за другим. Вы можете оставить вторую часть блока **Переключатель** пустой, как показано на рис. 6.14.

Запустите модифицированную программу, чтобы посмотреть, что произойдет. Если условие истинно (бампер нажат), робот должен произнести «Объект» и поехать в обратном направлении, а программа должна воспроизводить музыку. Если условие ложно (кнопка датчика не нажата), программа, не обнаружив блоков в нижней части блока **Переключатель** (Switch), немедленно перейдет к блоку **Звук** (Sound) после переключения.

Режимы отображения блока Переключатель

Как правило, блок **Переключатель** (Switch) отображается в режиме без вкладок, с двумя последовательностями вложенных блоков сверху и снизу. При разработке сложных программ, содержащих блоки **Переключатель** (Switch), легко запутаться в схеме работы вашей программы. В таких случаях вы можете выбрать режим отображения с вкладками, чтобы уменьшить размер блока **Переключатель** (Switch), как показано на рис. 6.16. Обе части блока **Переключатель** (Switch) остаются в программе, но находятся на отдельных вкладках, которые можно переключать, щелкая по ним мышью.

Многократное выполнение блоков Переключатель

Каждый раз, когда ваша программа достигает блока **Переключатель** (Switch), он проверяет состояние датчика

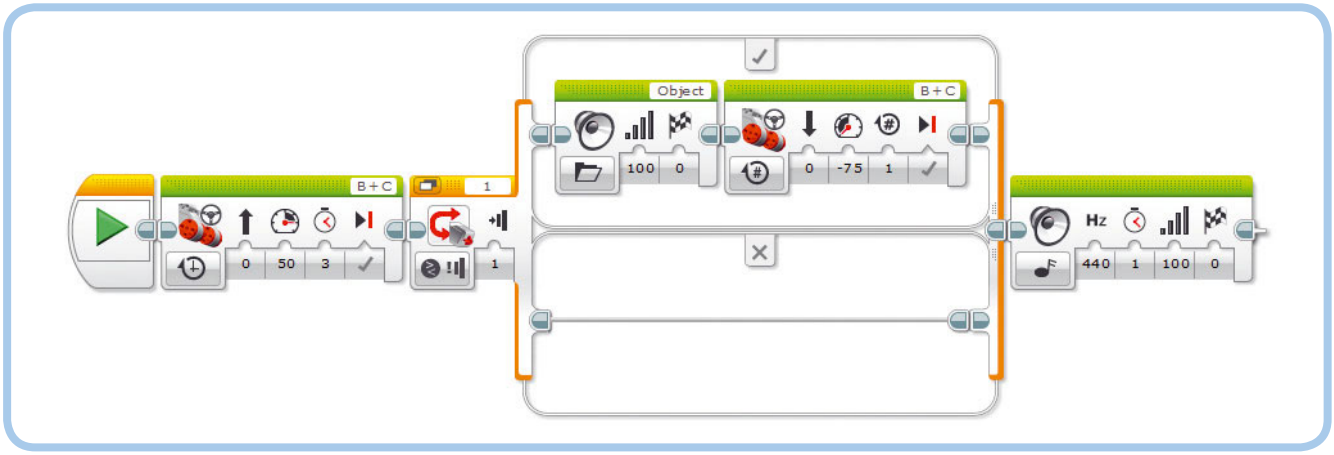


*Рис. 6.14. АМодифицированная версия программы TouchSwitch. Блок **Переключатель** не запускает каких-либо блоков, если условие ложно, поэтому по окончании движения в обратном направлении программа сразу же воспроизводит музыку, если кнопка датчика не нажата*

ПРАКТИКУМ № 27: СТОЙ ИЛИ ИДИ!

Сложность: **Время:**

Запрограммируйте робота стоять на месте в течение трех секунд. Затем, если кнопка датчика касания отпущена, робот должен развернуться и двигаться вперед на пять оборотов колес. Но, если кнопка датчика нажата, робот не должен делать ничего, и программа должна немедленно завершить работу.

касания, чтобы определить, из какой части следует запускать блоки, истинной или ложной. Чтобы робот проверил состояние более одного раза, вы можете вложить блок **Переключатель** (Switch) внутрь блока **Цикл** (Loop). Например, вы можете запрограммировать робота, чтобы он произносил: «Да», если кнопка датчика касания нажата, и «Нет», если отпущена. Если поместить блок **Переключатель** (Switch) с подобной конфигурацией в блок **Цикл** (Loop), робот будет постоянно проверять показания датчика и продолжать говорить «Да» или «Нет» соответственно. Теперь создадим программу RepeatSwitch, показанную на рис. 6.17.

Режимы Сравнение, Изменить и Измерение

При использовании датчика в программах с блоками **Ожидание** (Wait), **Цикл** (Loop) и/или **Переключатель** (Switch) часто требуется выбирать один из трех режимов работы каждого блока: **Сравнение** (Compare), **Изменить** (Change) и **Измерение** (Measure). В книге вы найдете много примеров использования этих режимов, но сначала рассмотрим внимательнее, как они работают.

ПРАКТИКУМ № 28: ТРУДНЫЕ РЕШЕНИЯ!

Сложность: **Время:**

Давайте еще поработаем с блоком **Переключатель** (Switch). Создайте программу для выполнения схемы решений, показанной на рис. 6.15. Как настроить блок **Переключатель** (Switch) и почему вы должны установить блок **Ожидание** (Wait) в конце программы?

Рис. 6.16. Схема программы для практикума № 28

*Рис. 6.15. При выборе режима отображения с вкладками размер блока **Переключатель** уменьшится. Режим отображения просто меняет вид блока; он не влияет на то, как программа работает*

Режим Сравнение

Режим **Сравнение** (Compare) позволяет сверять значение датчика с *условием*, указанным в настройках блока. Условием является утверждение, например, «кнопка датчика касания нажата», «измеренная яркость света составляет менее 37%» или «датчик цвета зафиксировал красный или синий».

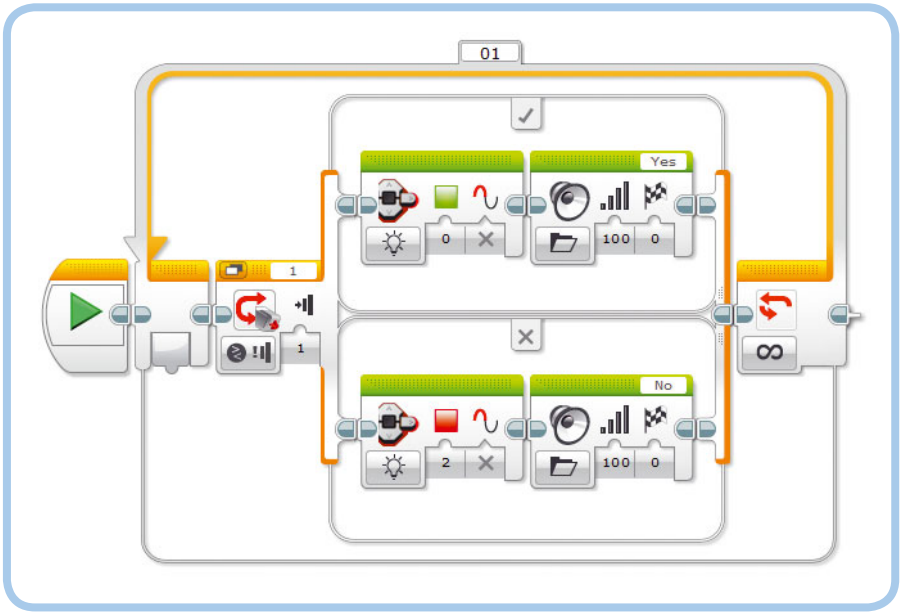


Рис. 6.17. Программа RepeatSwitch

- Блок **Ожидание** (Wait) в режиме **Сравнение** (Compare) продолжает снимать показания с датчиков, пока условие не станет истинным. Когда это происходит, программа переходит к следующему блоку в последовательности (см. рис. 6.6).
- Блок **Цикл** (Loop) в режиме **Сравнение** (Compare) проверяет новые значения датчика каждый раз при готовности запустить блоки внутри цикла. Если условие истинно, то программа переходит к блоку, расположенному после цикла; если оно ложно, то цикл запускается снова (см. рис. 6.10). Блоки **Цикл** (Loop) всегда находятся в режиме сравнения.
- Блок **Переключатель** (Switch) в режиме **Сравнение** (Compare) запускает последовательность блоков в верхней части, если условие истинно; в противном случае запускаются блоки, находящиеся внизу (рис. 6.13).

Режим Изменить

Режим **Изменить** (Change) доступен только в блоке **Ожидание** (Wait). Блок **Ожидание** (Wait) в режиме **Изменить** (Change) снимает первое и дальнейшие показания, пока не обнаружит такое, которое отличается от первого. Например, если при запуске блока кнопка датчика касания нажата, он ждет, пока кнопка датчика не будет отпущена. Затем программа переходит к следующему блоку (см. рис. 6.9).

Режим Измерение

Режим **Измерение** (Measure) доступен только в блоке **Переключатель** (Switch). Блок **Переключатель** (Switch) в этом режиме содержит набор блоков для запуска для каждого возможного значения датчика. Вы увидите, как это работает, в главе 7, когда вы будете создавать программу, выполняющую разные действия для каждого цвета, который может распознать датчик цвета.

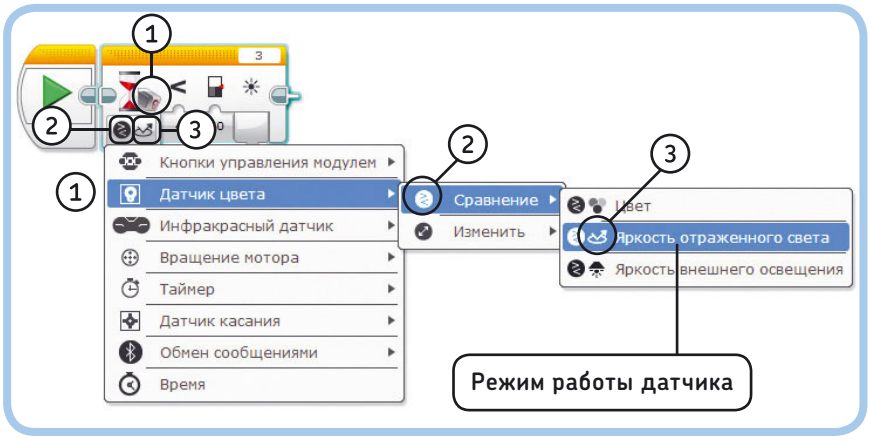


*Рис. 6.18. Если в примерах программ в этой книге используются блоки **Ожидание**, **Цикл** и **Переключатель**, выбирайте пункты меню так, чтобы они совпадали со значками, показанными на блоках. Начните с выбора подходящего датчика (1), выберите способ взаимодействия, **Сравнение**, **Изменить** или **Измерение** (2), и, наконец, выберите режим работы датчика (3)*

Настройка режимов

Как правило, из описания становится ясно, какой режим следует использовать для примеров программ, описанных в этой книге, кроме того, информацию вы можете видеть на схемах программ. Если вы не уверены, какой режим выбрать, просто взгляните на значки, указанные в каждом блоке, как изображено на рис. 6.18.

После того как вы выбрали датчик (1) и нужный режим (**Сравнение** (Compare), **Изменить** (Change) или **Измерение** (Measure)) (2), вы выбираете способ работы датчика (3). Датчик касания измеряет только один показатель (состояние красной кнопки), а датчик цвета имеет три режима работы, как показано на рис. 6.18. В последующих главах вы попрактикуетесь с каждым режимом и сможете создавать собственные программы с датчиками в кратчайшие сроки.

Дальнейшее изучение

В этой главе вы узнали, что роботы используют датчики для сбора информации из окружающей среды. Вы также узнали, как разрабатывать программы с датчиками, используя блоки **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch).

До сих пор вы работали только с датчиком касания. Для работы с другими датчиками вы можете применять все методы программирования, которые вы изучили в этой главе. В главе 7 вы научитесь работать с датчиком цвета. В главе 8 узнаете об инфракрасном датчике. В главе 9 вы разберетесь со встроенным датчиком вращения мотора и кнопками модуля EV3. Прежде чем продолжить работу, попрактикуйтесь в развитии своих навыков программирования с использованием датчика касания, решая представленные ниже практические задачи.

ПРАКТИКУМ № 29: ВЫБОР НАПРАВЛЕНИЯ!

Сложность: **Время:**

Дополните программу TouchAvoid, показанную на рис. 6.8, чтобы робот сразу после первого столкновения с объектом двигался вправо, а после повторного нажатия кнопки датчика — влево. Следующее препятствие снова должно приводить к повороту робота направо и так далее.

СОВЕТ **Удвойте количество блоков внутри блока Цикл (Loop), чтобы в цикле их стало восемь, и измените значение параметра Рулевое управление (Move Steering) во втором наборе блоков.**

ПРАКТИКУМ № 30: ОЖИДАНИЕ, ЦИКЛ ИЛИ ПЕРЕКЛЮЧАТЕЛЬ?

Сложность: **Время:**

Запрограммируйте робота ждать нажатия кнопки датчика касания. Затем, если кнопка датчика нажата дольше пяти секунд, пусть робот произносит: «Да». В противном случае он должен сказать «Нет».

СОВЕТ **Вам потребуется сочетание блоков Ожидание (Wait) и Переключатель (Switch).**

ПРАКТИКУМ № 31: КНОПКИ МОДУЛЯ!

Сложность: **Время:**

Из главы 9 вы узнаете, что большинство кнопок модуля EV3 вы можете использовать так же, как датчик касания. Не забегая вперед, можете ли вы сделать так, чтобы робот говорил «Влево» при нажатии кнопки «Влево» на модуле EV3 и «Вправо» — при нажатии кнопки «Вправо»? Можете ли вы запрограммировать робота произносить слова «Вверх» и «Вниз», когда вы нажимаете кнопки «Вверх» и «Вниз» соответственно?

СДЕЛАЙ САМ № 4: ОХРАННАЯ СИГНАЛИЗАЦИЯ!

Сборка: **Программирование:**

Можете ли вы перепрограммировать вашего робота на сигнал тревоги, который предупреждал бы вас о проникновении в комнату злоумышленников? Используйте датчик касания как переключатель, который активируется, когда кто-то входит в комнату. Когда это происходит, пусть робот воспроизведет громкий звуковой сигнал.

СОВЕТ **Положите что-нибудь тяжелое, например книгу, около передней части робота, чтобы бампер был нажат. Затем создайте хитроумную конструкцию, которая автоматически убирает книгу, когда кто-то входит в комнату. Ваш робот должен воспроизводить звук, как только кнопка датчика касания отпускается.**

СДЕЛАЙ САМ № 5: ВЫКЛЮЧАТЕЛЬ!

Сборка: **Программирование:**

Сконструируйте робота, который переключает выключатель света в вашей комнате, когда вы нажимаете кнопку датчика касания. Каждый раз, когда вы нажимаете кнопку датчика, робот должен включать или выключать свет. Чтобы робот был способен сделать это, вы можете добавить средний мотор к EXPLOR3R. Или вы можете создать совершенно нового робота, специально предназначенного для решения этой задачи.

7

Использование датчика цвета

В этой главе вы установите на EXPLOR3R (рис. 7.1) и научитесь использовать датчик цвета, с помощью которого робот сможет определять цвет бумаги, следовать по линии и реагировать на световые сигналы.

Датчик цвета может определить цвет поверхности (в режиме **Цвет** (Color)), яркость света, отраженного от поверхности (в режиме **Яркость отраженного света** (Reflected Light Intensity)), или яркость окружающего освещения (в режиме **Яркость внешнего освещения** (Ambient Light Intensity)), как показано на рис. 7.2.

Вы создадите программы для EXPLOR3R, чтобы опробовать каждый режим работы, используя блоки **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch), как ранее с датчиком касания. По мере чтения книги и сборки роботов, описанных в ней, вы увидите больше применений для этого датчика, например в виде модели LAVA R3X в главе 19, измеряющей яркость отраженного света с целью обнаружить рукопожатие.

Рис. 7.1. Используя датчик цвета, EXPLOR3R может определять цвета и следовать по линии

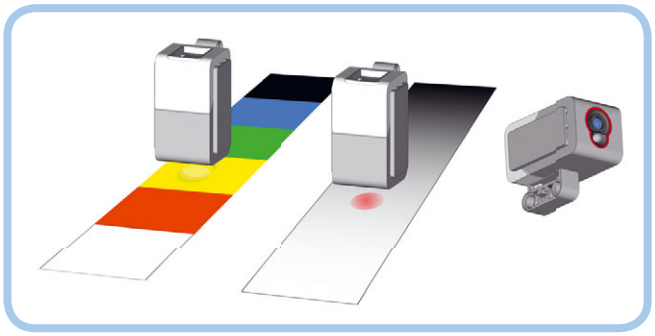


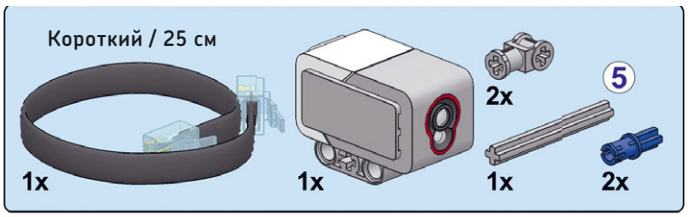
*Рис. 7.2. Три режима работы датчика цвета: **Цвет** (слева), **Яркость отраженного света** (средний) и **Яркость внешнего освещения** (справа). Датчик справа направлен вверх для определения яркости света в комнате*

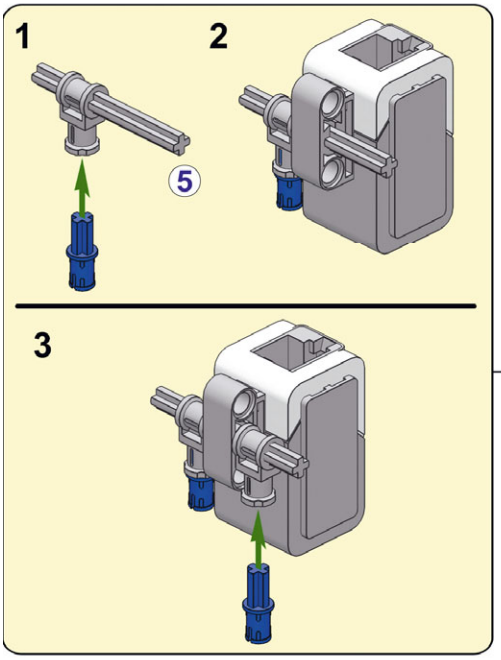
Подключение датчика цвета

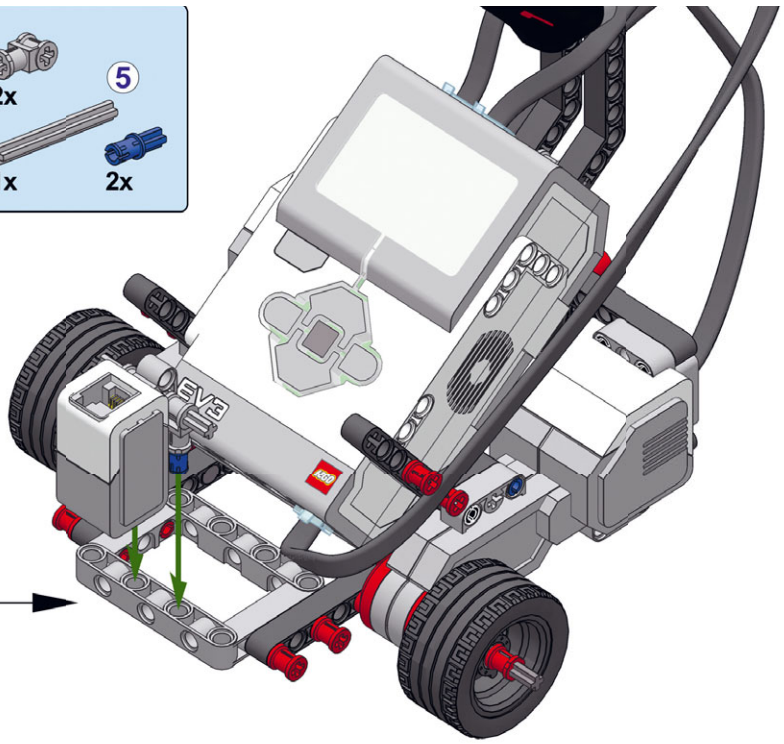
Перед тем как начать разработку программ, снимите подключенный датчик касания с робота (не убирайте его далеко, позднее он вам понадобится). Затем подключите датчик цвета, используя следующую инструкцию по сборке.

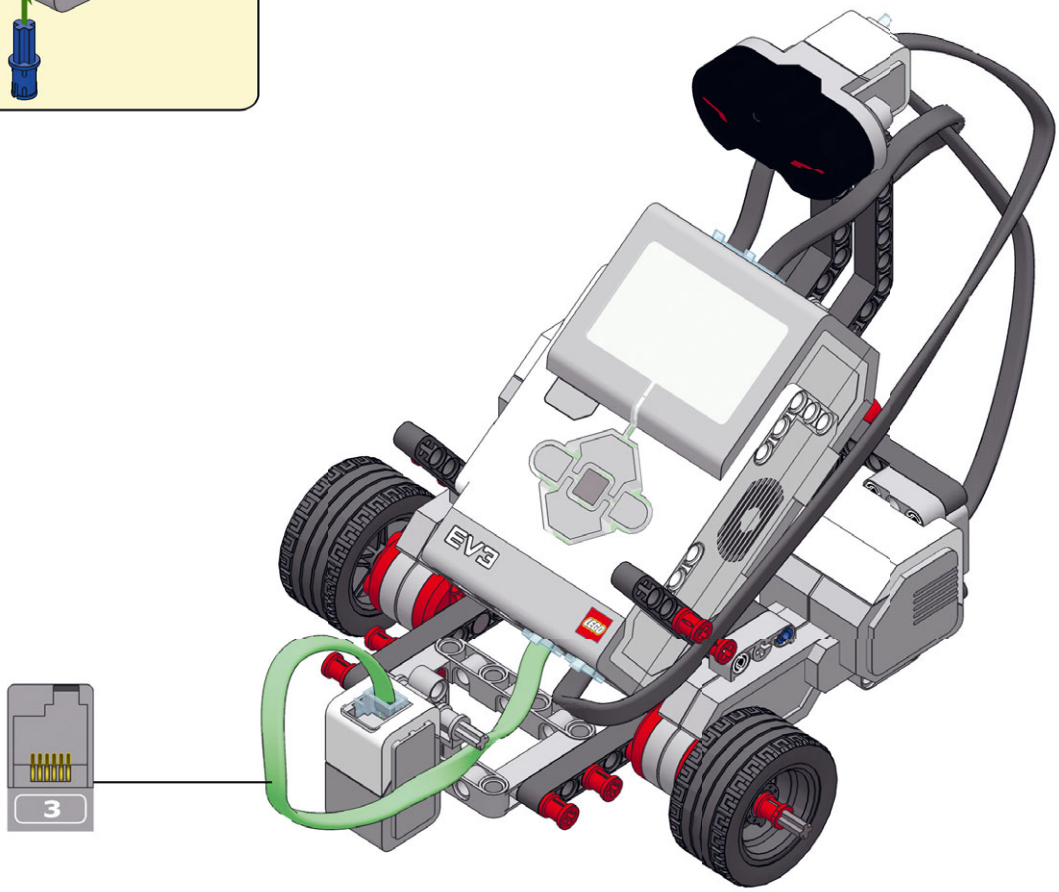
Цветовой режим

Первый режим работы, который вы будете использовать, это **Цвет** (Color). В этом режиме датчик может определять цвет поверхности на расстоянии около 1 см. Датчик устанавливается в направлении прямо вниз, поэтому он может определять цвет непосредственно под роботом. Для проверки определения цвета скачайте и распечатайте на цветном принтере файл *Датчик_цвета.pdf* по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip** и поставьте робота на


Короткий / 25 см
1x
1x
2x
5
1x
2x


1
2
5
3


EV3


EV3
3

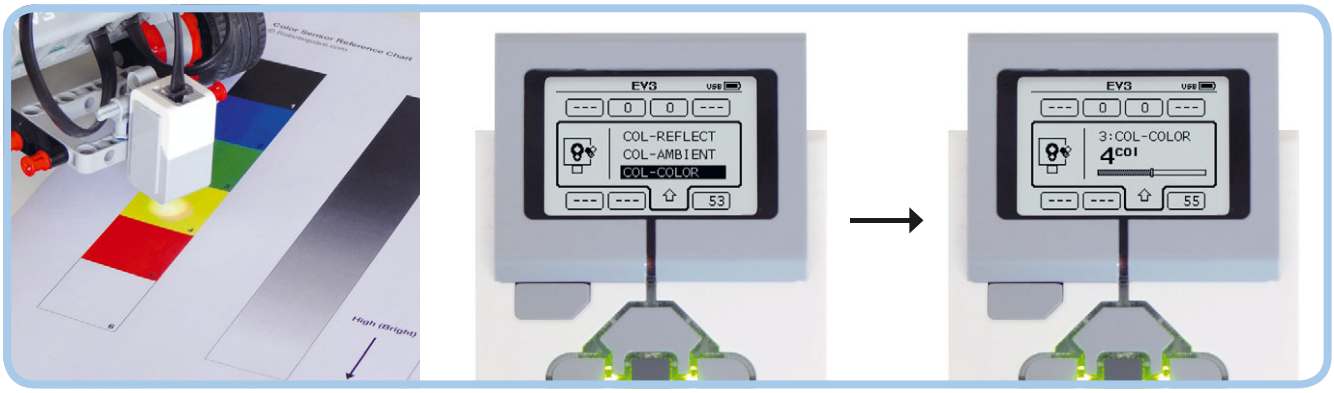


*Рис. 7.3. Запустите приложение **Port View**, чтобы увидеть результаты замеров, произведенных датчиком. Перейдите к датчику на входном порте 3 и нажмите центральную кнопку. Затем выберите пункт **COL-COLOR** и снова нажмите центральную кнопку. Теперь вы должны увидеть число в диапазоне от 0 до 7, которое обозначает цвет. В данном случае от датчика получено значение 4, что соответствует желтому цвету*

распечатку. Затем перейдите к приложению **Port View** на модуле EV3, чтобы увидеть значение обнаруженного цвета, указанное в виде числа (рис. 7.3). Если у вас нет цветного принтера или если датчик не определяет цвета правильно, попробуйте использовать трассу для выполнения миссий, которая поставляется в комплекте с набором EV3.

Датчик определяет следующие цвета: черный (1), синий (2), зеленый (3), желтый (4), красный (5), белый (6) и коричневый (7). Значение 0 указывает на то, что датчик не может определить цвет, потому что находится слишком далеко от поверхности или слишком близко к ней.

Нахождение внутри контура

Программы могут использовать блоки **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch) для выбора последовательности действий на основе информации, полученной от датчика. Например, вы можете сделать робота, который будет двигаться в пределах черного контура, показанного на рис. 7.4. Для этого робот должен ехать вперед, пока не обнаружит черную линию. Затем он должен дать задний ход, развернуться и двигаться вперед в другом направлении.

Создание тестовой трассы

Сначала вам нужно создать круговую трассу, которую вы будете использовать для тестирования датчика. Трасса состоит из набора элементов, напечатанных на стандартной бумаге формата A4. Скачайте и распечатайте документ *Тестовая_трасса.pdf* по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**. Эти элементы нужно вырезать по пунктирной

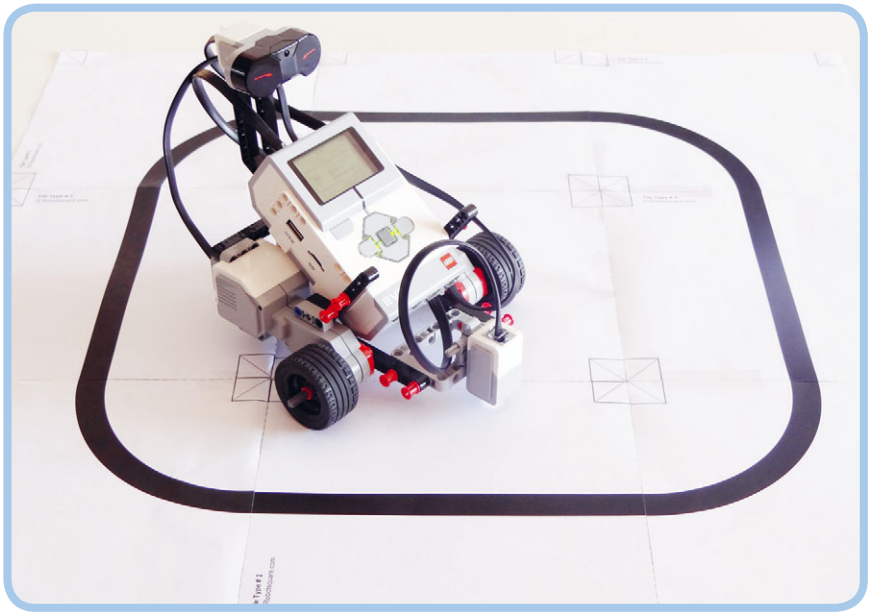


Рис. 7.4. Робот EXPLOR3R перемещается, не выезжая за пределы черного контура тестовой трассы

линии и склеить скотчем, чтобы получилось цельное полотно, как показано на рис. 7.4.

Если у вас нет возможности напечатать тестовую трассу, сделайте собственную, используя черный скотч на светлой поверхности, например на белой напольной плитке или большом листе фанеры.

Создание программы

На рис. 7.5 показана схема программы, которую вы должны создать, чтобы робот перемещался в пределах черного контура тестовой трассы. Она похожа на программу уклонения от препятствий, которую вы разрабатывали для датчика касания (см. рис. 6.7 в главе 6), только на этот раз программа отслеживает черную линию, а не нажатие кнопки.

Для создания этой программы вы будете использовать блок **Ожидание** (Wait) в режиме **Датчик цвета** ▸ **Сравнение** ▸ **Цвет** (Color Sensor ▸ Compare ▸ Color). В этой конфигурации вы можете выбрать цвета, которые датчик должен искать, с помощью параметра **Набор цветов** (Set of Colors). Когда обнаруживается какой-либо из них, блок

СДЕЛАЙ САМ № 6: БУЛЬДОЗЕР!

Сборка: **Программирование:**

С помощью программы StayInCircle робот будет перемещаться в разных направлениях. Если вы положите несколько деталей LEGO в круг, ваш робот может вытолкнуть их. Но сначала вам нужно снабдить EXPLOR3R специальным отвалом* бульдозера. Можете ли вы собрать такой отвал из деталей LEGO?

* Передняя часть бульдозера, которой он сгребает землю, мусор и т. п.

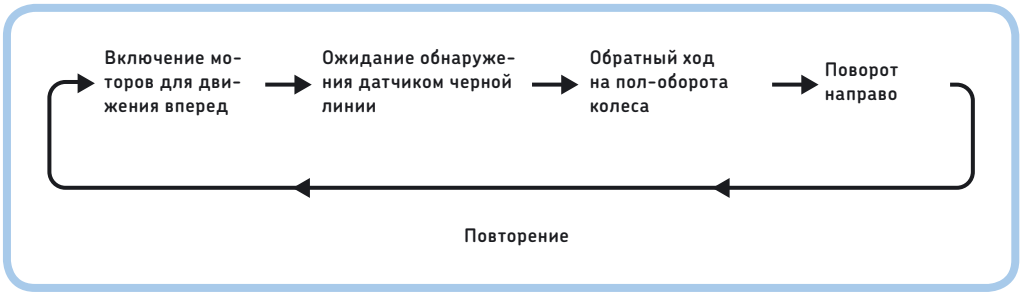


Рис. 7.5. Схема программы StayInCircle

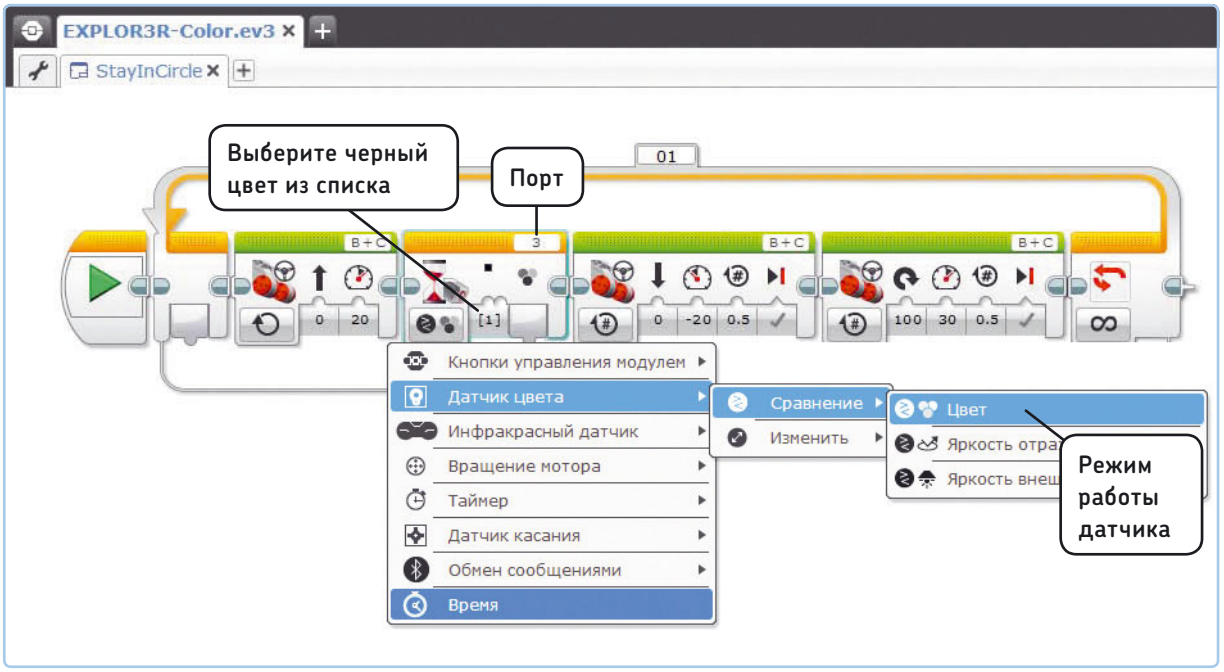


*Рис. 7.6. Новый проект EXPLOR3R-Color с одной программой StayInCircle. Чтобы настроить блок **Ожидание**, сначала выберите режим; затем откройте раскрывающийся список **Набор цветов** (Set of Colors) и выберите черный цвет (пункт **1**) из списка*

останавливается в ожидании, и программа переходит к следующему блоку. Рис. 7.6 демонстрирует готовую программу StayInCircle с блоком **Ожидание** (Wait), настроенным на ожидание черной линии. Поместите робота внутрь черного контура на тестовой трассе и запустите программу, чтобы увидеть ее в действии.

Движение по трассе

В своем следующем проекте вы будете использовать датчик цвета, чтобы создать робота, движущегося по трассе. Такой робот будет двигаться строго по линии тестовой трассы. Давайте рассмотрим принцип работы этой программы.

Когда робот следует по черной линии на белом фоне, датчик будет определять один из двух цветов: белый или черный. Поэтому, чтобы создать программу, определяющую трассу в черно-белой среде, вы можете использовать блок **Переключатель** (Switch), который отслеживает черный цвет. Когда датчик определяет черный цвет, блок **Переключатель** (Switch) запускает блок **Рулевое управление** (Move Steering) для выполнения одного движения; если блок **Переключатель** (Switch) определяет белый цвет, робот выполняет другое движение, как это показано на рис. 7.7.

Если робот обнаруживает белый цвет, он не будет знать, с какой стороны трассы он находится. Вы должны убедиться, что он всегда остается только с одной стороны линии; в противном случае он будет отклоняться от трассы в белую область. Вы можете сделать это, всегда поворачивая EXPLOR3R направо, когда он обнаруживает черный цвет, и налево, когда обнаруживает белый цвет. Если вы сделаете так, чтобы робот немного проезжал вперед, как бы подсказывая ему, и зациклите это поведение, робот будет следовать по трассе. Программа Colorline реализует эту стратегию, как показано на рис. 7.8.

ПРИМЕЧАНИЕ **Если ваш робот отклоняется от трассы в углах или на крутых поворотах, замедлите его движение (укажите 20% скорости вместо 25%).**

Теперь давайте посмотрим на поведение робота, когда он следует по линии трассы. Обратите внимание на то, что робот на самом деле следует по *краю* линии. Программа поддерживает настройки рулевого управления робота на основании данных, получаемых от датчика, заставляя его двигаться зигзагом у края линии: как только он обнаруживает черную линию, он пытается уйти от нее направо; как только он видит белую область, он пытается вернуться к трассе, двигаясь налево.

В результате робот движется таким образом, что линия трассы находится слева от датчика. Это означает, что, если поместить робота на трассе таким образом, чтобы он двигался по кругу по часовой стрелке, он придерживается внутреннего края окружности; если вы

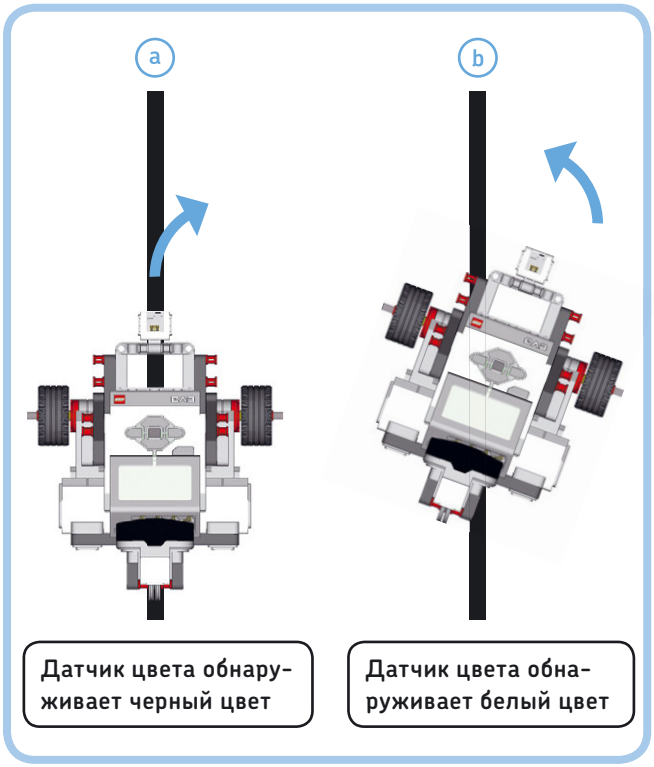


Рис. 7.7. Робот EXPLOR3R поворачивает вправо, если обнаруживает черную линию (a), и влево, если определяет белую область (b). Таким образом он движется вперед, поэтому, если зациклить это поведение, ваш робот в конечном итоге будет следовать по трассе

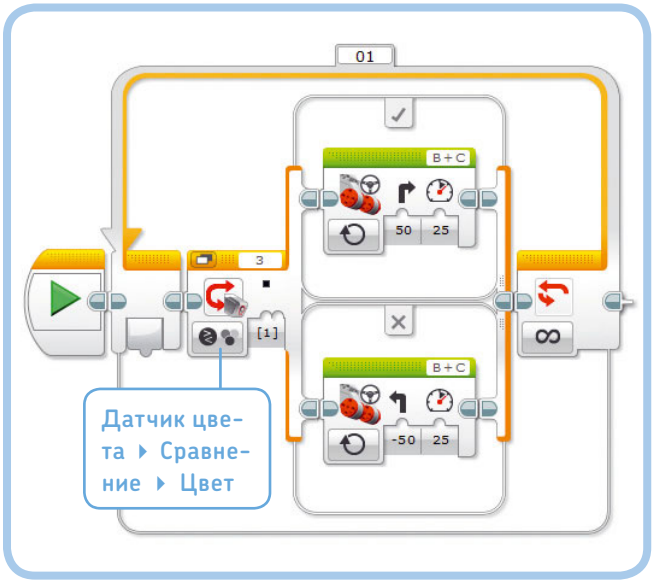


*Рис. 7.8. Программа Colorline. Обратите внимание, что блоки **Рулевое управление** находятся в режиме **Включить**. Как только робот начинает поворачивать, программа немедленно возвращается к началу для определения, был ли обнаружен другой цвет или поворот следует продолжить в том же направлении. Режим **Включить** лишь включает моторы, и программа продолжает выполняться*

пустите его против часовой стрелки, то он поедет по внешнему краю. Чтобы убедиться в этом, поверните робота во время работы программы, чтобы он двигался по линии в другую сторону; он должен начать двигаться по другому краю трассы.

Блок Переключатель в режиме Измерение

Блок **Переключатель** (Switch) в программе Colorline задает роботу движение прямо, если датчик обнаруживает черную линию. Если он определяет какой-либо другой цвет, например зеленый или красный, робот двигается влево. Изменив режим работы блока **Переключатель** (Switch) на **Датчик цвета** ▸ **Измерение** ▸ **Цвет** (Color Sensor ▸ Measure ▸ Color), вы можете настроить блок так, чтобы робот выполнял различные действия для каждого цвета.

Программа ShowColor на рис. 7.9 меняет цвет индикатора состояния модуля в зависимости от информации, полученной от датчика цвета. Блок **Переключатель** (Switch) в этой программе содержит четыре *варианта*, каждый из них включает один или несколько блоков. Каждый вариант соответствует различным замерам цвета: индикатор состояния подсвечивается зеленым цветом, если датчик обнаруживает зеленый; красным, если красный, и оранжевым, если желтый. Если цвет не обнаружен (⊘), индикатор состояния гаснет. Программа запускает блоки, относящиеся к тому варианту, который она определила, когда добралась до блока **Переключатель** (Switch).

Но что произойдет, если датчик обнаружил черный, синий, белый или коричневый? Если ни один из вариантов поведения не соответствует значению датчика, блок **Переключатель** (Switch) запускает вариант по умолчанию, который отмечен точкой, как показано на рис. 7.9. Здесь запускаются те же блоки, которые выполняются, когда цвет не обнаружен.

Теперь создайте программу и поместите датчик поверх распечатки файла *Датчик_цвета.pdf*, чтобы проверить работу датчика.

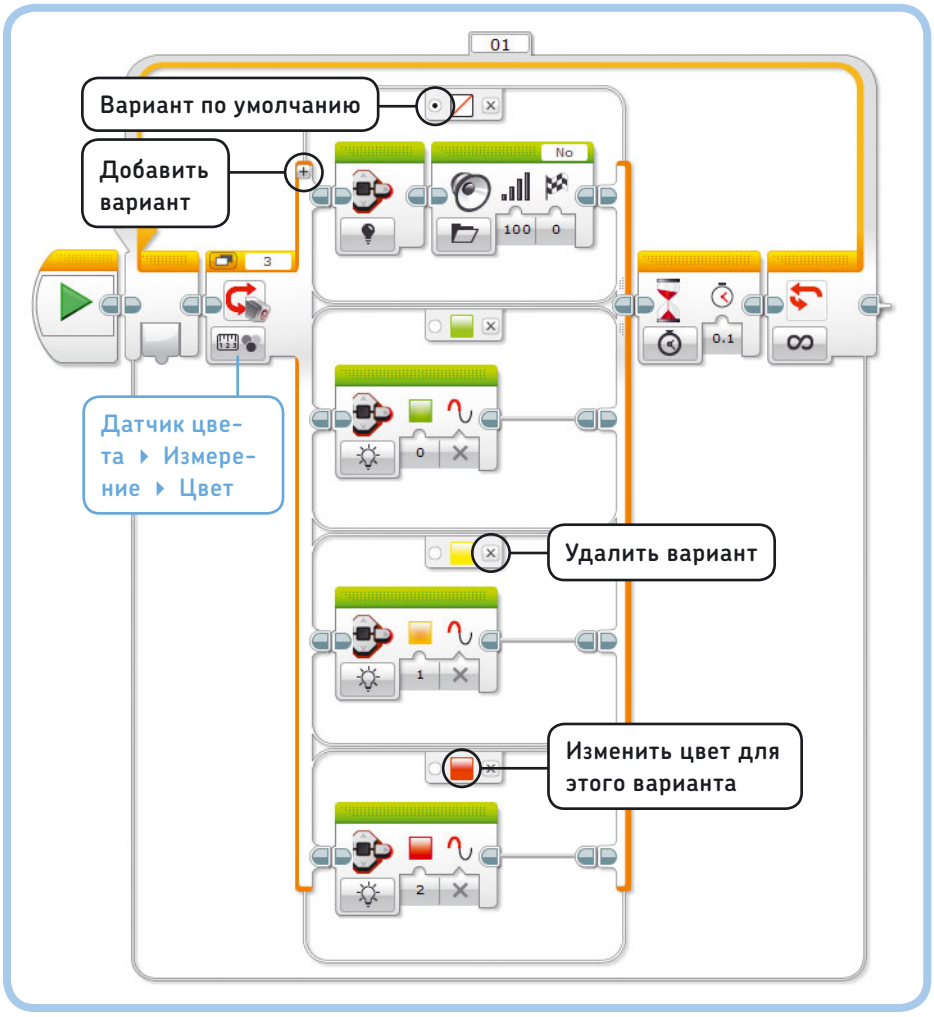


*Рис. 7.9. Программа ShowColor. Чтобы настроить варианты блока **Переключатель**, сначала выберите команду **Датчик цвета** ▸ **Измерение** ▸ **Цвет** и добавьте два случая, два раза нажав кнопку **+**. Теперь, когда ваш блок **Переключатель** содержит четыре варианта, выберите для каждого из них свой цвет. Не забудьте указать, что вариант, при котором цвет не обнаруживается, используется по умолчанию, установив переключатель в соответствующее положение (см. рисунок)*

ПРАКТИКУМ № 32: СОЗДАЙТЕ СОБСТВЕННУЮ ТРАССУ!

Сложность: **Время:**

Тестовая трасса, которую вы только что сделали, — отличное начало, но EXPLOR3R может освоить гораздо более сложные трассы. Перейдите по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**, чтобы скачать файл *Настраиваемая_трасса.pdf* и создать собственную трассу. Вы можете выбрать любые из тридцати типов элементов, включая прямые линии, углы и перекрестки. Напечатайте элементы трассы, которые вам нравятся, обрежьте их по пунктирным линиям и не забудьте использовать скотч, чтобы склеить поле воедино.

Для начала напечатайте четыре угла (четыре копии с. 3), зигзагообразную линию (с. 15), а также прямую линию, которую пересекает линия синего цвета (с. 18). С помощью этих элементов можно собрать трассу, показанную на рис. 7.10. Запустите программу Colorline, которую вы сделали, чтобы проверить EXPLOR3R на новой трассе.

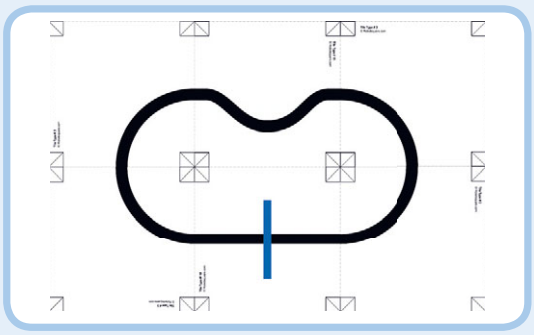


Рис. 7.10. Трасса для робота из практикумов № 32 и № 33

СОВЕТ **Некоторые принтеры позволяют печатать все необходимые страницы по очереди одной командой. Для этого надо ввести номера страниц в поле Страницы (Pages) настроек принтера следующим образом: 3,3,3,3,15,18.**

ПРАКТИКУМ № 33: ОСТАНОВИСЬ НА СИНИЙ!

Сложность: **Время:**

Измените программу Colorline так, чтобы робот следовал по черной линии трассы, которую вы сделали в практикуме № 32, пока она не пересечется с синей линией. Когда робот обнаружит синий цвет, он должен остановиться и воспроизвести звук.

СОВЕТ **Смените режим блока Цикл (Loop) на обнаружение синего цвета.**

ПРАКТИКУМ № 34: НАЗОВИ ЦВЕТ!

Сложность: **Время:**

Создайте программу, с помощью которой робот сможет сообщать вам, какой цвет он видит. Задайте такие параметры, чтобы он произносил «Blue», если датчик обнаружил синий цвет, и так далее. Если он не обнаружил цвет, настройте произнесение какой-либо другой фразы. Затем превратите блок **Переключатель** (Switch), который определяет цвет, в контейнер «Мой блок» с именем **SayColor**. Вы сможете использовать этот блок в любое время, когда захотите узнать, какой цвет обнаружил робот.

СОВЕТ **Ваша программа будет похожа на программу ShowColor.**

Режим Яркость отраженного света

Датчик цвета может измерять и яркость цвета в режиме **Яркость отраженного света** (Reflected Light Intensity). Например, датчик может определить разницу между белой, серой и черной бумагой, освещая бумагу и измеряя количество отраженного света. **Яркость отраженного света** измеряется в процентах от значения 0% (очень низкий коэффициент отражения: темный) до 100% (очень высокий коэффициент отражения: светлый).

Черная бумага отражает небольшое количество света, в результате чего замеры этого показателя будут ниже 10%. Белая бумага может показать результат измерения выше 60%. Чтобы проверить

ПРАКТИКУМ № 35: СУПЕРОТРАЖАТЕЛЬ!

Сложность: **Время:**

Подберите по крайней мере один материал, значение яркости отраженного света которого достигает 100%. Что это за материал и почему значение так высоко?

эти значения, запустите приложение **Port View** на модуле EV3, укажите порт **3**, а затем выберите пункт **COL-REFLECT** (см. рис. 7.3). Поставьте робота на распечатку файла *Датчик_цвета.pdf* и наблюдайте, как изменяется значение по мере передвижения робота по цветовой схеме с различными оттенками серого.

Установка порогового значения

В режиме **Цвет** (Color) датчик мог определять цвет тестовой трассы, черный или белый. Теперь наблюдайте за измерением отраженного света, передвигая робота руками (очень медленно!) от черной линии в белую область тестовой трассы. Вы заметите, что измеряемое значение постепенно увеличивается примерно от 6% (черный) до 62% (белый). Когда датчик частично фиксирует черную линию и частично белую бумагу, значение будет находиться в середине указанного диапазона, как будто датчик фиксирует серую поверхность. Это более детальное измерение вы можете использовать для улучшения вашего робота, движущегося по трассе.

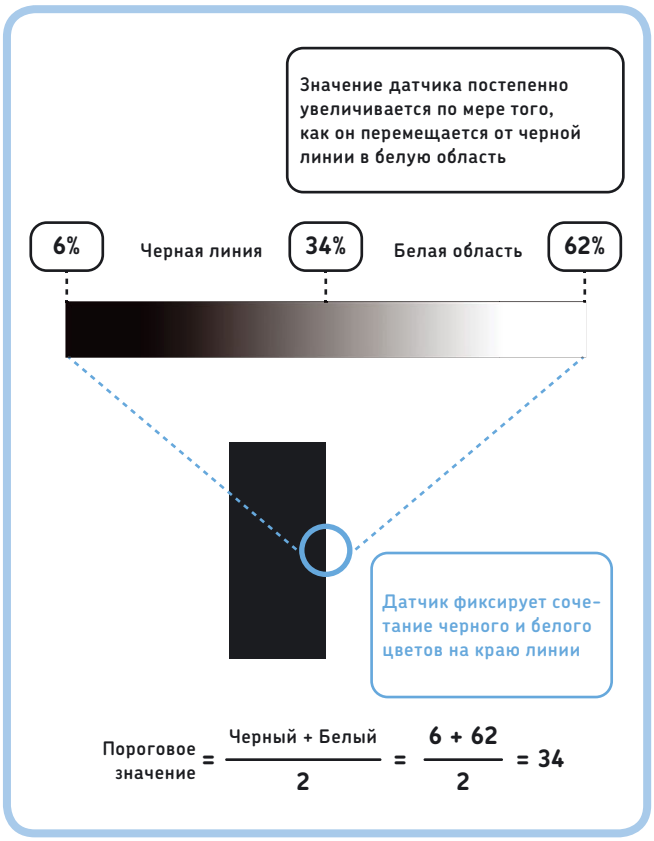


Рис. 7.11. На краю линии трассы датчик фиксирует сразу и черный и белый цвета и выдает среднее значение между ними, как будто перед ним серая поверхность. Пороговым считается среднее значение показаний датчика, которое он получил при измерении на черной линии (меньшее число) и на белой области (большее число). Чтобы вычислить среднее, сложите эти два значения и поделите сумму на 2

Чтобы сообщить роботу, какой цвет вы решили измерять — белую область или черную линию трассы, задайте белый и черный цвета в программе, используя *пороговое значение*. Измеренное значение выше порогового будет считаться белым цветом, а значение ниже этого порога — черным. Другими словами, вы принимаете темно-серые тона в качестве черных, а светло-серые в качестве белых, как показано на рис. 7.11.

Сравнение замеров датчика с пороговым значением

Теперь создадим для движения по трассе новую программу, которая использует режим **Яркость отраженного света** (Reflected Light Intensity) и пороговое значение, чтобы определить, где находится датчик — на черной линии или белой области. Как и прежде, робот должен повернуть направо, если он обнаружил черную линию, и налево — если белый цвет.

Для достижения этой цели используйте блок **Переключатель** (Switch) в режиме **Датчик цвета ▸ Сравнение ▸ Яркость отраженного света** (Color Sensor ▸ Compare ▸ Reflected Light Intensity), как показано на рис. 7.12. Введите ранее вычисленный порог в качестве значения параметра (Threshold Value). Параметр (Compare Type) определяет, какие показания датчика делают условие *истинным*, то есть какие значения будут запускать последовательность блоков в верхней части блока **Переключатель** (Switch). Вы можете выбрать для запуска верхние блоки, когда значение соответствует одному из следующих условий.

0. Равно (=) пороговому значению
1. Не равно (≠) пороговому значению
2. Больше чем (>) пороговое значение
3. Больше или равно (≥) пороговому значению
4. Меньше (<) порогового значения
5. Меньше или равно (≤) пороговому значению

Вам нужно, чтобы робот поворачивал направо, если датчик обнаруживает черный цвет (значение *меньше* порогового), поэтому вы можете выбрать это значение в качестве условия, запускающего верхнюю последовательность блоков. А теперь создайте и запустите программу ReflectedLine1 (см. рис. 7.12) и протестируйте робота точно так же, как в предыдущей программе.

ПРИМЕЧАНИЕ Обязательно рассчитайте собственные пороговые значения, а не используйте приведенные в диаграмме. У вас могут получиться другие результаты замеров для черного и белого цвета, в зависимости от таких факторов, как уровень освещенности в помещении, уровень заряда батареи робота и тип используемой бумаги.

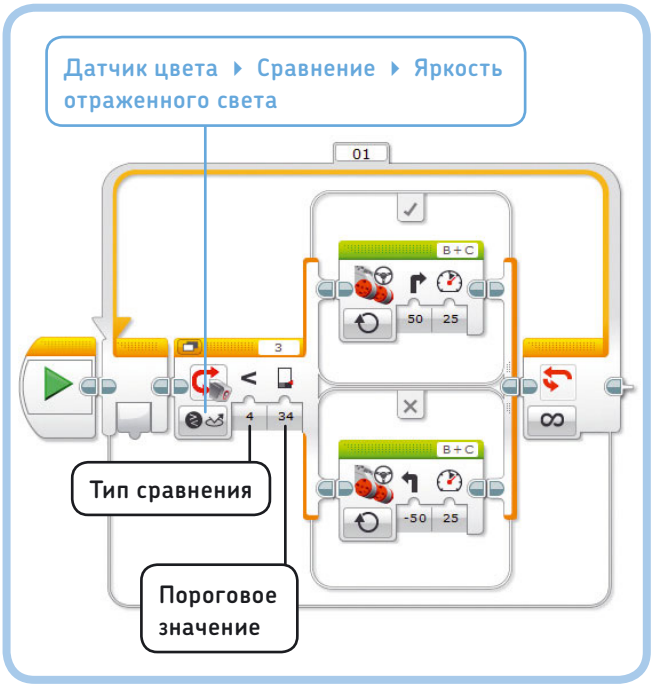


*Рис. 7.12. Согласно инструкциям программы ReflectedLine1 робот EXPLOR3R движется по линии тестовой трассы, используя режим **Яркость отраженного света**. Для параметра **Тип сравнения** используется значение **Меньше (<)** (под номером 4)*

Плавное движение по трассе

Преимущество режима **Яркость отраженного света** (Reflected Light Intensity) заключается в том, что робот, двигаясь по краю линии трассы, может измерять не только черный и белый, но и сочетание черного и белого цветов. В программе ReflectedLine1, если робот обнаруживает белый цвет, он делает резкий поворот влево, чтобы вернуться на трассу.

Но резкие повороты не нужны, если датчик находится близко к линии трассы. Если робот получил данные о светло-сером цвете, то для возвращения на трассу достаточно мягкого левого поворота. Все это ведет к тому, что робот плавно движется по трассе, а не дергается влево-вправо по нескольку раз. Иногда бывает недостаточно совершить мягкий поворот, чтобы вернуться к линии трассы, в таком случае робот должен сделать резкий поворот, когда он далеко от трассы и фиксирует белый цвет.

Чтобы определить, являются ли полученные роботом данные с датчика черным, темно-серым, светло-серым или белым цветом, нужно использовать два дополнительных пороговых значения, на полпути между тремя известными значениями, как показано на рис. 7.13.

Схема на рис. 7.14 иллюстрирует, как робот должен определить, в каком направлении повернуть и нужно ли сделать резкий или мягкий ход. Запустите программу ReflectedLine2 (рис. 7.15), чтобы убедиться, что робот движется по линии более плавно, чем раньше. Когда вы запустите программу, индикатор состояния модуля поможет вам определить, делает ли робот резкий поворот (индикатор подсвечивается красным цветом) или мягкий (зеленым цветом).

Режим Яркость внешнего освещения

Датчик цвета можно применить для определения уровня освещенности в помещении или яркости источника света, если выбрать режим **Яркость внешнего освещения** (Ambient Light Intensity). Этот режим можно использовать, чтобы робот

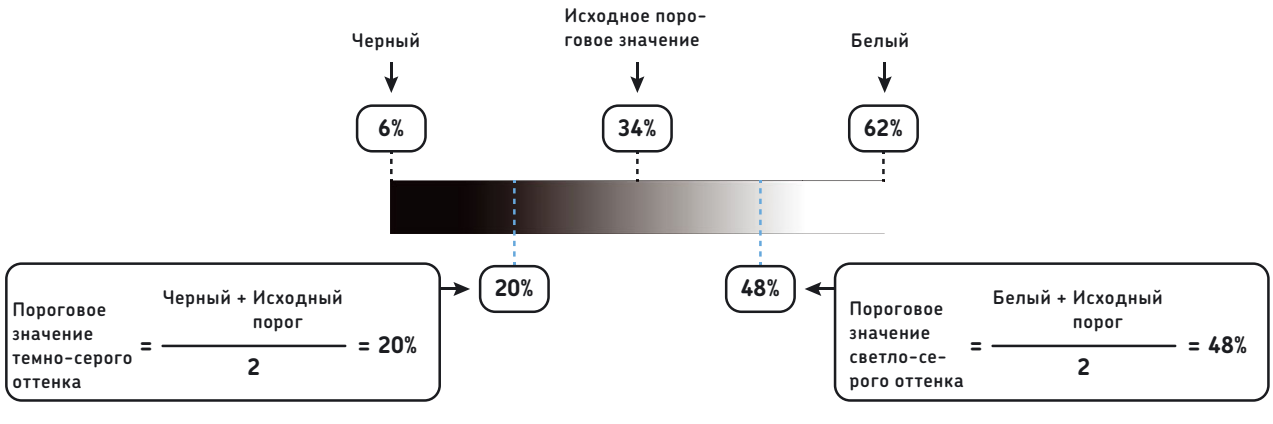


Рис. 7.13. Вам необходимо рассчитать два дополнительных пороговых значения, чтобы различать черный, темно-серый, светло-серый и белый цвета. Как и прежде, каждый порог представляет собой среднее от двух известных значений. Например, пороговое значение темно-серого цвета (20%) представляет собой среднее значение между черным (6%) и первоначальным пороговым значением (34%)

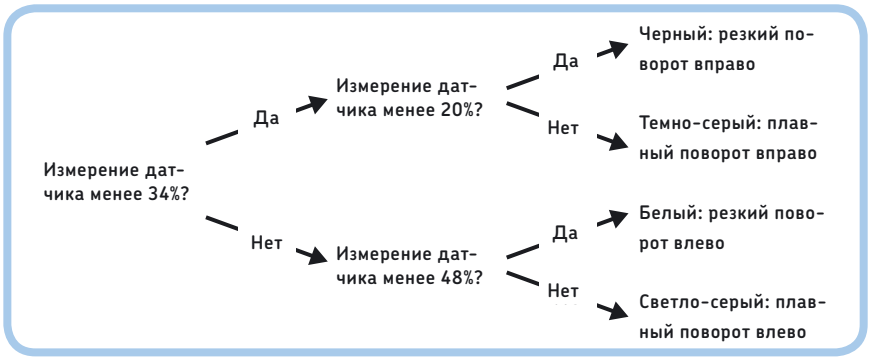


Рис. 7.14. Схема работы программы ReflectedLine2

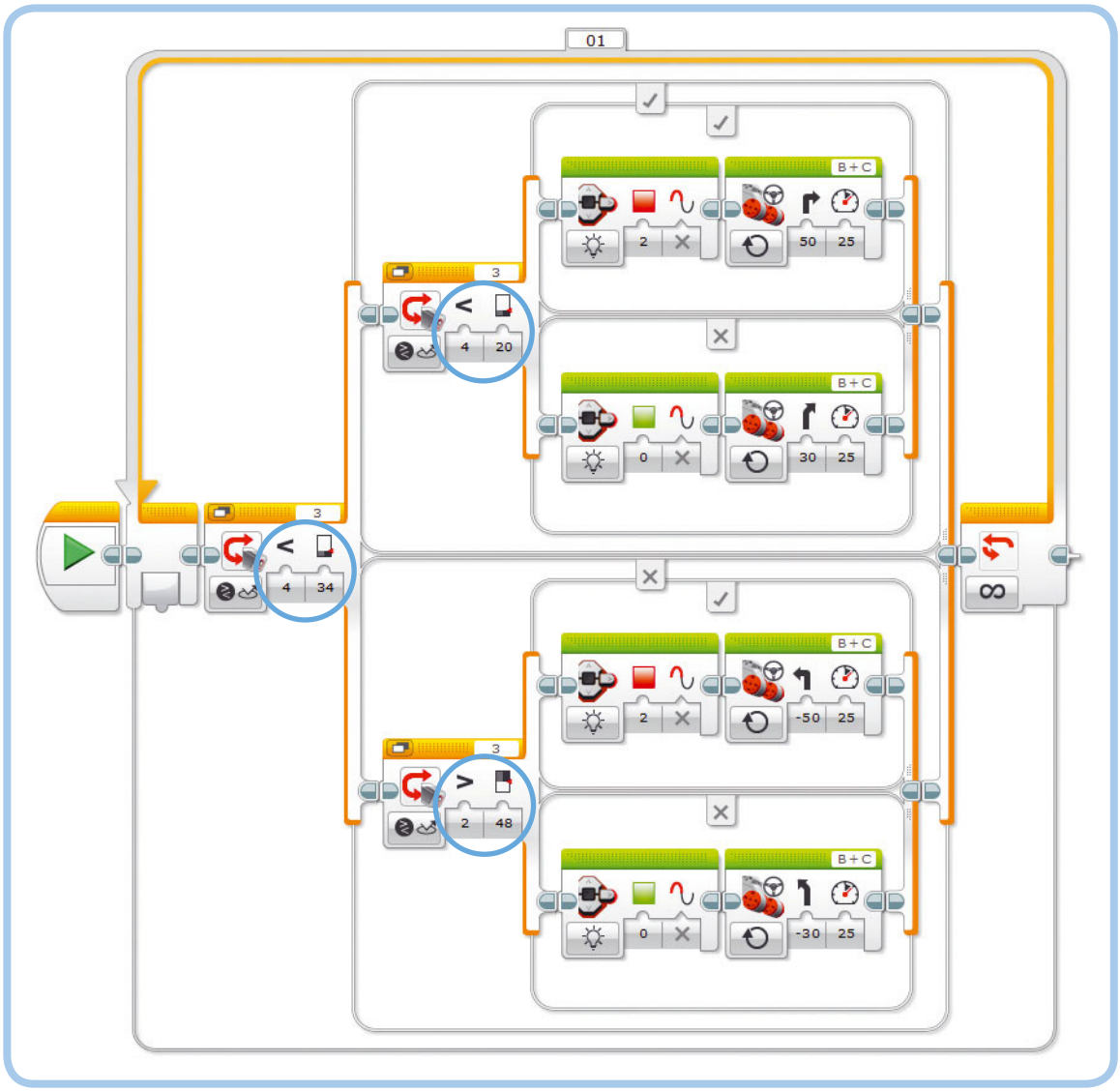


*Рис. 7.15. Программа ReflectedLine2. Обратите внимание, что блок **Переключатель** в нижней части сконфигурирован с возможностью запуска блоков в его верхней части, если показание датчика больше (>) светло-серого порогового значения. Для указания этого условия выберите вариант **2** в раскрывающемся списке **Тип сравнения***

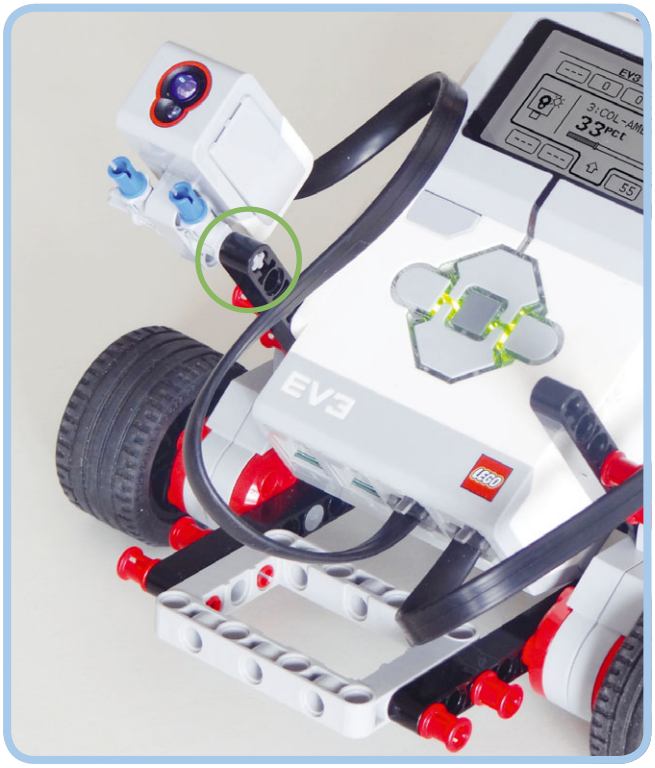


Рис. 7.16. Снимите датчик цвета с передней части робота и прикрепите его сбоку. Для этого вам не понадобятся дополнительные детали LEGO. Кабель остается подключенным к входному порту 3

учитывал, включен свет в комнате или выключен. Для полноценной работы рекомендуется закрепить датчик цвета так, как показано на рис. 7.16.

Измерение яркости внешнего освещения

Запустите программу **Port View** на модуле EV3 и выберите пункт **COL-AMBIENT**, чтобы просмотреть показания датчика о яркости окружающего освещения. Значение колеблется от 0% (очень темно) до 100% (очень ярко). Если вы, к примеру, прикроете датчик рукой, он передаст значение ниже 5%, а если поставите его рядом с лампой, то значение может составить около 70%.

В этой программе яркость окружающего света можно измерить теми же способами, которые применялись для замеров яркости отраженного света, но теперь вы будете использовать режим **Датчик цвета** ▸ **Сравнение** ▸ **Яркость внешнего освещения** (Color Sensor ▸ Compare ▸ Ambient Light Intensity). Чтобы определить, к примеру, включен ли свет в комнате, робот может использовать блок **Переключатель** (Switch), следящий, превышает ли показание датчика пороговое значение. В этой ситуации пороговым должно быть среднее значение между показаниями датчика, полученными при включенном свете и при выключенном.

ПРИМЕЧАНИЕ **В режиме Яркость внешнего освещения (Ambient Light Intensity) датчик излучает тусклый синий свет, но это никак не влияет на точность измерения. При этом отраженный свет не измеряется; датчик фиксирует только свет, поступающий из окружающей среды.**

Программа MorseCode

Теперь мы создадим программу, которая позволяет управлять роботом в темной комнате с помощью световых сигналов. Вы запрограммируете робота ехать вправо, если он фиксирует свет более двух секунд, и влево, если фиксируется короткий световой сигнал. Это позволяет управлять направлением движения робота путем включения и выключения света в комнате. Это упрощенная форма *азбуки Морзе* — способа связи, распространенного до изобретения телефона.

Вначале используется блок **Ожидание** (Wait), чтобы подождать поступления светового сигнала. После поступления сигнала выполняется ожидание с помощью другого блока **Ожидание** (Wait) в течение еще двух секунд, а затем программа использует блок **Переключатель** (Switch), чтобы определить, горит ли все еще свет. Если да, то робот поворачивает направо; в противном случае он поворачивает налево.

Создайте программу MorseCode, показанную на рис. 7.17, и запустите ее на роботе в темной комнате. Кроме того, вы

ПРАКТИКУМ № 36: УТРЕННИЙ БУДИЛЬНИК!

Сложность: **Время:**

Можете ли вы сделать так, чтобы ваш робот издавал сигнал тревоги, когда встает солнце? Поместите робота рядом с окном. В программе должен присутствовать блок **Ожидание** (Wait), который приостановит выполнение программы до тех пор, пока яркость внешнего освещения не поднимется выше порогового значения, которое вы вычислили. При достижении порога робот должен циклично воспроизводить громкий звук, пока вы не нажмете кнопку датчика касания, который действует как кнопка отключения.

СОВЕТ **Робот обычно выключается, если вы не используете его в течение 30 минут, поэтому он не разбудит вас утром. Чтобы ваш робот-будильник сработал, перейдите на вкладку Settings (Настройки) на экране модуля EV3; выберите пункт Sleep (Сон), а затем пункт Never (Никогда). На следующий день не забудьте вернуть параметру Sleep (Сон) значение 30 минут, чтобы батарея не разрядилась, если вы забудете выключить робота.**

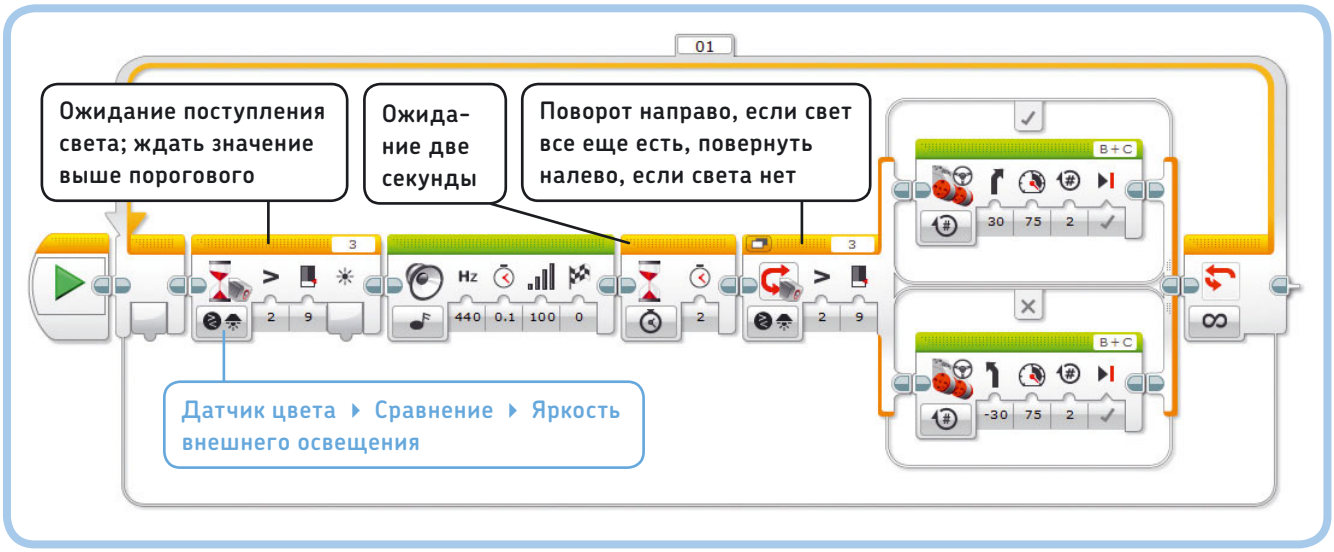


Рис. 7.17. Программа MorseCode. И первый блок ***Ожидание****, и блок* ***Переключатель*** *находятся в режиме* ***Датчик цвета ▸ Сравнение ▸ Яркость внешнего освещения***

можете попробовать протестировать ее в освещенной комнате, посылая световые сигналы ярким фонарем.

ПРИМЕЧАНИЕ **В моей комнате значение измерений датчика при выключенном свете составляет 2, и 16%, когда свет включен. Поэтому я выбрал пороговое значение 9% для программы MorseCode. Для ваших условий вы должны определить собственное пороговое значение.**

Дальнейшее изучение

Датчик цвета позволяет роботу «ощущать» окружающую среду путем обнаружения цвета, яркости отраженного света и яркости внешнего освещения. Робот использует информацию с этого датчика для выполнения задач различного вида. В этой главе вы запрограммировали EXPLOR3R на движение по трассе и реагирование на световые сигналы.

Вы также научились вычислять пороговые значения и сравнивать их с замерами датчика, чтобы ваш робот мог обнаруживать изменения в окружающей среде. Робот использует пороговые значения, чтобы определить, находится ли он на черной линии и включен ли свет в комнате. Пороговые значения применяются и для других датчиков, поэтому позднее мы снова будем их использовать.

Датчик цвета является универсальным устройством, и существует множество интересных способов его использования. Попытайтесь выполнить некоторые из практических задач, и вы увидите, что можно придумать!

ПРАКТИКУМ № 37: ЦВЕТОВЫЕ САЛКИ!

Сложность: **Время:**

Установите датчик цвета, как показано на рис. 7.16, и запрограммируйте робота на движение в разных направлениях в зависимости от того, какого цвета предмет вы подносите к датчику. Каждое движение робота должно длиться три секунды.

ПРАКТИКУМ № 38: СКАНЕР ОТПЕЧАТКОВ ПАЛЬЦЕВ!

Сложность: **Время:**

Можете ли вы разработать такую программу для робота, чтобы он поворачивал налево, если нажать кнопку датчика касания, и направо, если вы «активировали» датчик цвета? Снимите оба датчика с их креплений, чтобы вы могли держать их в руках. Затем подключите их к модулю EV3, используя самые длинные кабели, какие у вас есть. Как вы поступите, чтобы датчик цвета обнаруживал прикосновение ваших пальцев?

СОВЕТ **Что такое яркость отраженного света, когда вы поместите палец на датчик?**

ПРАКТИКУМ № 39: ЦВЕТОВОЙ ШАБЛОН!

Сложность: **Время:**

Дополните программу, которую вы сделали в практикуме № 37, чтобы робот начал реагировать на различные цветовые шаблоны. Например, если он фиксирует красный в течение двух секунд, пусть совершит резкий поворот влево, а если фиксирует красный в течение одной секунды, а затем синий в течение одной секунды, он должен совершить резкий поворот вправо.

СОВЕТ **Создайте программу аналогичную MorseCode.**

ПРАКТИКУМ № 40: ТРАССА С ПРЕПЯТСТВИЯМИ!

Сложность: **Время:**

Можете ли вы сделать робота, который движется по линии тестовой трассы и разворачивается, если встречает препятствие на пути? Снова закрепите датчик касания на роботе и прикрепите слева от него датчик цвета, как показано на рис. 7.18. Вы сможете установить его только на один из двух синих штифтов, но этого достаточно.

СОВЕТ **Измените блок Цикл (Loop) в программе движения по трассе так, чтобы цикл выполнялся, пока кнопка датчика касания нажата. Затем запрограммируйте робота на разворот, поиск линии трассы и движение по ней в противоположном направлении.**

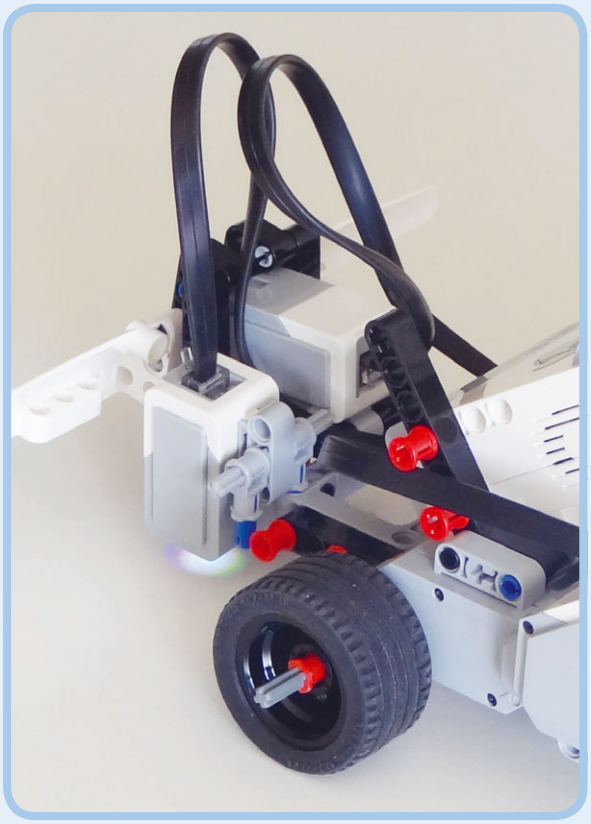


Рис. 7.18. Робот EXPLOR3R с двумя подключенными датчиками: цвета (порт 3) и касания (порт 1). Обратите внимание, что программа движения по трассе, которую вы создали ранее, должна работать с этой конфигурацией

ПРАКТИКУМ № 41: СУМАСШЕДШАЯ ДОРОЖКА!

Сложность: **Время:**

Откройте файл *Настраиваемая_трасса.pdf*, который можно скачать с сайта **eksmo.ru/files/Lego_Mindstorms_Primers.zip**, и распечатайте два угловых элемента трассы (две копии с. 3), трехсторонний перекресток (с. 3), круговой элемент (с. 17), желтый смайлик (с. 26) и зеленую звезду (с. 28), чтобы создать трассу, показанную на рис. 7.19. Запрограммируйте робота следовать по линии трассы до тех пор, пока он не встретит зеленую звезду, независимо от его исходного положения.

СОВЕТ **Запрограммируйте робота разворачиваться и следовать по трассе в противоположном направлении, если он встречает желтый смайлик.**

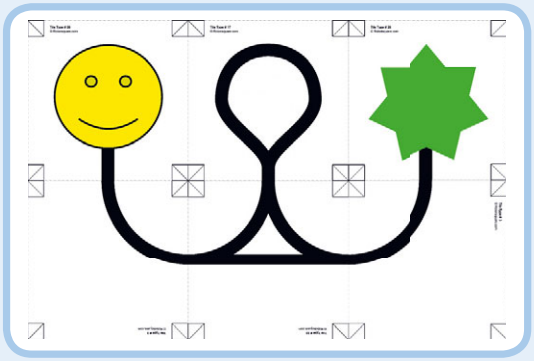


Рис. 7.19. Трасса для практикума № 41

СДЕЛАЙ САМ № 7: ДВЕРНОЙ ЗВОНОК!

Сборка: **Программирование:**

Можете ли вы сделать так, чтобы модуль EV3 воспроизводил звук, когда кто-то проходит через дверной проем? Установите датчик цвета на одной стороне дверной коробки и поместите фонарик на другой стороне, направив его прямо на датчик цвета. Как надо запрограммировать робота, чтобы он определял, когда кто-то проходит через дверь?

СОВЕТ **Когда кто-то блокирует свет от фонаря, проходя через дверной проем, яркость окружающего освещения падает.**

СДЕЛАЙ САМ № 8: ВКЛАД В БЕЗОПАСНОСТЬ!

Сборка: **Программирование:**

Смогли бы вы создать сейф для хранения ценностей, который бы открывался с помощью цветной карты безопасности, как показано на рис. 7.20? Один мотор используйте для того, чтобы перемещать каждый цветной квадрат на карточке безопасности мимо датчика цвета, а второй — чтобы открывать и закрывать сейф после успешного сканирования карты. Вы можете сканировать эту конкретную карту, используя следующие шаги.

1. Поверните колесо, пока датчик цвета не зафиксирует один из следующих цветов: красный, желтый, зеленый или синий.
2. Извлеките карту, если цвет не красный. Если красный — продолжайте.
3. Включите мотор, чтобы переместиться к следующему цвету.
4. Извлеките карту, если цвет не желтый. Если желтый — продолжайте.
5. Включите мотор, чтобы переместиться к следующему цвету.
6. И т. д.

СОВЕТ **Для каждого цветного квадрата используйте собственный блок Переключатель (Switch). Первый блок Переключатель (Switch) определяет, является ли первый цвет красным. Если это так (Истина (True)), мотор вращается и запускается следующий блок Переключатель (Switch), определяющий, является ли следующий цвет желтым. Если это так (Истина (True)), мотор вращается и так далее. Вариант Ложь (False) каждого блока Переключатель (Switch) должен содержать блоки, позволяющие извлечь карту.**

Рис. 7.20. Вы можете распечатать копию этой карты, скачав файл Карта_безопасности.pdf *с сайта eksmo.ru/files/Lego_Mindstorms_Primers.zip*

8

Использование инфракрасного датчика

Инфракрасный датчик позволяет роботу «видеть» окрестности путем измерения приблизительного расстояния до объекта с помощью инфракрасных лучей. Кроме того, датчик собирает данные от *удаленного инфракрасного маяка*. Датчик определяет, какие кнопки на маяке вы нажимаете, приблизительное расстояние до маяка, а также примерное направление, или курс, от робота к маяку.

Вы можете реализовать каждую из этих функций в ваших программах с использованием инфракрасного датчика в четырех различных режимах: **Приближение** (Proximity), **Удаленный** (Remote), **Приближение маяка** (Beacon Proximity) и **Направление маяка** (Beacon Heading) (рис. 8.1). Инфракрасный свет от маяка можно увидеть, глядя на него через цифровую камеру, например смартфона. Кроме того, можно увидеть слабый свет светодиода датчика, если тот находится в режиме **Приближение** (Proximity). В этой главе вы узнаете, как используется каждый режим, как разработать программы для EXPLOR3R, чтобы он избегал препятствий, реагировал на удаленные команды, а также находил маяк.

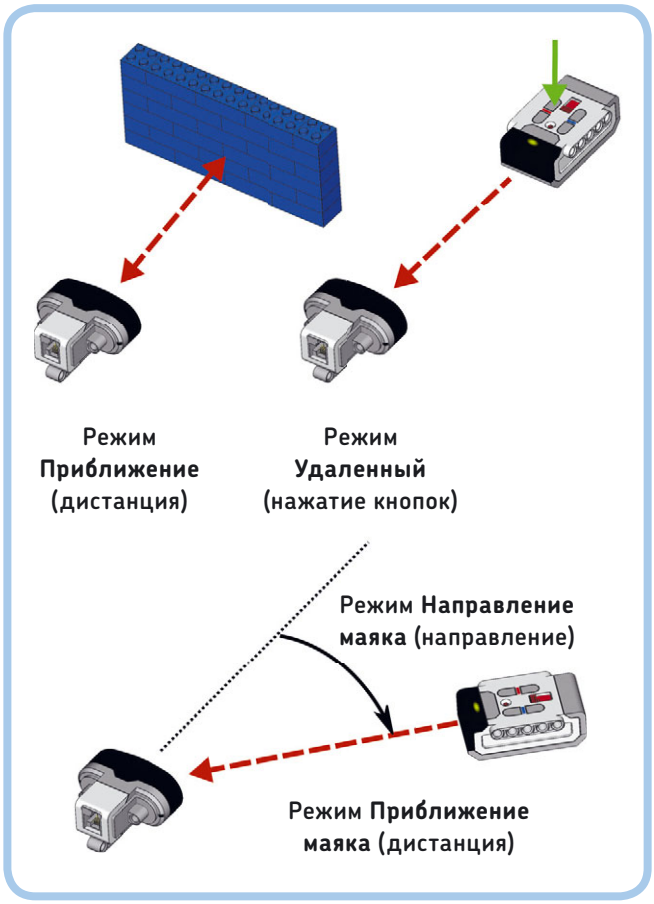


Рис. 8.1. Режимы работы инфракрасного датчика. Красные пунктирные линии обозначают невидимые лучи инфракрасного света. Если вы преградите путь между датчиком и маяком, датчик не сможет взять правильные замеры

Режим Приближение

В режиме **Приближение** (Proximity) робот измеряет расстояние от датчика до объекта. Но замеры производятся не в метрах или сантиметрах. Датчик определяет расстояние в процентах от 0% (очень близко) до 100% (очень далеко). Чтобы просмотреть показания датчика, запустите приложение **Port View** на модуле EV3, выберите входной порт 4, а затем — пункт **IR-PROX** (сокращение от словосочетания infrared proximity — инфракрасное приближение).

Значение определяется путем измерения, какая часть инфракрасных лучей, которые испускает датчик, отражается

обратно. Чем ближе объект к датчику, тем больше света возвращается. Некоторые поверхности отражают инфракрасный свет лучше, чем другие, поэтому кажется, что они находятся ближе. Например, расстояние до белой стены может казаться меньше, чем до черной, даже если они на самом деле на одинаковом расстоянии от датчика.

Избегание препятствий

Хотя датчик измеряет расстояние до объекта не совсем точно, он хорошо определяет, есть препятствие на пути или нет. Значение показаний датчика составляет 100%, когда он ничего не обнаружил, а если робот приближается к стене, значение уменьшается примерно до 30%. Чтобы робот избегал любые препятствия, нужно запрограммировать его на движение вперед, но, если он фиксирует объект, находящийся ближе, чем на 65% видимости (пороговое значение), он должен объехать его. В программе ProximityAvoid, показанной на рис. 8.2, используется блок **Ожидание** (Wait) для отслеживания значений менее 65%.

Совместное использование датчиков

Вы не ограничены применением только одного датчика. На самом деле, вы можете использовать все датчики, подключенные к EV3, в одной программе. Так вы получите более точные результаты замеров.

К примеру, инфракрасный датчик не всегда «видит» мелкие предметы на пути робота. Обнаружить их поможет датчик касания. С другой стороны, датчик касания не определит некоторые мягкие препятствия, например шторы, а инфракрасный датчик с этой задачей справится. Если комбинировать измерения датчиков, вероятность столкновения с препятствием у робота EXPLOR3R будет намного ниже.

Один из способов комбинации измерений датчиков заключается во вложении одного блока **Переключатель** (Switch) в другой, как показано на рис. 8.3. Подготовьте и запустите программу CombinedSensors, которая воспроизведет эту схему на практике, как показано на рис. 8.4.

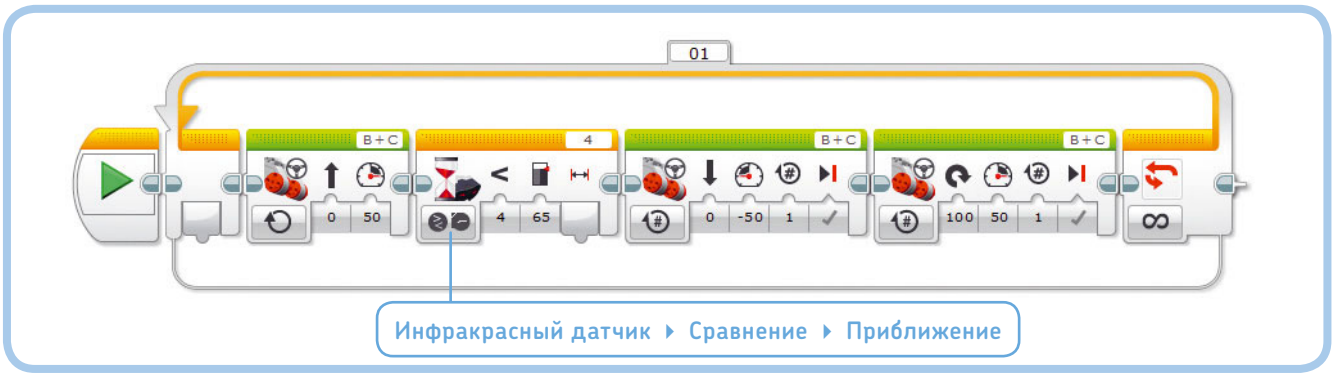


Рис. 8.2. Создайте новый проект под названием EXPLOR3R-IR и программу ProximityAvoid. Настройте блоки, как показано на рисунке. Обратите внимание на сходство с программой TouchAvoid, которую вы разработали в главе 6

ПРАКТИКУМ № 42: ИДУ НА СБЛИЖЕНИЕ!

Сложность: **Время:**

Запрограммируйте робота повторять слово «Обнаружено»*, если он заметил объект ближе 50%, в противном случае он должен произносить «Поиск»**. Потом попробуйте использовать другие пороговые значения, например 5 или 95%, чтобы разобраться, насколько близкие/далекие препятствия датчик может уверенно обнаруживать. Датчик не определяет точное расстояние, и вы увидите, что результаты варьируются в зависимости от того, какой тип объекта вы пытаетесь зафиксировать.

СОВЕТ **Вам нужно вложить блок Переключатель (Switch) в блок Цикл (Loop).**

ПРАКТИКУМ № 43: ТРИ ДАТЧИКА!

Сложность: **Время:**

Дополните программу CombinedSensors третьим датчиком. Задайте роботу такое поведение, чтобы он стоял на месте, если датчик цвета фиксирует синий объект, а при удалении синего объекта начинал движение, избегая препятствий.

* Пункт **Detected** в каталоге звуков Lego.
** Пункт **Searching** в каталоге звуков Lego.

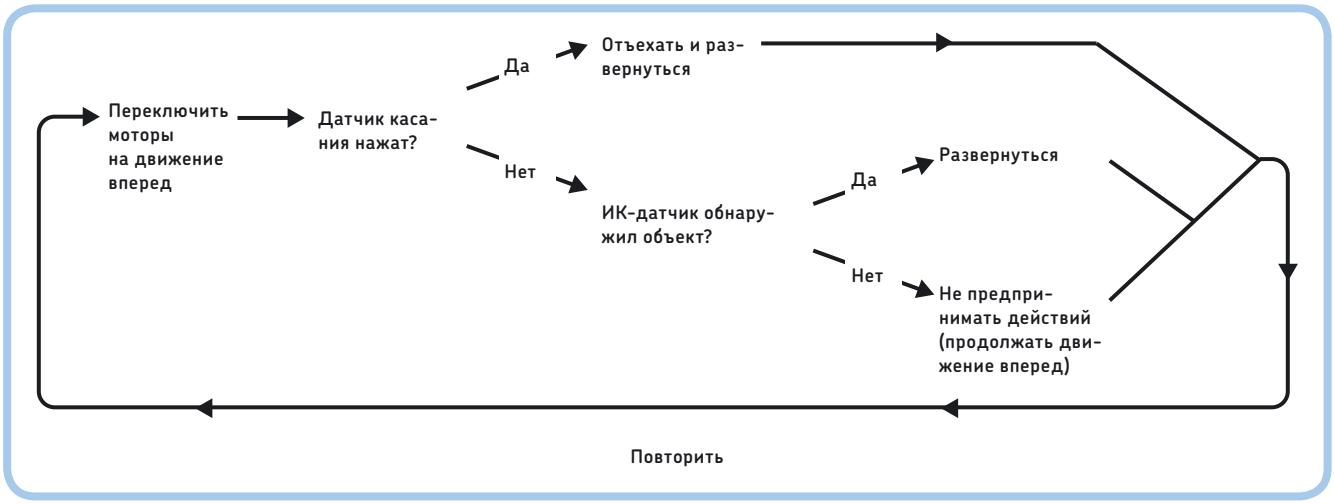


Рис. 8.3. Схема работы программы CombinedSensors

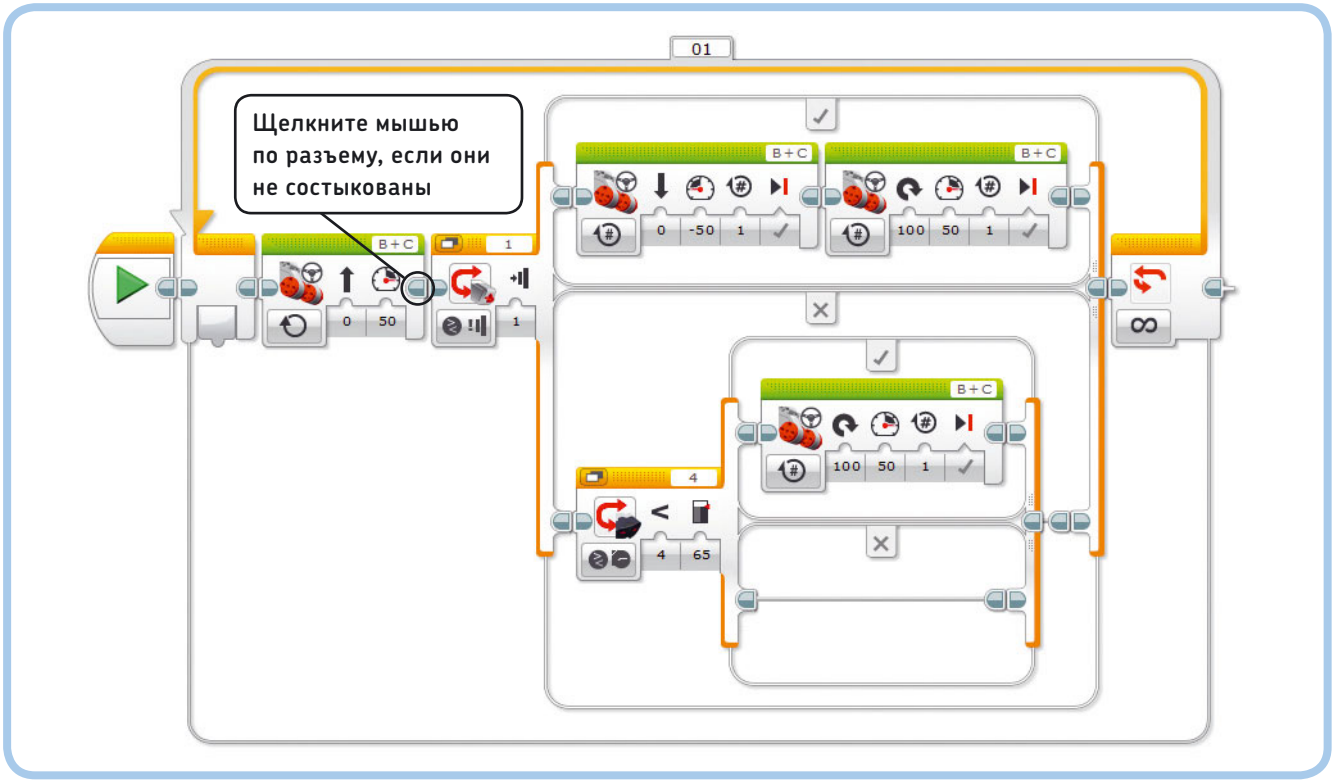


Рис. 8.4. Программа CombinedSensors. Измените размер блоков ***Переключатель*** *и при необходимости зациклите*

ПРИМЕЧАНИЕ **Не забудьте подключить датчик касания, которым вы оборудовали робота в главе 6. Датчик касания должен быть подключен к входному порту 1.**

Режим Удаленный

В режиме **Удаленный** (Remote) инфракрасный датчик фиксирует, какие кнопки нажаты на удаленном инфракрасном маяке, что позволяет разрабатывать программы, которые выполняют разные операции при нажатии каждой кнопки. Это похоже на то, как вы удаленно управляли роботом в главе 2. Приложение **IR Control**, предустановленное в модуле EV3, позволяет роботу резко поворачивать в разных направлениях согласно нажатиям кнопок на маяке. Датчик различает 12 комбинаций кнопок, или *идентификаторов*, как показано на рис. 8.5.

Блок **Переключатель** (Switch) в режиме **Измерение** (Measure) позволяет выбрать последовательность блоков, которую нужно запустить, для каждого значения, полученного от датчика, — каждого *идентификатора*. Программа CustomRemote использует этот переключатель, чтобы робот EXPLOR3R двигался вперед, когда нажаты две верхние кнопки (идентификатор 5). Робот поворачивает налево, когда нажата верхняя левая кнопка (идентификатор 1), и направо, если нажата верхняя правая кнопка (идентификатор 3). Если ни одна кнопка не нажата (идентификатор 0), робот прекращает движение. Бездействие — состояние по умолчанию, поэтому робот не двигается и при нажатии неопределенной комбинации кнопок.

Поскольку программа настроена для ожидания сигнала маяка на канале 1, вам нужно выбрать его на маяке (см. рис. 2.10). Если у вас есть еще один робот EV3, переключите его на другой канал (2, 3 или 4), чтобы избежать помех. Теперь создайте и запустите программу (рис. 8.6) и начните управлять вашим роботом дистанционно с помощью маяка.

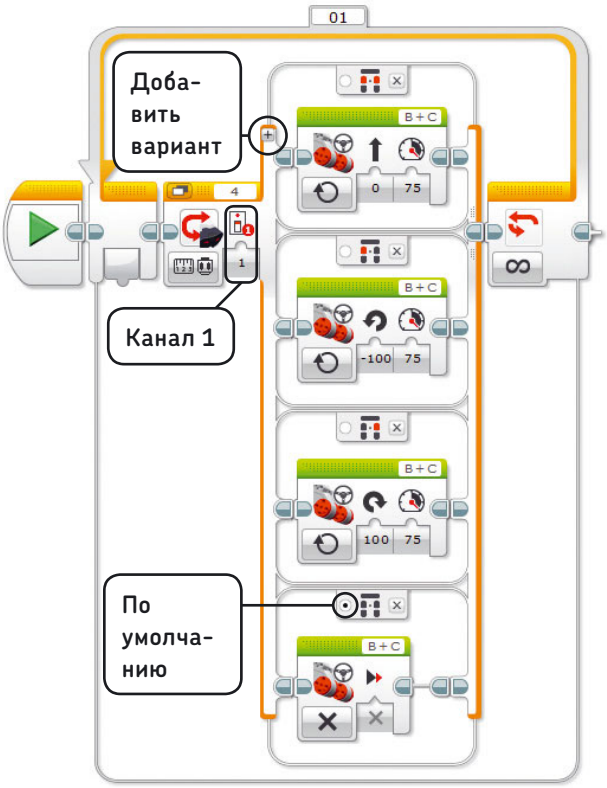


Рис. 8.6. Программа CustomRemote

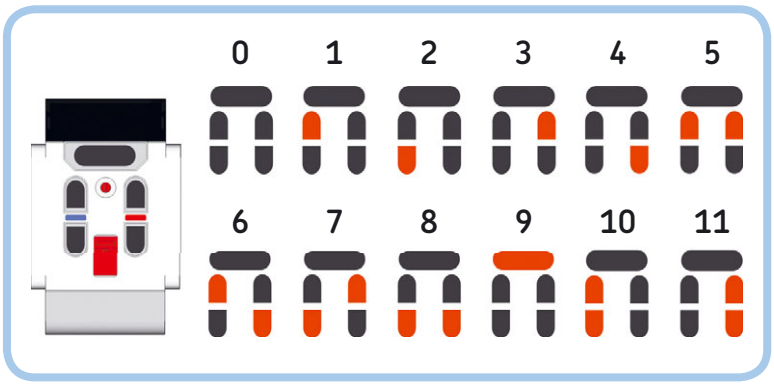


Рис. 8.5. Инфракрасный датчик определяет 12 комбинаций кнопок (идентификаторов), *которые можно нажать на удаленном инфракрасном маяке. Нажатые кнопки окрашены в красный цвет*

ПРАКТИКУМ № 44: РАЗБЛОКИРУЙ ДИСТАНЦИОННОЕ УПРАВЛЕНИЕ!

Сложность: **Время:**

Можете ли вы защитить вашу программу с помощью секретной комбинации кнопок? В программе CustomRemote прямо перед блоком **Цикл** (Loop) добавьте два блока **Ожидание** (Wait). Настройте эти блоки на ожидание нажатия идентификаторов 10 и 11 в указанном порядке, до того как остальная часть программы может быть выполнена. В качестве дополнительного задания попробуйте сделать программу более защищенной, используя способ, описанный в практикуме «Сделай сам № 8» на с. 110.

Этот способ особенно полезен (и весел!), потому что теперь вы можете сделать программу дистанционного управления для всех ваших роботов. Например, в главе 12 вы соберете гоночную машину EV3, которая ездит и управляется рулем иначе, чем EXPLOR3R. Стандартное приложение **IR Control** не сможет работать с этим роботом. Но вы можете решить эту проблему, создав собственную программу, позволяющую управлять этой машиной.

Режим Приближение маяка

Помимо определения кнопок, которые вы нажимаете на маяке, инфракрасный датчик может распознать мощность сигнала и направление, в котором сигнал поступает при нажатии любой кнопки. Робот может использовать эту информацию, чтобы определить расположение маяка и двигаться к нему.

В режиме **Приближение маяка** (Beacon Proximity) датчик использует мощность сигнала маяка для расчета относительного расстояния в диапазоне от 1% (маяк очень близко к датчику) до 100% (очень далеко). Для достижения наилучших результатов держите маяк примерно на такой же высоте или чуть выше датчика и направьте его в глаза робота (см. рис. 8.1).

Зеленый индикатор на маяке указывает на то, что он отсылает сигнал. Не имеет значения, какую кнопку вы нажимаете в режиме приближения или направления, но удобнее всего использовать кнопку в верхней части (идентификатор 9). Зеленый индикатор включается при однократном нажатии кнопки и выключается при повторном нажатии, поэтому вам не нужно удерживать кнопку.

Теперь вы создадите программу BeaconSearch1, согласно которой робот неоднократно произносит слово «Поиск»*, пока замеряемое инфракрасным датчиком расстояние составляет менее 10%. Другими словами, робот будет повторять слово «Поиск», пока маяк находится очень близко от датчика. Вы можете сделать это с помощью блока **Цикл** (Loop), настроенного, как показано на рис. 8.7. После завершения цикла робот говорит «Обнаружено»**.

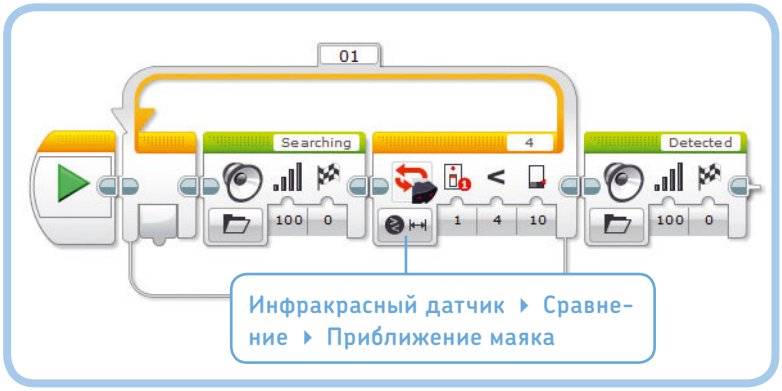


Рис. 8.7. Программа BeaconSearch1

ПРИМЕЧАНИЕ **Датчик может сообщать, обнаружен ли сигнал вообще. Вы можете сделать, к примеру, так, что робот остановится, если датчик перестал фиксировать сигнал. Подобная программа требует применения некоторых незнакомых вам блоков, которые мы рассмотрим в главе 14.**

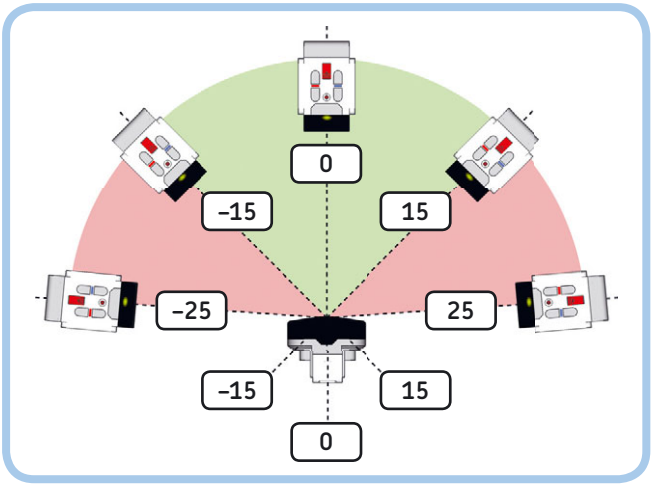


*Рис. 8.8. Диапазон допустимых значений в режиме **Направление маяка** находится в пределах от –25 до 25. Отрицательные значения указывают, что маяк слева от датчика; положительные — что он справа. Значение, близкое к нулю, говорит о том, что робот обнаружил маяк прямо впереди или позади себя*

Режим Направление маяка

Инфракрасный датчик может измерять направление сигнала маяка, используя режим **Направление маяка** (Beacon Heading). Этот режим дает роботу приблизительное представление об угле между маяком и датчиком. Результат измерения представляет собой число в диапазоне от –25 до 25, как показано

* Пункт **Searching** в каталоге звуков Lego.
** Пункт **Detected** в каталоге звуков Lego.

на рис. 8.8. Датчик не определяет угол с точностью, но делает это достаточно хорошо, чтобы распознать, находится ли маяк слева от робота (отрицательные значения) или справа (положительные значения).

Датчик способен обнаружить маяк во всех направлениях, даже если он находится позади, но самое надежное измерение получается в зеленой области, схематически представленной на рис. 8.8. Значение, зафиксированное датчиком, будет стремиться к нулю, если маяк находится прямо перед датчиком или прямо позади него, но оно равно нулю и тогда, когда датчик вообще не обнаруживает сигнал.

Информация о том, находится ли маяк слева или справа от робота, — все, что вам нужно, чтобы робот направился в сторону маяка. Когда датчик определяет, что маяк слева, робот поворачивает налево, а если справа, то направо. Вы можете использовать блок **Переключатель** (Switch), чтобы определять замер датчика меньше (<) 0, информирующий о том, что маяк находится слева.

Если постоянно корректировать направление движения, основываясь на измерениях датчика и одновременно двигаясь вперед, ваша программа успешно справится с поиском маяка. Удалите блоки **Звук** (Sound) из предыдущей программы и добавьте один блок **Переключатель** (Switch) и два блока **Рулевое управление** (Move Steering) для доработки программы BeaconSearch2, как показано на рис. 8.9. Используемый в программе блок **Цикл** (Loop) инструктирует робота продолжать поиск маяка до тех пор, пока расстояние до него будет не менее 10%; при достижении этого результата программа завершает работу.

ПРАКТИКУМ № 45: ПЛАВНОЕ ПРЕСЛЕДОВАНИЕ!

Сложность: **Время:**

Можете ли вы дополнить программу BeaconSearch2 так, чтобы робот двигался к маяку плавнее? Робот совершает плавные повороты (значение 25% параметра **Рулевое управление** (Move Steering)), если маяк находится в зеленой зоне (см. рис. 8.8), и резкие повороты (значение 50% параметра **Рулевое управление** (Move Steering)), если маяк находится в красной зоне.

СОВЕТ **Используйте способы, которые вы изучили в разделе «Плавное движение по трассе» в главе 7. Вам не нужно рассчитывать пороговые значения; они приведены на рис. 8.8.**

ПРИМЕЧАНИЕ **Не забудьте сделать так, чтобы маяк передавал сигнал непрерывно, либо удерживайте одну из кнопок нажатой, либо переключите кнопку в верхней части (идентификатор 9), чтобы зеленый индикатор горел постоянно.**

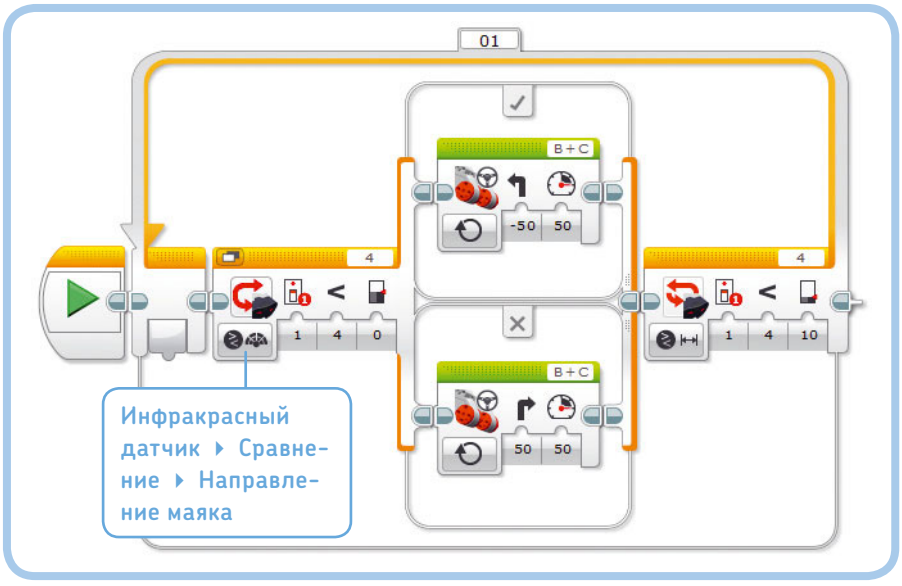


Рис. 8.9. Программа BeaconSearch2 заставляет робота двигаться к маяку. Когда маяк окажется близко — робот остановится. Если маяк все время перемещать, робот будет следовать за вами

Комбинированный режим работы датчика

Объединение нескольких режимов работы инфракрасного датчика в одной программе чревато тем, что ваш робот будет вести себя непредсказуемо, поскольку датчику нужно время, чтобы переключиться с одного режима на другой. **Приближение маяка** (Beacon Proximity) и **Направление маяка** (Beacon Heading) — это единственные режимы работы инфракрасного датчика, которые можно использовать вместе без задержек.

К примеру, нужно изменить блок **Цикл** (Loop) в программе CustomRemote (см. рис. 8.6), повторяя цикл до тех пор, пока датчик не определит значение приближения менее 10%. Блок **Переключатель** (Switch) определяет нажатие кнопки в режиме **Удаленный** (Remote), а затем блок **Цикл** (Loop) считывает значение приближения в режиме **Приближение** (Proximity). Но, поскольку датчик должен каждый раз переходить из одного режима в другой, программа будет работать очень медленно, и ваш робот не сможет должным образом реагировать на маяк. (Попробуйте, если вас это устраивает.)

Если время не имеет для ваших задач решающего значения, то использование различных режимов в одной программе — это замечательно. Например, программа MultiMode, показанная на рис. 8.10, работает именно так. Сначала она ждет, пока вы нажмете правую верхнюю кнопку на маяке (идентификатор 3), после чего воспроизводится звуковой сигнал. Потом робот говорит: «Да»*, если значение приближения ниже 30%; в противном случае он говорит: «Нет»**. Между звуковым сигналом и произнесенным словом будет пауза — это задержка, вызванная переключением режимов с **Удаленный** (Remote) на **Приближение** (Proximity).

Дальнейшее изучение

Инфракрасный датчик позволяет вашему роботу обнаруживать окружающие его объекты на расстоянии. В сочетании с удаленным инфракрасным маяком датчик может выступать

ПРАКТИКУМ № 46: СЛЕДУЙ ЗА МНОЙ!

Сложность: **Время:**

Можете ли вы запрограммировать робота EXPLOR3R следовать за вами по прямой, сохраняя постоянную дистанцию? Используйте инфракрасный датчик в режиме **Приближение** (Proximity) для определения расстояния до вашей руки (держите ее перед роботом). Робот должен следовать за вами, когда вы перемещаете руку; если вы поднесете руку ближе к роботу, он должен откатиться назад. Пусть робот стоит на месте, если он «видит» вашу руку на расстоянии от 35 до 45%.

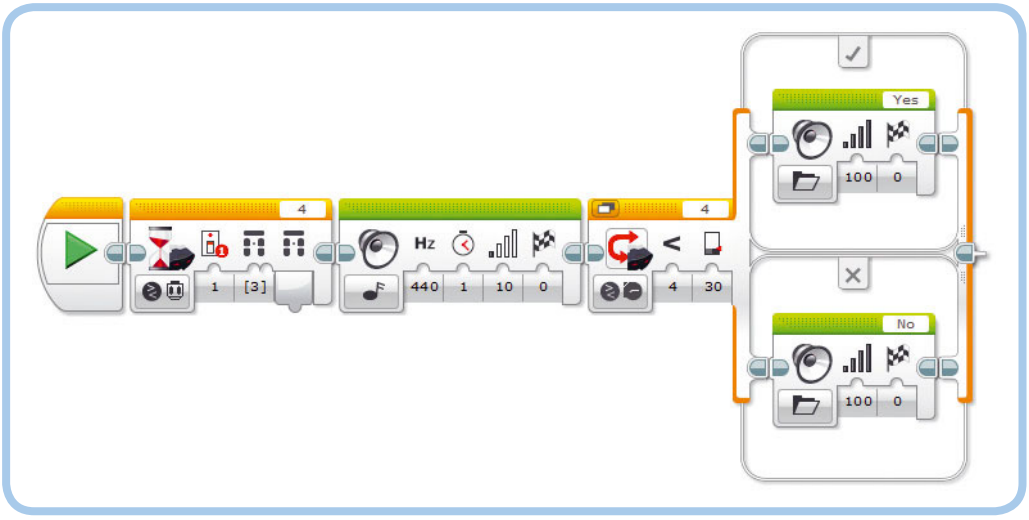


*Рис. 8.10. Программа MultiMode. В этом случае время выполнения программы не имеет решающего значения, поэтому допустимо использовать режимы **Удаленный** и **Приближение** в одной программе*

* Пункт **Yes** в каталоге звуков Lego.

** Пункт **No** в каталоге звуков Lego.

ПРАКТИКУМ № 47: АКУСТИЧЕСКИЙ ЛОКАТОР!

Сложность: **Время:**

Можете ли вы сделать, чтобы EV3 издавал звуки так, чтобы привести вас к маяку с закрытыми глазами? Настройте его издавать тональные звуки разной частоты и громкости в зависимости от местоположения маяка. Если маяк находится слева от датчика, следует воспроизвести звук низкой частоты (400 Гц), а если справа — звук высокой частоты (1000 Гц). Чем ближе вы подходите к маяку, тем громче должен быть звук.

СОВЕТ **Во-первых, используйте блок Переключатель (Switch), чтобы определить, с какой стороны находится маяк — слева или справа. Во-вторых, в обеих частях этого переключателя поместите блок Переключатель (Switch), чтобы определить, далеко находится маяк или близко. После этого у вас будет четыре варианта размещения блока Звук (Sound), каждый из которых настроен с возможностью воспроизведения одного из тонов: низкого и громкого, низкого и тихого, высокого и громкого и высокого и тихого.**

в качестве приемника инфракрасного сигнала и пеленгатора маяка. Кроме того, вы узнали, как объединить датчик касания и инфракрасный датчик, чтобы робот с большей эффективностью избегал препятствий. Чтобы еще больше усложнить программу, вы можете добавить датчик цвета. Прежде чем переходить к следующей главе, проверьте свои навыки по сборке и программированию, выполнив следующие практические задания!

СДЕЛАЙ САМ № 9: ЖЕЛЕЗНОДОРОЖНЫЙ ПЕРЕЕЗД!

Сборка: **Программирование:**

Сможете ли вы разработать автоматизированный железнодорожный переезд для модели поезда LEGO? Используйте мотор, чтобы поднимать шлагбаум, который препятствует автомобилю пересекать железную дорогу, когда проходит поезд. Используйте инфракрасный датчик или датчик цвета, чтобы определить, когда поезд приближается и шлагбаумы должны быть опущены, а когда они должны быть снова подняты.

СДЕЛАЙ САМ № 10: НАДЕЖНЫЙ СИГНАЛ ТРЕВОГИ!

Сборка: **Программирование:**

Можете ли вы применить все три датчика из набора MINDSTORMS EV3 для создания аварийного сигнала о вторжении, который срабатывает всегда? Используйте датчик касания для определения открытия двери (практикум «Сделай сам № 4» на с. 96), датчик цвета для обнаружения людей, перешагнувших порог (практикум «Сделай сам № 7» на с. 110), а также инфракрасный датчик в режиме **Приближение** (Proximity) для распознавания движения вблизи определенного объекта, например смартфона.

СОВЕТ **Настройте вашего робота и программу таким образом, чтобы вы (и только вы!) могли войти в комнату без тревожного сигнала.**

9

Использование кнопок модуля EV3 и датчиков вращения мотора

Помимо датчиков касания, цвета и инфракрасного датчика, EV3 содержит два типа встроенных датчиков: кнопки модуля и датчики вращения мотора. Вы можете использовать кнопки на модуле EV3, чтобы управлять программой в процессе ее работы. Например, программа может вывести запрос нажать кнопку, предлагая выбрать один из вариантов дальнейшего поведения робота.

Каждый мотор EV3 содержит встроенный датчик вращения, который определяет позицию мотора. Это позволяет точно управлять колесами или другими механизмами. Кроме того, датчик измеряет скорость вращения мотора, позволяя выявить ситуации, когда мотор вращается быстрее или медленнее, чем предполагалось.

Использование кнопок модуля EV3

Вы можете использовать кнопки «Вверх», «Вниз», «Влево», «Вправо» и центральную кнопку модуля EV3 в ваших программах по аналогии с датчиком касания. Например, вы можете сделать так, чтобы ваш робот реагировал воспроизведением звука на нажатие конкретной кнопки. А можно настроить робота на ожидание момента, когда кнопка будет отпущена или щелкнута (нажата и сразу же отпущена).

Еще один интересный способ использования кнопок модуля EV3заключается в создании экранного меню, позволяющего выбрать следующее действие в программе. Программа ButtonMenu на рис. 9.1 воспроизводит один из трех звуков в зависимости от того, какая кнопка нажата.

Два блока **Экран** (Display) отображают простое меню на экране, предлагая пользователю выбрать, следует ли роботу сказать: «Привет», «О'кей» или «Да»*. Затем блок **Ожидание** (Wait) (в режиме **Кнопки управления модулем** ▸ **Сравнение** (Brick Buttons ▸ Compare)) приостанавливает выполнение

* Пункты **Hello**, **Okay** и **Yes** в каталоге звуков Lego.

ПРАКТИКУМ № 48: ДЛИННОЕ СООБЩЕНИЕ!

Сложность: **Время:**

При отображении длинного сообщения на экране модуля EV3 оно может не вместиться полностью. Создайте программу, которая позволит использовать кнопку «Вниз» для прокрутки сообщения.

СОВЕТ **Научите робота показывать на экране какой-либо новый текст при каждом нажатии кнопки.**

ПРАКТИКУМ № 49: ПОЛЬЗОВАТЕЛЬСКОЕ МЕНЮ!

Сложность: **Время:**

Можете ли вы дополнить программу ButtonMenu, чтобы ваш робот выполнял какие-то полезные действия, помимо воспроизведения звуков? Возьмите три программы, которые вы сделали ранее, превратите их в контейнеры «Мой блок», а затем поместите их в блок **Переключатель** (Switch) программы ButtonMenu. Измените настройки блоков **Экран** (Display), чтобы они демонстрировали то, что происходит при нажатии на каждую кнопку.

СОВЕТ **Этот метод часто используется в соревнованиях робототехники, поскольку обеспечивает быстрый способ запуска различных программ. Чтобы изменить действия каждой подпрограммы, просто измените блоки во всех контейнерах «Мой блок».**

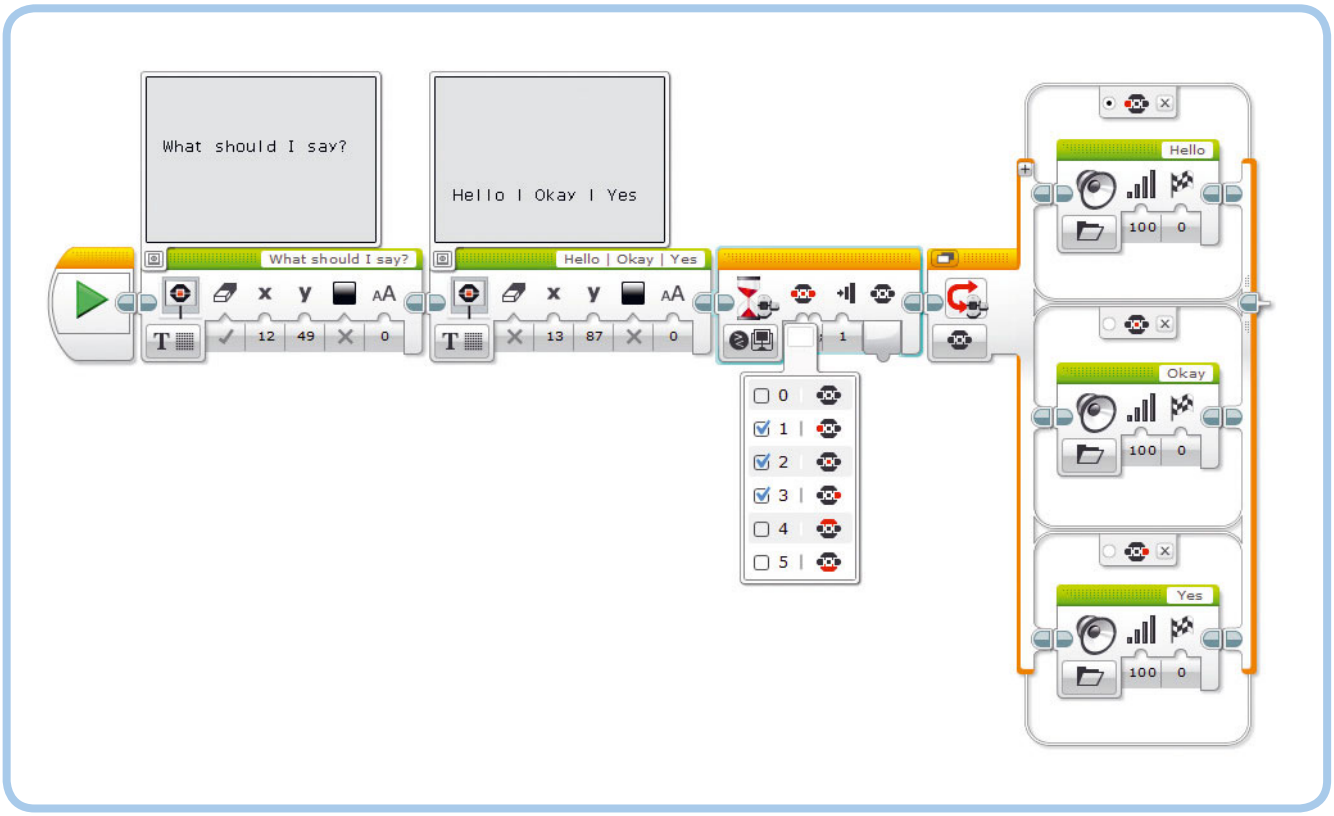


Рис. 9.1. Программа ButtonMenu. Блок ***Ожидание*** *настроен на режим* ***Кнопки управления модулем*** ▸ ***Сравнение*** ▸ ***Кнопки управления модулем****, а блок* ***Переключатель*** *— на режим* ***Кнопки управления модулем*** ▸ ***Измерение*** ▸ ***Кнопки управления модулем***

программы до тех пор, пока пользователь не нажмет одну из кнопок: «Влево», «Вправо» или центральную.

Затем блок **Переключатель** (Switch) (в режиме **Кнопки управления модулем** ▸ **Измерение** (Brick Buttons ▸ Measure)) определяет, какая кнопка нажата, и робот воспроизводит требуемый звук. После того как блок **Ожидание** (Wait) завершил работу, блок **Переключатель** (Switch) запускается настолько быстро, что уже во время нажатия кнопки он проверяет ее состояние. Даже если вы тотчас отпустили ее.

Использование датчика вращения мотора

Когда вы инструктируете робота двигаться вперед на три оборота мотора посредством блока **Рулевое управление** (Move Steering), он «знает», когда надо прекратить движение, поскольку датчик вращения в каждом моторе сообщает модулю EV3, сколько выполнено оборотов. Программа также может сообщить, насколько быстро в настоящее время крутится мотор, измеряя, как быстро изменяется его позиция.

Вы можете использовать блоки **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch) в режиме **Вращение мотора** (Motor Rotation) для измерения позиции мотора (режим **Градусы** (Degrees) или режим **Обороты** (Rotations)) и скорости вращения мотора (режим **Текущая мощность** (Current Power)).

Положение мотора

Положение мотора говорит вам о том, сколько он совершил оборотов с того момента, как вы запустили программу. Используйте приложение Port View в модуле EV3, перейдите к мотору B или C и вращайте моторы вручную, чтобы увидеть замеры датчика.

При первом запуске программы Port View (или собственной программы) значение датчика вращения будет равно нулю. Значение становится положительным при вращении мотора вперед и отрицательным, если вы включите его в обратном направлении после позиции 0, как показано на рис. 9.2. Например, если вращать мотор вперед на 90 градусов, а затем назад на один оборот (360 градусов), мотор передаст данные о позиции –270 градусов.

Вы можете использовать замеры позиции мотора, чтобы создать программу, которая воспроизводит звук, если вы

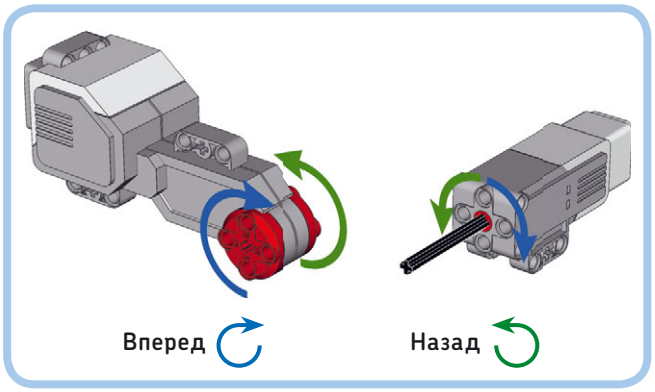


Рис. 9.2. Если вы программируете большой или средний мотор на движение вперед, он вращается в направлении, показанном синей стрелкой, и значение датчика вращения мотора становится положительным

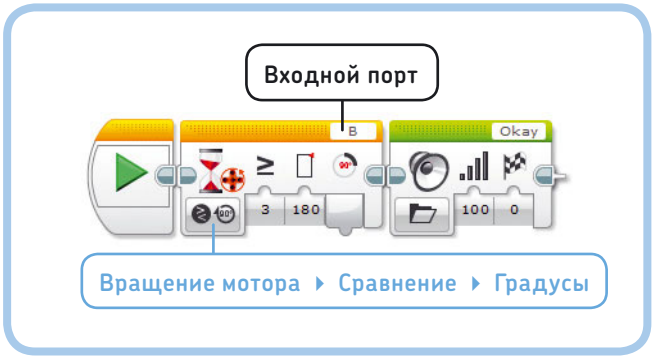


*Рис. 9.3. Программа HandRotate инструктирует робота произнести слово «О'кей», как только мотор B повернется в прямом направлении на 180 градусов. Обратите внимание, что блок **Ожидание** выполнит то же самое, если выбрать режим **Вращение мотора** ▸ **Сравнение** ▸ **Обороты** со значением параметра **Пороговое значение** равным 0,5*

вручную вращаете одно колесо на 180 градусов вперед, как показано на рис. 9.3. Блок **Ожидание** (Wait) в режиме **Вращение мотора** ▸ **Сравнение** ▸ **Градусы** (Motor Rotation ▸ Compare ▸ Degrees) ожидает, пока значение датчика вращения мотора не станет больше или равно (≥) 180 градусам. Поскольку эти датчики встроены в моторы EV3, они всегда подключены к выходным портам. В этой программе мотор используется на выходном порте B.

Сброс показаний датчика вращения мотора

Теперь предположим, что вы хотите повторить действия, описанные в программе HandRotate, с блоком **Цикл** (Loop) так, чтобы звук воспроизводился снова, когда вы провернете колесо еще на 180 градусов. Первый звук воспроизводится при вращении мотора вперед на 180 градусов. Но во время второго запуска цикла звук будет воспроизводиться сразу, потому что значение датчика уже больше 180 градусов, а это совсем не то, что нужно в результате.

Решение состоит в том, чтобы сбросить значение датчика вращения до нуля в начале цикла, используя для этого блок **Вращение мотора** (Motor Rotation) в режиме **Сброс** (Reset), как показано на рис. 9.4. Другие особенности этого и иных блоков датчиков описаны в главе 14.

Запустите программу и убедитесь, что вы слышите звук однократно каждый раз, когда колесо поворачивается на 180 градусов вперед.

Скорость вращения

Датчик вращения определяет, как быстро мотор вращается в диапазоне значений от –100 до 100%, основываясь на скорости, с которой изменяется позиция мотора. Значение положительно, когда мотор вращается вперед (синяя стрелка

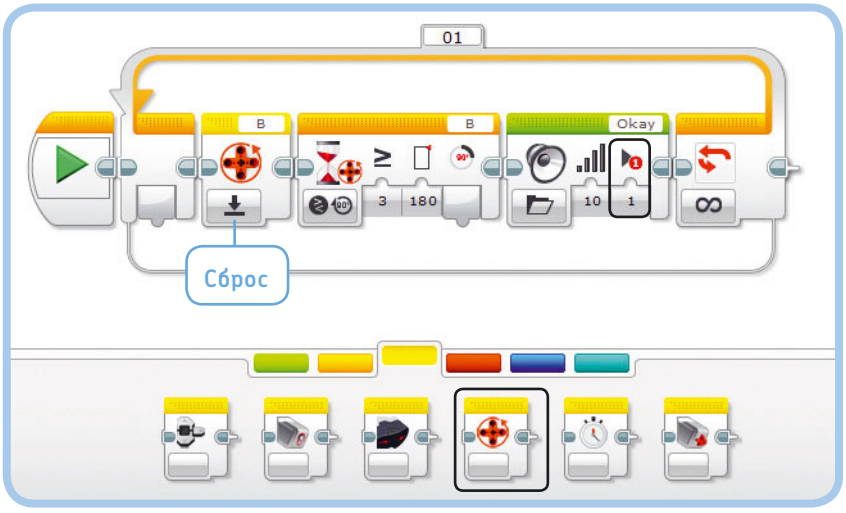


*Рис. 9.4. Программа HandRotateReset присваивает датчику значение 0 в начале каждого цикла с помощью блока **Вращение мотора**, работающего в режиме **Сброс**. Обратите внимание, что параметру **Тип воспроизведения** в блоке **Звук** присвоено значение **Воспроизвести один раз (1)**, поэтому программа не дожидается окончания воспроизведения звука*

ПРАКТИКУМ № 50: ВЕРНЕМСЯ К НАЧАЛУ!

Сложность: **Время:**

Можете ли вы создать программу, которая возвращает мотор в ту позицию, в какой он был при запуске программы? Робот должен предоставлять вам пять секунд, чтобы вы могли повернуть мотор в случайное положение вручную, а затем мотор должен вернуться к исходной точке. Используйте схему, показанную на рис. 9.5, в качестве руководства по разработке программы.

Рис. 9.5. Блок-схема для практикума № 50. Как робот определяет, что мотор был повернут в обратном направлении?

на рис. 9.2), и отрицательно, когда мотор вращается назад (зеленая стрелка). Нулевое значение датчика появится в том случае, когда мотор не вращается.

Для большого мотора значение **Текущая мощность** (Current Power), равное 50%, соответствует скорости вращения 85 оборотов в минуту (об/мин). Вы можете достичь этой скорости, вращая мотор вручную или с помощью блока **Рулевое управление** (Move Steering) со значением параметра **Мощность** (Power), равным 50%.

ПРИМЕЧАНИЕ **В режиме Текущая мощность (Current Power) измеряется скорость вращения; потребление тока или мощность не измеряется!**

Вычисление скорости вращения

Вы можете рассчитать скорость, измеренную в оборотах в минуту (об/мин), используя текущее значение параметра **Текущая мощность** (Current Power) следующим образом:

большой мотор
скорость вращения (об/мин) = значение датчика × 1,70

средний мотор
скорость вращения (об/мин) = значение датчика × 2,67

Например, если текущее значение параметра **Текущая мощность** (Current Power) большого мотора составляет 30%, мотор вращается со скоростью 30 × 1,70 = 51 оборот в минуту. Поскольку один оборот в минуту соответствует шести градусам в секунду, вы можете рассчитать скорость вращения в градусах в секунду (град/с) следующим образом:

скорость вращения (град/с) = скорость вращения (оборотов в минуту) × 6

В этом примере вы получите 51 × 6 = 306 градусов в секунду.

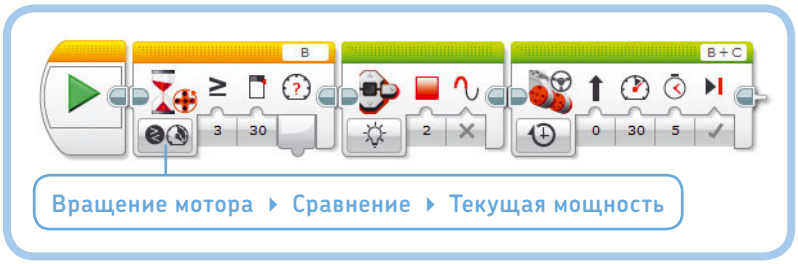


Рис. 9.6. Программа PushToStart

ПРАКТИКУМ № 51: ЦВЕТНАЯ СКОРОСТЬ!

Сложность: **Время:**

Создайте программу, которая непрерывно меняет цвет индикатора состояния модуля на зеленый, если мотор B вращается вперед, на оранжевый, если он вращается в обратном направлении, и на красный, если мотор остановлен. Поверните мотор B вручную, чтобы протестировать вашу программу.

СОВЕТ **Вам понадобится блок Цикл (Loop), два блока Переключатель (Switch) и три блока Индикатор состояния модуля (Brick Status Light).**

Измерение скорости вращения в программе

Для измерения скорости вращения в программе используется режим **Текущая мощность** (Current Power) датчика вращения, как показано на рис. 9.6. Программа PushToStart содержит блок **Ожидание** (Wait), настроенный на приостановку программы, пока мотор B не достигнет значения датчика 30% (51 об/мин). Затем запускается блок **Рулевое управление** (Move Steering). Запустите программу и вручную направляйте EXPLOR3R вперед, пока он не начнет двигаться самостоятельно.

Управление скоростью

До сих пор вы использовали по-разному настроенные зеленые блоки **Рулевое управление** (Move Steering), чтобы заставить робота двигаться. С помощью этих блоков моторы вращаются с постоянной, *регулируемой* скоростью. Когда вращение моторов замедляется из-за препятствия или наклонной поверхности, EV3 подает некоторую дополнительную мощность на мотор, чтобы сохранить нужную скорость вращения. Настройка параметра **Мощность** (Power) этих блоков фактически определяет *скорость*, которую моторы стараются поддерживать. То есть большой мотор, вращающийся на скорости 20% (34 оборота в минуту), при преодолении препятствия может потреблять больше энергии, чем при движении без затруднений на скорости 40% (68 оборотов в минуту).

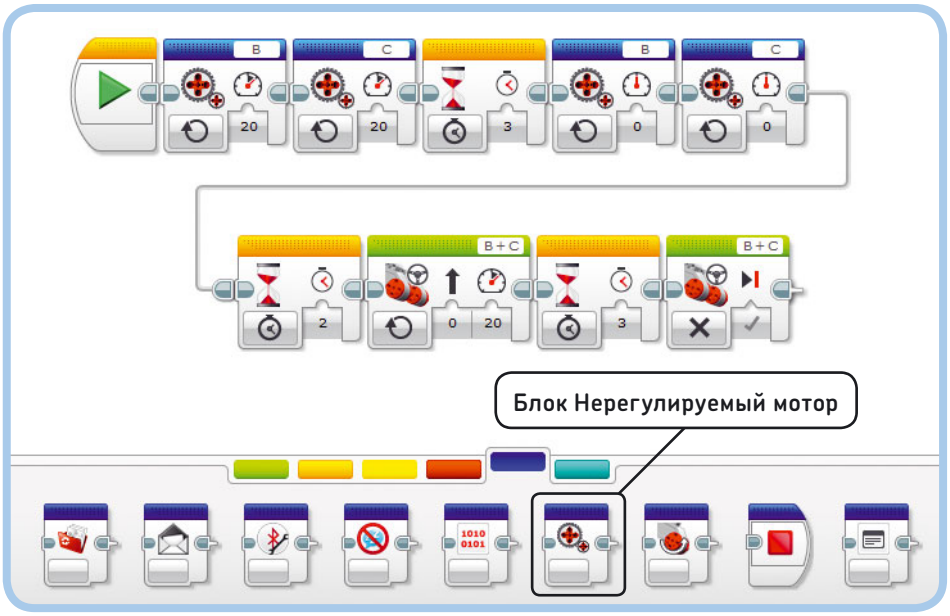


Рис. 9.7. Программа SteepSlope. Обратите внимание, что я использовал соединитель, чтобы разделить программу на две части для большей наглядности, но вам это делать необязательно

Если вы не хотите, чтобы EV3 использовал эту дополнительную мощность для поддержания постоянной скорости вращения, вы можете использовать *нерегулируемую скорость*.

Регулирование скорости в действии

Чтобы увидеть разницу между регулируемой скоростью и нерегулируемой скоростью, давайте создадим программу, с помощью которой робот въезжает вверх по наклонной поверхности, например по столу, одна сторона которого приподнята. Сначала в течение трех секунд робот едет с нерегулируемой скоростью, а затем в течение трех секунд он движется с регулируемой скоростью.

Вам понадобятся два блока **Нерегулируемый мотор** (Unregulated Motor) (по одному для каждого мотора), которые доступны на вкладке **Дополнения** (Advanced) палитры программирования. Присвойте параметру **Мощность** (Power) значение 20%, как показано на рис. 9.7. После ожидания в течение трех секунд остановите моторы, присвоив параметру **Мощность** (Power) значение 0%. Для движения с регулируемой скоростью 20% (34 оборота в минуту) в программе используется блок **Рулевое управление** (Move Steering), который вы видели раньше.

Поместите робота EXPLOR3R на наклонную поверхность и запустите программу SteepSlope. Вы увидите, что в течение первых трех секунд робот едет вверх медленно, а в течение следующих трех секунд он движется быстрее.

На первом этапе движения EV3 включает оба мотора и в течение следующих трех секунд не взаимодействует с ними. Робот движется медленно, потому что для подъема в горку требуется больше энергии, чем для езды по ровной местности. На втором этапе движения датчик вращения сообщает роботу, что он движется медленно, и это вынуждает модуль EV3 передавать некоторое количество дополнительной мощности, чтобы скорость движения не снижалась.

Остановка заглохшего мотора

Если во время движения замедлить одно из колес, когда запущен блок **Рулевое управление** (Move Steering), вы почувствуете, что робот пытается с усилием восстановить скорость вращения. Для колесных транспортных средств это хорошо. Но нежелательно для механизмов, которые не совершают полных оборотов, например для клешневого устройства.

Чтобы избежать этой проблемы, вы можете применить блок **Нерегулируемый мотор** (Unregulated Motor) и датчик вращения, определяющий,

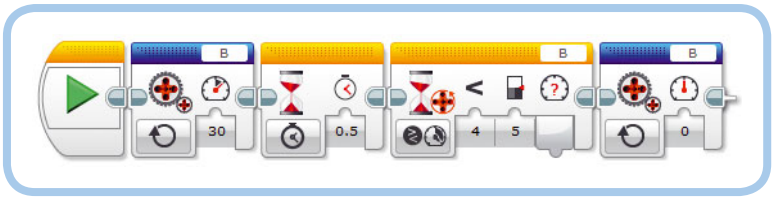


*Рис. 9.8. Программа WaitForStall крутит мотор B, пока он не заглохнет. Обратите внимание, что первый блок **Ожидание** требуется, чтобы предоставить мотору некоторое время для набора скорости, в противном случае при измерении скорости значение частоты вращения будет равно нулю, в результате чего программа немедленно завершит работу*

когда мотор заглох (блокируется), как показано в программе WaitForStall (рис. 9.8). Программа включает мотор B на 30% мощности и ожидает, когда скорость упадет ниже 5%, что свидетельствует о том, что мотор заглох. Если вы запустите эту программу на EXPLOR3R, ваш робот будет ездить по кругу, пока вы не остановите его, преградив ему путь.

Дальнейшее изучение

Теперь, когда вы научились работать со всеми датчиками из набора MINDSTORMS EV3, вы можете создавать роботов, которые взаимодействуют с окружающей средой. EXPLOR3R — это, разумеется, только один пример. По мере изучения этой книги вы научитесь собирать разных роботов с датчиками, и каждый из них будет применять датчики в своих целях.

До сих пор вы учились использовать компоненты, которые необходимы для создания работающих роботов: EV3, моторы, датчики и программное обеспечение. В следующих главах мы рассмотрим каждый компонент более подробно, так что у вас появится возможность создавать все более и более сложных (и интересных!) роботов. В следующей главе вы узнаете, как использовать детали конструктора Technic в наборе MINDSTORMS EV3 для создания собственных роботов.

Следующие практические задания помогут вам изучить больше возможностей датчиков, с которыми вы познакомились в этой главе.

ПРАКТИКУМ № 52: ДИСТАНЦИОННОЕ УПРАВЛЕНИЕ С ПОМОЩЬЮ МОДУЛЯ EV3!

Сложность: **Время:**

Снимите модуль EV3 с вашего робота (оставьте кабели подключенными) и создайте программу, которая позволяет перемещать робота с помощью кнопок на корпусе модуля EV3. Задайте роботу движение вперед при нажатии кнопки «Вверх», влево — при нажатии кнопки «Влево» и так далее.

СОВЕТ **Используйте блок Переключатель (Switch) в режиме Кнопки управления модулем ▸ Измерение ▸ Кнопки управления модулем (Brick Buttons ▸ Measure ▸ Brick Buttons).**

ПРАКТИКУМ № 53: СНИЖЕНИЕ СКОРОСТИ ПРИ ОБНАРУЖЕНИИ ПРЕПЯТСТВИЯ!

Сложность: **Время:**

Можете ли вы запрограммировать робота на движение по комнате и избегание препятствий без использования датчиков касания, цвета или инфракрасного датчика? Задайте роботу движение вперед, используя блоки **Нерегулируемый мотор** (Unregulated Motor), пока он не обнаружит препятствие. Затем робот должен отъехать, развернуться и продолжить движение в новом направлении.

СОВЕТ **Скорость вращения мотора снижается, когда робот натыкается на препятствие.**

СДЕЛАЙ САМ № 11: АВТОМАТИЧЕСКИЙ ДОМ!

Сборка: **Программирование:**

Вы когда-нибудь строили дома из обычных деталей LEGO? Теперь, когда вы знаете, как использовать моторы и датчики и как создавать рабочие программы, как насчет того, чтобы попытаться построить роботизированный дом с помощью EV3?

СОВЕТ **Используйте мотор для автоматического открывания двери, когда кто-то нажимает на дверной звонок (датчик касания), и установите сигнал о вторжении, который звучит, когда инфракрасный датчик обнаруживает кого-то. Используйте другой мотор, чтобы закрывать и открывать ставни, основываясь на уровне освещенности, измеренной с помощью датчика цвета.**

ЧАСТЬ III

Способы конструирования роботов

10

Конструирование с балками, осями, фиксаторами и моторами

Вы уже узнали довольно много о программировании роботов, но и учиться собирать роботов так же важно. Опыт сборки приходит с практикой, и в этой части книги вы получите основную информацию о конструировании роботов из деталей вашего набора MINDSTORMS EV3. В этой главе вы узнаете, как создавать прочные конструкции с помощью балок, рам, штифтов, фиксаторов и осей, которые вы можете увидеть на рис. 10.1. Вы поймете, как размерная сетка LEGO поможет вам собирать собственные конструкции из балок и фиксаторов. В главе 11 вы узнаете, как работают зубчатые колеса (шестеренки).

Каждая конструкция (но не все сразу), приведенная в этой главе, может быть построена с использованием деталей из вашего набора MINDSTORMS EV3. Изучите примеры, чтобы получить представление о том, насколько прочна каждая конструкция и какие из них будут полезны для ваших роботов.

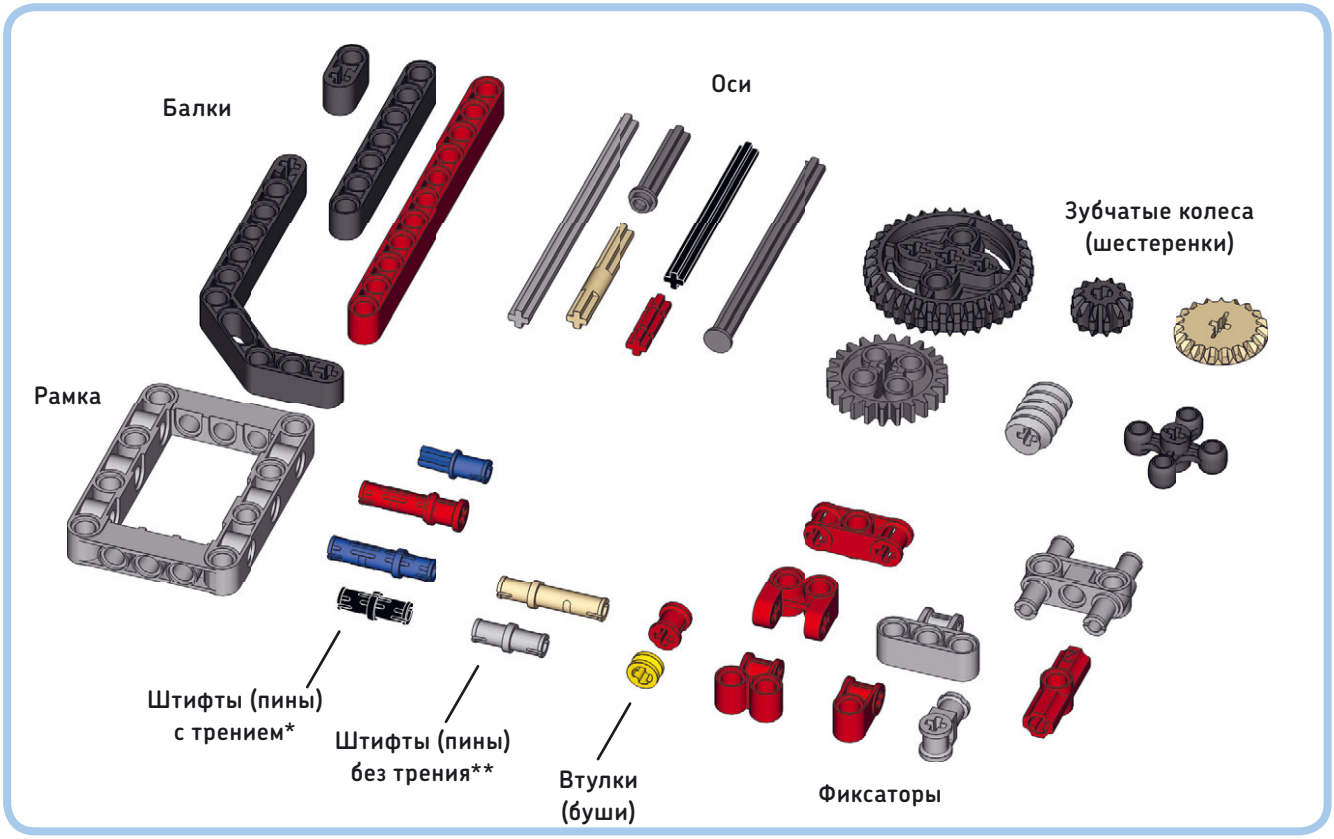


Рис. 10.1. Набор MINDSTORMS EV3 содержит множество различных балок, рам, осей, зубчатых колес, фиксаторов и штифтов. Полный список деталей вы найдете на внутренней стороне обложки

* Также называются фрикционными.

** Также называются бесфрикционными.

ПРИМЕЧАНИЕ **Если хотите, вы можете разобрать робота EXPLOR3R; для изучения остальной части книги он вам не понадобится.**

Использование балок и рамок

Вы уже многое знаете о применении моторов, датчиков и модуля EV3. *Балки* используются для создания конструкций, которые скрепляют все эти детали вместе. Длина балок и других деталей измеряется в *единицах* LEGO, иногда называемых *модулями*, как показано на рис. 10.2. Длина самой короткой прямой балки в наборе MINDSTORMS EV3 составляет две LEGO единицы, или 2M; самая длинная балка — 15M.

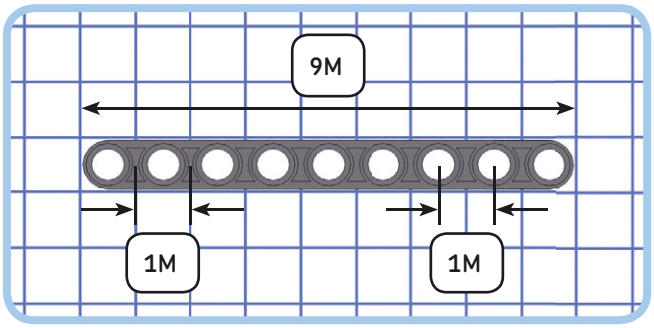


Рис. 10.2. Длина балок и других деталей измеряется в единицах LEGO. Расстояние между центральными точками двух соседних отверстий составляет одну LEGO единицу, или 1M. Следовательно, расстояние между центральными точками крайнего левого отверстия и крайнего правого отверстия этой балки длиной 9M составляет 8M

Удлинение балок

Балки можно увеличить в длину и ширину путем объединения нескольких балок с помощью штифтов с трением (с жестким креплением). Для неподвижного соединения вам потребуется не менее двух штифтов, и по крайней мере три отверстия должны перекрывать друг друга, как показано на рис. 10.3.

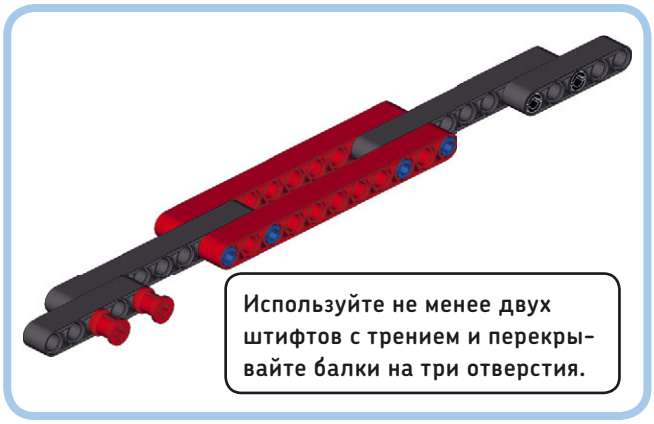


Рис. 10.3. Вы можете нарастить балки с помощью штифтов с трением (красные, синие и черные штифты в наборе)

Использование рамок

Набор MINDSTORMS EV3 содержит два типа рамок, они изображены на рис. 10.4. Рамки позволяют легко создавать большие конструкции со множеством отверстий для крепления других деталей, например балок и моторов. Рамки могут быть использованы для подсоединения балок под прямым углом.

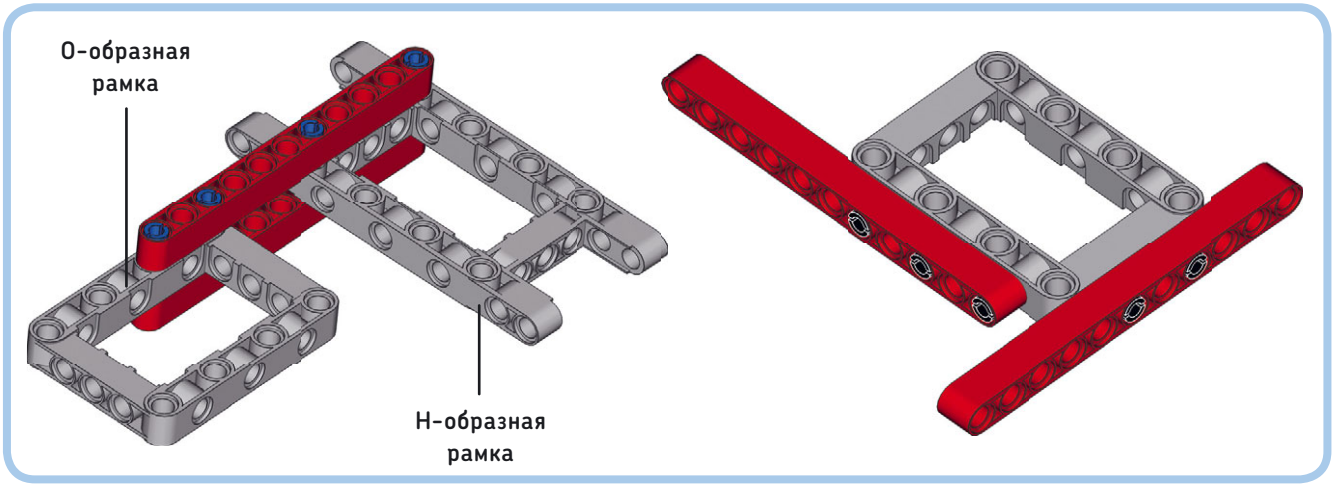


Рис. 10.4. Рамки используются для создания больших и прочных конструкций с большим количеством отверстий для прикрепления других деталей (слева) и для соединения балок под прямым углом (справа)

Использование балок для укрепления конструкций

Балки могут быть использованы не только для создания новых конструкций, но и для укрепления или усиления уже существующих. Например, рассмотрим верхнюю конструкцию на рис. 10.5, состоящую из двух рамок. Ее легко разрушить, потянув рамки в разные стороны, потому что они соединены только в двух местах.

Нижняя конструкция усилена двумя балками 3М. Такое соединение очень трудно разрушить и разъединить две рамки. Еще больше усилить такую конструкцию возможно, заменив балки 3М на балки 11М, которые охватывают всю ширину рамки.

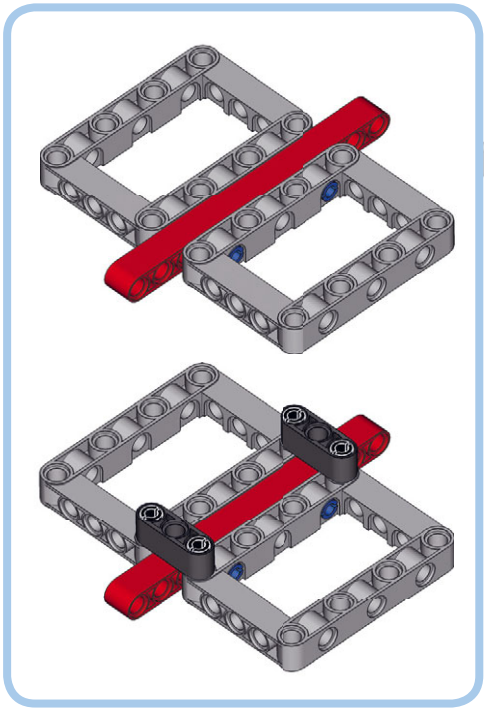


Рис. 10.5. Балки можно использовать для укрепления конструкций. Сравните прочность обеих конструкций, пытаясь разъединить две рамки. Вы поймете, что конструкция внизу гораздо прочнее

Использование угловых балок

Набор MINDSTORMS EV3 содержит множество *угловых балок* различных типов и с разными углами. В том числе четыре типа угловых балок с углом 90 градусов (или с *прямым углом*), как показано на рис. 10.6. Балки такого типа используются для соединения прямых балок под прямым углом.

Кроме балок, согнутых под углом 90 градусов, ваш набор содержит два типа балок с углом 53,13 градуса,

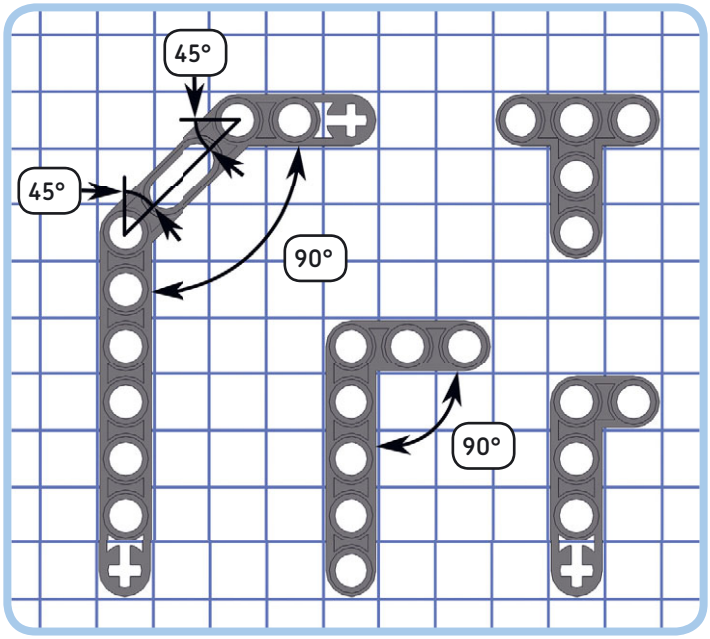


Рис. 10.6. Четыре типа балок с прямым углом (90 градусов)

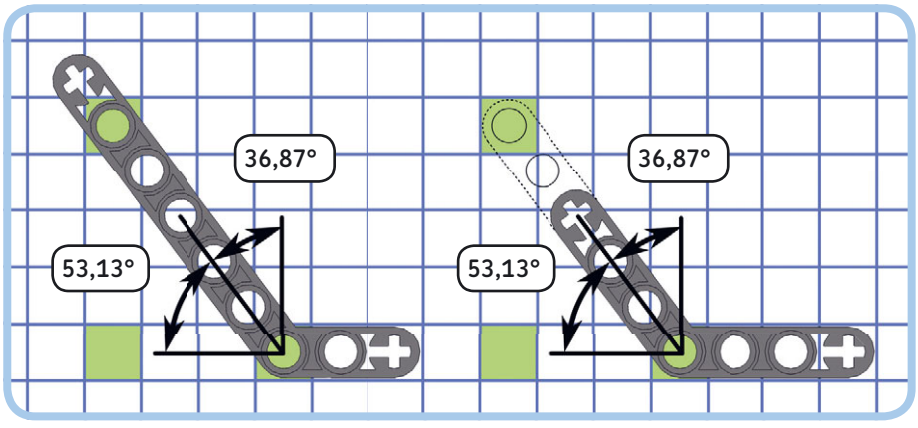


Рис. 10.7. Две балки с углом 53,13 градуса. Поскольку углы обеих балок одинаковы, вы можете нарастить более короткую балку (справа), присоединив прямую балку, чтобы ее можно было использовать для достижения таких же целей, как и длинную (слева)

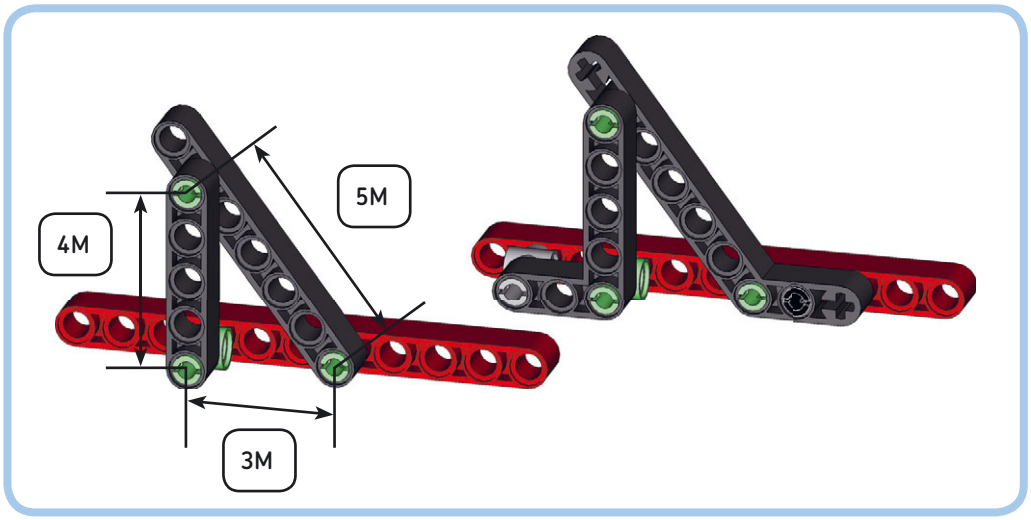


Рис. 10.8. Создание двух одинаковых прямоугольных треугольников с использованием прямых балок (слева) и угловых балок (справа). Штифты на самом деле не зеленого цвета, они окрашены для книги, чтобы показать точки соединения, отмеченные зеленым цветом на рис. 10.7. Обратите внимание, что длина стороны треугольника измеряется между центральными точками отверстий на каждом углу, а не путем подсчета количества отверстий на балке

которые изображены на рис. 10.7. Такой угол может показаться странным, но на самом деле его удобно использовать, потому что он может образовывать угол какого-нибудь общего треугольника. В частности, вы можете использовать этот угол, чтобы создать треугольник Пифагора (прямоугольный), стороны которого составляют 3M, 4M и 5M, как показано на рис. 10.8.

ПРАКТИКУМ № 54: БОЛЬШИЕ ТРЕУГОЛЬНИКИ!

Сложность: **Время:**

Существует еще один полезный прямоугольный треугольник, который можно собрать из деталей набора MINDSTORMS EV3. Его размер в два раза превышает размер треугольника, показанного на рис. 10.8. Можете ли вы собрать такой треугольник? А треугольник в три раза больше?

Размерная сетка LEGO

Слева на рис. 10.9 изображена сетка с квадратами размером 1M. Эта размерная сетка LEGO поможет вам в сборке прочных роботов. Если присоединять новые детали к балкам так, чтобы их отверстия совпадали с этой сеткой, позже будет легче добавлять новые детали (а).

Если вы отклонитесь от сетки, размещая детали под углом, соединение большого количества деталей становится затруднительным, так как детали LEGO имеют ограниченное количество фиксированных размеров (б). Такой тип соединения не стоит использовать для сборки несущей конструкции вашего робота, хотя его вполне можно применить для декоративных деталей, таких как хвост робота-животного.

Сборка без использования сетки чревата тем, что балки будут растягиваться или сгибаться, что может привести к повреждению деталей LEGO (в). Поэтому вы всегда должны избегать такого варианта сборки. Не используйте соединений, которые вы не можете сделать без небольшого сгибания или растягивания деталей. Если вы не уверены, нужно ли будет сгибать детали в угловой конструкции, лучше придерживаться сетки, используя для соединения прямые углы.

Вы можете положить на сетку балки с углом 53,13 градуса, чтобы определить соединительные отверстия, отмеченные зеленым на рис. 10.7. Таким образом, вы можете использовать угловые балки для создания прямоугольных конструкций, как показано на рис. 10.10.

ПРИМЕЧАНИЕ Свою копию сетки LEGO вы можете распечатать, открыв файл *Размерная_сетка.pdf*, и использовать распечатку для создания ваших собственных конструкций. Не забудьте выбрать *фактический размер*, или 100%, в настройках режима печати. Балка 15M на печатной схеме должна быть такого же размера, как реальная балка 15M. Если размер получился не такой, измените настройки масштаба и распечатайте сетку еще раз.

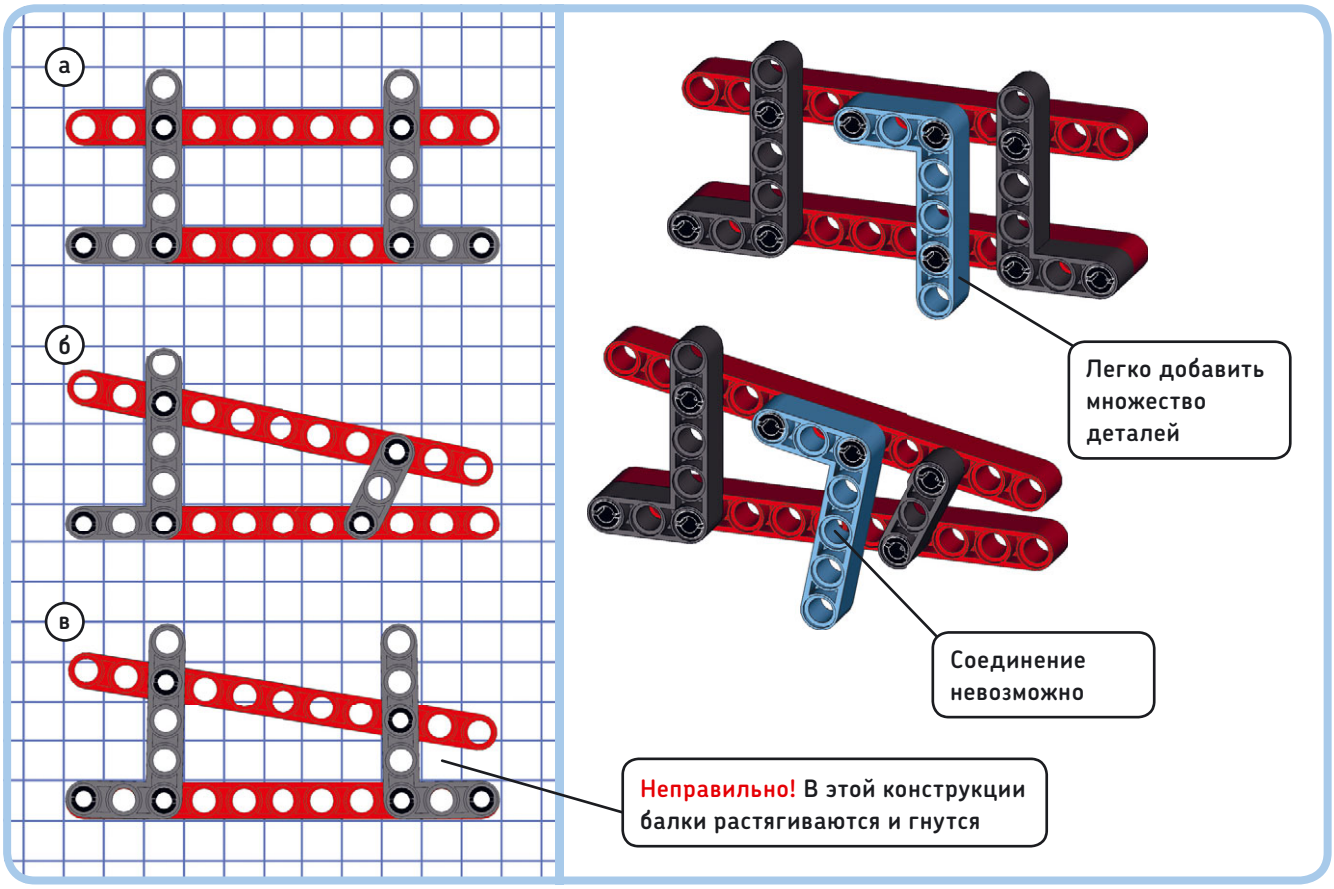


Рис. 10.9. Рекомендуется выстраивать конструкции на сетке (а). При необходимости можно собирать конструкции, отклоняясь от сетки (б) до тех пор, пока вы не найдете подходящее соединение балок (в)

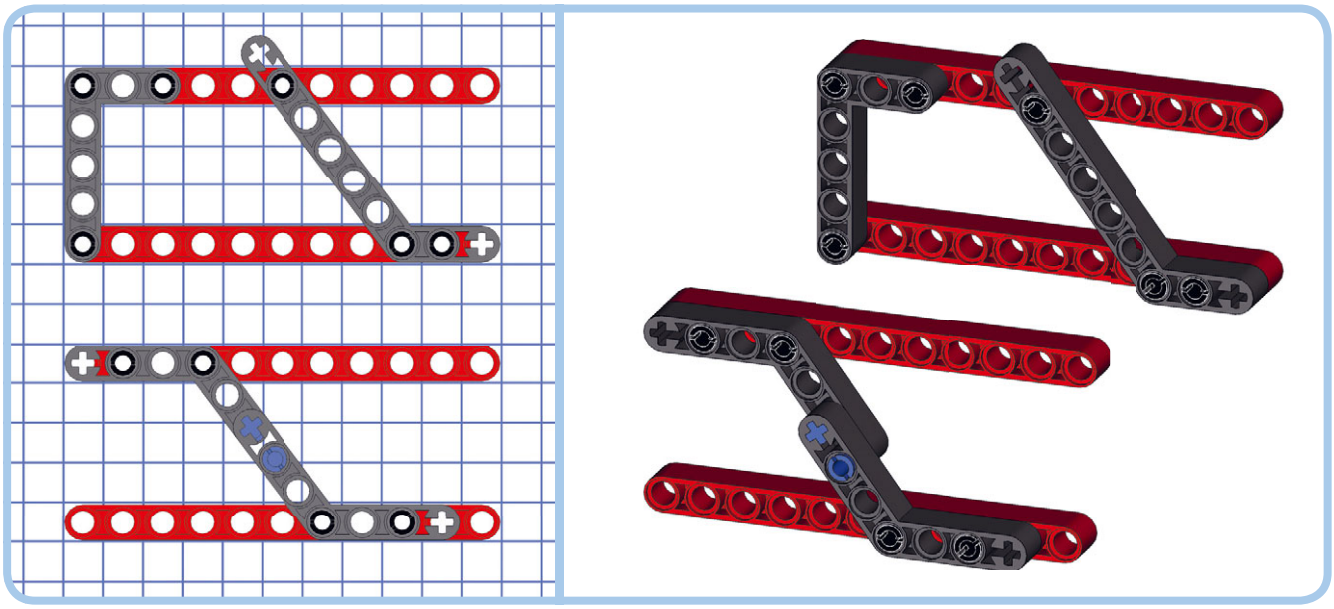


Рис. 10.10. Два способа использовать угловые балки 53,13 градусов, с раскладкой на сетке

ПРАКТИКУМ № 55: УГЛОВЫЕ КОМБИНАЦИИ!

Сложность: **Время:**

Можете ли вы найти комбинацию из двух угловых балок (53,13 градуса) для соединения двух параллельных балок 11М, показанных на рис. 10.11?

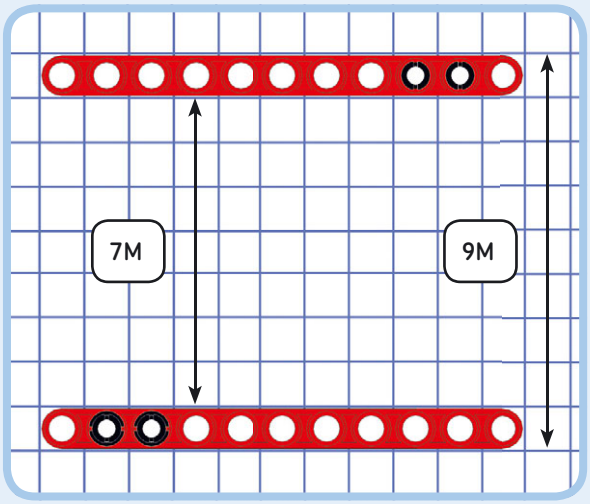


Рис. 10.11. Две прямые балки из практикума № 55

Использование осей и крестовых отверстий

Ось представляет собой стержень, на который можно установить вращающиеся детали, например колёса или шестеренки (зубчатые колеса). В круглых отверстиях оси свободно вращаются, но вы можете использовать их для неподвижных соединений, установив в крестовые отверстия, как показано на рис. 10.12. Самая короткая ось в наборе MINDSTORMS EV3 имеет длину 2М; самая длинная — 9М.

Чтобы ось не выпадала из круглого отверстия, добавьте втулку, чтобы закрепить ось, или используйте оси с ограничителем, как показано на рис. 10.13.

Осевой штифт с трением, изображенный на рис. 10.14, представляет собой полезный фиксатор. Один конец штифта с трением можно установить в круглое отверстие, где он будет вращаться с некоторым сопротивлением. Другой

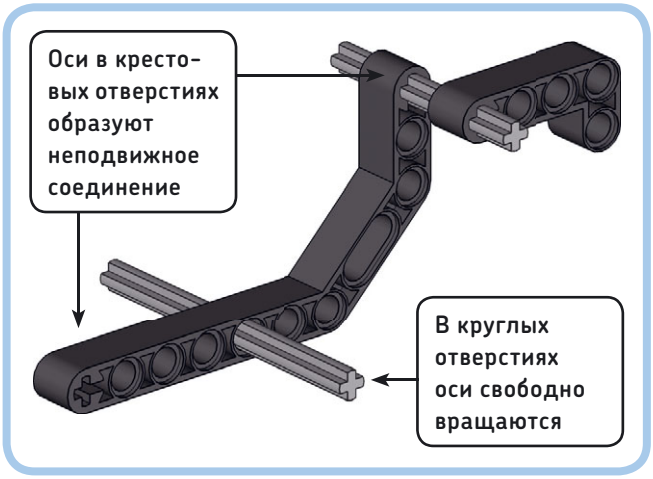


Рис. 10.12. Оси свободно вращаются в круглых отверстиях, а в крестовых образуют жесткое соединение

Рис. 10.13. Вы можете закрепить ось в отверстии балки с помощью втулки. Если ось имеет ограничитель на одном конце, вам понадобится только одна втулка

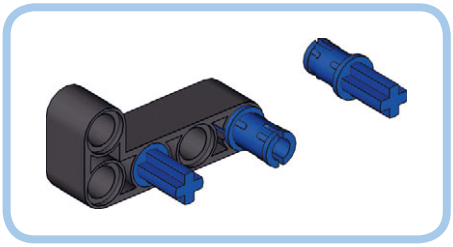


Рис. 10.14. Синие осевые штифты с трением соединяют круглое отверстие с крестообразным

конец представляет собой ось, которая может быть установлена в крестовое отверстие, где она будет оставаться неподвижной. Некоторые наборы LEGO содержат подобные желтовато-коричневые или серые осевые штифты без трения; такой штифт плавно вращается в круглом отверстии.

Использование фиксаторов

Для установки балок, осей, моторов и датчиков под разными углами вы можете использовать *фиксаторы*. Каждый тип фиксатора в наборе MINDSTORMS EV3 может быть использован разными способами. В этом разделе приведено несколько полезных примеров, с которых можно начать.

Удлинение осей

Некоторые фиксаторы можно использовать для соединения двух осей, как показано на рис. 10.15. Это позволяет совместить оси под углом или сделать их длиннее.

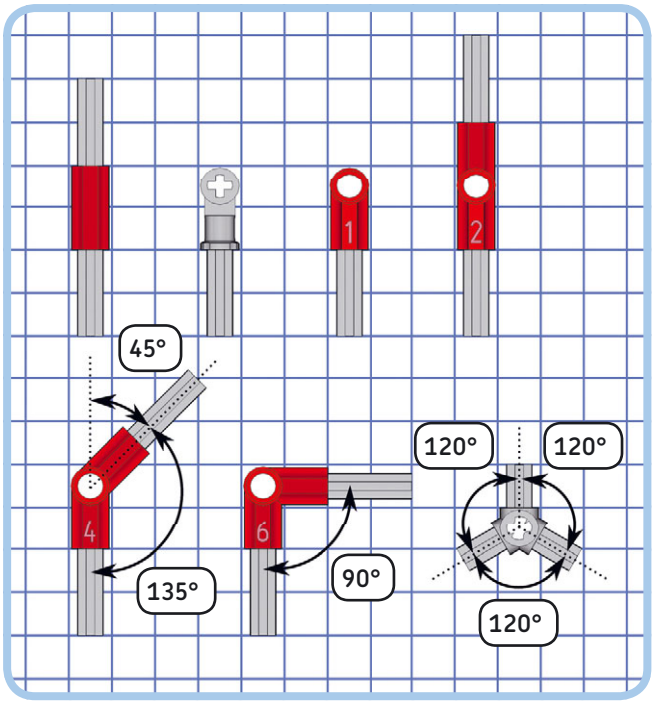


Рис. 10.15. Удлинение осей с помощью фиксаторов

Соединение параллельных балок

Чтобы соединить две параллельные балки, вы можете использовать как рамки или балки, так и комбинацию фиксаторов, как показано на рис. 10.16 и 10.17. Такое соединение будет уместно, когда ограничено пространство или если использование балок или рамок нецелесообразно. Используйте размерную сетку в качестве эталона для своих конструкций. Например, если вам необходимо соединить две параллельные балки длиной 3М, вы можете использовать вариант *f*, показанный на рис. 10.16.

В левой части рис. 10.16–10.18 показано, как комбинировать разные фиксаторы, а справа изображены примеры, иллюстрирующие эти комбинации.

ПРИМЕЧАНИЕ Эти примеры иллюстрируют, как соединить две балки с помощью фиксаторов. Те же принципы вы можете применить к другим деталям с отверстиями. Например, вы можете использовать эти комбинации фиксаторов, чтобы прикрепить датчик к отверстиям мотора или модуля EV3.

Соединение балок под прямым углом

Многие фиксаторы имеют круглые или крестообразные отверстия, расположенные перпендикулярно друг к другу. Это позволяет подсоединить балки под прямым углом или расположить их параллельно, но так, чтобы их отверстия находились перпендикулярно друг к другу, как показано на рис. 10.18.

Укрепление параллельных балок

Вы уже видели ранее, что с помощью балок можно укрепить конструкцию (см. рис. 10.5). Но в зависимости от ориентации отверстий балки может возникнуть необходимость добавить фиксаторы, прежде чем вы сможете подсоединить балки для укрепления, как показано на рис. 10.19.

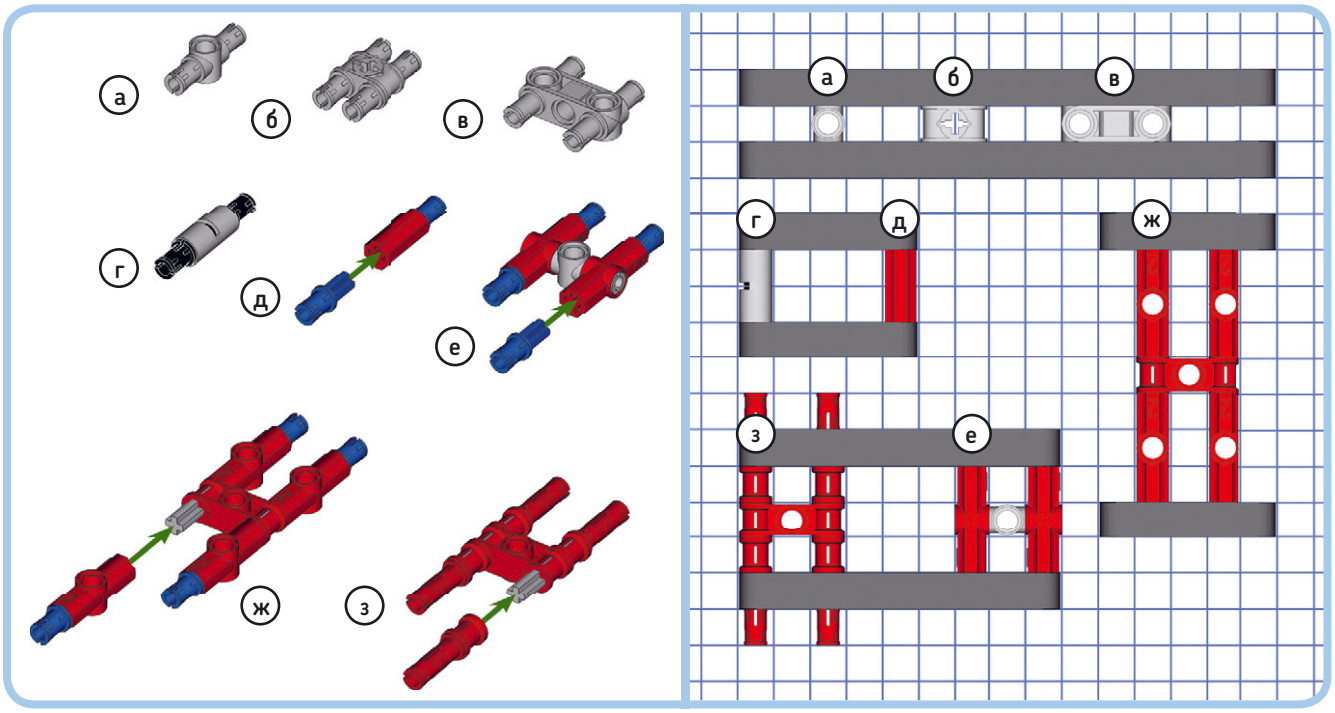


Рис. 10.16. Соединение параллельных балок в случае, если их отверстия обращены друг к другу

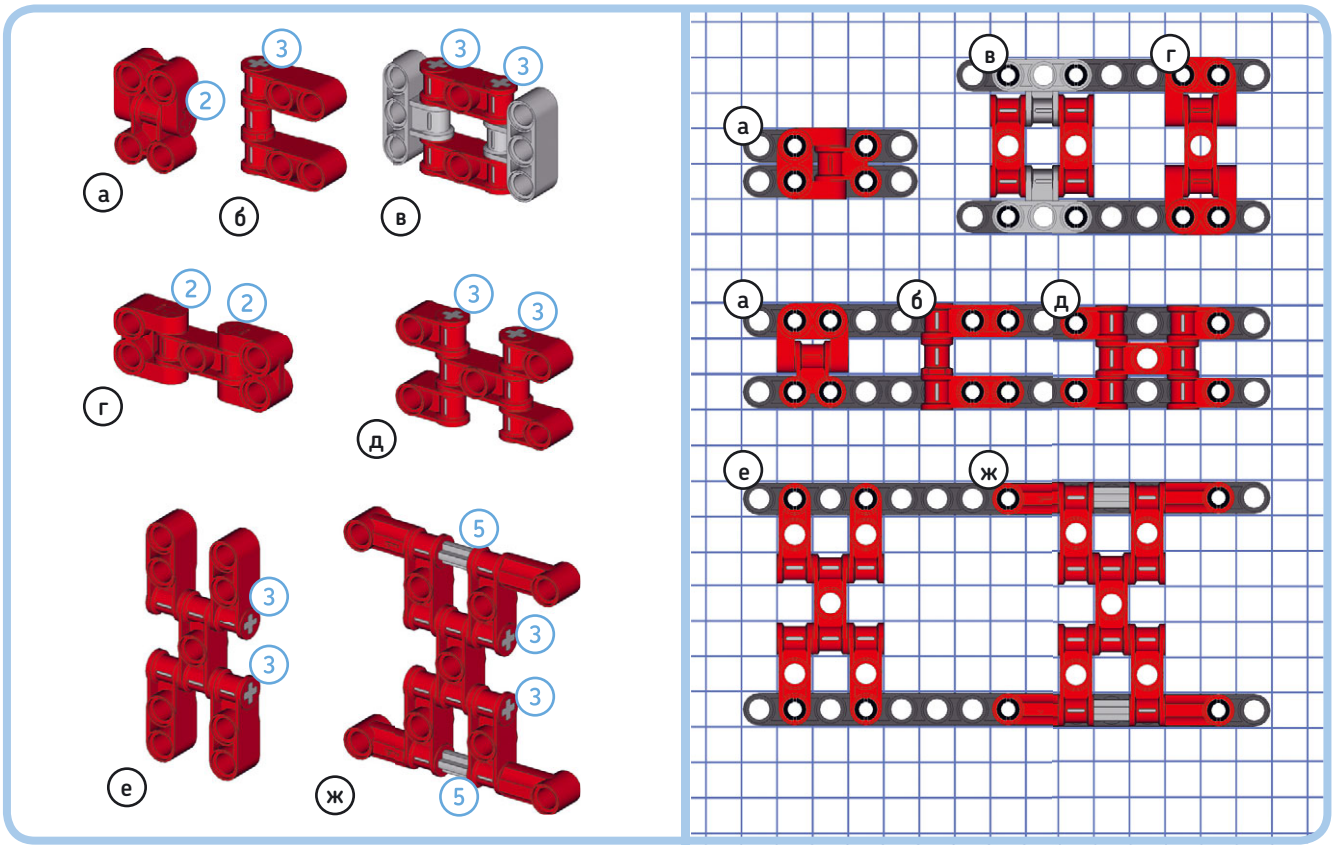


Рис. 10.17. Соединение параллельных балок, если их плоские стороны обращены друг к другу. Числа в кружочках обозначают длину осей, используемых в конструкциях

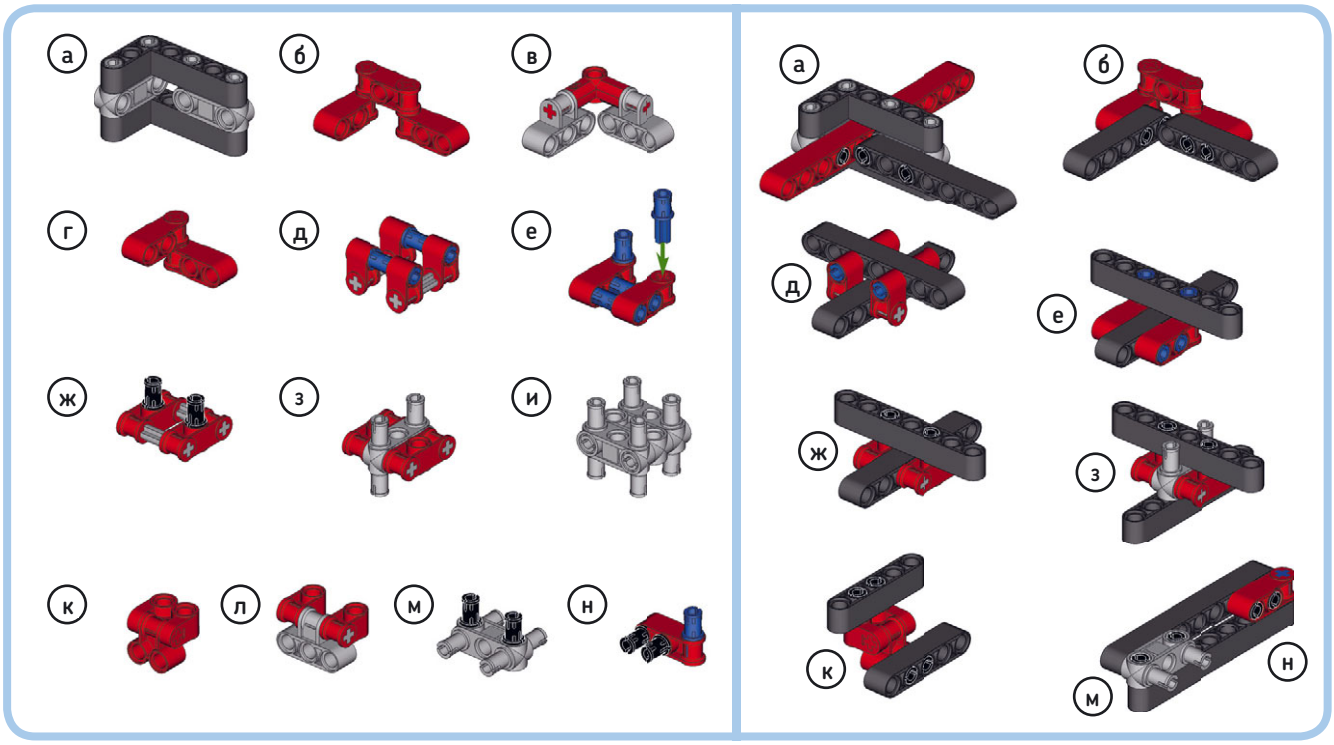


Рис. 10.18. Использование фиксаторов для подсоединения балок под прямым углом. Серые оси на рисунке имеют длину 3М

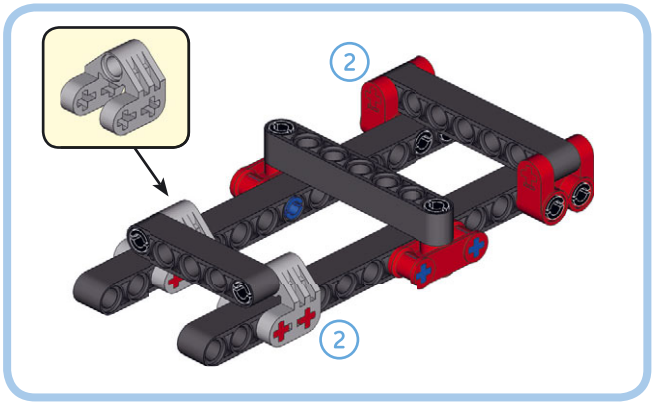


Рис. 10.19. Вы можете применить фиксаторы, чтобы создать точки крепления для балок, которые усилят конструкцию. Обратите внимание, что эти конструкции не являются жесткими, когда используются по отдельности. Этот способ укрепления конструкций нужно использовать так же, как изображенные на рис. 10.16

ПРАКТИКУМ № 56: КОНСТРУКТИВНЫЕ ФИКСАТОРЫ!

Сложность: **Время:**

Можете ли вы скомбинировать фиксаторы так, чтобы создать прочное крепление балок, показанных на рис. 10.20? Дополните примерами, взятыми с рис. 10.16–10.19, или создайте собственные комбинации.

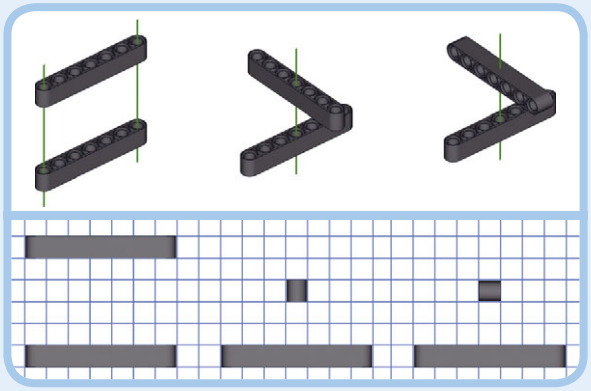


Рис. 10.20. Зеленые линии показывают, как две балки расположены по отношению друг к другу

Детали с половинчатым размером

Некоторые фиксаторы заканчиваются на половине клетки, как показано на рис. 10.21. При правильном использовании они предоставляют больше возможностей без ущерба для конструкции. Например, вы можете создавать конструкции, имеющие размер 7,5М, а не 7М или 8М.

Обратите внимание, что такие конструкции укреплять балками не так просто, поскольку расстояние между двумя отверстиями балки всегда должно быть целым числом. То есть применительно к данной конструкции невозможно использовать другую балку, чтобы соединить верхнюю балку с той, что в середине.

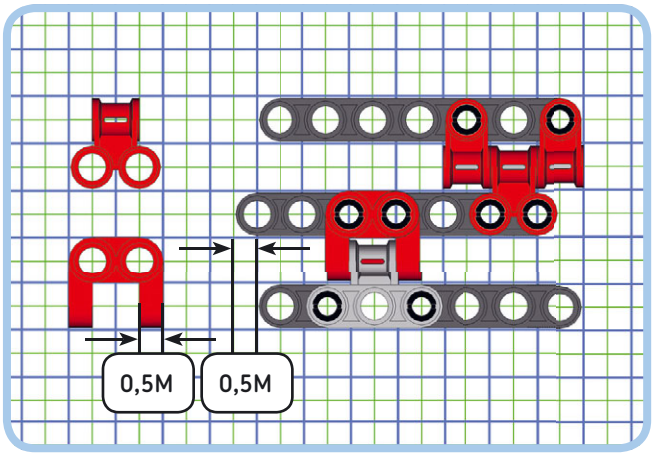


Рис. 10.21. Некоторые фиксаторы приводят к смещению деталей на 0,5М относительно квадрата размерной сетки. Балка в середине смещена на 0,5М относительно двух других балок

ПРАКТИКУМ № 57: ПОЛОВИНЧАТЫЕ БАЛКИ!

Сложность: **Время:**

Можете ли вы соединить две балки, чтобы сделать балку длиной 18,5М, с помощью фиксаторов?

СОВЕТ **Используйте фиксаторы, которые показаны на рис. 10.21.**

Использование тонких деталей

Большинство деталей в наборе MINDSTORMS EV3 имеют точную ширину, равную 1М, но есть и тонкие детали шириной 0,5М, как показано на рис. 10.22. Тонкие детали могут быть использованы, когда для установки обычных не хватает места. В наборе MINDSTORMS EV3 мало тонких деталей, поэтому возьмите их из набора LEGO Technic.

Одной из деталей, обозначенных здесь, является кулачок мотора. Он пригодится для создания вращающегося механизма, который нажимает кнопку датчика касания один раз за один оборот. Об этом вы подробнее узнаете в главе 13.

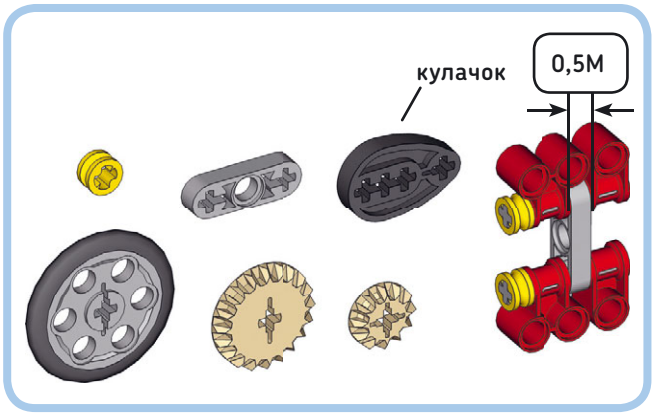


Рис. 10.22. Тонкие детали в наборе MINDSTORMS EV3

Создание гибких конструкций

Для создания шарниров и гибких механизмов вместо жестких конструкций нужно использовать штифты без трения (серые и желтовато-коричневые штифты в наборе MINDSTORMS EV3). Штифты без трения в механизме на рис. 10.23 позволяют легко повернуть зубчатое колесо.

Набор MINDSTORMS EV3 содержит тяги двух типов (6М и 9М), которые обычно используются для создания рулевых механизмов в автомобилях LEGO Technic. Эти тяги можно применить в некоторых конструкциях в качестве замены для балок. Хотя тяги и создают менее прочное соединение, чем балки, они могут быть использованы для соединения деталей, которые находятся в разных плоскостях. То есть, если вы расширите движущуюся балку предыдущего механизма, вы не сможете использовать балку для соединения с зубчатым колесом. Вместо нее можно использовать тягу, как показано на рис. 10.24.

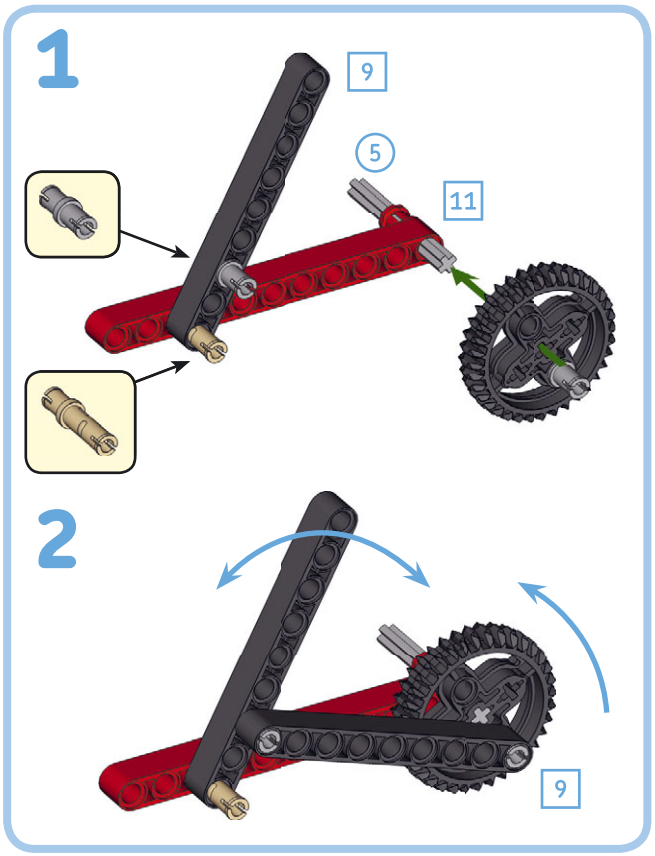


Рис. 10.23. В этой динамической конструкции используются штифты без трения, поэтому ее легко превратить в механизм, который заставляет балку двигаться вперед и назад. Для сравнения используйте штифты с трением, и вы увидите, что повернуть зубчатое колесо стало намного сложнее

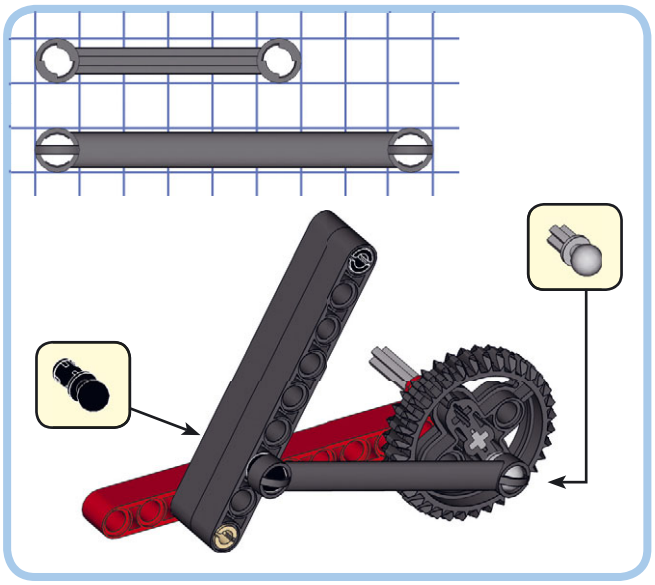


Рис. 10.24. В этой модифицированной версии динамической конструкции вместо балки используется тяга. Вы присоединяете тягу через круглое или крестообразное отверстие, используя штифт с трением и с шаром, как показано на рисунке

Конструкции с моторами и датчиками

Теперь мы рассмотрим, как, учитывая размер и форму моторов, использовать их в качестве центрального компонента вашего робота. Моторы довольно велики, и у них есть множество точек крепления для штифтов и осей. Поэтому чаще всего при создании такого механизма, как роботизированная клешня или гусеничный движитель, сборку лучше начинать как раз с мотора.

Такой способ позволяет тестировать каждый механизм, или *модуль*, по отдельности. После того как вы убедились, что все модули прекрасно работают по отдельности, вы можете собрать из них целого робота.

Конструкции с большим мотором

Геометрия большого мотора (рис. 10.25) позволяет легко соединять два мотора с использованием рамки и штифтов с трением. Это дает преимущество в создании транспортного робота, как показано на рис. 10.26. Все, что вам нужно сделать, это добавить колеса или гусеницы и модуль EV3.

Подключение колес и гусениц

Большой мотор достаточно мощный и быстрый, чтобы вращать колеса непосредственно, как показано на рис. 10.27. Вы также можете напрямую управлять гусеницами танков, присоединив их с помощью двух балок 13M. Рис. 10.28 демонстрирует базовую схему сборки. Другой пример можно увидеть в главе 18, там, где собирается SNATCH3R. Как подключить зубчатые колеса к большому мотору, вы узнаете в следующей главе.

Подключение балок к валу мотора

Подсоединить колеса и шестеренки к красному валу большого мотора вы можете с помощью крестового отверстия. Кроме того, чтобы балки и другие детали вращались, их можно вставить в круглые отверстия на валу. Например, чтобы создать возвратно-поступательный механизм, вы можете сделать так, чтобы мотор непрерывно вращал балку 3M, как показано на рис. 10.29.

Конструкции со средним мотором

Средний мотор (рис. 10.30) меньше по размеру, чем большой, что позволяет использовать его в небольших конструкциях, таких как рулевой механизм гоночного автомобиля.

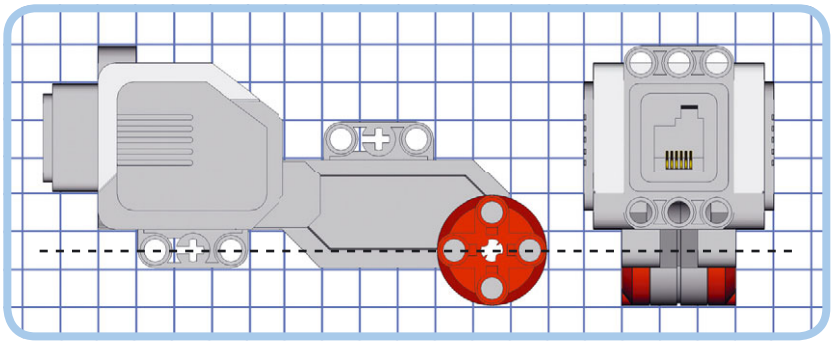


Рис. 10.25. Внешний вид большого мотора

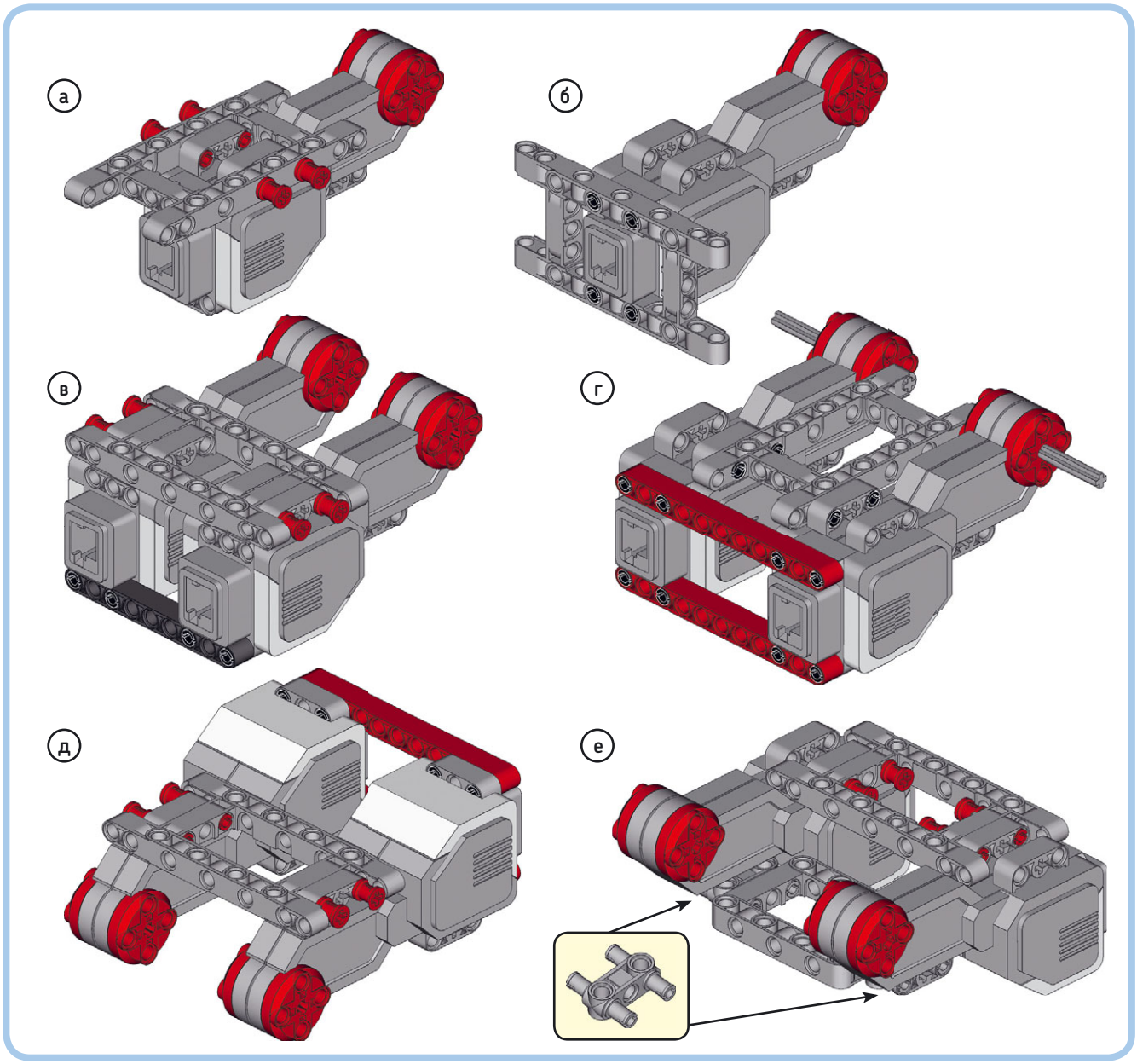


Рис. 10.26. Упростить крепление мотора к роботу можно, установив на него рамки. Например, вы можете создать основу для транспортного робота, объединив два больших мотора с помощью рамок, балок и штифтов с трением различными способами

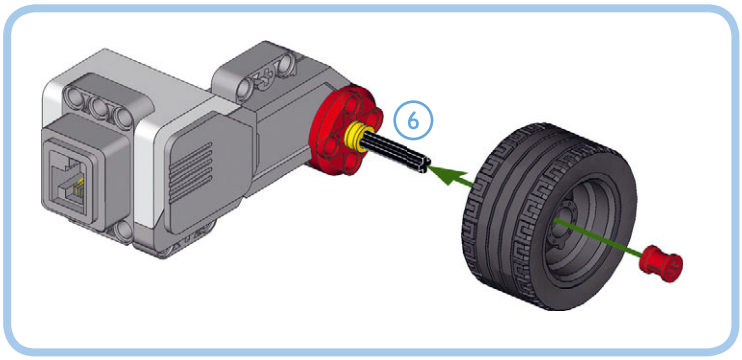


Рис. 10.27. Колеса можно подключить непосредственно к большому мотору с помощью оси 6M. Половинчатая втулка создает небольшое пространство между мотором и колесом, а обычная втулка нужна, чтобы колеса не соскальзывали с оси

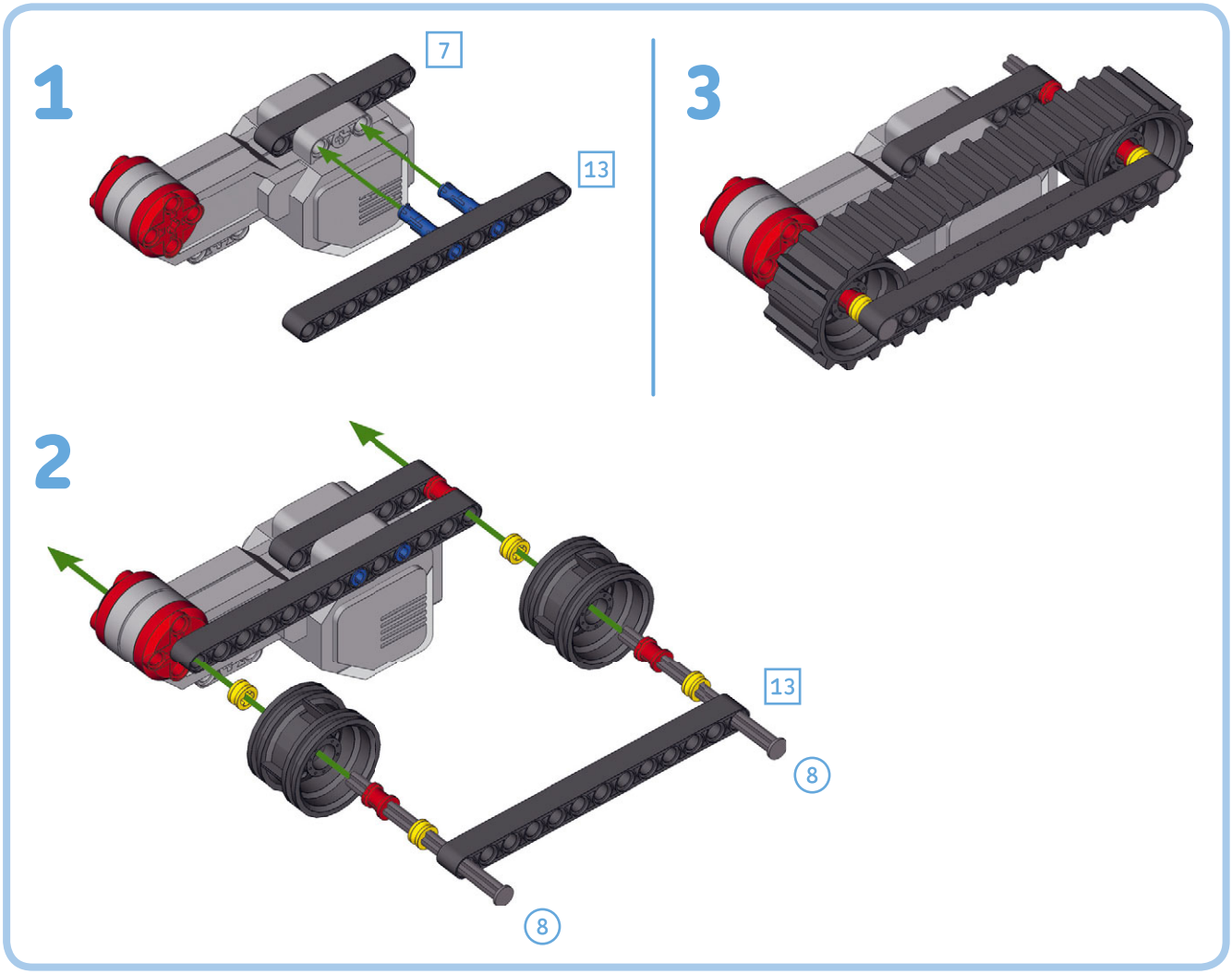


Рис. 10.28. Вы можете подключить диски гусеничного движителя к большому мотору с помощью двух балок длиной 13M и двух осей 8M с ограничителями

На передней стороне мотора есть круглые соединительные отверстия. А добавив рамку, можно создать больше точек подключения в задней части, как показано на рис. 10.31. Ну а если добавить рамку, как показано на рис. 10.32, вы сможете установить его между двумя большими моторами транспортного робота, например чтобы привести в движение механизм вилочного погрузчика.

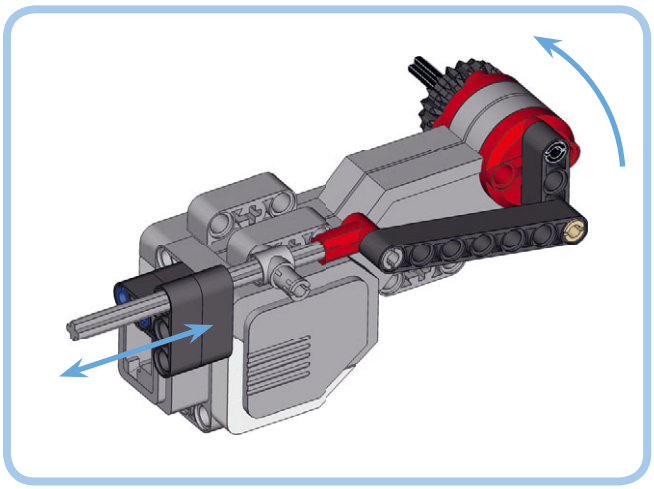


Рис. 10.29. Вы можете подключить шестеренки и колеса к валу мотора с помощью оси или фиксирующих балок, используя круглые отверстия. Этот механизм двигает серую ось 9М вперед и назад за один оборот вала мотора

Конструкции с датчиками

Каждый датчик в наборе MINDSTORMS EV3 имеет точки крепления для одной оси и двух штифтов, как показано на рис. 10.33. Кроме того, инфракрасный датчик имеет два круглых отверстия в задней части. Чтобы создать жесткое соединение, вам нужно использовать либо два штифта и балку, либо ось и балку с крестообразным отверстием.

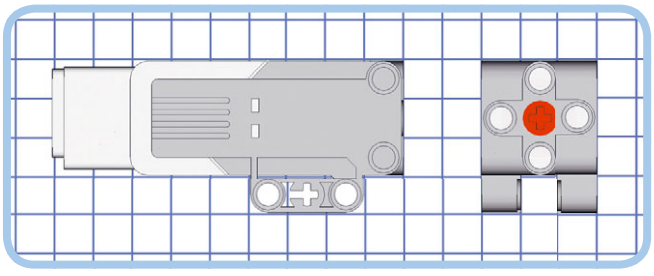


Рис. 10.30. Внешний вид среднего мотора

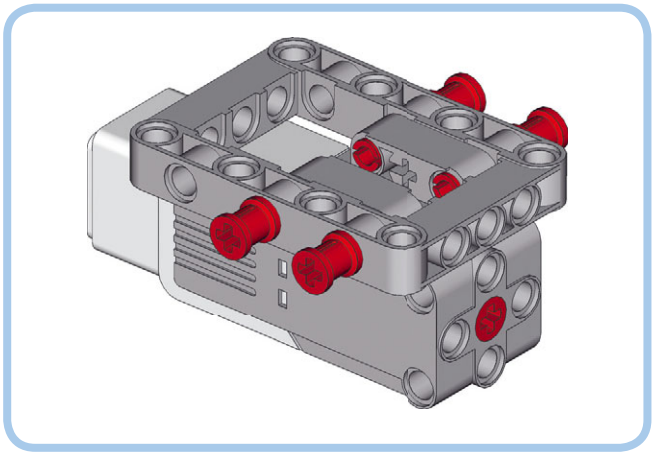


Рис. 10.31. Добавление точек подключения к среднему мотору с помощью рамки

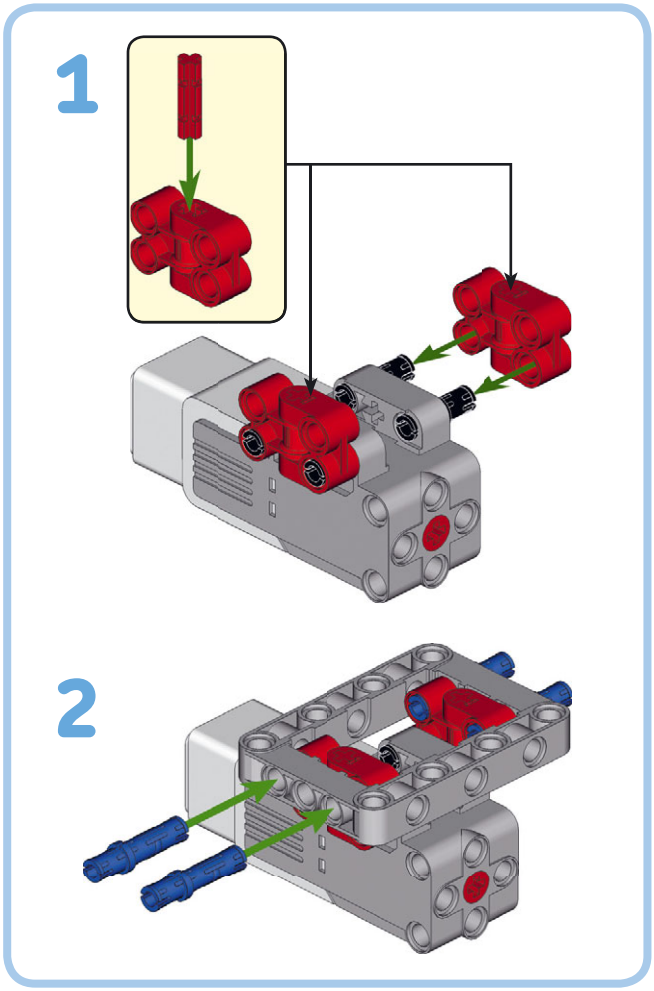


Рис. 10.32. Вы можете прикрепить средний мотор к раме так, что его легко будет установить между двумя большими моторами транспортного робота. Например, вы можете удалить рамку в примере е на рис. 10.26 и установить вместо нее эту конструкцию

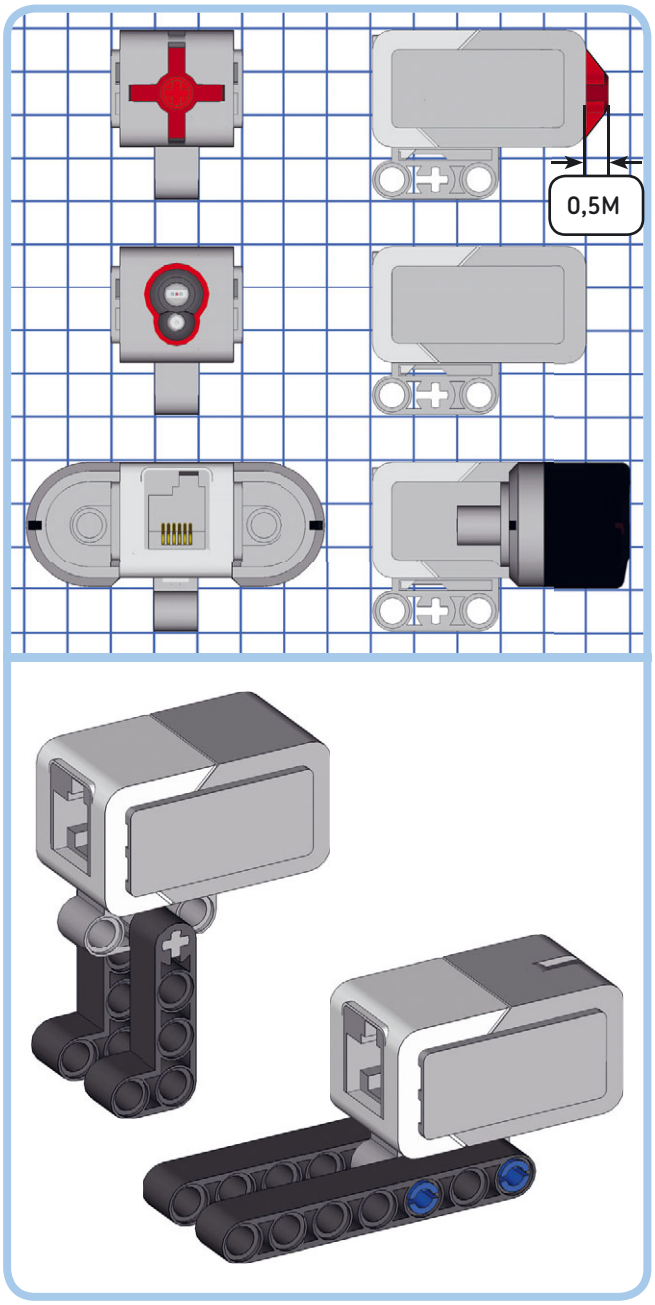


Рис. 10.33. Внешний вид датчиков в наборе MINDSTORMS EV3 (вверху) и присоединение их к роботу (внизу)

Прочие детали

В дополнение к деталям, обсуждаемым в этой части книги, набор содержит другие разнообразные элементы, такие как мечи и маски монстров, которые можно использовать для украшения ваших роботов. И наконец, ваш набор включает в себя детали магазина и механизма для стрельбы шариками. Вы найдете инструкции по строительству такой пушки, изучив раздел «Сборка и программирование» робота EV3RSTORM в программе EV3 (центральный робот на рис. 3.2 в главе 3).

Дальнейшее изучение

В этой главе вы научились использовать балки, рамки, штифты, оси, фиксаторы и моторы, с помощью которых создаются компоненты ваших роботов. Вы узнали, как размерная сетка LEGO может помочь в сборке прочных конструкций. Рецепта для создания идеального робота не существует. А лучшим способом получить опыт в разработке собственных роботов была и остается практика. Вы можете конструировать проекты, представленные в этой книге, а затем изменять их, создавая своих роботов, выполняя по возможности все задания «Сделай сам».

В следующей главе мы рассмотрим, как работают зубчатые колеса и как вы можете использовать их с моторами EV3.

СДЕЛАЙ САМ № 12: ПЕРЕСЯДЕМ НА ГУСЕНИЦЫ!

Сборка: **Программирование:**

Робот EXPLOR3R перемещается на двух основных колесах и опорном колесе сзади. Можете ли вы создать версию EXPLOR3R на гусеницах? Проверьте свое творение, управляя им с помощью инфракрасного маяка.

СОВЕТ **Используйте пример гусеничного движителя, показанный на рис. 10.28. Почему нужно убрать опорное колесо EXPLOR3R?**

СДЕЛАЙ САМ № 13: РОБОТ-УБОРЩИК!

Сборка:  **Программирование:** 

Можете ли вы создать робота, который ездит по столу, не падая с него? Сконструируйте робота, который будет сметать любые детали LEGO, оказавшиеся перед ним, с помощью среднего мотора, освобождая таким образом пространство во время движения.

Установите на переднюю часть робота инфракрасный датчик, закрепив его на расстоянии около 25 см от передних колес, и направьте его вниз, чтобы он «видел» поверхность стола. Как сделать так, чтобы робот определял, что он приближается к краю стола?

СОВЕТ **Если вы уже построили робота, описанного в практикуме «Сделай сам № 12» на с. 141, вы можете использовать его в качестве отправной точки для этого задания.**

СДЕЛАЙ САМ № 14: РОБОТ, ОТКРЫВАЮЩИЙ ШТОРЫ!

Сборка:  **Программирование:** 

Можете ли вы создать робота, который автоматически открывает шторы на восходе солнца и закрывает на закате? Используйте датчик цвета для измерения яркости окружающего света, а также добавьте возможность отмены автоматического поведения с помощью удаленного инфракрасного маяка.

СОВЕТ **Если вы используете только одно пороговое значение, робот может открывать и закрывать шторы по несколько раз, если уровень освещенности варьируется около порогового. Чтобы избежать этой проблемы, вы можете использовать два разных пороговых значения. Что робот должен делать, когда уровень освещенности находится между этими значениями?**

11

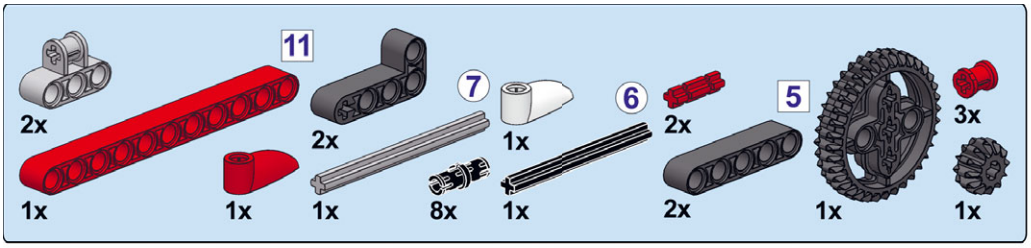
Конструирование с зубчатыми колесами

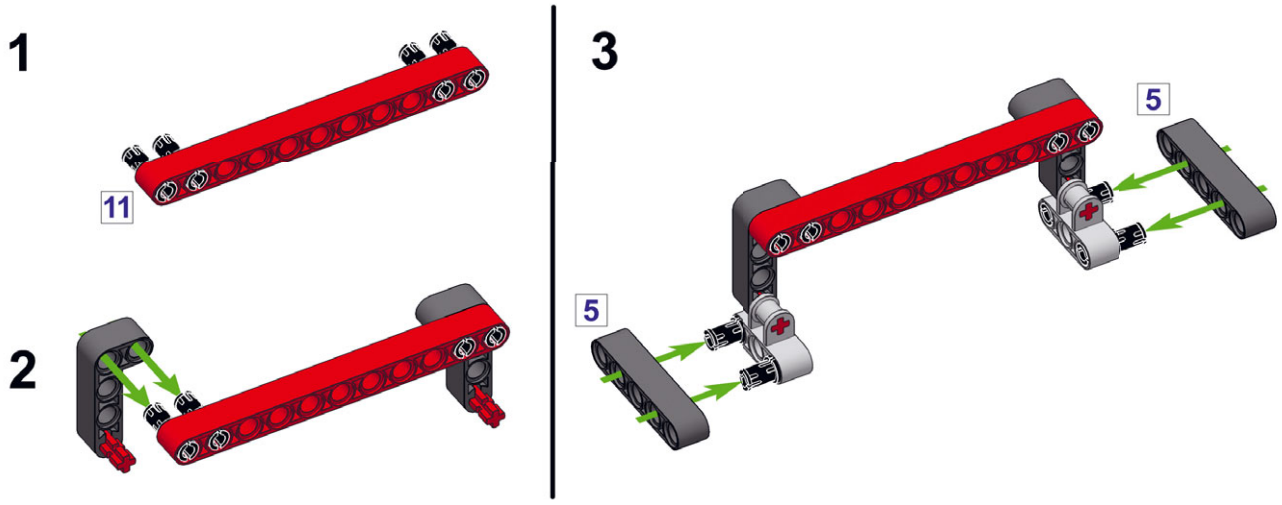
Вы можете использовать зубчатые колеса для передачи крутящего момента от одной вращающейся оси к другой. Например, можно передать крутящий момент вращающегося мотора на колеса робота, чтобы сделать его движущимся. Кроме того, зубчатые колеса можно использовать для изменения скорости на выходе и крутящего момента вращающейся оси.

Группа зубчатых колес, которая применяется для передачи движения, называется *зубчатой передачей*. В этой главе вы начнете изучать, как работают зубчатые колеса, проводя эксперименты с простой зубчатой передачей. После этого вы увидите, как соотношение зубчатых колес влияет на производительность зубчатой передачи. Наконец, вы исследуете все зубчатые колеса в наборе MINDSTORMS EV3 и узнаете, как их можно эффективно использовать при создании разных роботов.

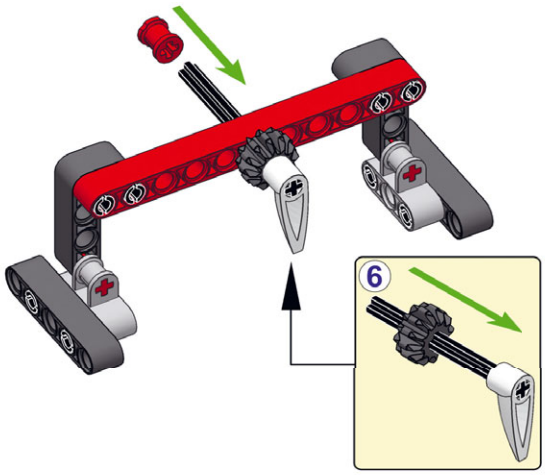
Простые зубчатые передачи

Прежде всего, создайте механизм с двумя зубчатыми колесами, следуя приведенной ниже схеме сборки. Этот механизм мы будем использовать, чтобы изучить принципы работы зубчатой передачи. Обязательно попробуйте другие примеры, когда



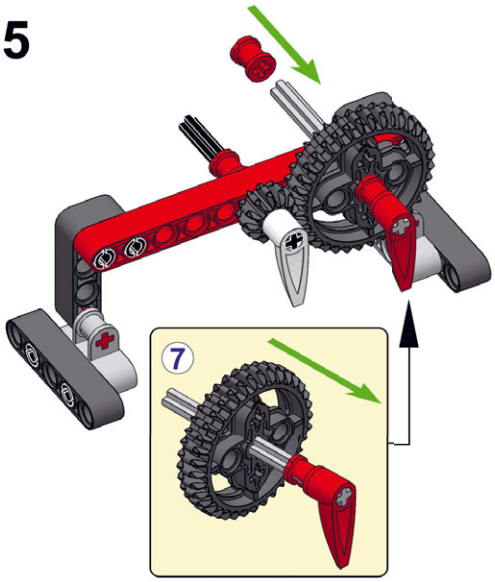


4



Не прикрепляйте втулки слишком плотно — зубчатые колеса должны поворачиваться плавно

5



Убедитесь, что оба указателя направлены вниз, как показано на рисунке, прежде чем присоединить втулки

доберетесь до них в книге, — это самый лучший способ точно уяснить, как используются зубчатые колеса.

Прежде чем детально рассматривать зубчатые колеса, давайте повернем их вручную и посмотрим, что происходит.

- Поворот одного зубчатого колеса влечет за собой вращение другого. Независимо от того, какое зубчатое колесо вы крутите, второе всегда вращается в *противоположном* направлении.
- На каждый полный оборот красной стрелки приходится *три* оборота белой. Чтобы убедиться в этом, в начале эксперимента установите обе стрелки так, чтобы они указывали вниз, а затем посчитайте, сколько сделает оборотов белая стрелка за то время, пока красная сделает один поворот.
- Зубчатое колесо меньшего размера всегда вращается быстрее, чем большое. В данном случае маленькое зубчатое колесо поворачивается в три раза быстрее, чем большое.
- Если вы руками заблокируете серую ось, прикрепленную к большому зубчатому колесу, вы обнаружите, что все еще можете вращать черную ось, прикрепленную к малому зубчатому колесу, но с некоторым усилием. Однако, если вы заблокируете черную ось, повернуть серую будет очень сложно.

Для каждого из этих наблюдений вы найдете объяснение по мере прочтения книги.

ПРАКТИКУМ № 58: НАБЛЮДЕНИЕ ЗА ЗУБЧАТЫМИ КОЛЕСАМИ!

Сложность: **Время:**

Если вращать зубчатые колеса медленно, вы увидите, что обе стрелки указывают в одном направлении несколько раз. За то время, пока красная стрелка сделает один полный оборот, сколько раз обе стрелки укажут в одинаковом направлении? Можете ли вы объяснить, почему так происходит?

Подробно о зубчатых колесах

Если вы посмотрите на зубчатые колеса в нашем примере более внимательно, то увидите, что малое колесо имеет 12 *зубцов* (мы обозначим его как *12T*), а большое — 36 (*36T*). В *точке контакта* зубцы обоих колес зацепляются, как показано на рис. 11.1. Если вы вручную немного повернете маленькое колесо, его зубцы зацепят зубцы большого колеса, и оно начнет вращаться в противоположном направлении. Зубчатое колесо, которое мы вращали вручную, будем называть *ведущим*. Зубчатое колесо, на которое передается крутящий момент от ведущего колеса, будем именовать *ведомым*.

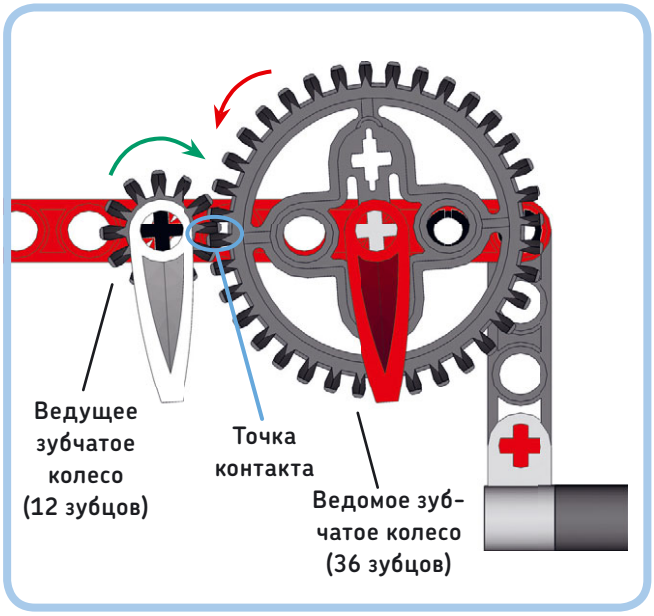


Рис. 11.1. Если вы внимательно посмотрите на устройство зубчатой передачи, то увидите, что зубцы колес зацеплены друг за друга. Ведущее зубчатое колесо передает вращение на ведомое зубчатое колесо, оказывая давление на зубцы ведомого колеса в точке контакта в направлении, указанном стрелкой

За каждым зубцом малого колеса, проходящим через точку контакта, следует один зубец большого колеса. За то время, пока малое зубчатое колесо (12T) делает три полных оборота, каждый его зубец проходит через точку контакта три раза, а значит, в общей сложности через точку контакта проходит 36 зубцов (3 × 12 = 36). В течение этого времени все 36 зубцов большого зубчатого колеса (36T) проходят через точку контакта, и оно, таким образом, совершает один оборот.

Расчет передаточного числа для двух зубчатых колес

Как вы только что видели, три оборота зубчатого колеса 12T (белая стрелка) приводят к одному обороту зубчатого колеса 36T (красная стрелка). Такую конфигурацию можно описать с помощью *передаточного числа*. Передаточное число — это коэффициент, который показывает *уменьшение выходной скорости* по отношению к входной скорости. Помимо этого, передаточное число — это коэффициент, который показывает *увеличение выходного крутящего момента* по сравнению с входным крутящим моментом. Больший крутящий момент облегчает подъем по наклонной поверхности для транспортного средства. Позже мы поговорим о том, что такое крутящий момент, более подробно.

Соотношение рассчитывается следующим образом:

$$\text{Передаточное число} = \frac{\text{Число зубцов на } \textit{ведомом} \text{ зубчатом колесе}}{\text{Число зубцов на } \textit{ведущем} \text{ зубчатом колесе}} \quad (1)$$

Ведомое зубчатое колесо имеет 36 зубцов, а ведущее — 12 зубцов, так что в нашем примере формула следующая:

$$36 \div 12 = 3$$

В нашем случае передаточное число — это коэффициент, который отражает уменьшение выходной скорости, за счет чего ведомое зубчатое колесо вращается в три раза медленнее, чем ведущее. Другими словами, три оборота ведущего зубчатого колеса соответствуют 1 обороту ведомого колеса.

Передаточное число иногда записывается как количество зубцов на выходе и на входе, разделенное двоеточием, в данном случае 36:12. Если сократить это соотношение, то получится выражение 3:1, которое имеет тот же смысл. Читая слева направо, вы увидите, что 3 входных (ведущих) оборота вызывают 1 выходной (ведомый) оборот.

Вычисление выходной скорости

После того как вы рассчитали передаточное число существующей конструкции, вы можете использовать полученное соотношение для расчета выходной скорости (т.е. скорости вращения *ведомого* колеса), если вы знаете входную скорость (т.е. скорость вращения *ведущего* колеса):

$$\text{Выходная скорость вращения} = \frac{\text{Входная скорость вращения}}{\text{Передаточное число}} \quad (2)$$

Если передаточное число равно 3 и скорость вращения ведущего зубчатого колеса равна 30 оборотам в минуту (об/мин), то ведомое колесо будет вращаться со скоростью 30 ÷ 3 = 10 оборотов в минуту. Этот результат подтверждает, что скорость уменьшается в 3 раза.

Расчет требуемого передаточного числа

Вы можете переписать предыдущую формулу для расчета необходимого передаточного числа для вашей конструкции, если знаете входную скорость и желаемую скорость вращения ведомого зубчатого колеса:

$$\text{Передаточное число} = \frac{\text{Входная скорость вращения}}{\text{Выходная скорость вращения}} \quad (3)$$

Например, если вы хотите, чтобы ведомое зубчатое колесо вращалось со скоростью 120 оборотов в минуту при наличии мотора, который вращает ведущее колесо с постоянной скоростью 72 оборота в минуту, необходимо следующее передаточное число: 72 ÷ 120 = 0,6. Вы можете выполнить это соотношение с помощью ведущего зубчатого колеса 20T и ведомого колеса 12T (12 ÷ 20 = 0,6).

Не каждое передаточное число может быть реализовано с помощью зубчатых колес, входящих в набор MINDSTORMS EV3. Используйте одну из комбинаций передач, приведенных в этой главе, и формулу [2] для расчета итоговой скорости вращения ведомого зубчатого колеса.

ПРИМЕЧАНИЕ **Обязательно используйте одни и те же единицы измерения для входной и выходной скоростей вращения. Если вы измеряете входную скорость в оборотах в минуту, выходная скорость тоже должна быть в оборотах в минуту.**

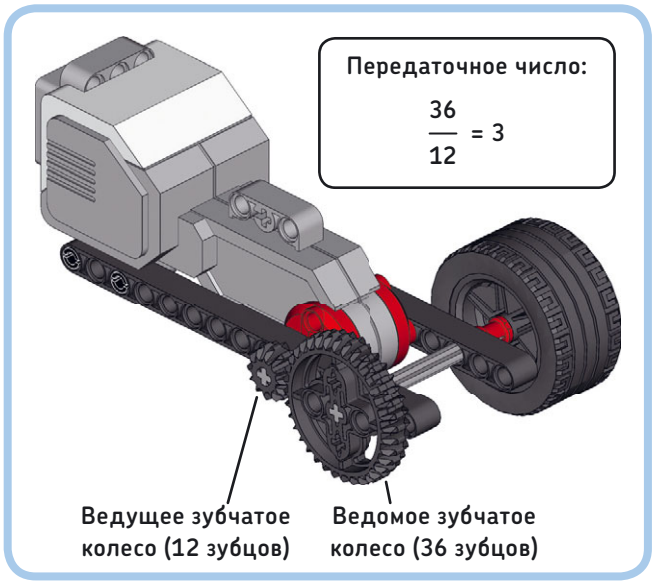


Рис. 11.2. Уменьшение выходной скорости вращения в 3 раза при одновременном увеличении крутящего момента на коэффициент 3. Передаточное число равно 3 (или 3:1)

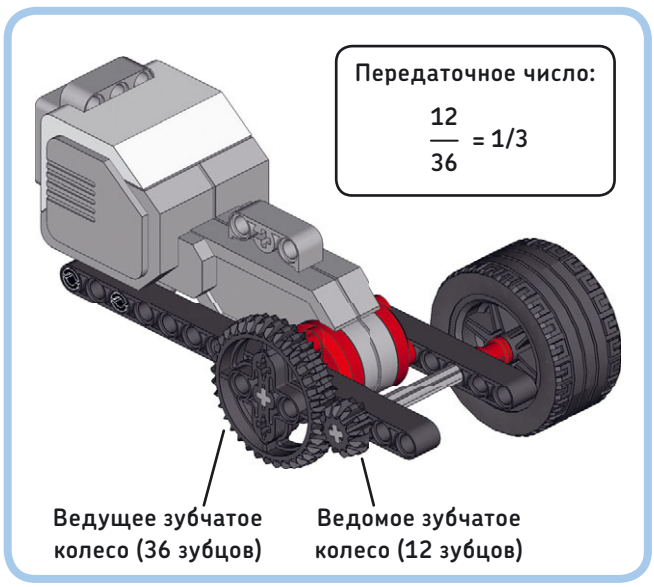


Рис. 11.3. Увеличение выходной скорости вращения в 3 раза, тогда как крутящий момент уменьшается на коэффициент 3. Передаточное отношение 1/3 (или 1:3)

Уменьшение и увеличение скорости вращения

Теперь давайте рассмотрим, как применять зубчатую передачу. Можно использовать зубчатые колеса для изменения скорости вращения на выходе, например скорости вращения колеса по отношению к входной скорости мотора. Чтобы замедлить, или уменьшить, скорость, ведомое колесо должно иметь больше зубцов, чем входное, чтобы передаточное число было выше 1, как показано на рис. 11.2.

Такая конфигурация уменьшает скорость вращения колеса на коэффициент 3. В результате выходной крутящий момент возрастает в 3 раза.

Теперь давайте посмотрим, что произойдет, если поменять местами два зубчатых колеса, как показано на рис. 11.3. Зубчатое колесо 36T стало ведущим от мотора, а шестеренка 12T стала ведомой, соединенной с колесом. Передаточное число 12 ÷ 36 = 1/3, или приблизительно 0,333.

Таким образом, скорость уменьшается на коэффициент 1/3, или, иначе говоря, скорость увеличивается на коэффициент 3. Если вращать ведущее зубчатое колесо со скоростью 30 оборотов в минуту, формула [2] позволяет получить 30 ÷ 0,333 = 90 оборотов в минуту — выходную скорость вращения, которая на самом деле в три раза выше.

Увеличение выходной скорости вращения называется ускорением. Увеличение выходной скорости означает, что

ПРАКТИКУМ № 59: ЗУБЧАТАЯ МАТЕМАТИКА!

Сложность: **Время:**

Какое передаточное число у каждого набора зубчатых колес, показанных на рис. 11.4? Если вращать ведущие зубчатые колеса со скоростью 10 оборотов в минуту, какова будет скорость вращения ведомых колес?

СОВЕТ **Подтвердить свои расчеты можно на практике, собрав зубчатые передачи. Прикрепите к зубчатым колесам стрелки, чтобы легче было увидеть, сколько оборотов совершает каждое.**

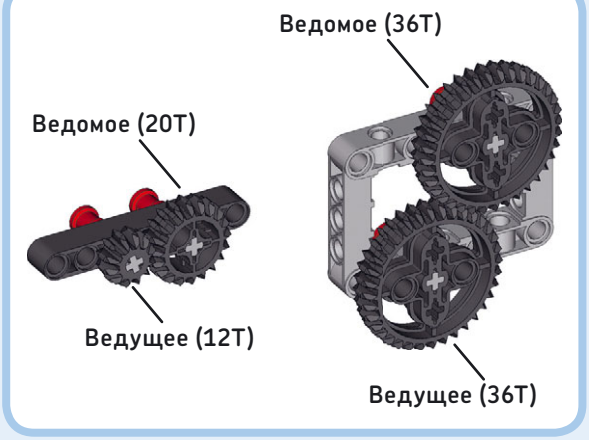


Рис. 11.4. Каковы передаточные числа этих зубчатых передач?

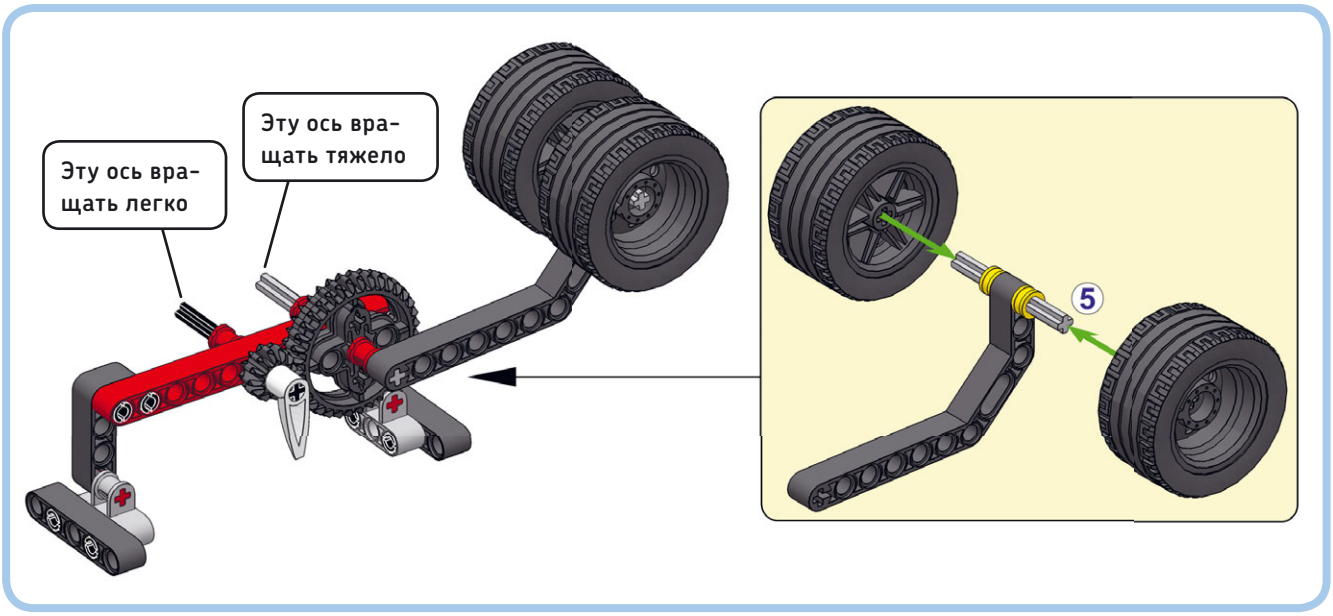


Рис. 11.5. Поднять вес, поворачивая черную ось, легче, потому что требуемый крутящий момент составляет лишь треть крутящего момента, необходимого для вращения серой оси

выходной крутящий момент уменьшается, поэтому подниматься по наклонной поверхности будет труднее.

При использовании двух колес с одинаковым числом зубцов передаточное число будет составлять 1, а скорость вращения и крутящий момент останутся неизменными.

Крутящий момент

Вы разобрались, как можно увеличить крутящий момент, но что это такое? Почему его увеличение может быть полезно? Чтобы на практике получить представление о том, что такое крутящий момент, замените красную стрелку на два колеса, как показано на рис. 11.5. Теперь попытайтесь поднять вес, вращая серую ось вручную. Когда вы это делаете, ваша рука должна применить *крутящий момент* на ось, чтобы уравновесить крутящий момент, созданный весом колес, расположенных на некотором расстоянии от оси.

Крутящий момент является произведением силы и расстояния между позицией приложения силы и осью. В данном случае — силы тяжести, действующей на колеса. Если увеличить вес за счет добавления дополнительных колес или если вы установите колеса дальше от оси, используя более длинную балку, крутящий момент, созданный колесами, будет увеличиваться, и вы должны будете применить больший крутящий момент к серой оси, чтобы поднять их. Теперь попытайтесь поднять вес, поворачивая *черную* ось. Когда вы делаете это, зубчатые колеса увеличивают создаваемый вашими руками крутящий момент в 3 раза, так что поднять вес значительно проще. Тем не менее вы должны повернуть черную ось дальше, чем серую, чтобы полностью поднять вес. (Вы должны вращать ее в три раза больше, чтобы достичь того же эффекта.)

Увеличение крутящего момента

Увеличение крутящего момента с помощью зубчатой передачи полезно, если мотор не может обеспечить достаточный крутящий момент для решения конкретной задачи, например поднятия тяжелого веса. Если ваш мотор недостаточно мощный для выполнения определенных действий, необходимых в вашей программе, вы можете использовать механизмы, чтобы увеличить выходной крутящий момент и уменьшить нагрузку на мотор (рис. 11.2).

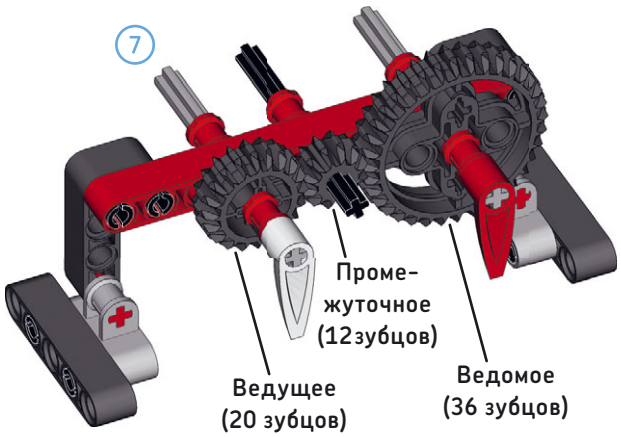


Рис. 11.6. Снимите белую стрелку с черной оси и установите ось 7M и зубчатое колесо 20T, как показано на рисунке. Убедитесь, что обе стрелки направлены вниз

Максимальный крутящий момент большого мотора примерно в три раза превышает максимальный крутящий момент среднего мотора, поэтому для выполнения сложных задач лучше использовать мотор большой мощности. Если оба больших мотора уже используются, скажем, для перемещения вашего робота, вы можете применить зубчатые колеса, чтобы увеличить выходной крутящий момент среднего мотора. Посетите сайт **gears.sariel.pl** для получения подробной информации о взаимосвязи между крутящим моментом и скоростью вращения моторов EV3.

Уменьшение крутящего момента

В редких случаях будет необходимо уменьшить выходной крутящий момент мотора, чтобы защитить хрупкий механизм. Конечно, вы можете уменьшить его с помощью зубчатых передач, используя технику, показанную на рис. 11.3. Но легче ограничить выходной крутящий момент в вашей программе, используя блок **Нерегулируемый мотор** (Unregulated Motor) на низком уровне мощности, например 30%, о чем вы читали в главе 9. Если вы используете этот блок, мотор не увеличивает крутящий момент, когда внешняя сила замедляет его.

Конструирование сложных зубчатых передач

Зубчатые передачи, которые вы видели до сих пор, содержат только два зубчатых колеса, но вы можете добавить дополнительные, чтобы передавать движение на большее расстояние. Например, добавьте зубчатое колесо 20T, как показано на рис. 11.6, и используйте его в качестве ведущего. Давайте более подробно рассмотрим этот новый механизм.

Зубчатая передача теперь состоит из ведущего (20T), ведомого (36T) и *промежуточного* колеса (12T), расположенного в середине. Промежуточное зубчатое колесо передает движение от ведущего колеса к ведомому. Кроме того, оно изменяет направление, в котором вращается ведомое колесо, таким образом, что ведущее и ведомое колеса крутятся в одну сторону (рис. 11.7).

ПРИМЕЧАНИЕ Каждое зубчатое колесо в передаче вращается в направлении, противоположном тому, в котором вращается соседнее колесо. Следовательно, ведущее и ведомое колеса в зубчатой передаче с *нечетным* числом передач вращаются в *одном* направлении; а ведущее и ведомое зубчатые колеса передачи с четным *числом* передач вращаются в *противоположном* направлении.

Вычисление составного передаточного числа

Вы можете определить отношение между входной и выходной скоростью зубчатой передачи, вычислив общее передаточное число, или *составное передаточное число*. Чтобы вычислить его, сначала необходимо определить

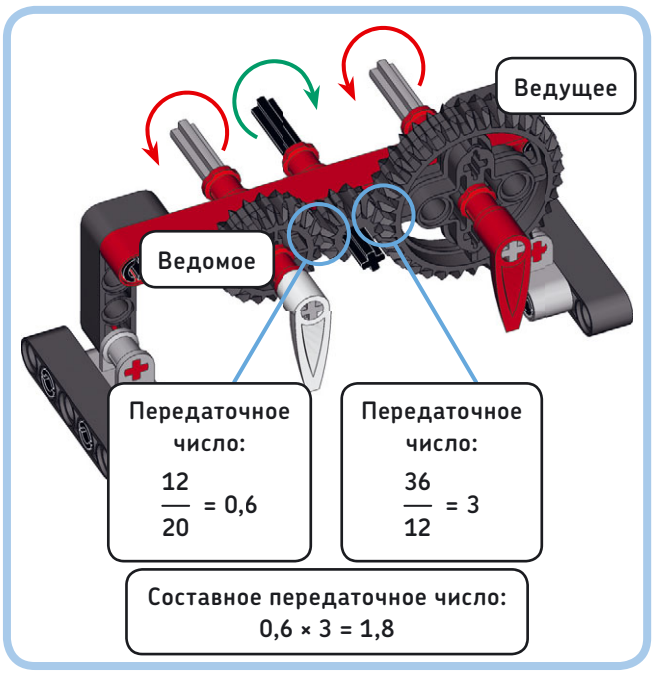


Рис. 11.7. Расчет составного передаточного числа

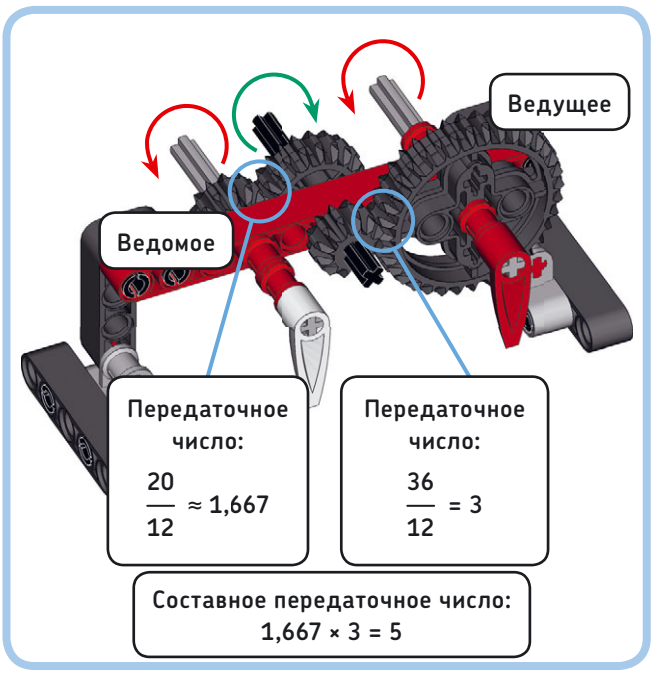


Рис. 11.8. Соединение двух пар зубчатых колес для достижения большего передаточного числа. Если вы не округлили промежуточные значения передаточного числа, составное передаточное число в этом примере будет ровно 5

передаточное число каждой пары соседних зубчатых колес, а затем умножить эти числа, как показано на рис. 11.7.

Механизм в примере имеет две пары смежных зубчатых колес. Сначала ведущее зубчатое колесо передает движение на промежуточное колесо с передаточным числом 0,6. Затем промежуточное зубчатое колесо передает движение на ведомое колесо с передаточным числом 3. Обратите внимание, что промежуточное зубчатое колесо служит в качестве ведомого для первой пары и в то же время для второй пары оно представляет собой ведущее зубчатое колесо.

При расчете составного передаточного числа мы получаем 0,6 × 3 = 1,8. Это означает, что выходная скорость уменьшается в 1,8 раза, а крутящий момент увеличивается в 1,8 раза. Таким образом, 1,8 оборота ведущего зубчатого колеса равно 1 обороту ведомого колеса. Чтобы убедиться в этом на практике, направьте обе стрелки вниз, как показано на рис. 11.7, и поверните белую стрелку 9 раз. Красная стрелка должна сделать 5 оборотов (9 ÷ 1,8 = 5), после чего обе стрелки снова должны оказаться направленными вниз.

Интересно отметить, что, так как центральное зубчатое колесо является промежуточным звеном, оно не влияет на составное передаточное число. То же самое число вы получаете путем деления только ведущего и ведомого зубчатых колес, что дает в результате 36 ÷ 20 = 1,8. Это происходит потому, что число зубцов на промежуточном зубчатом колесе (12T) сокращается при расчете составного передаточного числа:

$$\frac{\cancel{12}}{20} \times \frac{36}{\cancel{12}} = \frac{36}{20} = 1{,}8$$

Дальнейшее увеличение крутящего момента и уменьшение частоты вращения

Иногда для увеличения крутящего момента недостаточно двух зубчатых колес. В этом случае вы можете дополнительно увеличить передаточное число и, следовательно, крутящий момент, сочетая несколько пар зубчатых колес, которые имеют передаточное число больше 1. Чтобы понять, как это работает, измените свою зубчатую передачу, ориентируясь на то, как это изображено на рис. 11.8.

В приведенном выше примере первая пара зубчатых колес имеет передаточное число 20 ÷ 12 ≈ 1,667. Вторая пара имеет отношение 36 ÷ 12 = 3. Вы получаете составное передаточное число зубчатой передачи путем умножения этих чисел, которое дает 1,667 × 3 = 5. Следовательно, выходная частота вращения в 5 раз меньше, чем входная. Иными словами, 5 оборотов белой стрелки соответствуют одному полному обороту красной стрелки.

В результате крутящий момент увеличивается на коэффициент 5. Если заменить красную стрелку грузом, как показано на рис. 11.5, то из-за увеличенного крутящего момента будет легче поднять груз, поворачивая белую стрелку.

Баланс частоты вращения и крутящего момента

Если вы используете зубчатое колесо с красной стрелкой в качестве входа механизма, показанного на рис. 11.8, белая стрелка будет вращаться в пять раз быстрее. В принципе вы можете увеличивать выходную скорость и дальше, добавив еще больше зубчатых колес, но таким образом вы уменьшите выходной крутящий момент. В конце концов, вы достигнете точки, где выходной крутящий момент будет недостаточным для преодоления трения в зубчатой передаче, и зубчатые колеса не будут вращаться.

Кроме того, вы не можете использовать зубчатые колеса, чтобы бесконечно увеличивать скорость гоночного автомобиля, поскольку крутящего момента будет недостаточно, чтобы разогнать автомобиль с места.

В общем, вам придется экспериментировать с различными комбинациями зубчатых колес, чтобы найти

ПРАКТИКУМ № 60: ПРЕДСКАЗУЕМЫЕ ДВИЖЕНИЯ!

Сложность: **Время:**

Можете ли вы проанализировать зубчатую передачу, показанную на рис. 11.9, прежде чем соберете ее? Как быстро вращается красная стрелка справа по сравнению с белой стрелкой? И насколько быстрее вращается белая стрелка по сравнению с красной стрелкой слева? В каком направлении крутится каждая стрелка? После того как вы дали ответы на эти вопросы, соберите зубчатую передачу и проверьте свои предположения.

Рис. 11.9. Зубчатая передача с колесами 36T (слева), 12T (в центре) и второго 36T (справа)

ПРАКТИКУМ № 61: ОБЩЕЕ НАПРАВЛЕНИЕ!

Сложность: **Время:**

Каково составное передаточное число зубчатой передачи на рис. 11.10? Чем отличается эта зубчатая передача от изображенной на рис. 11.1? Чем может быть полезно добавление зубчатого колеса 24T?

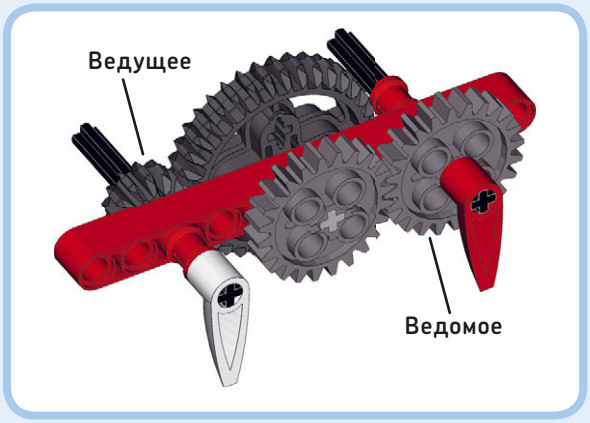


Рис. 11.11. Чему равно составное передаточное число этой зубчатой передачи?

правильный баланс между крутящим моментом и скоростью вращения в вашей конструкции.

- Во-первых, решите, действительно ли необходимо использовать зубчатые колеса. Возможно, достичь нужной скорости и крутящего момента получится путем изменения значения параметра **Мощность** (Power) в блоках вашей программы.
- Если максимальная скорость вашего мотора недостаточно высока, попробуйте увеличить ее, используя передаточное число меньше 1, а также убедитесь, что крутящего момента достаточно для того, чтобы ваш робот мог работать должным образом.
- Если мотор на пределе пытается выполнить тяжелую задачу, можно увеличить крутящий момент, используя передаточное отношение больше 1, за счет снижения скорости.

Трение и люфт

Существуют две важные особенности зубчатых колес, которые могут привести к снижению производительности зубчатой передачи. Во-первых, каждое зубчатое колесо привносит в механизм некоторое количество *трения*. Трение приводит к тому, что вращающиеся объекты замедляются при контакте с другими объектами, а это уменьшает выходной крутящий момент. Вы можете ощутить трение, прижав зубчатые колеса и втулки, изображенные на рис. 11.8, плотно к балке. Вы почувствуете, что теперь требуется больший крутящий момент, чтобы повернуть оси, чем когда втулки и зубчатые колеса соединены свободно. Чтобы уменьшить трение в вашей конструкции, можно укрепить зубчатые колеса между двумя балками. Пример такой сборки вы увидите в разделе «Сборка прочных зубчатых передач» далее в этой главе.

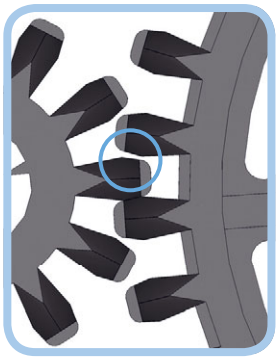


Рис. 11.10. Люфт обусловлен наличием небольшого зазора между зубцами колес

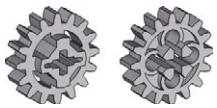
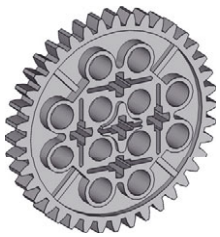
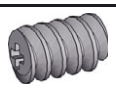
Во-вторых, каждое зубчатое колесо вносит небольшой зазор, или *люфт*, который показан на рис. 11.11. Даже если заблокировать левое зубчатое колесо, правое все еще может немного двигаться, поскольку между зубцами существует небольшое пространство. Это означает, что вы потеряете управление над положением ведомого зубчатого колеса. Независимо от того, насколько точно вы двигаете ведущее зубчатое колесо, ведомое колесо всегда может немного двигаться вперед и назад. Чем длиннее зубчатая передача, тем больше будет люфт.

Выходным валам моторов EV3 также присущ некоторый люфт из-за использования зубчатой передачи внутри каждого мотора.

Использование зубчатых колес из набора MINDSTORMS EV3

Набор EV3 содержит прямозубые, конические и двойные конические зубчатые колеса, а также особый тип передачи — колеса с ручками (кноб-колеса) и червячные колеса. Все это можно посмотреть в табл. 11.1. *Прямозубые зубчатые колеса* могут быть использованы для передачи движения между параллельными осями, а *конические* — между перпендикулярными. *Двойные конические зубчатые* колеса могут быть применены и для параллельных, и для перпендикулярных конфигураций. Понятие «перпендикулярно» здесь означает, что оси расположены под прямым углом друг к другу.

Таблица 11.1. Зубчатые колеса LEGO Technic

Категория	Количество	Зубчатое колесо	Количество зубьев	Радиус
Прямозубые зубчатые колеса	0*		8	0,5M
	0*		16	1M
	2		24	1,5M
	0*		40	2,5M
(Одинарные) конические зубчатые колеса	1		12	Н/Д
	1		20	Н/Д
Двойные конические зубчатые колеса	2		12	0,75M
	4		20	1,25M
	5		36	2,25M
Кноб-колесо	4		Н/Д	Н/Д
Червячное колесо	2		1	Н/Д

* Эти зубчатые колеса не входят в набор EV3, но, зная их характеристики, вы можете выбрать подходящие детали из наборов Technic.

Размерная сетка LEGO

Когда вы подбираете зубчатые колеса для создания зубчатой передачи, необходимо разместить их на нужном расстоянии, чтобы зубцы цеплялись должным образом. Если поместить зубчатые колеса слишком близко друг к другу, они не смогут вращаться; а если слишком далеко, то зубцы будут *проскальзывать*. Когда зубчатое колесо проскальзывает, оно вращается, не цепляя зубцы другого колеса. В этом случае вы услышите характерный треск.

Оставляя нужные промежутки, вы можете использовать любую комбинацию из двух прямозубых зубчатых колес, чтобы создать зубчатую передачу. Кроме того, вы можете объединять двойные конические зубчатые колеса. Комбинировать прямозубые и двойные конические зубчатые колеса тоже можно.

Требуемое расстояние между *центральными точками двух зубчатых колес* является суммой их радиусов, как показано на рис. 11.12. Радиусы прямозубых и двойных конических зубчатых колес измеряются в единицах LEGO (M) и приведены в табл. 11.1. Так, радиус зубчатого колеса 12T составляет 0,75M, а колеса 36T — 2,25M, поэтому расстояние между центральными точками передач должно быть 0,75M + 2,25M = 3M. Поскольку 3 является целым числом, то установить эти зубчатые колеса на балке с помощью осей не составляет труда. Для этого расположите их точно на расстоянии 3M друг от друга.

Зубчатые колеса и половина единицы LEGO

Сложение радиусов двух зубчатых колес иногда в результате дает дробные числа, например 1,5M или 2,5M. К примеру, расстояние между двумя зубчатыми колесами 20T составляет 1,25 M + 1,25 M = 2,5M. Для соединения зубчатых колес с подобным суммарным радиусом вы можете использовать фиксаторы, как показано на рис. 11.13.

ПРИМЕЧАНИЕ **Вместо вычисления требуемого расстояния между передачами самостоятельно вы можете использовать калькулятор передаточного числа, который доступен по адресу gears.sariel.pl. На схеме можно разместить оси, а калькулятор вычислит, какие зубчатые колеса подходят для этого расстояния.**

Угловые зубчатые передачи

Зубчатую передачу можно соорудить вдоль угловой балки, поместив одно зубчатое колесо в угловом отверстии, как показано на рис. 11.14. При таком соединении нужно укрепить зубчатую передачу между двумя балками, как вы увидите далее в этой главе.

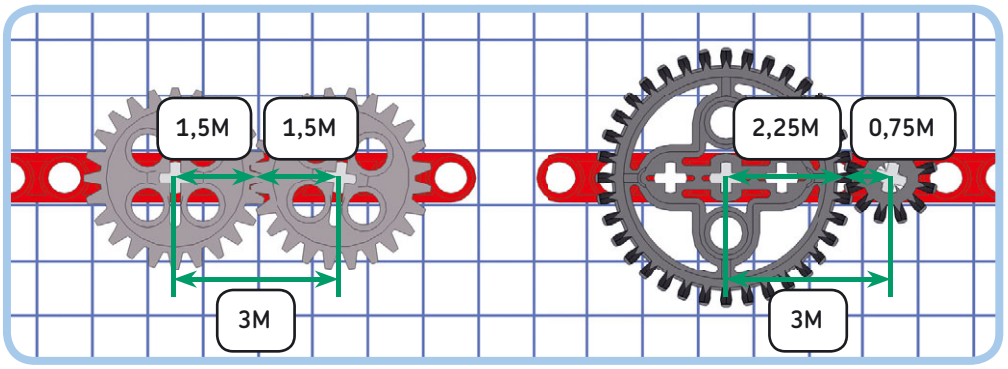


Рис. 11.12. Вычисление требуемого расстояния между центральными точками двух зубчатых колес. Если сумма двух радиусов является целым числом, зубчатые колеса можно установить на балке

Рис. 11.13. Если сумма радиусов представляет собой дробное число, вы должны будете использовать фиксаторы для смещения на 0,5M, как описано в главе 10

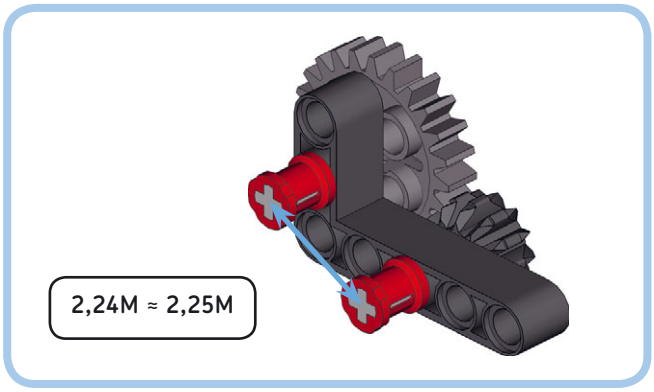


Рис. 11.14. Создание зубчатой передачи на угловой балке. Оба зубчатых колеса 20T вращаются в одном направлении с одинаковой скоростью. Промежуточное зубчатое колесо 12T не оказывает никакого влияния на передаточное число

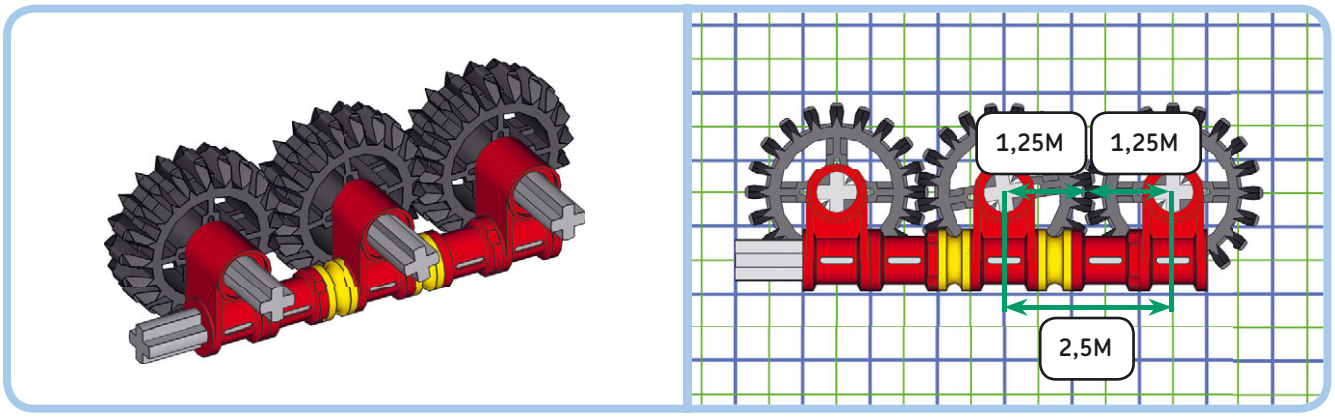


Рис. 11.15. Хотя вы не можете легко достичь расстояния 2,25M между двумя зубчатыми колесами, эта конфигурация обеспечивает расстояние 2,24M, которое вполне подходит для наших целей. Такая комбинация зубчатых колес бывает полезной, потому что значение передаточного числа составляет ровно 2, что позволяет удвоить крутящий момент и уменьшить скорость в 2 раза (или наоборот)

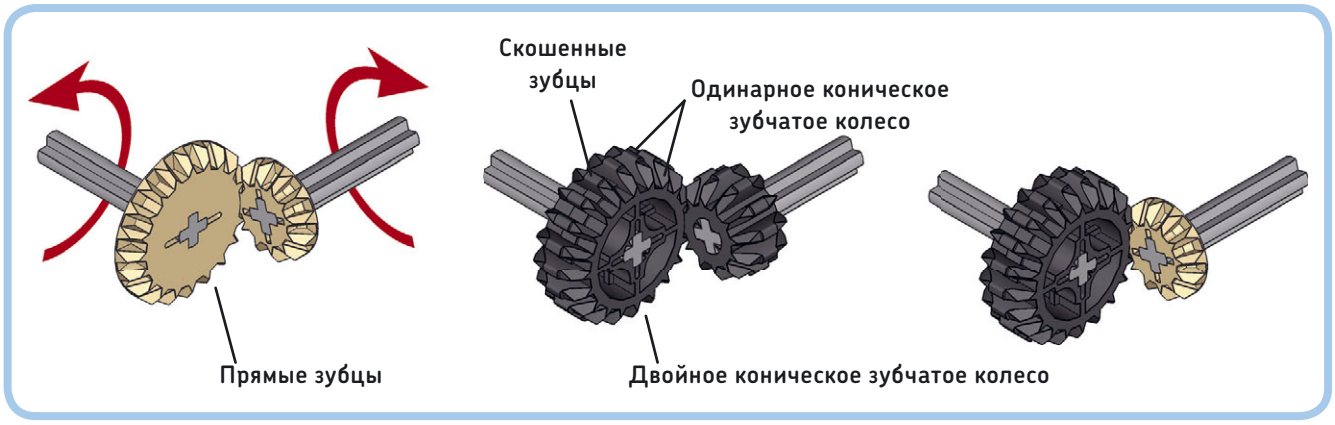


Рис. 11.16. Для передачи движения между двумя перпендикулярными осями используются одинарные и двойные конические зубчатые колеса. Передаточное число каждой комбинации, показанной на рисунке, одинаково. Если вы используете зубчатое колесо 20T в качестве ведущего и колесо 12T в качестве ведомого, передаточное число составит 12 ÷ 20 = 0,6

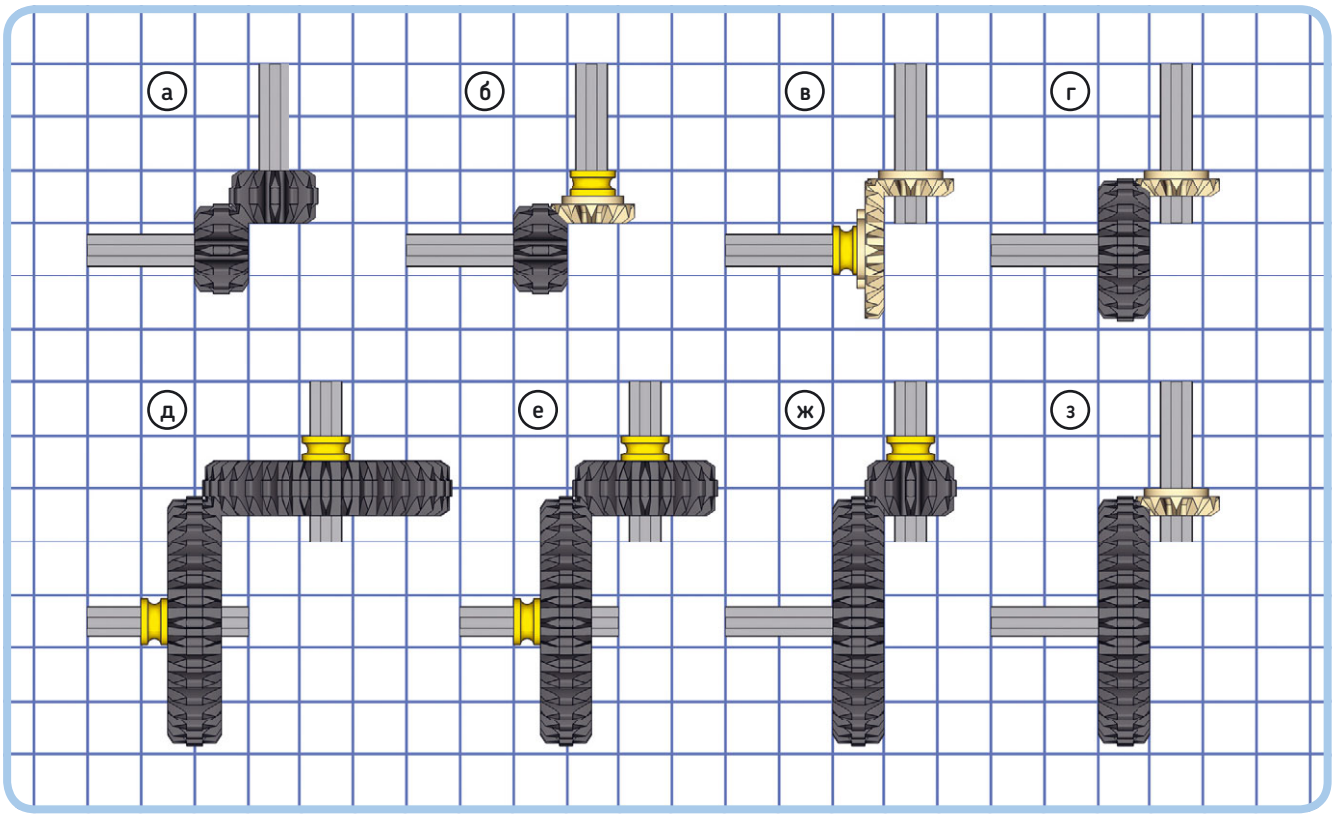


Рис. 11.17. Для передачи движения между перпендикулярными осями вы можете использовать любую комбинацию из двух одинарных и двойных конических зубчатых колес. В некоторых случаях вам понадобятся желтые втулки, позволяющие выполнить смещение на 0,5M

Ошибочные комбинации

Вы можете комбинировать прямозубые и двойные конические зубчатые колеса, но разместить их на подходящем расстоянии затруднительно, поскольку при сложении их радиусов не получается целое число или число с половинным значением. Например, двойное зубчатое колесо 12T и прямозубое зубчатое колесо 24T должны быть помещены на расстоянии 0,75M + 1,5M = 2,25M друг от друга.

Это расстояние, в том числе и со смещением 0,5M, нельзя задать на размерной сетке, но вы можете приблизиться к нему, используя угол прямоугольной балки, как показано на рис. 11.15. Вы можете рассчитать расстояние между отверстиями на угловой балке с помощью теоремы Пифагора или измерить расстояние линейкой (1M равен 8 мм).

Вы всегда должны тщательно проверять зубчатую передачу, если зубчатые колеса не находятся на абсолютно точном расстоянии. Зубчатые колеса должны плавно вращаться и не проскальзывать, даже если вы вручную заблокируете одно зубчатое колесо. Если вы не уверены, будет ли работать конкретное неправильное сочетание, лучше всего подобрать комбинацию зубчатых колес, чьи радиусы при сложении дают целое число.

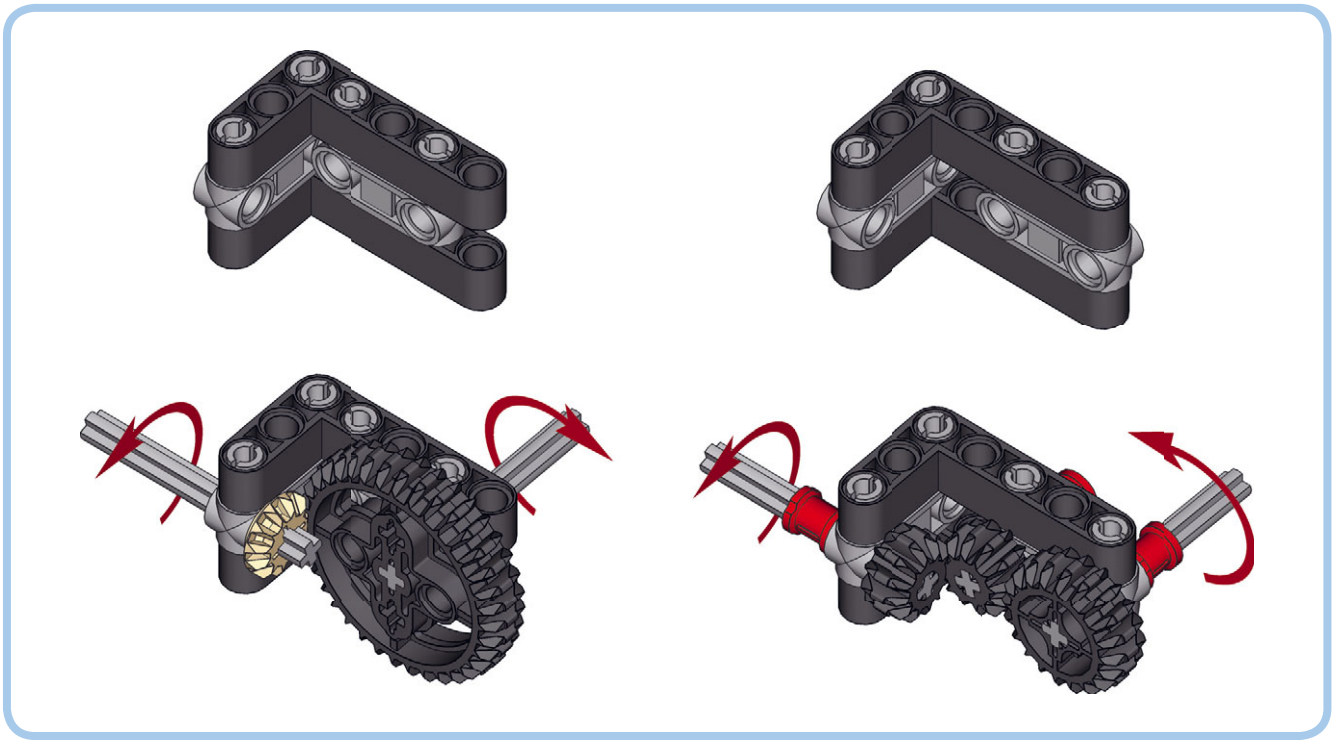


Рис. 11.18. Использование двух Г-образных балок и двух фиксаторов для надежного крепления двух перпендикулярных осей. Зубчатая передача справа содержит и перпендикулярное соединение, и соединение между двумя параллельными осями; зубчатое колесо 12T в середине выступает в роли промежуточного звена

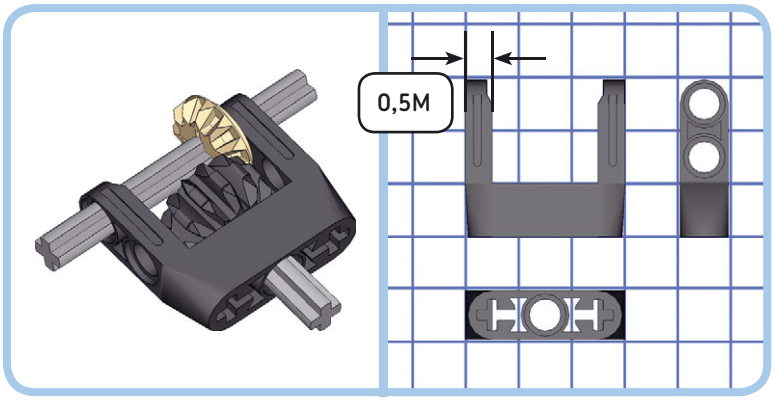


Рис. 11.19. Создание компактного перпендикулярного соединения

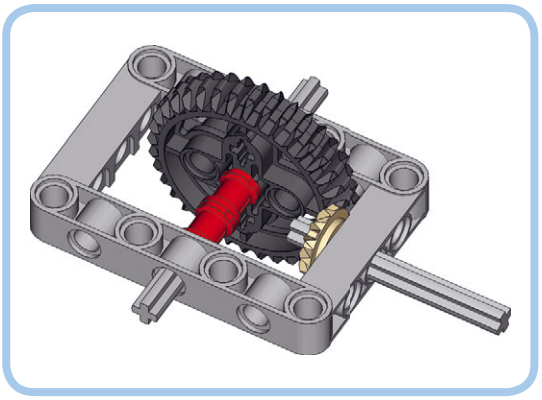


Рис. 11.20. Соединение перпендикулярных осей с помощью рамки

Одинарные и двойные конические зубчатые колеса

Для передачи движения между двумя перпендикулярными осями можно использовать одинарные и двойные конические зубчатые колеса, как показано на рис. 11.16.

Двойное коническое зубчатое колесо — это на самом деле сочетание прямозубого зубчатого колеса и двух конических зубчатых колес. До сих пор мы использовали только прямозубые колеса для передачи движения между двумя параллельными осями, но вы можете использовать конические зубцы с обеих сторон, чтобы создать перпендикулярное соединение. Например, одинарное коническое зубчатое колесо 20T может быть соединено с одной скошенной стороной двойного конического зубчатого колеса 20T. Так же можно соединить зубчатые колеса 12T, поэтому каждая передача, показанная на рис. 11.16, имеет то же передаточное число.

Перпендикулярные соединения на сетке LEGO

Любая комбинация одинарных и двойных конических зубчатых колес может быть составлена с помощью размерной сетки LEGO. На рис. 11.17 показаны некоторые особенно полезные комбинации. Для передачи движения между перпендикулярными осями вы можете выбрать один из этих примеров и использовать размерную сетку в качестве эталона для разработки конструкции, которая удерживает оси на месте.

ПРИМЕЧАНИЕ Метод расчета расстояния между двумя зубчатыми колесами с помощью их радиусов работает только для параллельных конфигураций колес, что вы можете увидеть на рис. 11.12. Для перпендикулярных конфигураций используйте размерную сетку LEGO, как показано на рис. 11.17.

Соединение перпендикулярных осей

При передаче движения между двумя перпендикулярными осями важно создать прочную конструкцию, чтобы зубчатые колеса не проскальзывали. Рис. 11.18 демонстрирует, как добиться этого с помощью угловых балок и фиксаторов.

Набор MINDSTORMS EV3 содержит особые детали для соединения двух небольших зубчатых колес на перпендикулярных осях, как показано на рис. 11.19. Эти детали можно легко подключить к среднему мотору (см. рис. 11.28 (*b*) далее в этой главе). Кроме того, для надежного соединения перпендикулярных осей можно использовать рамку, как показано на рис. 11.20.

Кноб-колеса

Кноб-колеса, или *колеса с ручками*, можно использовать для передачи движения между двумя параллельными или перпендикулярными осями, как показано на рис. 11.21.

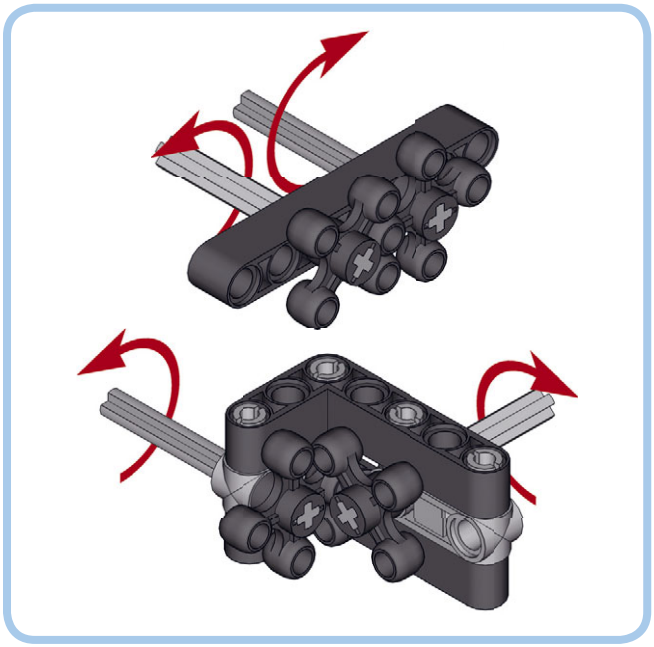


Рис. 11.21. Используйте кноб-колеса для передачи движения с параллельными осями (вверху) и с перпендикулярными (внизу)

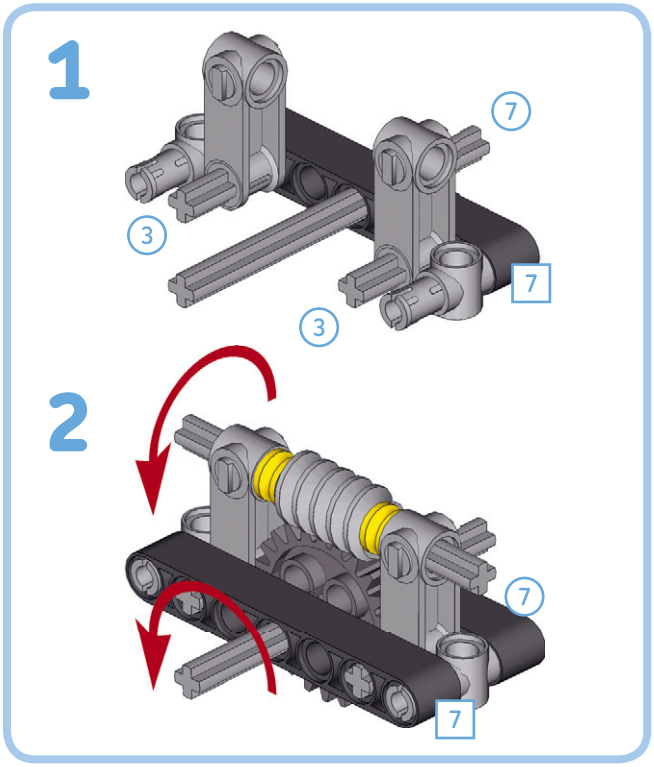


Рис. 11.22. Вы можете двигать зубчатое колесо 24T с помощью червячной передачи, чтобы уменьшить скорость вывода в 24 раза. Специализированные фиксаторы серого цвета поддерживают червячное колесо на нужном расстоянии, чтобы зубчатое колесо 24T могло поворачиваться. Вы можете использовать эту схему в качестве отправной точки для собственных конструкций с червячной передачей

ПРАКТИКУМ № 62: ПЕРПЕНДИКУЛЯРНЫЕ ВАРИАНТЫ!

Сложность: **Время:**

Существует вторая комбинация конических зубчатых колес, которая подходит для элемента, показанного на рис. 11.19. Какие для нее используются зубчатые колеса и каково их передаточное число?

ПРАКТИКУМ № 63: МОЩНЫЕ ЗУБЧАТЫЕ ПЕРЕДАЧИ!

Сложность: **Время:**

Можете ли вы создать зубчатую передачу с составным передаточным числом 15 из зубчатых колес, представленных в наборе MINDSTORMS EV3? Когда вы соберете такую передачу, протестируйте увеличение крутящего момента, пытаясь поднять вес двух шин (см. рис. 11.5).

СОВЕТ **Какое составное передаточное число вы получите, если объедините зубчатые передачи, показанные на рис. 11.8 и 11.20, в одну длинную зубчатую передачу?**

Кноб-колеса могут обрабатывать бóльшие крутящие моменты без проскальзывания, чем конические зубчатые колеса, поэтому они лучше подходят для использования с перпендикулярными осями с высокой нагрузкой. Благодаря уникальной форме кноб-колесо можно использовать для движения только другого кноб-колеса; они не могут двигать прямозубые или одинарные/двойные конические зубчатые колеса. Передаточное число между двумя кноб-колесами всегда равно 1.

Червячные колеса

Червячное колесо может двигать прямозубые зубчатые колеса (рис. 11.22), при этом добиваясь существенного снижения скорости. При расчете передаточного числа вы можете рассмотреть червячное колесо в качестве ведущего с одним зубом. Когда ведомым является зубчатое колесо 24T, передаточное число составляет 24 ÷ 1 = 24, тем самым скорость вывода снижается на коэффициент 24. В принципе, крутящий момент увеличивается в 24 раза, но, поскольку червячная конфигурация обладает бóльшим трением, чем передачи с обычными зубчатыми колесами, некоторый прирост крутящего момента теряется.

В отличие от тех конфигураций зубчатых колес, которые вы видели до сих пор, здесь передача движения работает только одним способом: вы можете вращать червячное колесо,

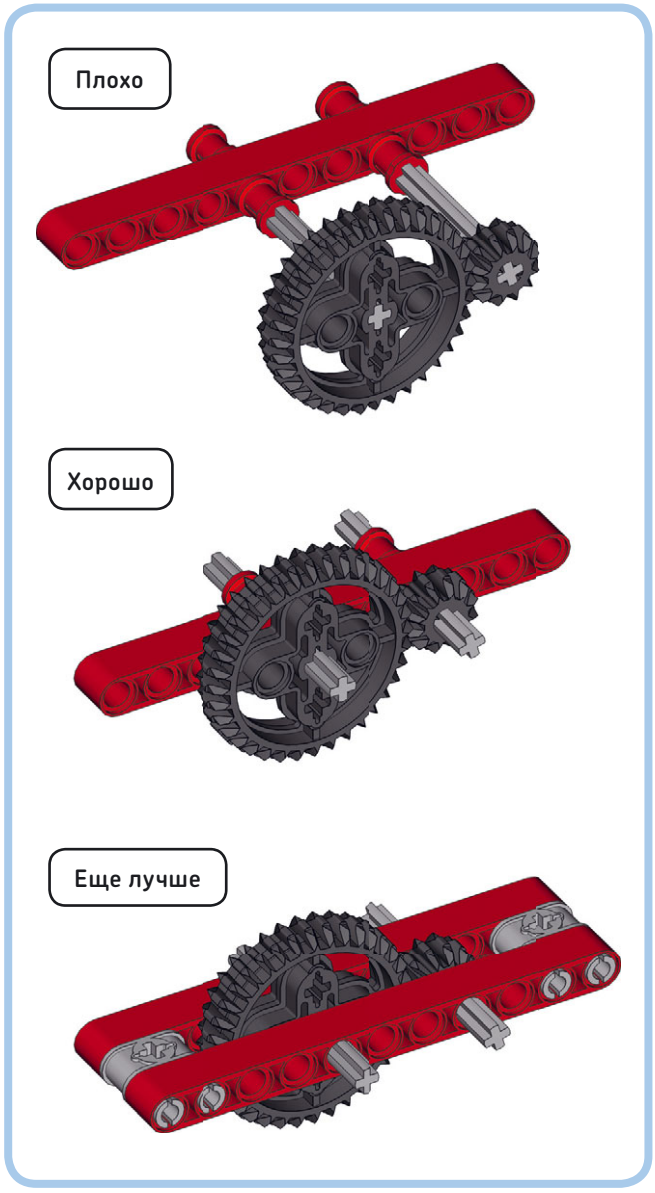


Рис. 11.23. Укрепление зубчатой передачи балками гарантирует, что зубцы не будут проскальзывать

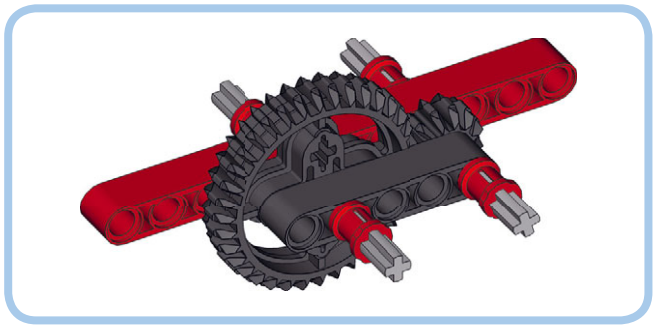


Рис. 11.24. Зубчатая передача, укрепленная короткой балкой

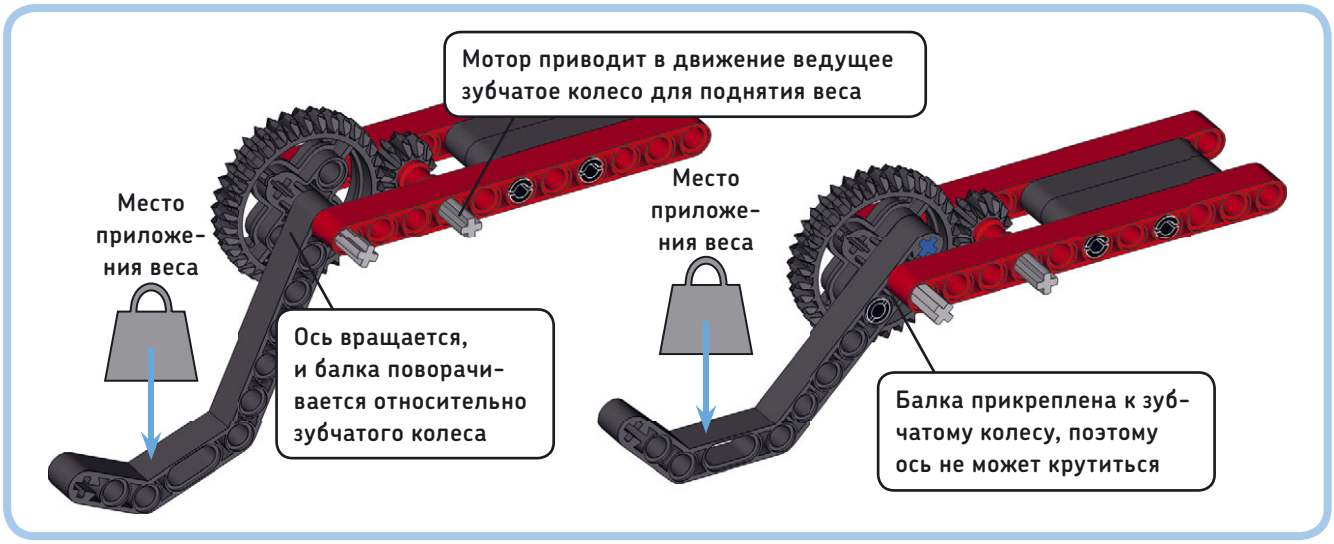


Рис. 11.25. Высокие значения крутящего момента могут привести к движению (кручению) оси (слева). Вы можете уменьшить нагрузку на выходную ось, присоединив изогнутую балку, которая перенесет большую нагрузку непосредственно на монтажные отверстия ведомого зубчатого колеса (справа)

ПРАКТИКУМ № 64: ЧЕРВЯЧНОЕ ДВИЖЕНИЕ!

Сложность: **Время:**

Можете ли вы создать зубчатую передачу, которая уменьшает скорость вывода в 8 раз?

СОВЕТ **Сначала уменьшите скорость в 24 раза, а затем увеличьте ее в 3 раза. Почему результат идентичен?**

чтобы передать движение на прямозубое зубчатое колесо, но не наоборот! Этот факт может быть преимуществом вашей конструкции. Например, если вы используете червячную передачу для управления роботизированной рукой, рука не будет опускаться обратно, когда вы прекратите подавать питание на мотор. Если использовать обычные зубчатые колеса, сила тяжести, действующая на руку, приведет к тому, что зубчатые колеса повернутся в противоположном направлении и рука опустится.

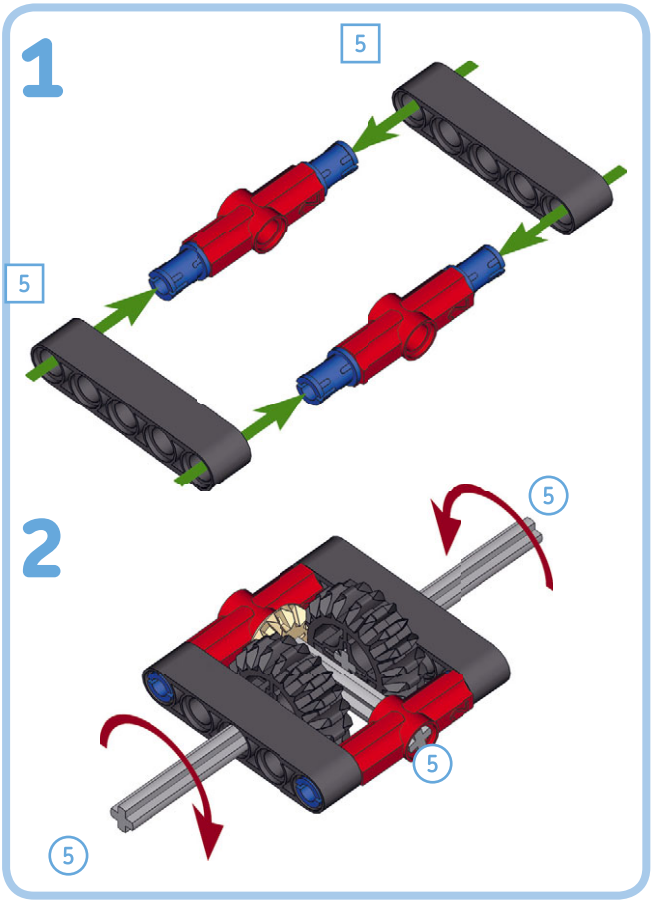


Рис. 11.26. Изменение направления вращения с помощью зубчатых колес

Сборка прочных зубчатых передач

После того как вы выбрали зубчатые колеса для конструирования передачи, вам необходимо установить их на робота. Для каждого робота доступны различные возможности установки осей с зубчатыми колесами, но всегда важно обращать внимание на надежность — оси не должны быть согнутыми, или скрученными, или установленными так, что зубцы не скользят.

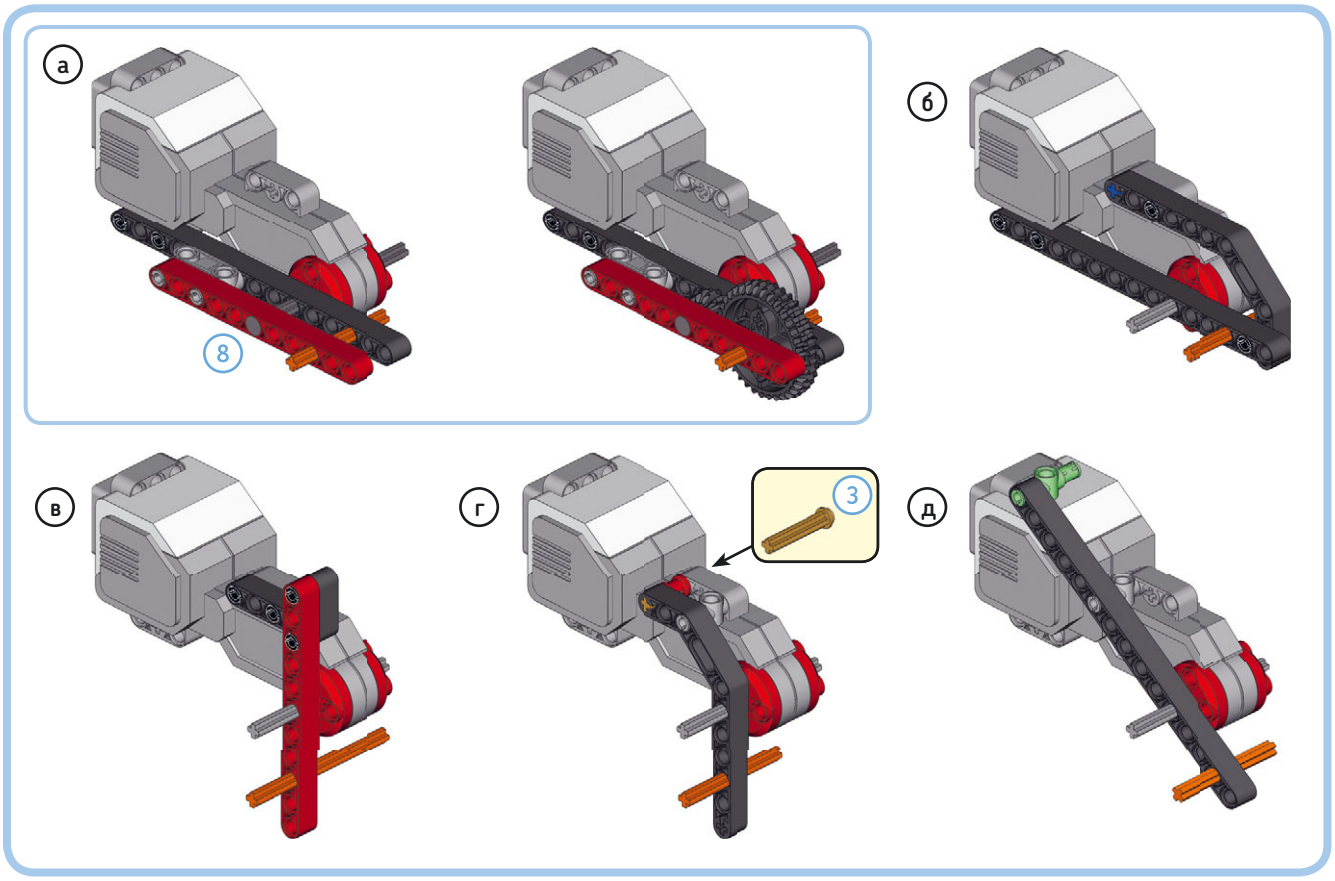


Рис. 11.27. Вы можете дополнительно установить балки на большой мотор для создания точек крепления для осей и зубчатых колес. Серые оси соединены с валом мотора, оранжевые оси показывают, куда прикрепить зубчатое колесо 36T для достижения передаточного отношения 3 (пример a). Балка в примере e расположена под углом 53,13 градуса, как описано в главе 10. Фиксатор, показанный зеленым, совпадает по размерной сетке LEGO

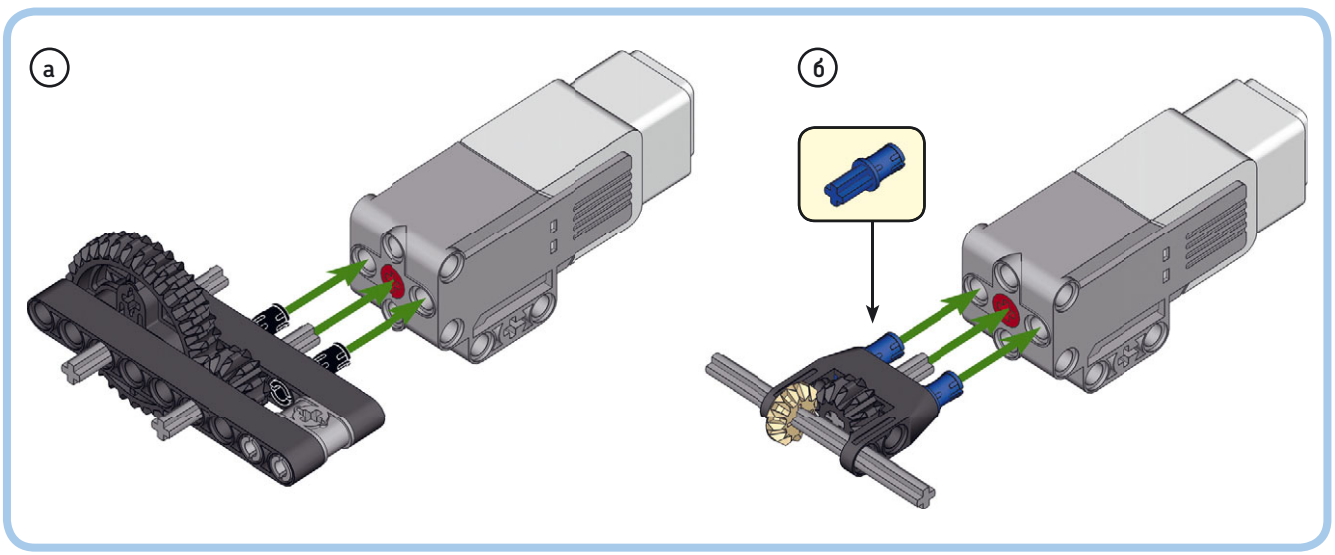


Рис. 11.28. Подключение зубчатых колес к среднему мотору. Выходная ось в примере a параллельна валу мотора и перпендикулярна ему в примере b

Крепление зубчатых колес с помощью балок

Механическое взаимодействие зубцов двух колес смещает друг от друга оси, на которых они установлены. Если из-за этого расстояние между зубчатыми колесами увеличится слишком сильно, они больше не будут цепляться должным образом, и зубцы станут проскальзывать. В этом случае можно уменьшить отклонение осей, установив рядом с зубчатой передачей балку, как показано на рис. 11.23. А если вы закрепите передачу между двумя балками, она станет еще более надежной. Механическое взаимодействие между зубцами по-прежнему существует, но поскольку балки удерживают оси с зубчатыми колесами на месте, то зубцы шестеренок крепко цепляются и не проскальзывают.

Если в вашей конструкции не представляется возможным закрепить зубчатые колеса второй большой балкой, вы можете добавить короткую балку или фиксатор (рис. 11.24). Это решение не столь надежно, но тоже защитит зубцы колес от проскальзывания.

Предотвращение прокручивания оси

При увеличении крутящего момента с помощью зубчатых колес можно добиться таких его значений, что оси придут в движение, как показано на левой части рис. 11.25. Вы можете избежать движения выходной оси, присоединив ту часть механизма, которая несет тяжелый груз, непосредственно на двойное коническое зубчатое колесо 36T, а не на ось, как показано на правой части рисунка.

По той же причине хорошим решением будет подсоединение балки непосредственно к отверстиям на валу мотора, а не только соединение ее с осью (см. рис. 10.29 в главе 10).

Изменение направления вращения

Перевернув мотор на входе, вы можете изменить направление вращения оси. Сделать это можно и с помощью зубчатых передач, как показано на рис. 11.26. Это полезно, если один мотор приводит в движение два механизма на одной оси, которые должны вращаться в противоположных направлениях.

Конструирование с зубчатыми колесами и моторами EV3

Вы часто будете использовать зубчатые колеса для передачи движения от мотора к механизму, такому как роботизированная рука. У большого мотора не так много точек крепления вблизи вращающегося вала, но вы можете добавить их самостоятельно, используя балки, как показано на рис. 11.27.

На рис. 11.28 показано, как можно приводить в движение оси, расположенные параллельно (a) и перпендикулярно (b) к валу среднего мотора.

Дальнейшее изучение

В этой главе вы разобрались, как работают зубчатые колеса и как их использовать, чтобы изменить скорость и крутящий момент, передаваемые моторами из набора EV3. Вы узнали, как передаточное отношение, трение и люфт влияют на производительность зубчатой передачи. Кроме того, вы изучили,

СДЕЛАЙ САМ № 15: ДРЭГСТЕР!

Сборка: **Программирование:**

Вы можете создать очень быстрого робота для дрэг-рейсинга? Сконструируйте робота с четырьмя колесами и используйте большой мотор, чтобы обеспечить движение двух из них. В этом задании нет необходимости добавлять функцию рулевого управления. Используйте зубчатые колеса для ускорения вашего робота. Какое передаточное отношение максимально увеличивает скорость?

СОВЕТ **Установите инфракрасный датчик на передней панели и запрограммируйте робота остановиться, если вблизи обнаружится препятствие.**

СДЕЛАЙ САМ № 16: РОБОТ-УЛИТКА!

Сборка: **Программирование:**

Какого наибольшего передаточного числа вы можете достичь, используя зубчатые колеса из набора MINDSTORMS EV3? Определите это число и примените его для создания самого медленного робота из всех. Разумеется, он должен двигаться!

СОВЕТ **Включите червячное колесо в зубчатую передачу.**

СДЕЛАЙ САМ № 17: РОБОТ-АЛЬПИНИСТ!

Сборка: **Программирование:**

Можете ли вы создать робота, который способен подниматься вертикально между двумя стенами, словно он карабкается вверх по дымоходу? Чтобы сделать «дымоход», установите книжную полку или стеллаж параллельно стене на расстоянии около 30 см. Обязательно вниз положите подушку на случай, если ваш робот неожиданно упадет.

Каким образом робот может подняться вертикально? Можно ли использовать колеса, чтобы въехать вверх по стенам?

СОВЕТ **Посетите страницу goo.gl/nL1EZV, чтобы посмотреть, как я построил такого робота из предыдущего поколения LEGO MINDSTORMS. Можете ли вы сделать то же самое, используя EV3?**

как создавать прочные зубчатые передачи, используя разные виды зубчатых колес: цилиндрические, а также одинарные и двойные конические. В следующей части книги вы сможете приложить свои навыки сборки и программирования к созданию гоночного автомобиля и насекомого-робота. Но сначала решите несколько заданий «Сделай сам», приведенных ниже, чтобы накопить опыт работы с зубчатыми колесами.

Если вы хотите больше узнать о том, как устроены зубчатые передачи, и о других методах сборки, я настоятельно рекомендую вам прочитать книгу Павла Кмиека, известного под ником Sariel, «The Unofficial LEGO Technic Builder's Guide», которая охватывает конструирование из деталей LEGO Technic намного более подробно.

СДЕЛАЙ САМ № 18: ПОВОРОТНАЯ ПЛАТФОРМА!

Сборка: **Программирование:**

Можете ли вы построить автоматизированную поворотную платформу? Она может транспортировать и вращать тяжелые предметы, например поезда или автомобили. Платформа может служить в качестве стационарной основы для таких механизмов, как роботизированная рука, или для робота, который раскладывает детали LEGO по различным отсекам для хранения. Используйте один мотор, чтобы поворотная платформа могла вращаться как по часовой стрелке, так и против.

СОВЕТ **Сконструируйте платформу с помощью балок, а потом добавьте четыре колеса под ней, расположив их, как показано на рис. 11.29. В каком направлении должно вращаться каждое колесо? Нужно ли вам подключать к мотору каждое колесо и чем здесь может быть полезен механизм, изображенный на рис. 11.26?**

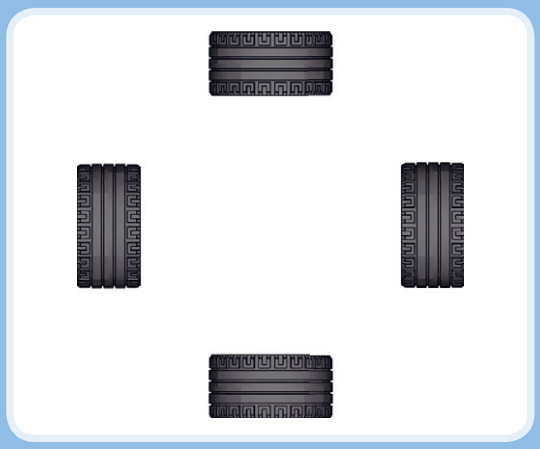


Рис. 11.29. Конфигурация колес для практикума «Сделай сам № 18»

СДЕЛАЙ САМ № 19: РОБОТИЗИРОВАННАЯ РУКА!

Сборка: **Программирование:**

Можете ли вы сделать роботизированную руку, которая будет способна захватывать и поднимать предметы? Используйте один мотор, чтобы робот мог поворачиваться на месте. Другой мотор должен опускать и поднимать руку. А третий — открывать и закрывать механизм захвата. Разработайте программу, которая позволяет управлять каждым мотором с помощью удаленных команд.

СОВЕТ **Используйте вращающуюся платформу, которая описана в практикуме «Сделай сам № 18», в качестве основы робота.**

ЧАСТЬ IV

Роботы-животные и транспортные средства

12

FORMULA EV3: робот-гонщик

Теперь, когда вы научились разрабатывать программы для модуля EV3, позволяющие управлять моторами и датчиками, можно приступать к сборке более сложных роботов, таких как автономные транспортные средства, роботизированные животные и сложные механизмы. В этой главе представлен гоночный автомобиль FORMULA EV3, показанный на рис. 12.1.

В отличие от робота EXPLOR3R, которого вы сконструировали ранее, у гоночного автомобиля три мотора. Два больших мотора в задней части приводят автомобиль в движение, а средний мотор позволяет управлять передними колесами. Представьте, что задние моторы — это двигатель автомобиля, а передний — его рулевое колесо.

Сконструировав гоночный автомобиль, приступайте к программированию.

Создайте несколько контейнеров «Мой блок», чтобы можно было легко запрограммировать автомобиль ездить и рулить. Затем объедините эти блоки в программу, которая позволит управлять автомобилем дистанционно,

Рис. 12.1. Гоночный автомобиль FORMULA EV3

и в программу, которая позволит роботу ездить самостоятельно и объезжать препятствия. В заключение вы получите задание добавить функциональности текущей модели и увеличить его скорость передвижения с помощью зубчатых колес.

Сборка гоночного автомобиля FORMULA EV3

Соберите гоночный автомобиль, следуя инструкциям, приведенным далее. Перед началом сборки выберите необходимые детали, которые показаны на рис. 12.2.

Передвижение и рулевое управление

В гоночном автомобиле FORMULA EV3 используются два разных механизма для передвижения и рулевого управления. Для перемещения вы включаете большие моторы сзади. Управлять автомобилем можно, поворачивая передние колеса влево или вправо с помощью среднего мотора, расположенного спереди. Объединив передвижение и рулевое управление в одной программе, вы сможете перемещать автомобиль в любом направлении.

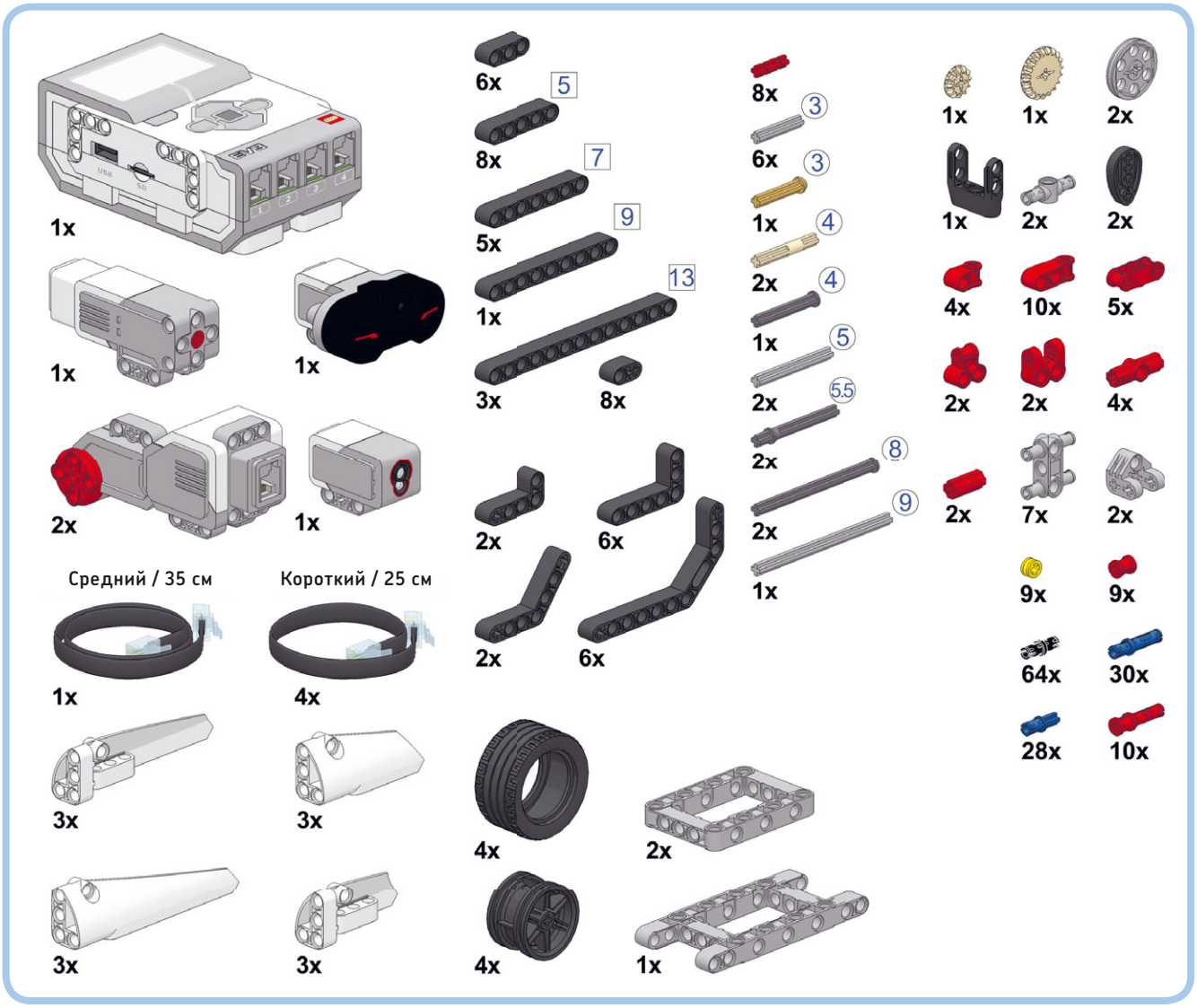
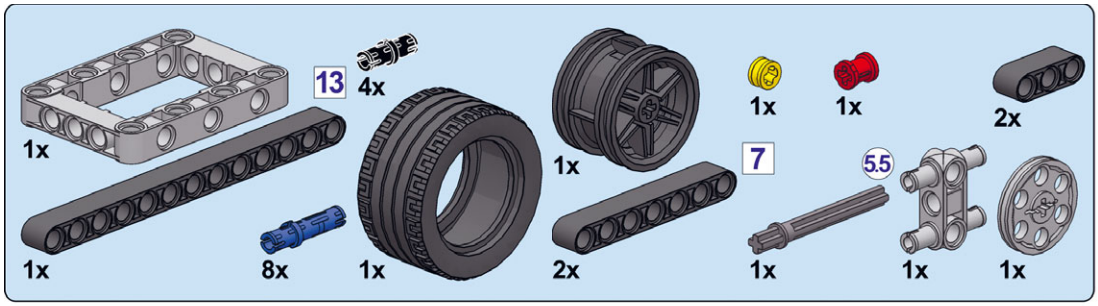
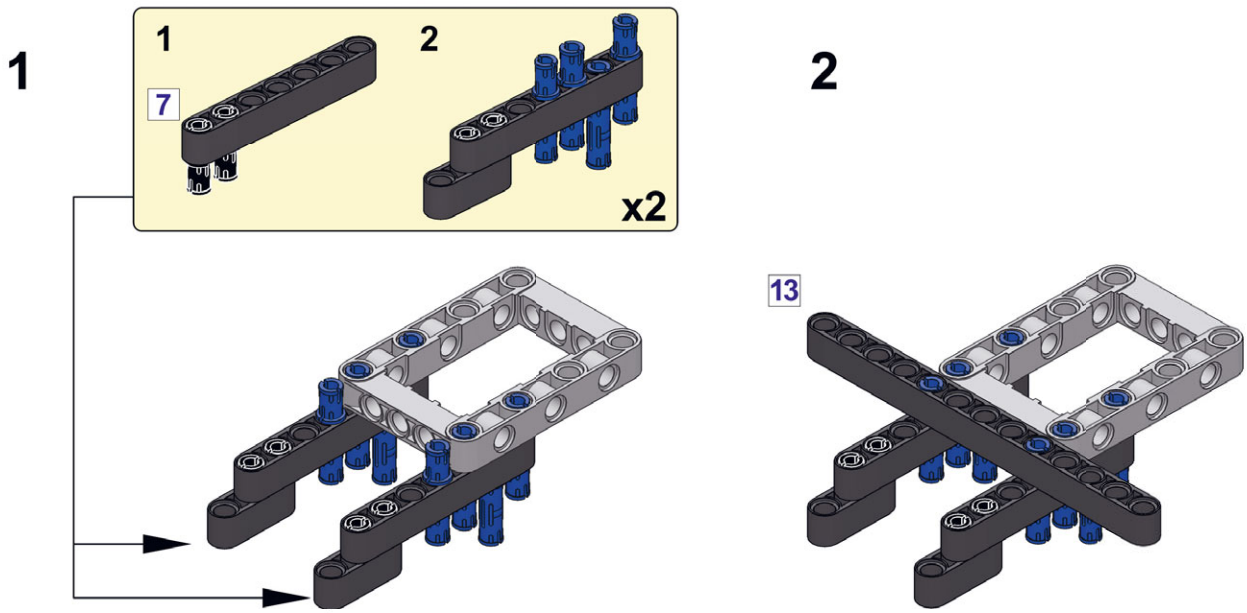
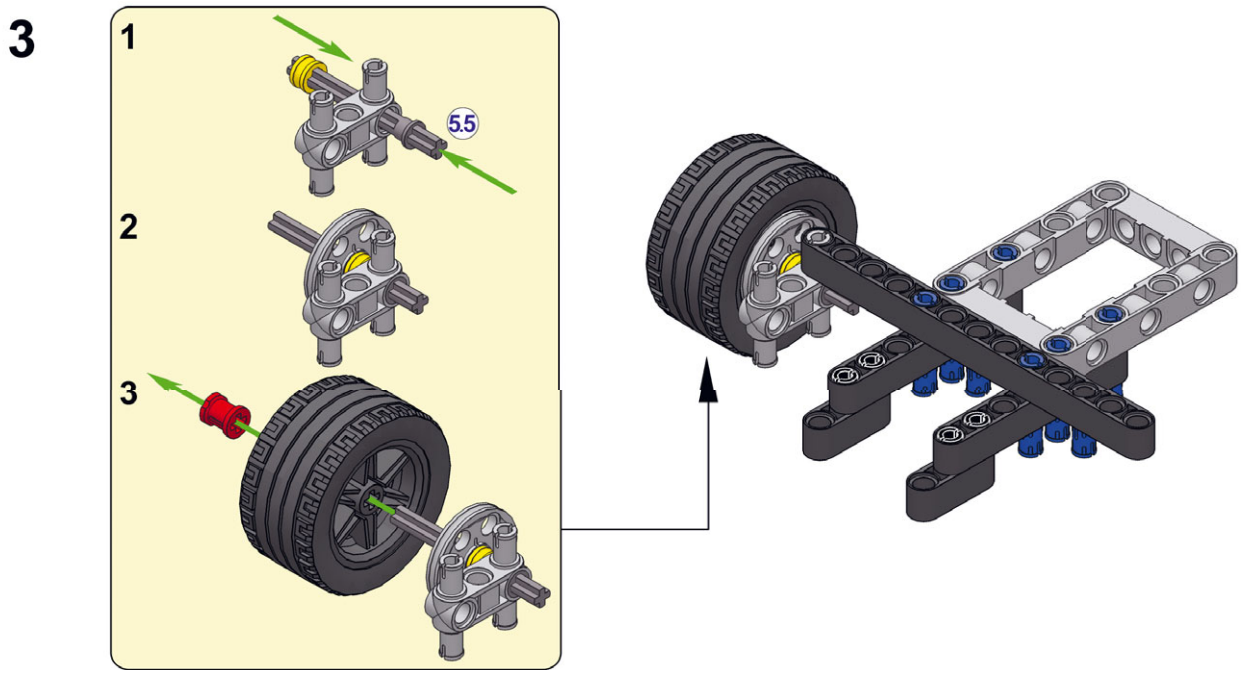
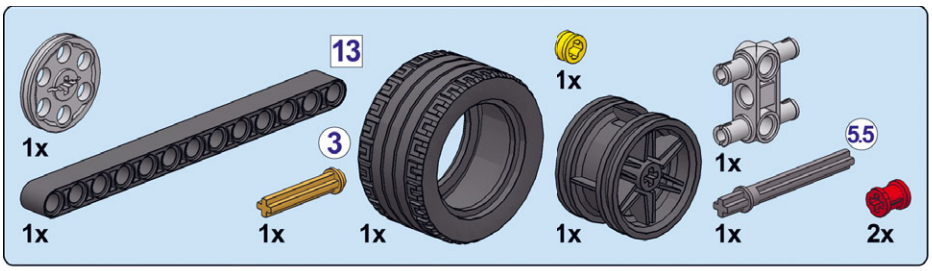


Рис. 12.2. Детали, необходимые для сборки гоночного автомобиля FORMULA EV3

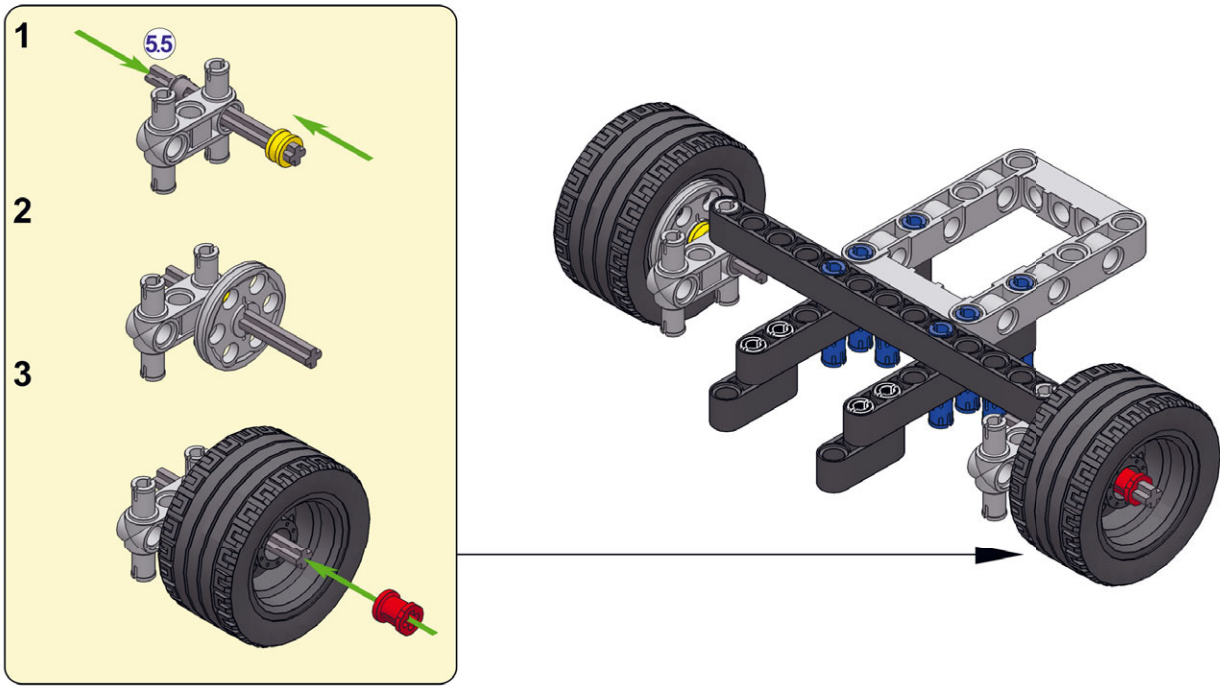

1x
13
4x
1x
1x
1x
1x
7
5.5
2x
1x
8x
1x
2x
1x
1x
1x


1
1
7
2
x2
2
13

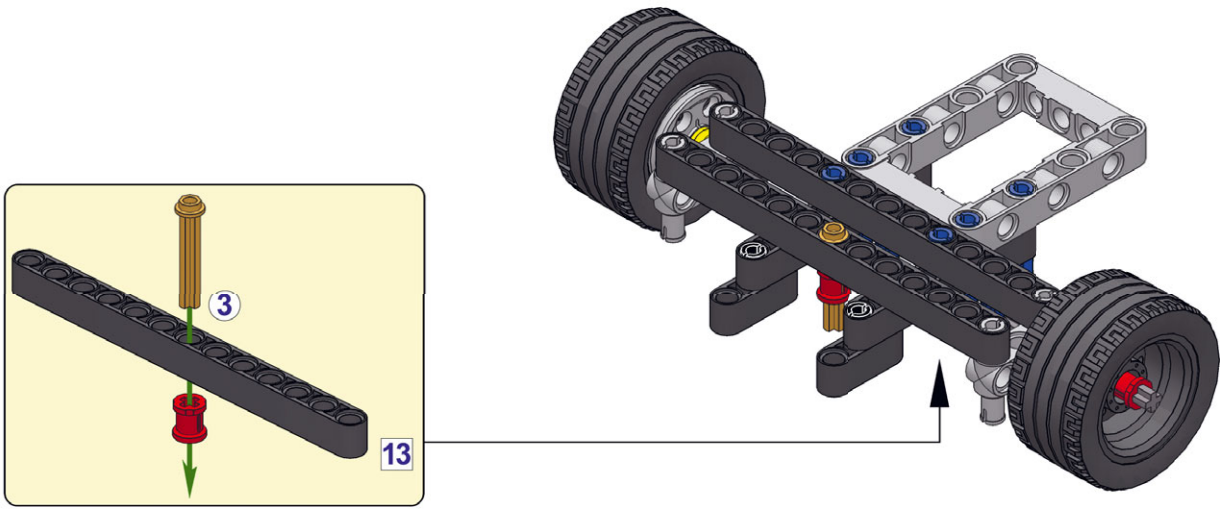

3
1
5.5
2
3

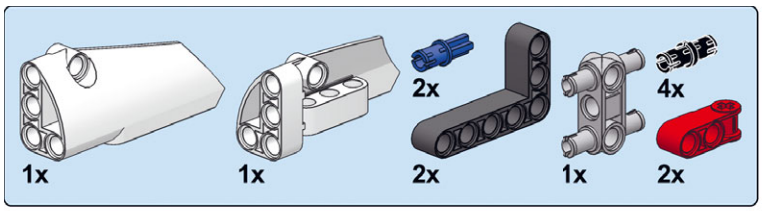


4

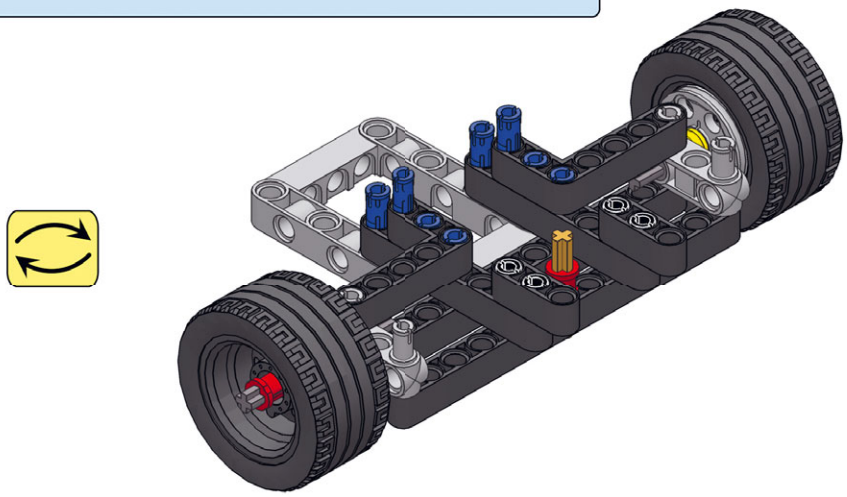


5

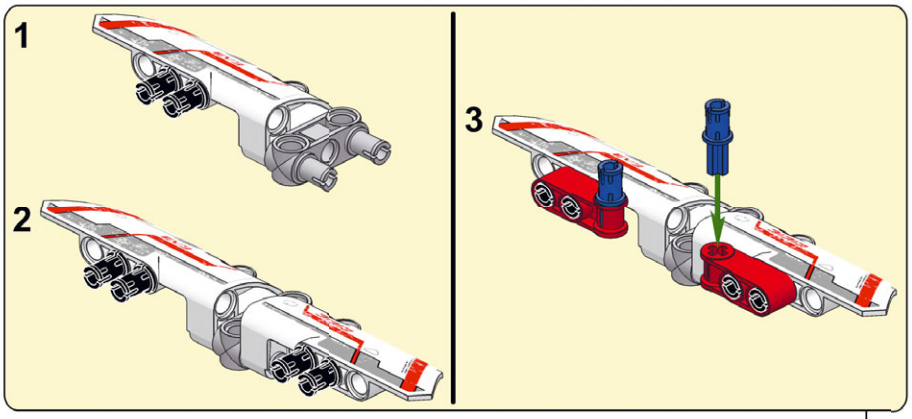


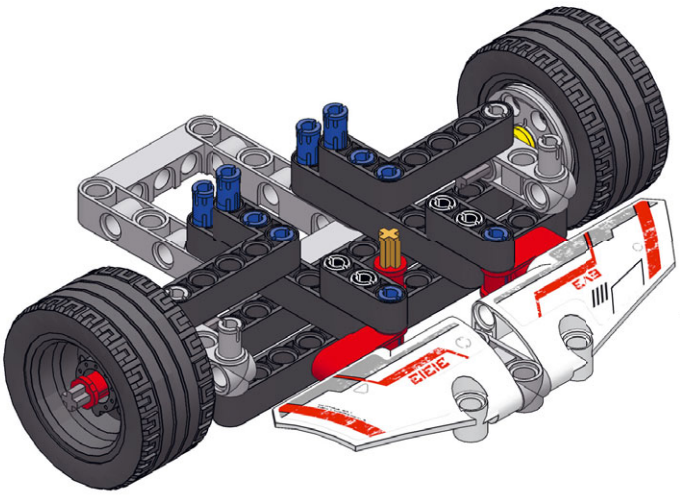


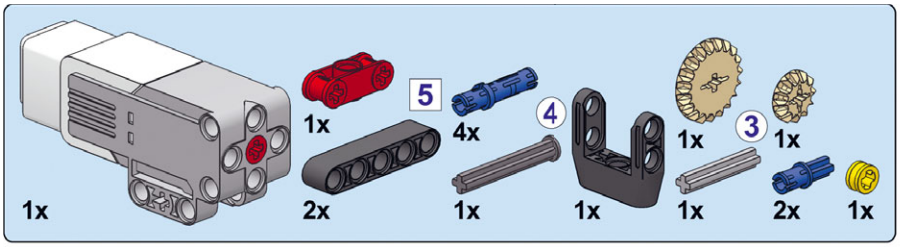
6

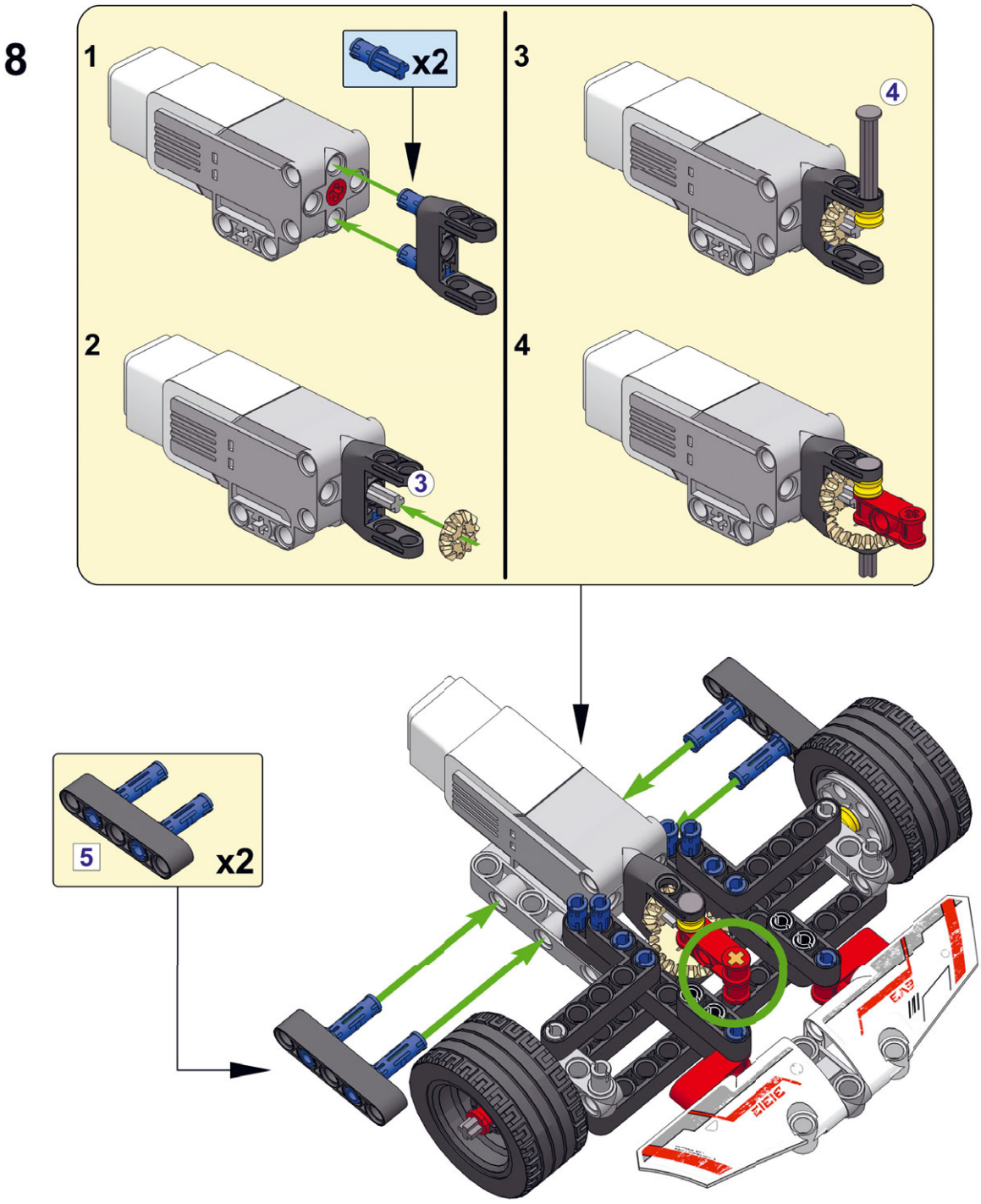


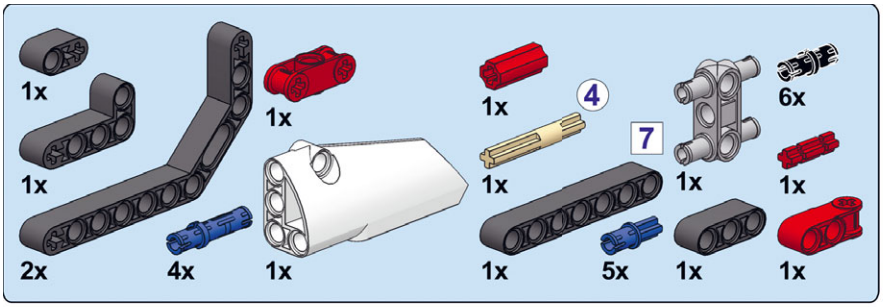
7

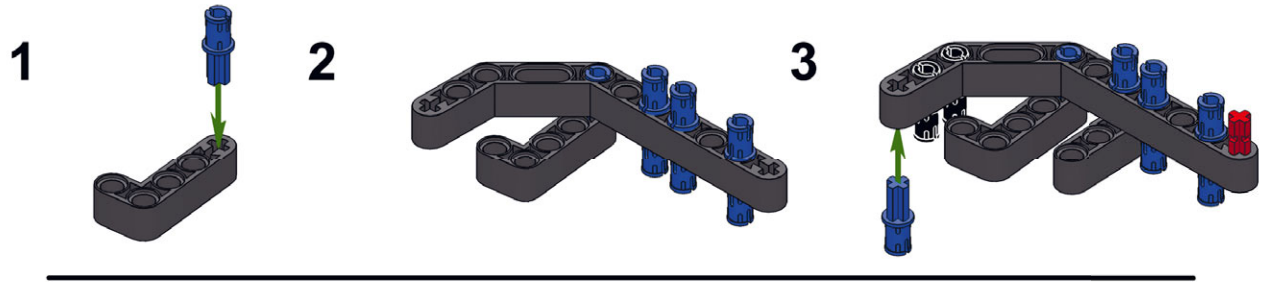




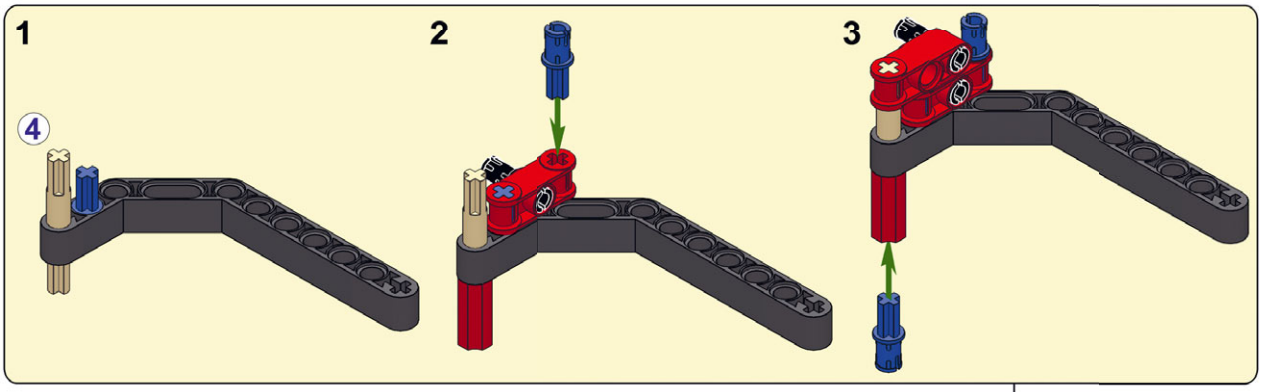

1x
1x
2x
5
4x
1x
4
1x
1x
3
1x
2x
1x


8
1
x2
3
4
2
3
4
5
x2

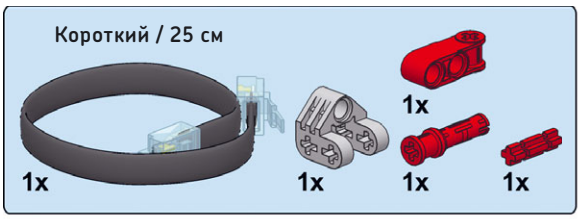

1x
1x
2x
4x
1x
1x
1x
5x
4
7
1x
1x
1x
6x
1x


1
2
3

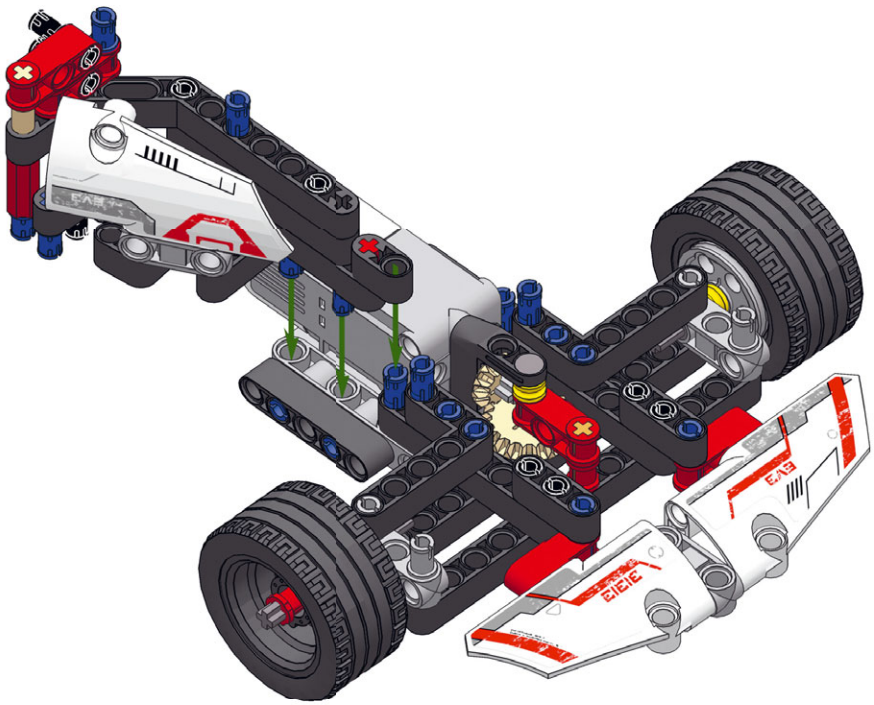

4
5
7


1
4
2
3

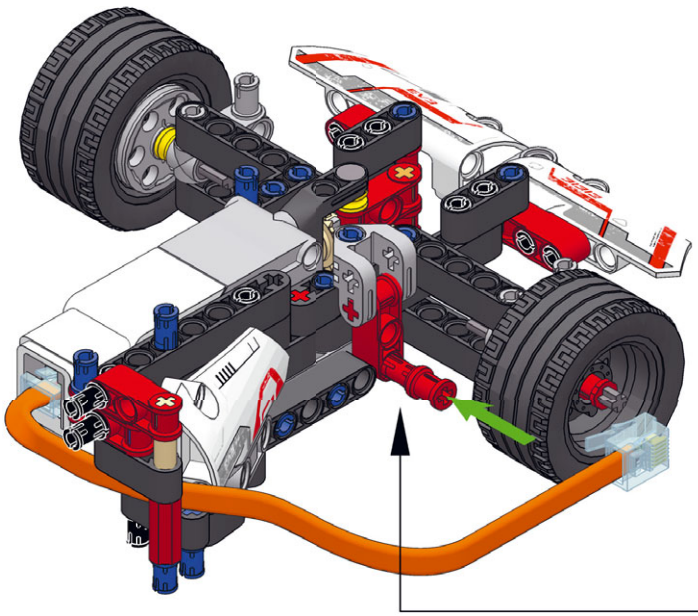

6
7

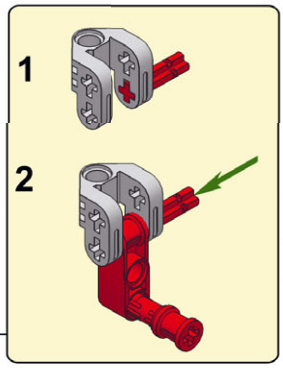


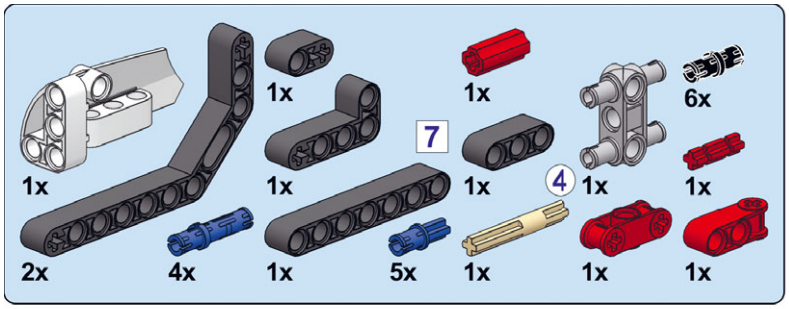
8

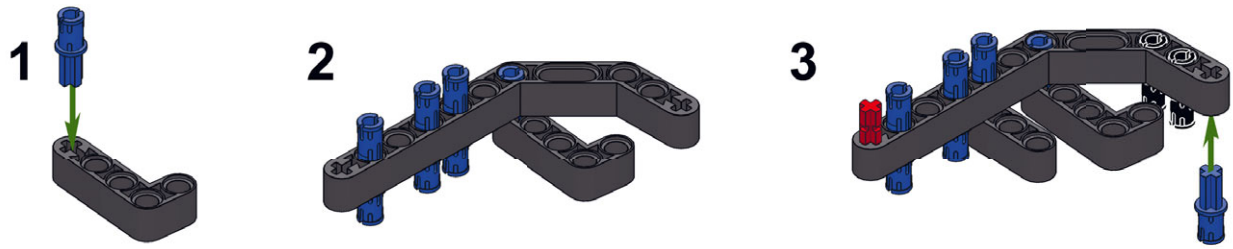


9

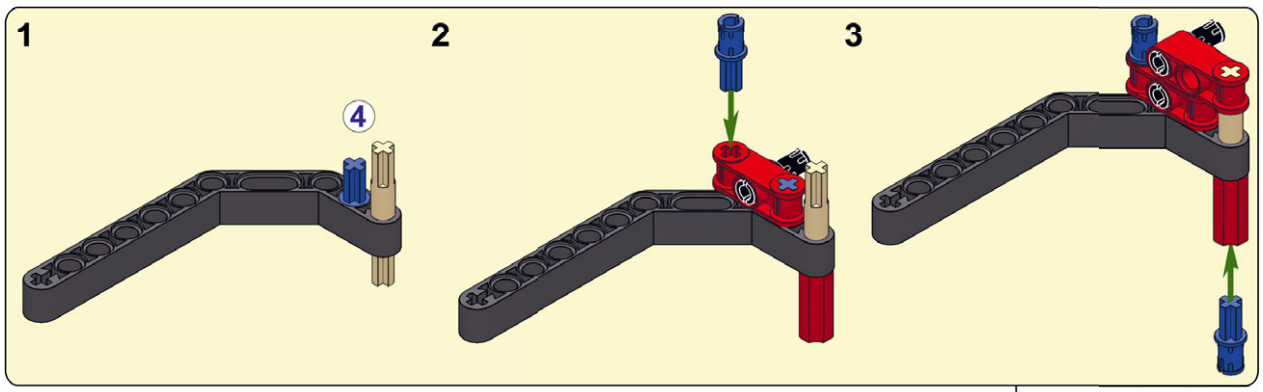


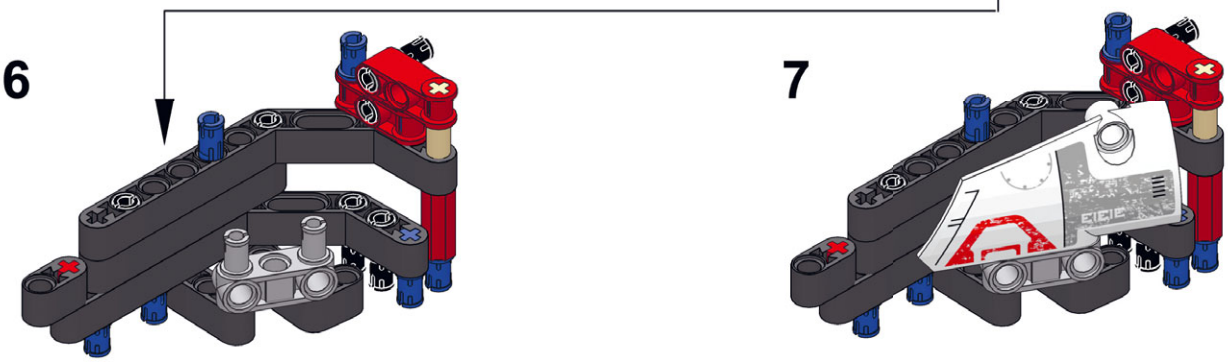


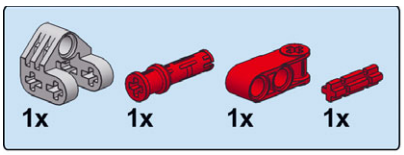

1x
1x
7
1x
6x
1x
4
1x
1x
2x
4x
1x
5x
1x
1x
1x


1
2
3

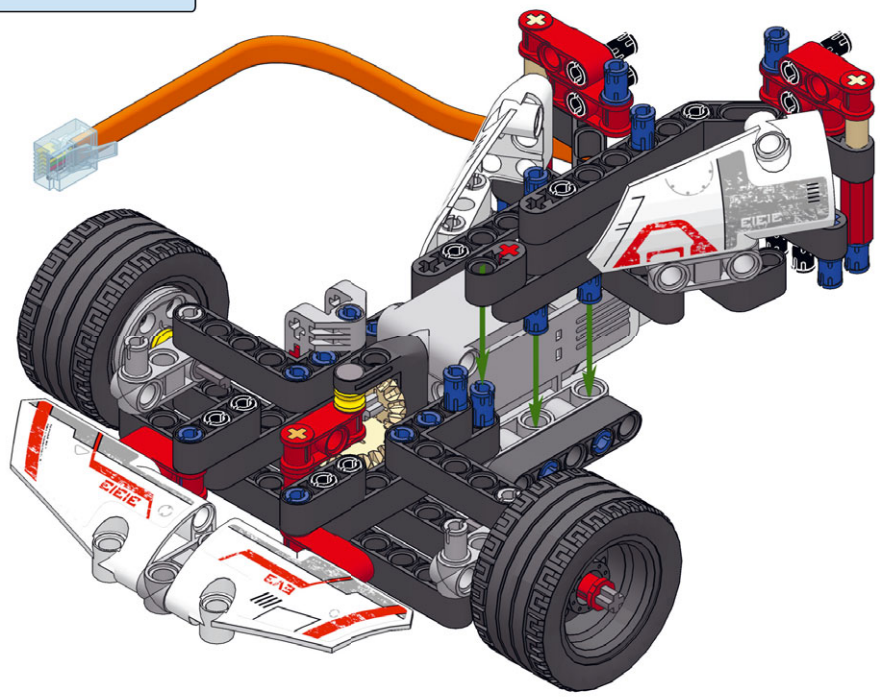

4
5
7


1
4
2
3

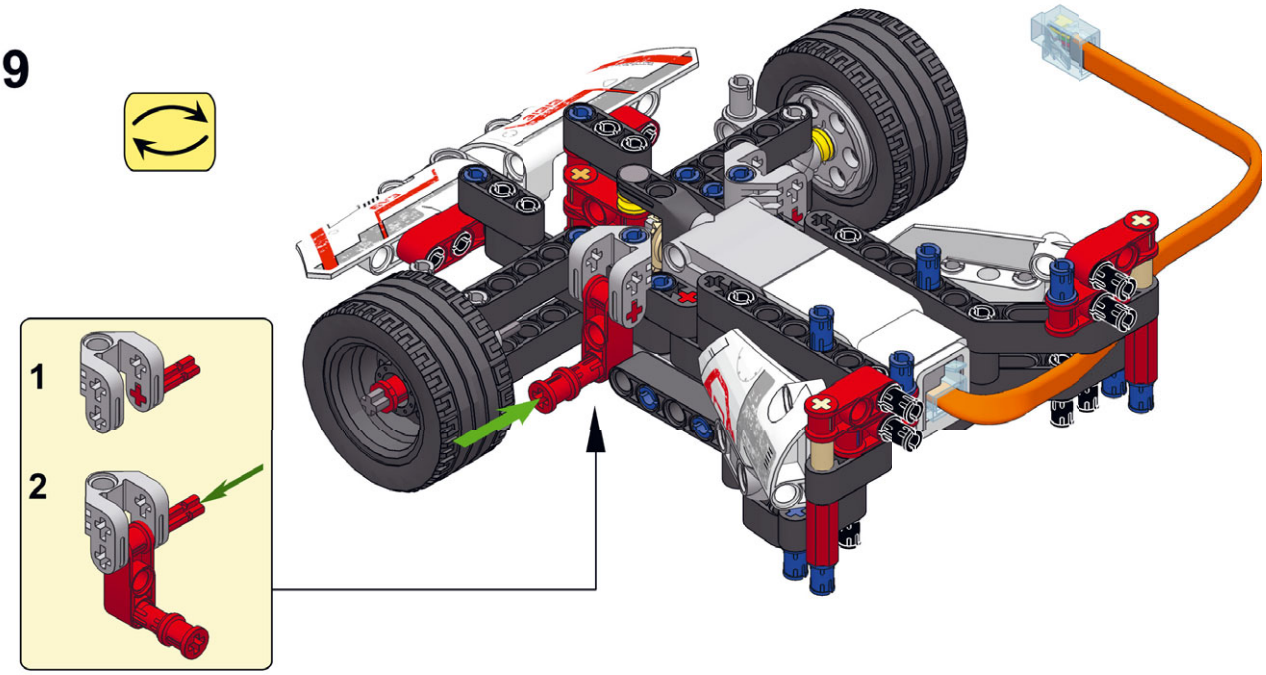

6
7

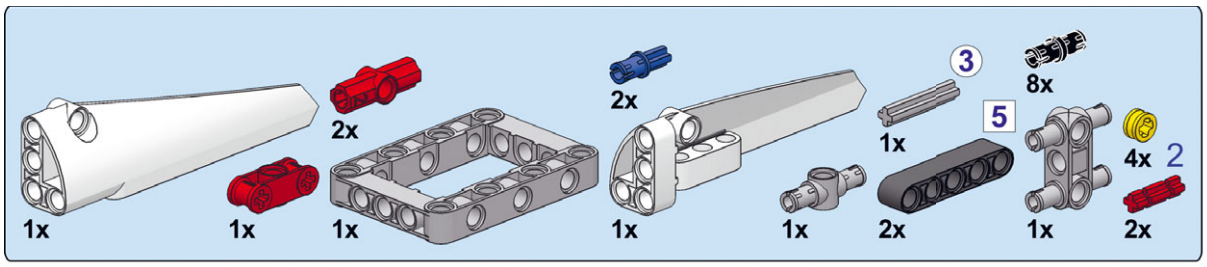


8

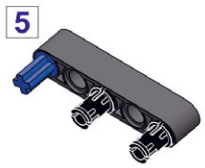


9

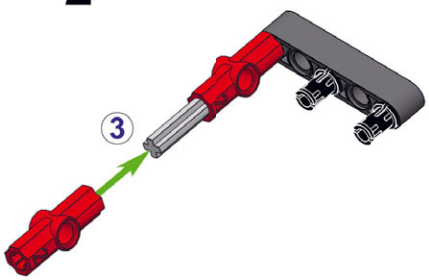




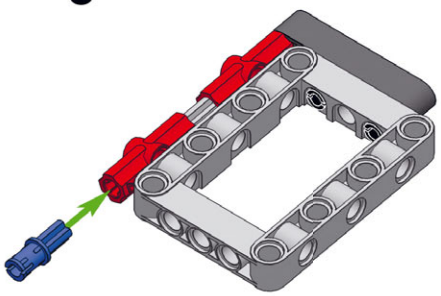
1



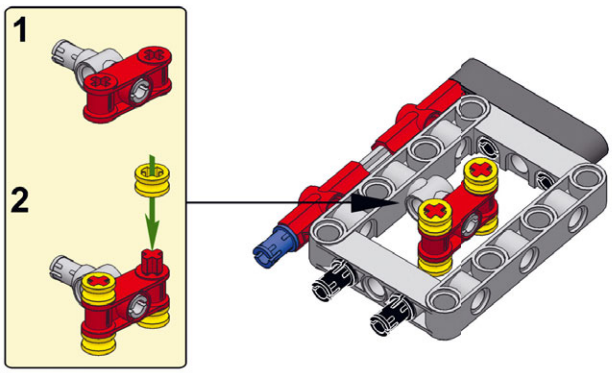
2



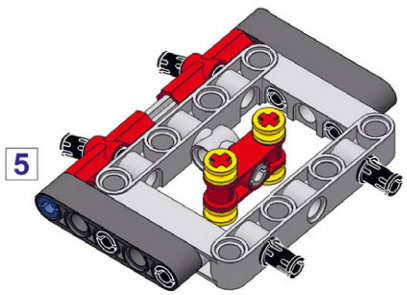
3



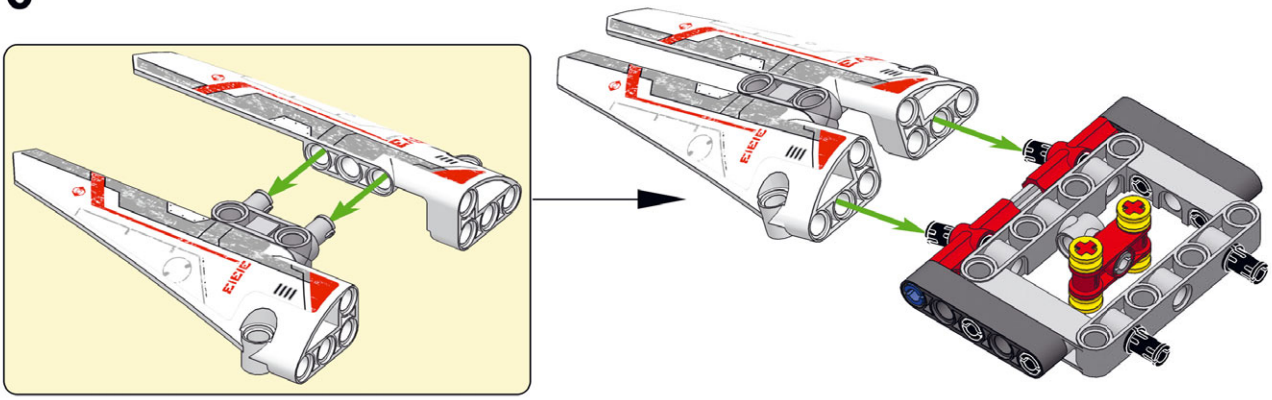
4

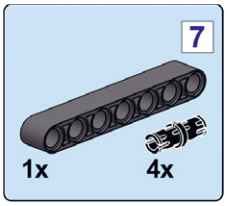


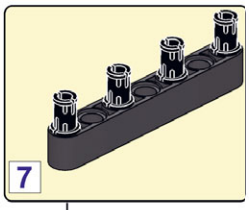
5



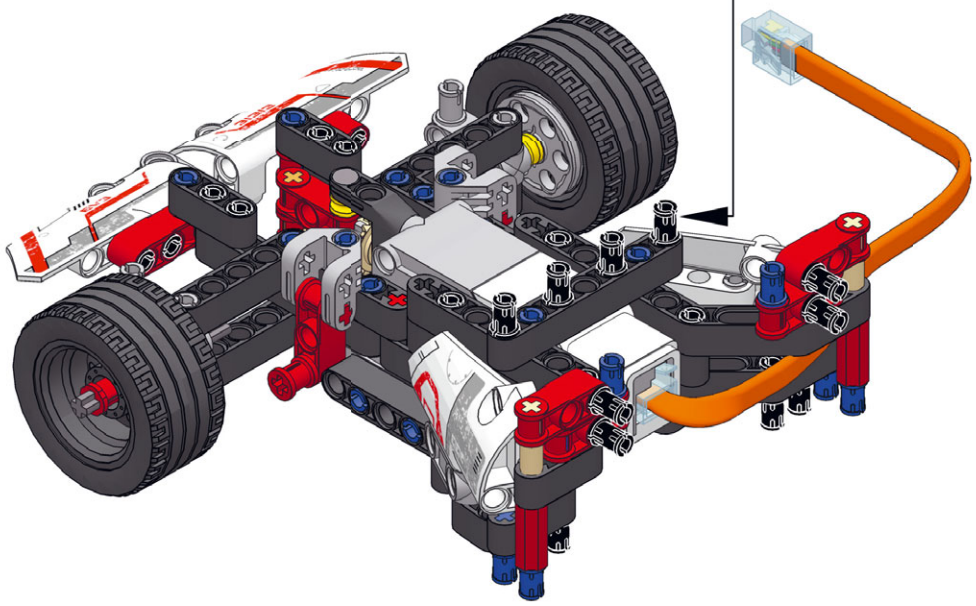
6



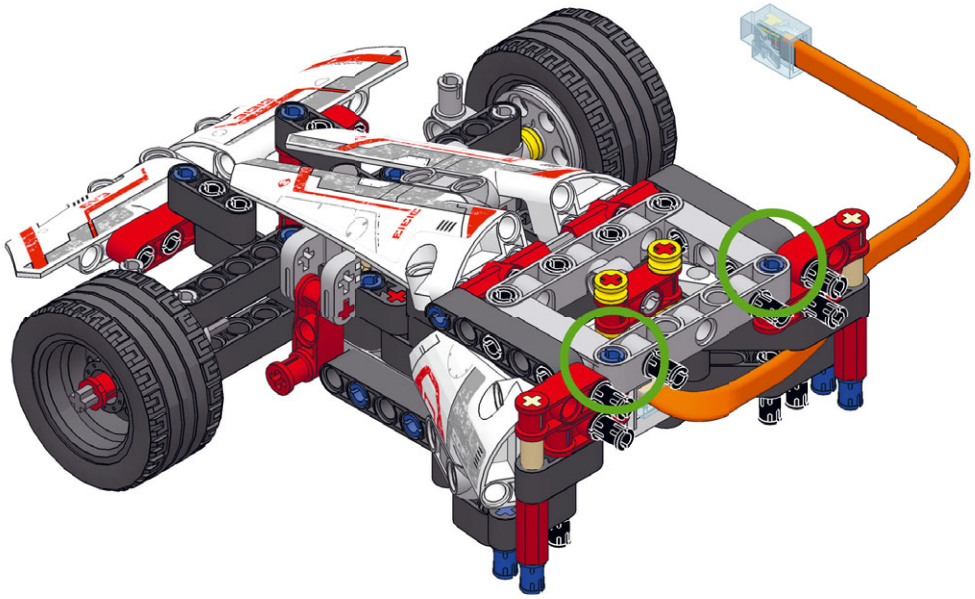


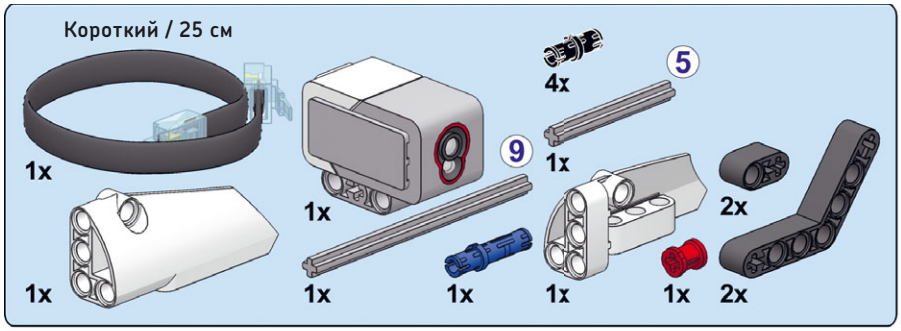


7



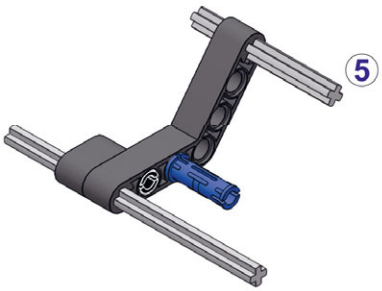
8



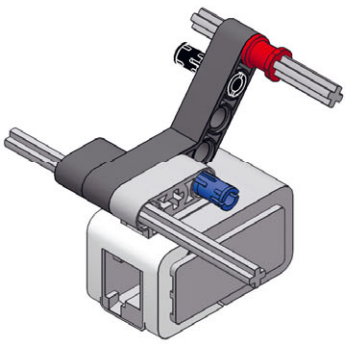


1

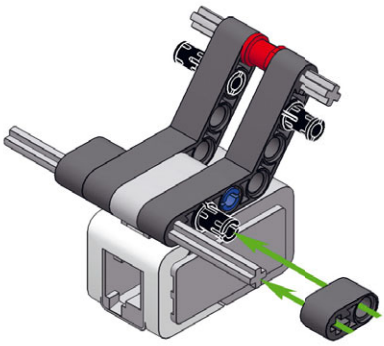
2



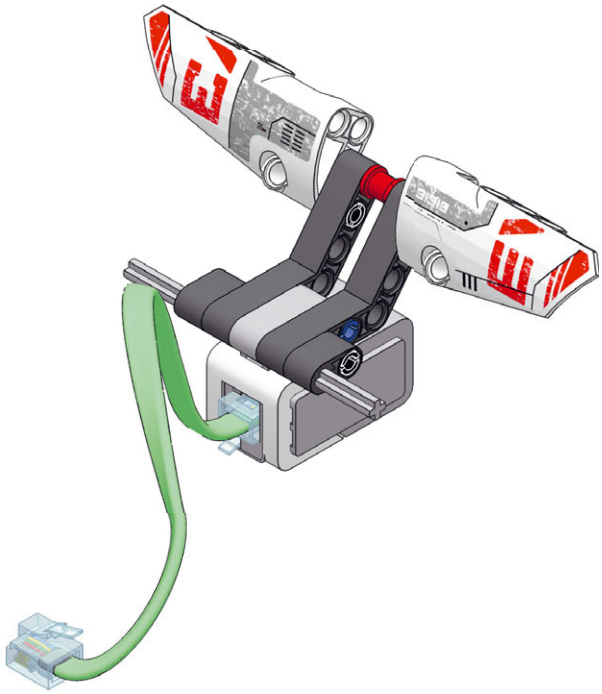
3

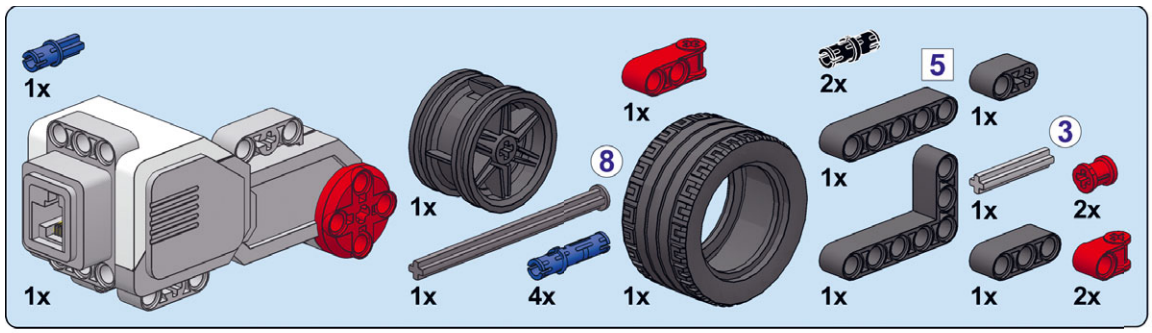


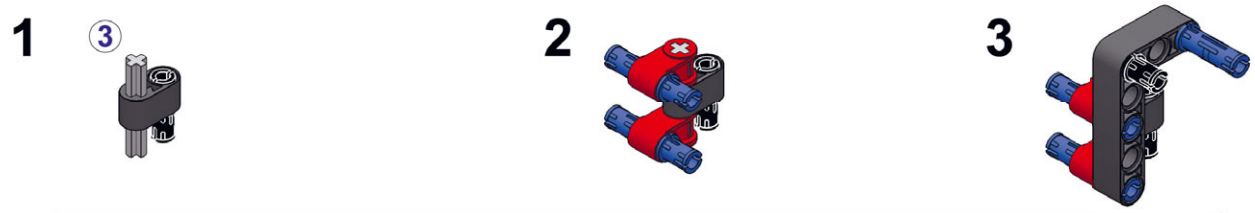
4

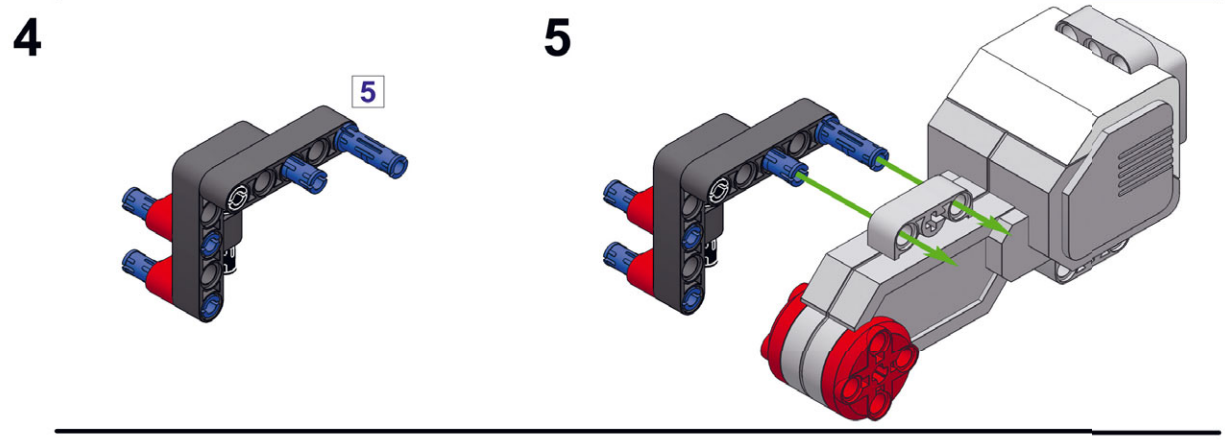


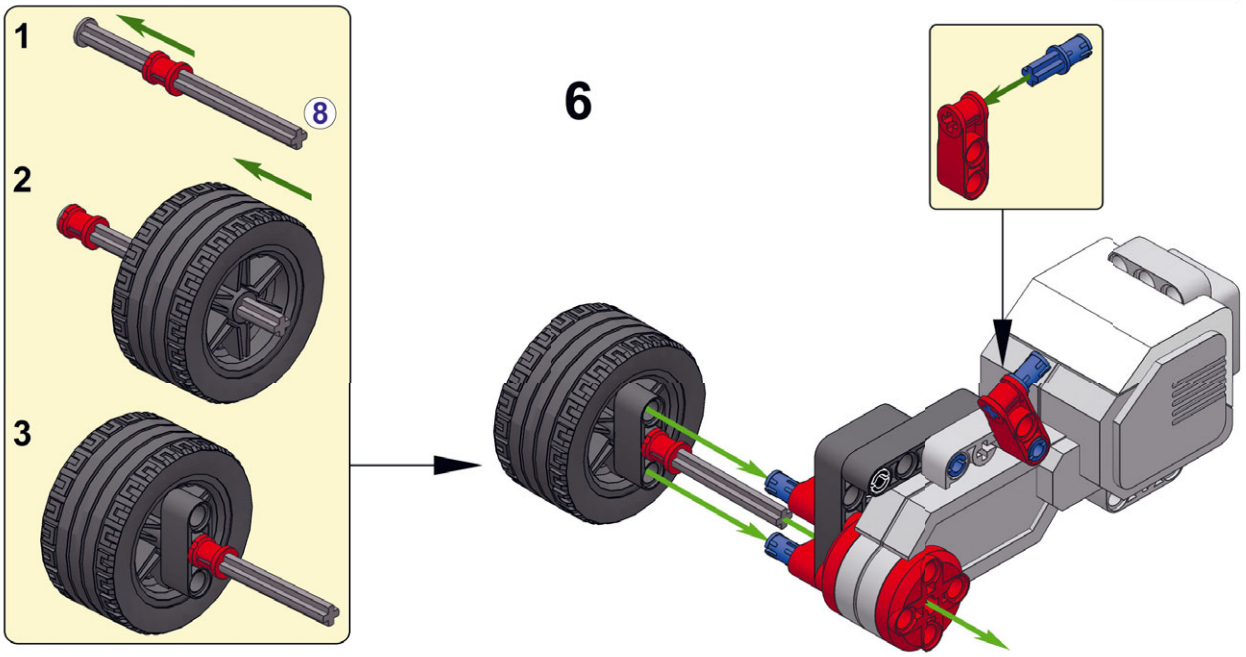
5

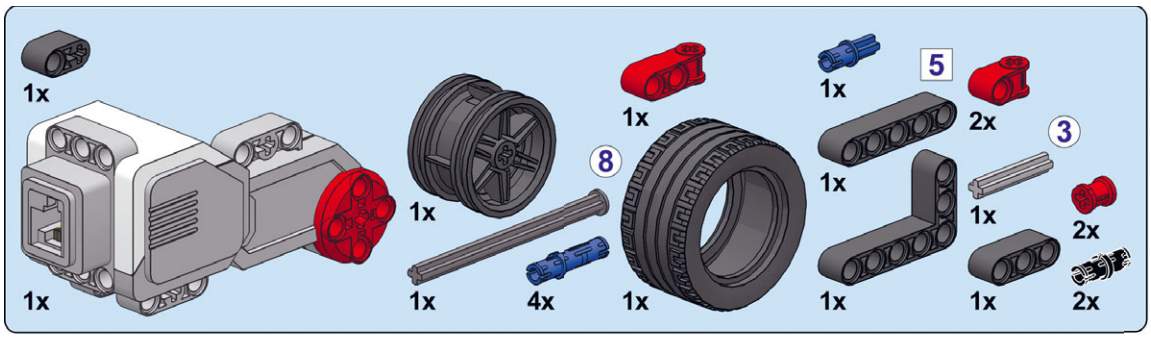


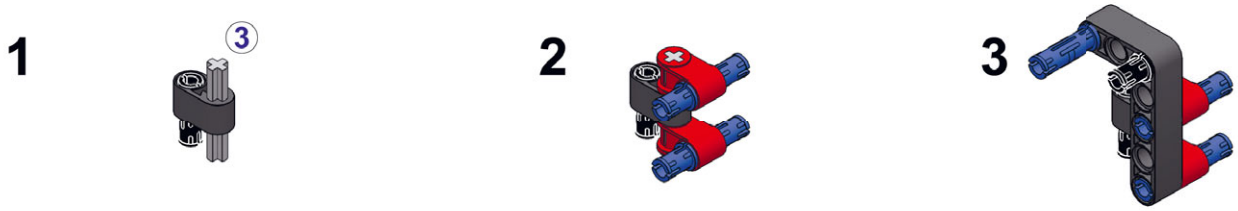

1x
1x
1x
8
1x
4x
1x
2x
5
1x
1x
3
1x
2x
1x
1x
1x
2x

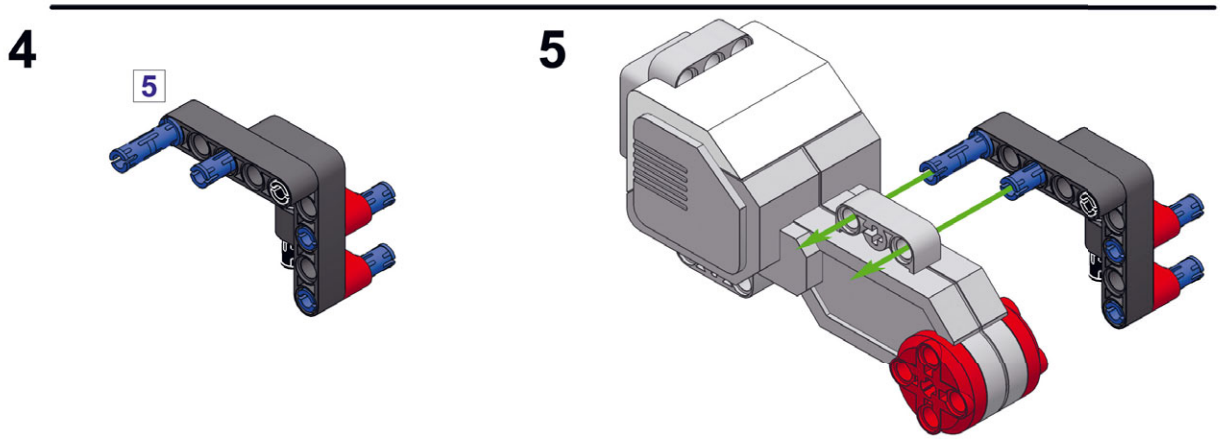

1
3
2
3

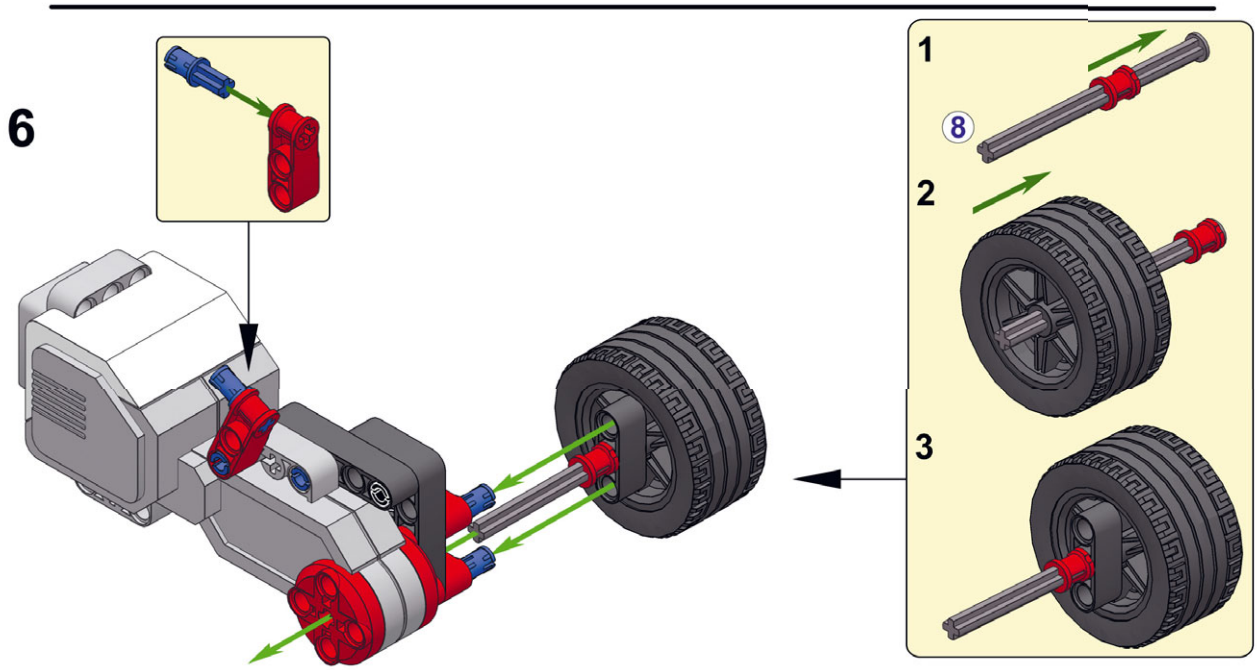

4
5
5

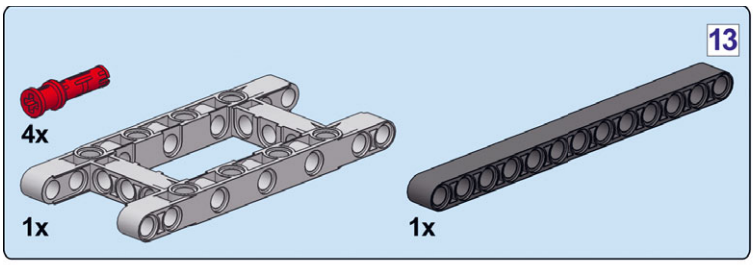

1
8
2
3
6


1x
1x
1x
1x
4x
8
1x
1x
1x
5
2x
1x
3
1x
2x
1x
2x

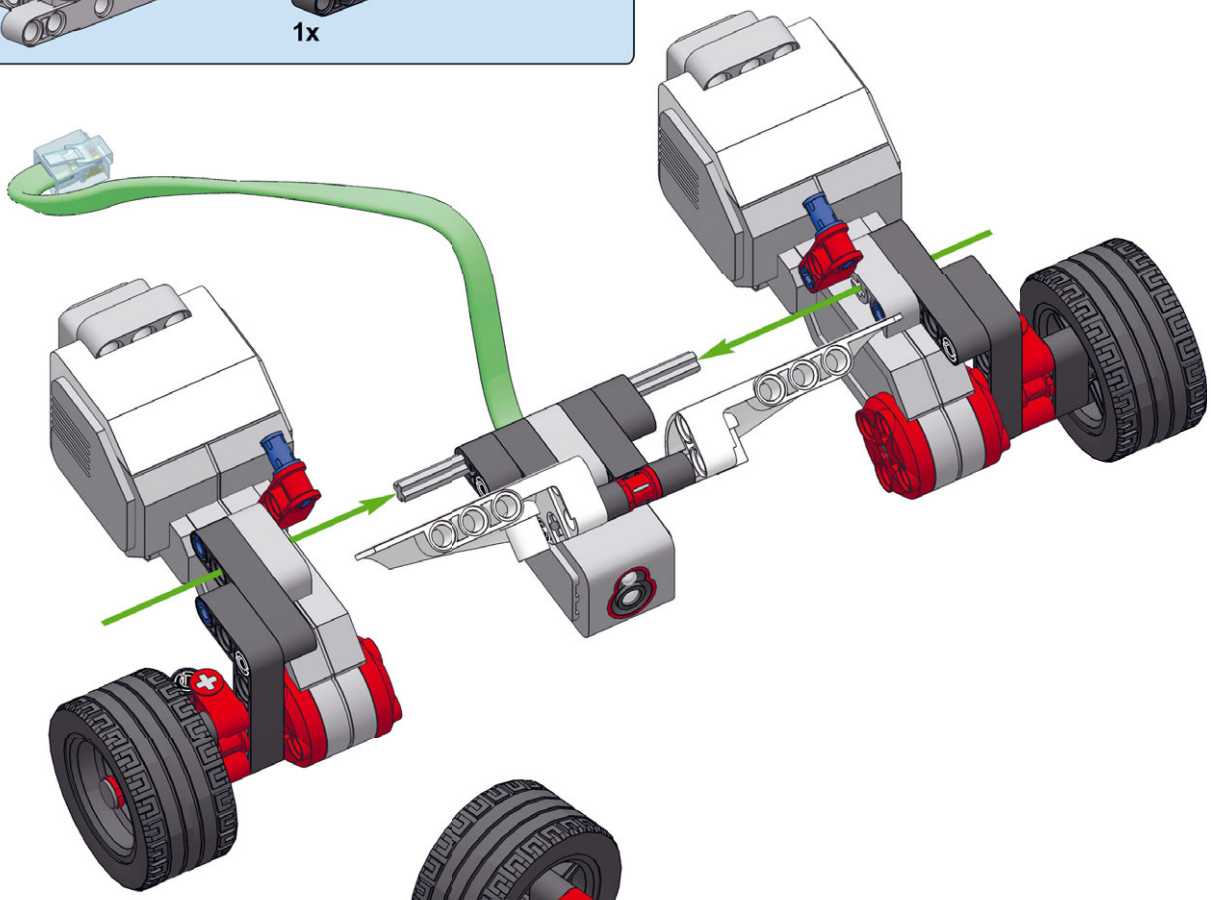

1
3
2
3


4
5
5

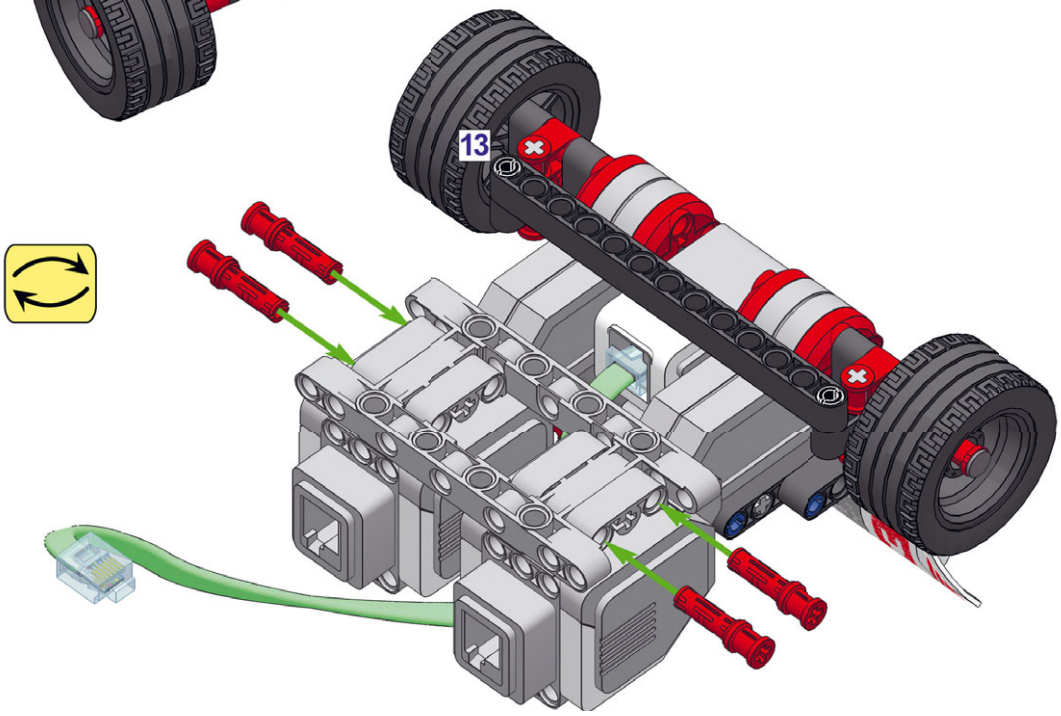

6
1
8
2
3

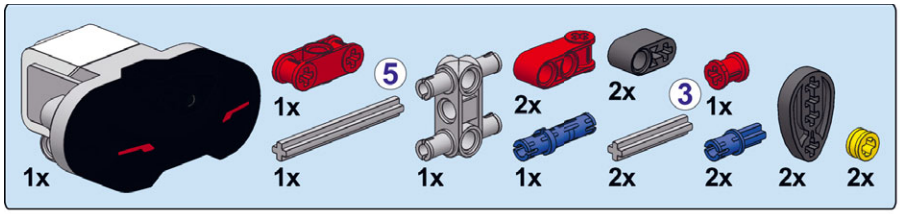


7

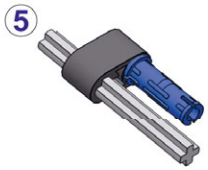


8

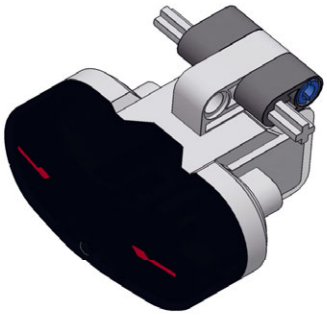




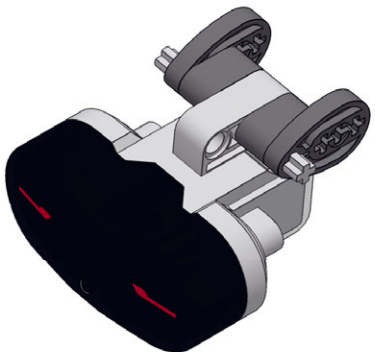
1

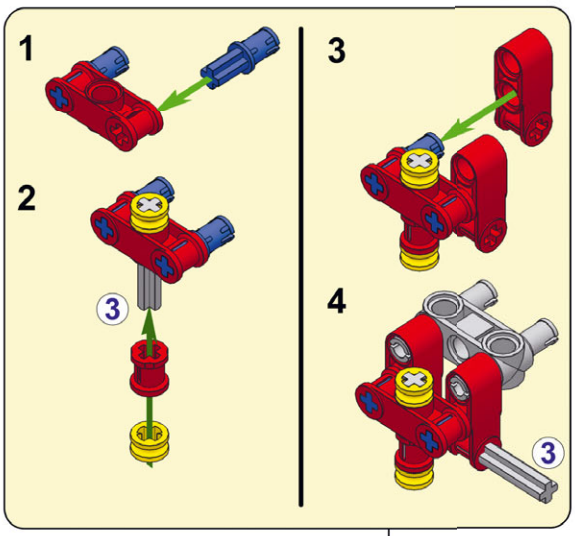


2

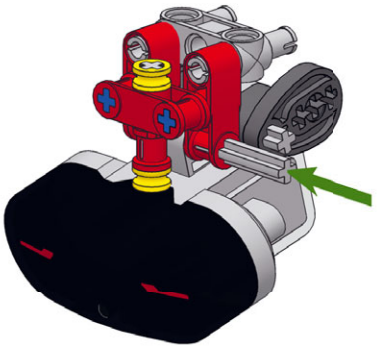


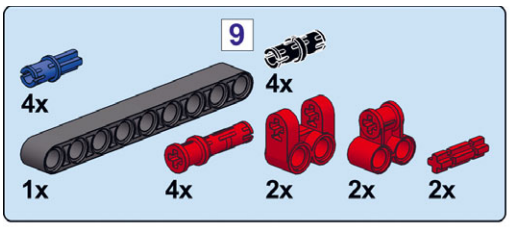
3

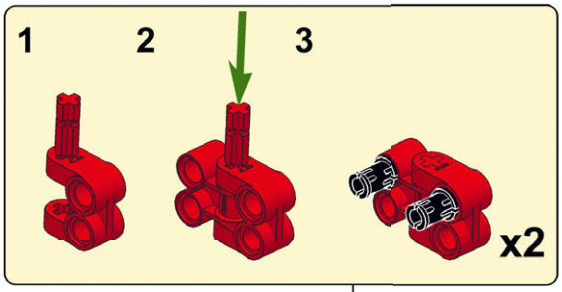




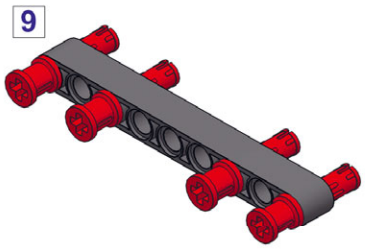
4



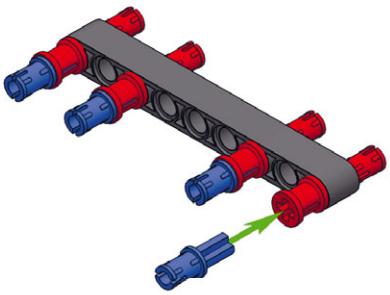




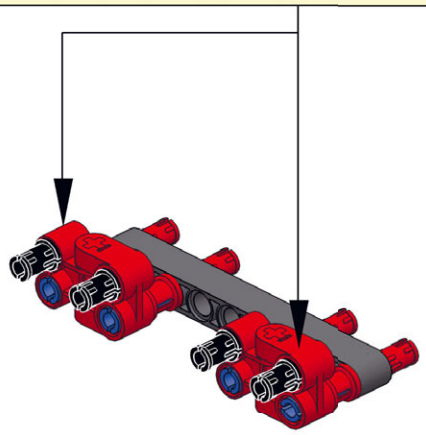
1



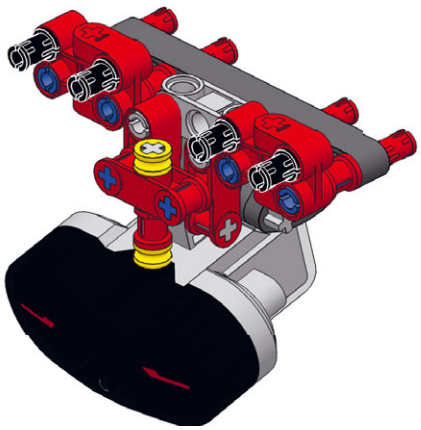
2



3

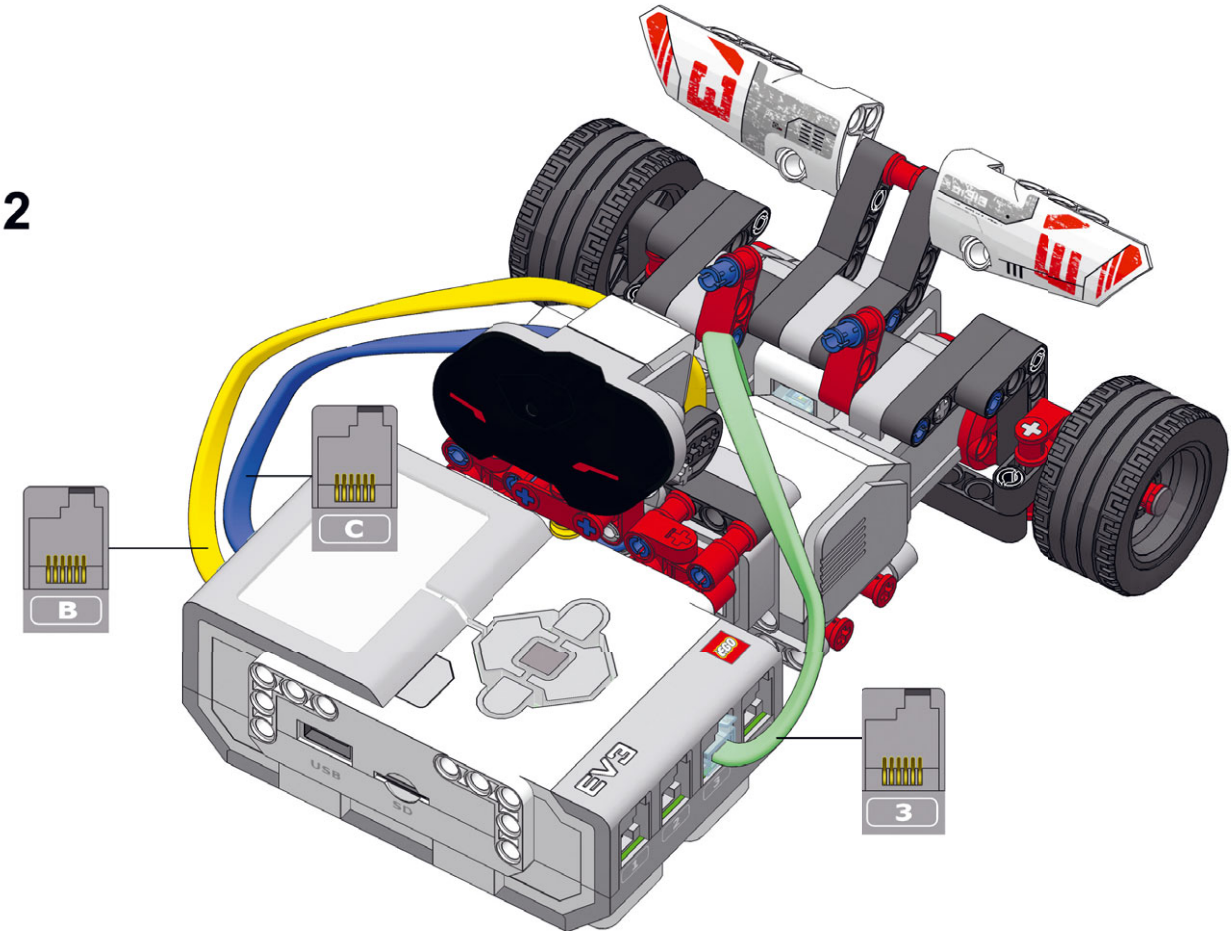


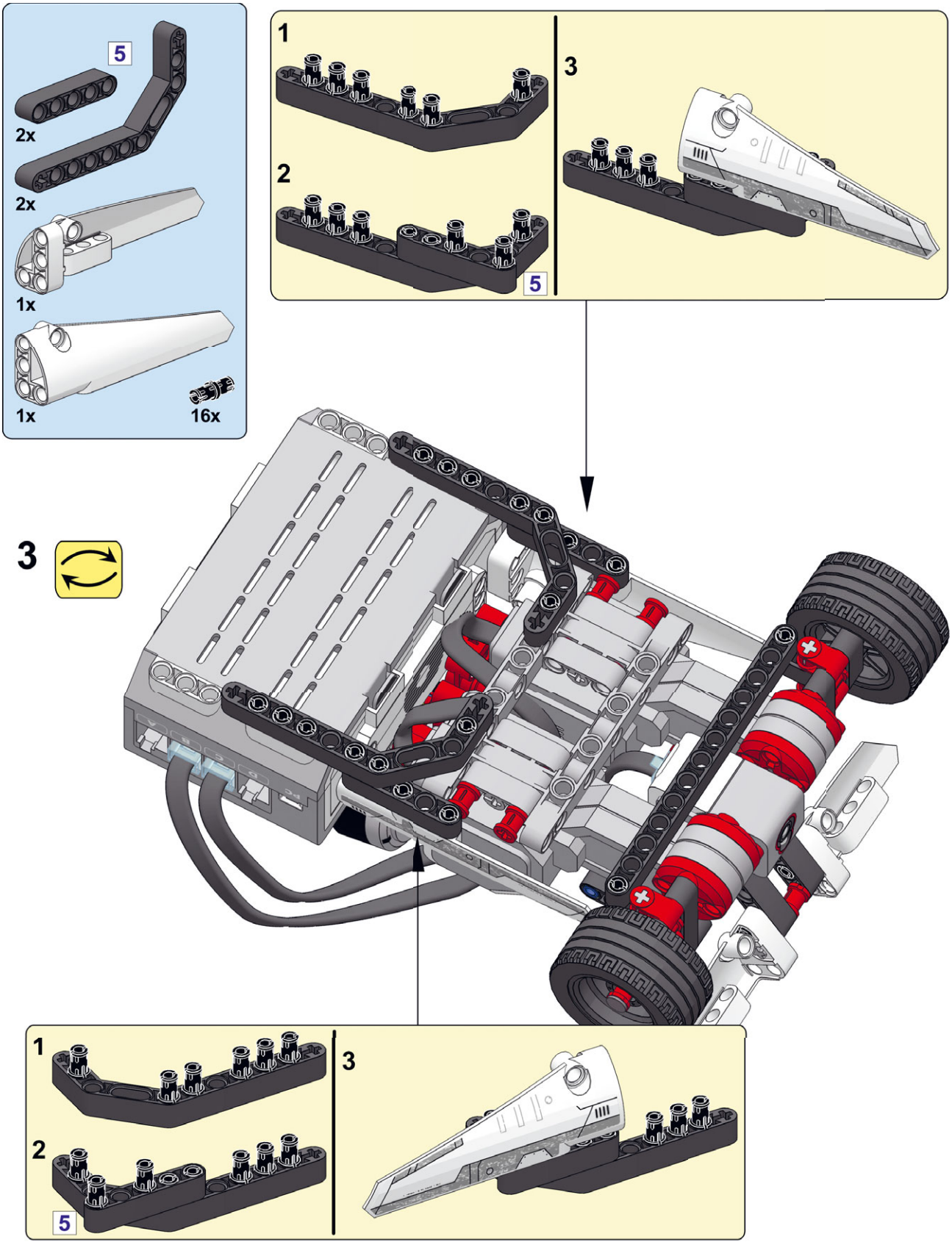
4

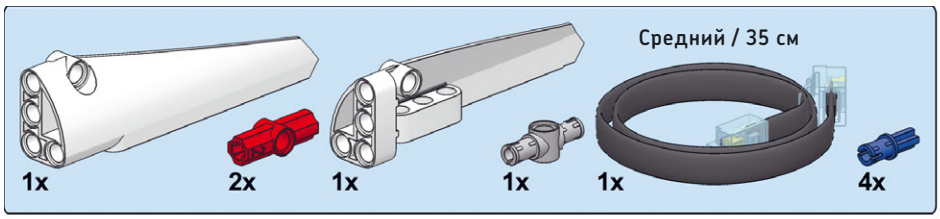


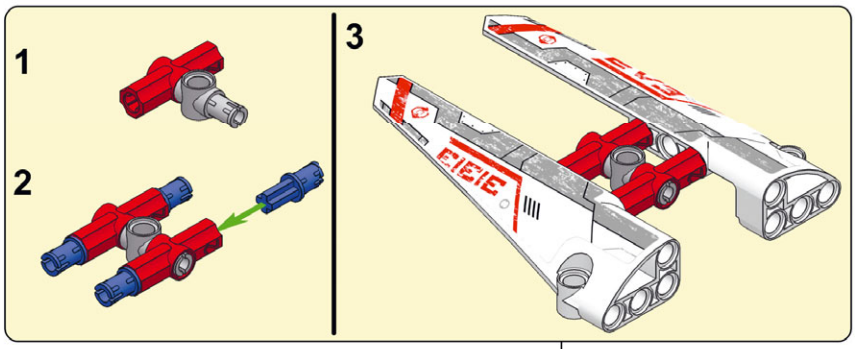
1

2

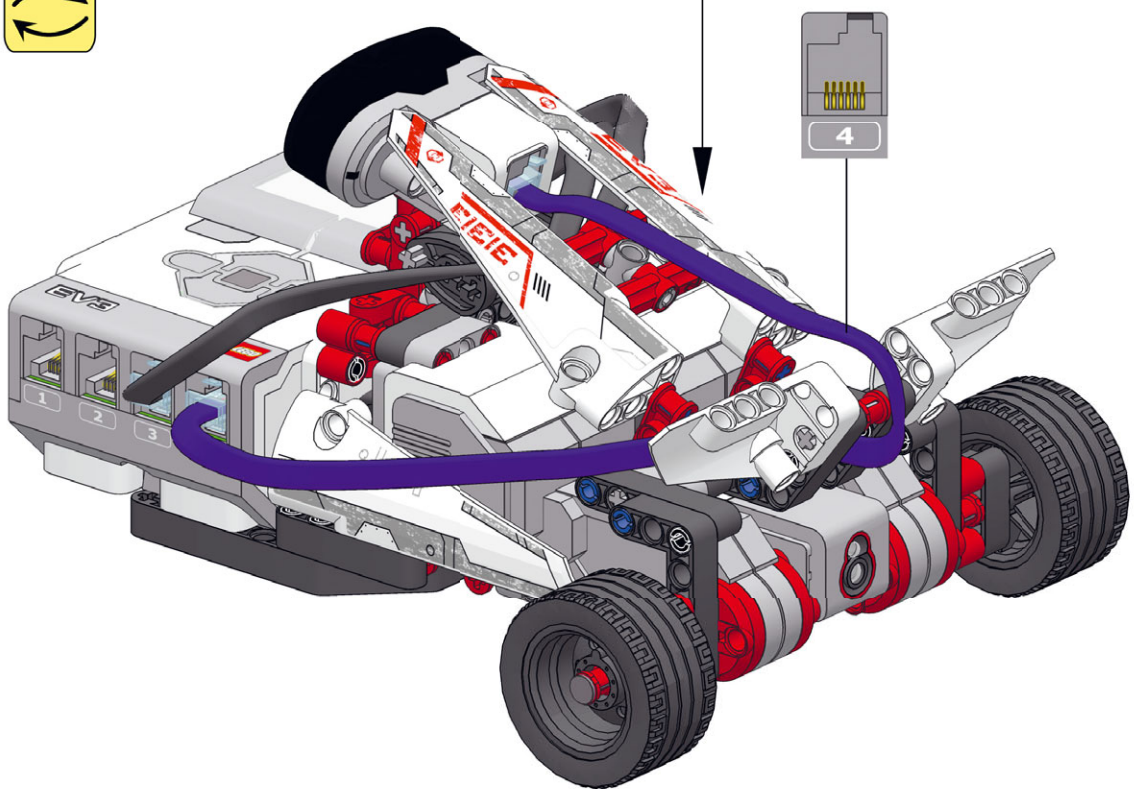


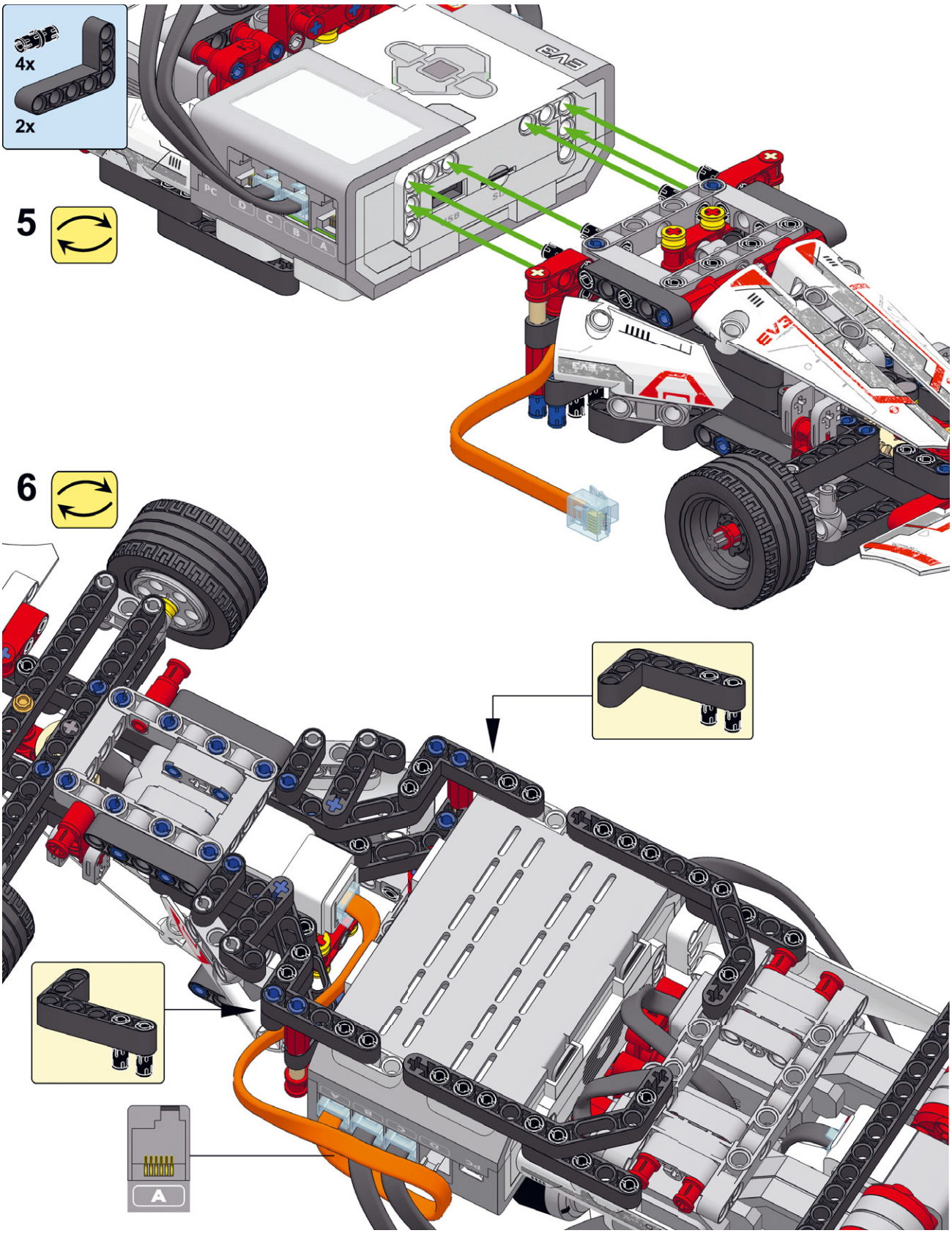

5
2x
2x
1x
1x
16x
1
2
5
3
3
1
2
5
3





4




4x
2x
5
6
A

Создание контейнеров «Мой блок» для рулевого управления

Вскоре мы создадим несколько контейнеров «Мой блок», с помощью которых можно будет легко запрограммировать гоночный автомобиль FORMULA EV3 на повороты в разные стороны. Но сначала выясним, как работает рулевой механизм. Программа будет использовать датчик вращения, находящийся внутри среднего мотора, чтобы точно контролировать ориентацию передних колес, определяющих направление движения робота.

Чтобы увидеть, как она работает, вручную установите передние колеса в центральное положение, как показано на рис. 12.3. Затем перейдите к приложению **Port View** в модуле EV3 и посмотрите, как меняются значения датчика вращения (порт A) при повороте колес влево и вправо вручную. Вы увидите, что значение поворота влево равно примерно 60 градусам, положения по центру 0 градусов, а поворота вправо –60 градусам. Обратите внимание, что эти значения указывают на положение мотора. Колеса поворачиваются вправо и влево на меньшую величину из-за зубчатых колес в механизме рулевого управления.

Как показано на рис. 12.3, чтобы робот повернул влево, средний мотор должен повернуться к точке, которую датчик определяет как 60 градусов. Чтобы повернуть вправо, мотор должен повернуться в противоположную сторону до значения –60 градусов, а чтобы робот ехал прямо — вернуться к значению 0 градусов. Для поворота колес в каждое из этих положений вам понадобятся три контейнера «Мой блок» с именами **Left**, **Right** и **Center**.

Эти блоки будут работать, только если значение датчика при центральном положении колес равно 0 градусов. Так как выравнивать колеса вручную при каждом запуске программы не хочется, нужно создать еще один контейнер «Мой блок» с именем **Reset**, который будет делать это вместо вас. Этот контейнер выравнивает колеса и задает значение датчика вращения 0. Поместите его в начале всех программ робота.

Контейнер «Мой блок» № 1: сброс

В момент запуска программы передние колеса могут находиться в любом положении, поэтому вам придется привести их в необходимое положение прежде, чем выравнивать по центру.

Для этого поверните колеса до упора влево, заставив средний мотор вращаться вперед, пока тот не заглохнет.

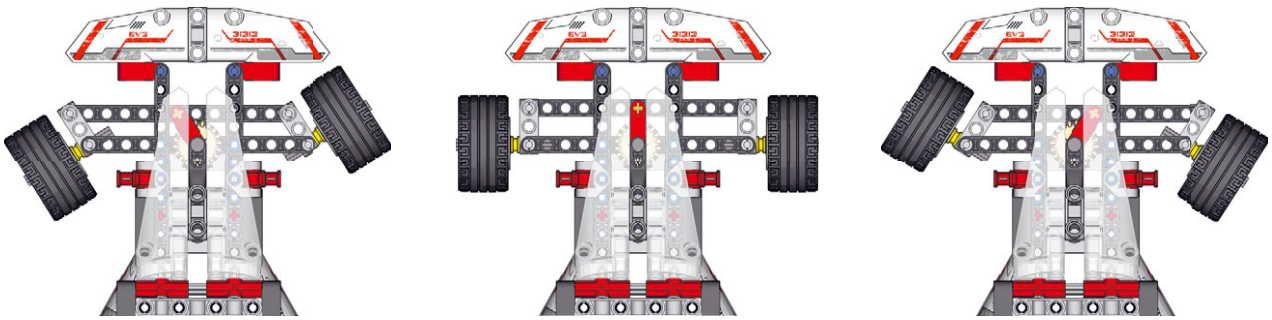
Положение заглохшего мотора должно быть 78 градусов от центра, а значит, нужно повернуть колеса до упора влево, а затем с помощью среднего мотора повернуть их на 78 градусов в обратном направлении, чтобы выровнять по центру. Как только передние колеса будут выровнены должным образом, установите значение датчика вращения равным нулю.

С этого момента значения датчика вращения близкие к нулю будут указывать, что передние колеса выровнены по центру.

Начните с создания нового проекта под названием FORMULA EV3 для всех программ гоночного автомобиля. Затем создайте контейнер «Мой блок» с именем **Reset**, как показано на рис. 12.4.

Контейнер «Мой блок» № 2: влево

Следующий контейнер «Мой блок» заставляет передние колеса вращаться влево, поворачивая средний мотор вперед, пока значение датчика вращения не составит 60 градусов, но только в том случае, если колеса еще не направлены влево. Чтобы определить, направлены ли колеса влево, вам понадобится блок **Переключатель** (Switch). Если это так, мотор выключится, если нет — будет вращаться вперед, пока не достигнет положения 60 градусов.



Влево: 60 градусов | **По центру: 0 градусов** | **Вправо: –60 градусов**

*Рис. 12.3. Значения датчика вращения для различных положений передних колес. Убедитесь, что колеса находятся в центральном положении, прежде чем запускать приложение **Port View** для просмотра показателей датчика*

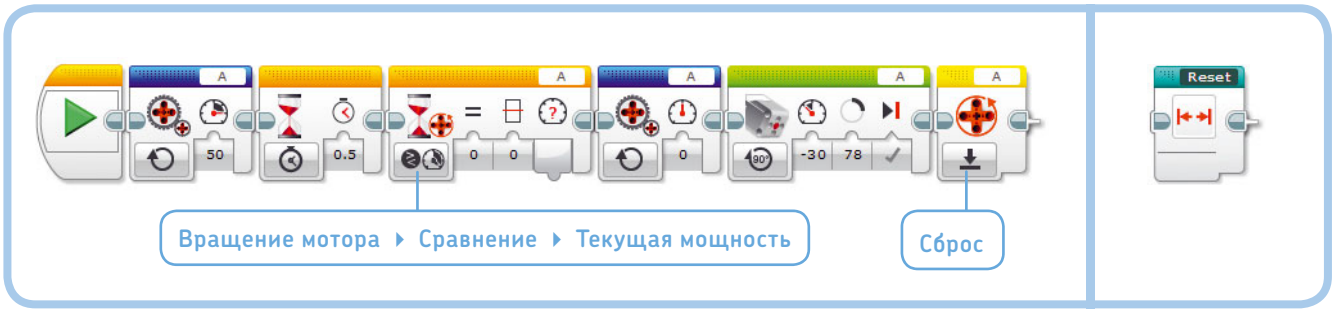


Рис. 12.4. Контейнер «Мой блок» с именем Reset перемещает рулевое колесо по центру и устанавливает значение датчика вращения равным 0. Завершенный контейнер «Мой блок» показан справа

Это работает независимо от того, выровнены колеса в данный момент по центру или повернуты вправо, потому что мотор будет вращаться до тех пор, пока не достигнет нужного положения, а не на фиксированное количество градусов. Параметру **Тормозить в конце** (Brake at End) должно быть присвоено значение **Истина** (True), блокирующее мотор в этом положении до следующего движения. Разместите и настройте блоки, как показано на рис. 12.5, а затем объедините их в контейнер «Мой блок» с именем **Left**.

ПРИМЕЧАНИЕ Блоки Переключатель (Switch) и Ожидание (Wait) в контейнерах «Мой блок» с именами Left, Right и Center работают в режиме Вращение мотора ▸ Сравнение ▸ Градусы (Motor Rotation ▸ Compare ▸ Degrees).

Контейнер «Мой блок» № 3: вправо

Контейнер «Мой блок» с именем **Right** выполняет те же действия, но в другую сторону. Сначала он определяет, не повернуты ли колеса вправо. Если повернуты, то мотор выключается, если нет, то мотор вращается назад, пока не окажется под углом –60 градусов. Для этого средний мотор переключается на отрицательную скорость (–30%), блок **Ожидание** (Wait) приостанавливает выполнение программы, пока значение датчика вращения не достигнет –60 градусов, а затем мотор выключается.

Соберите контейнер «Мой блок» с именем **Right**, как показано на рис. 12.6.

Контейнер «Мой блок» № 4: по центру

Передние колеса идеально выровнены по центру, если значение датчика равно нулю, но в этом блоке любое положение колес в диапазоне от –5 до 5 градусов считается достаточно близким к центру. Если уменьшить диапазон, мотор будет «проскакивать» центральное положение при каждой попытке выравнивания.

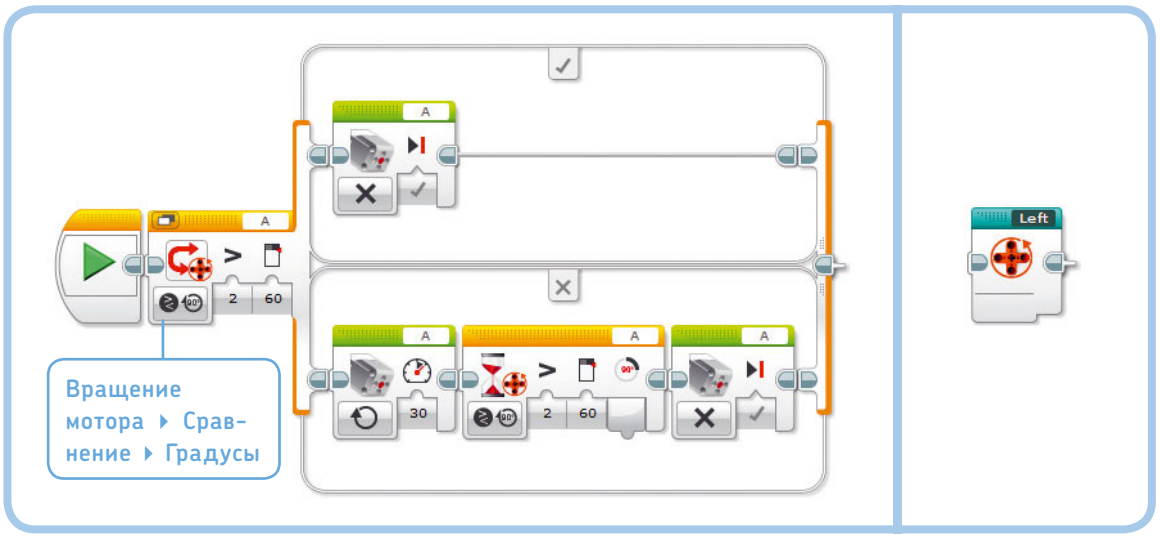


*Рис. 12.5. Контейнер «Мой блок» с именем **Left** заставляет передние колеса поворачиваться налево. Завершенный контейнер «Мой блок» показан справа*

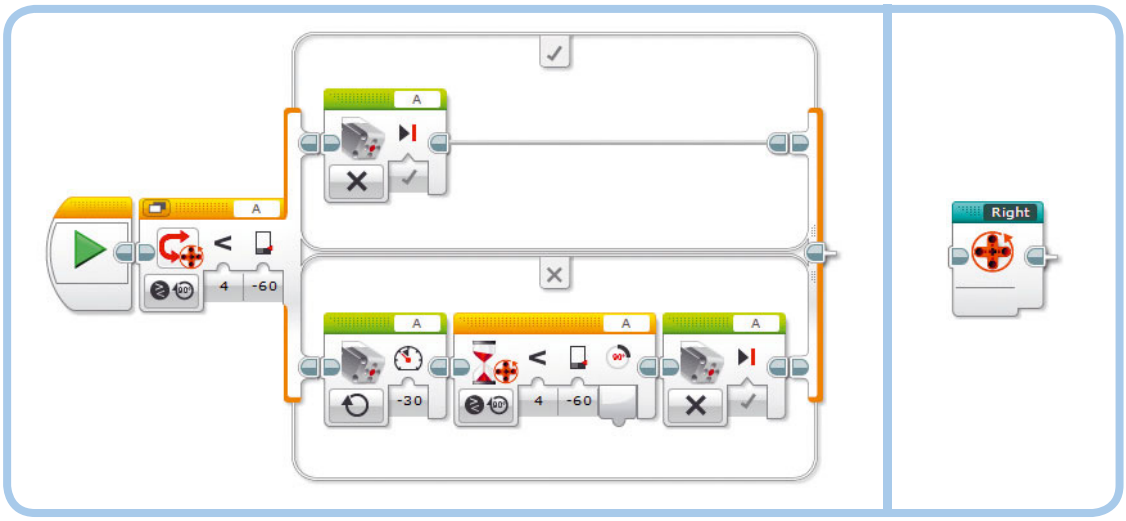


*Рис. 12.6. Контейнер «Мой блок» с именем **Right** заставляет передние колеса поворачиваться направо. Завершенный контейнер «Мой блок» показан справа*

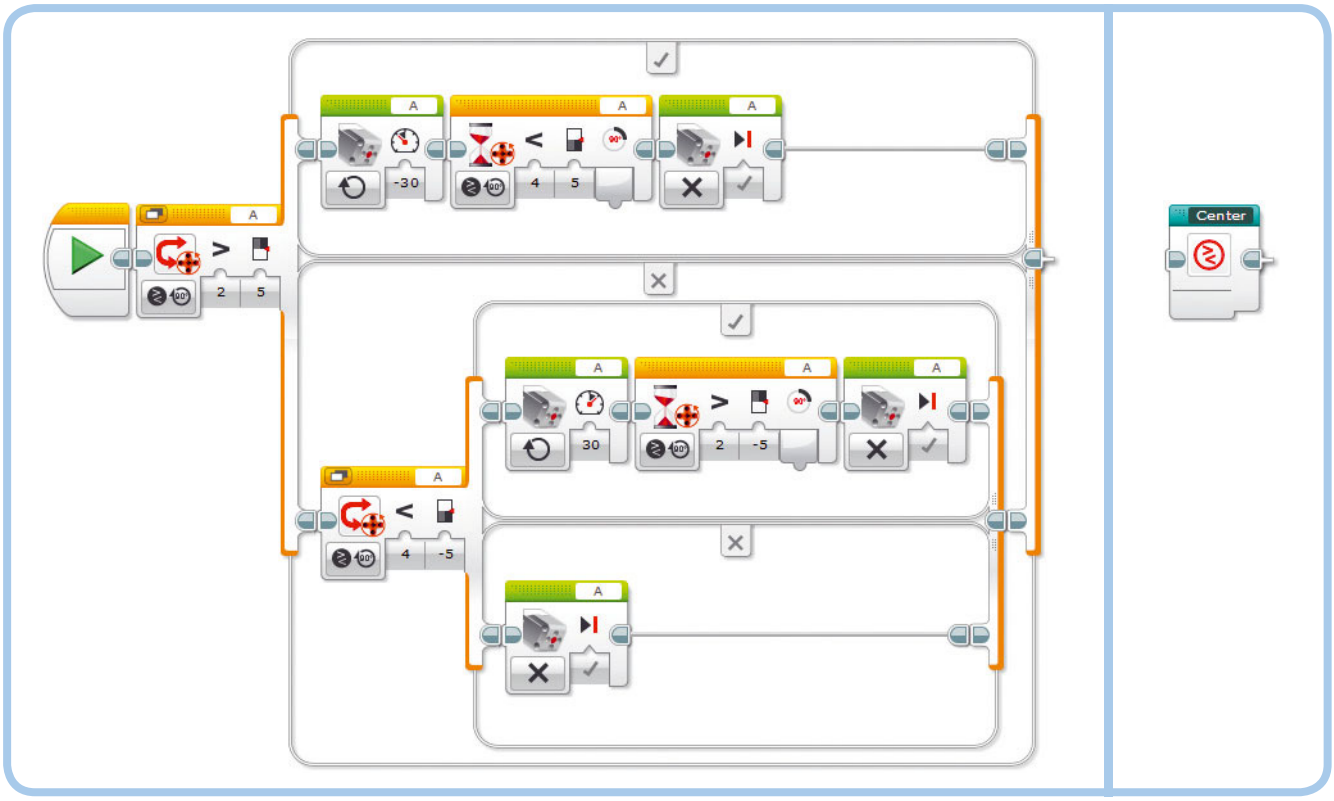


*Рис. 12.7. Контейнер «Мой блок» с именем **Center** выравнивает передние колеса по центру независимо от их текущего положения. Завершенный контейнер «Мой блок» показан справа*

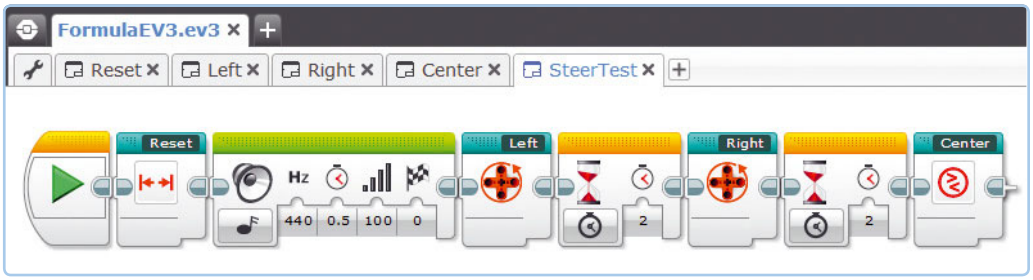


Рис. 12.8. С помощью программы SteerTest протестируйте контейнеры «Мой блок». Запустите программу несколько раз, устанавливая колеса в разные исходные положения

Если мотор уже находится по центру, то при активации этого контейнера он выключится. Если колеса повернуты влево (значение датчика превышает 5 градусов), они будут поворачиваться вправо, пока не окажутся в положении по центру (отклонение менее 5 градусов).

Если колеса повернуты вправо (значение менее –5 градусов), они будут поворачиваться влево, пока не окажутся по центру (значение больше –5 градусов). Для определения положения мотора используйте два блока **Переключатель** (Switch). Пользуясь рис. 12.7, создайте контейнер «Мой блок» с именем **Center**.

Проверка контейнеров «Мой блок»

Перед тем как собирать программу целиком, нужно протестировать контейнеры «Мой блок», чтобы выяснить, правильно ли они работают. Скомпонуйте и запустите программу SteerTest, показанную на рис. 12.8. Передние колеса должны автоматически выровняться в одну линию с задними колесами при запуске контейнера «Мой блок» с именем **Reset**. Затем они должны повернуться влево, вправо и снова выровняться по центру.

Создание программы дистанционного управления

Теперь, когда вы сделали контейнеры «Мой блок» для рулевого управления, будет несложно создать программу дистанционного управления, применив методы, изученные в главе 8. Ваша следующая программа заставит автомобиль ездить

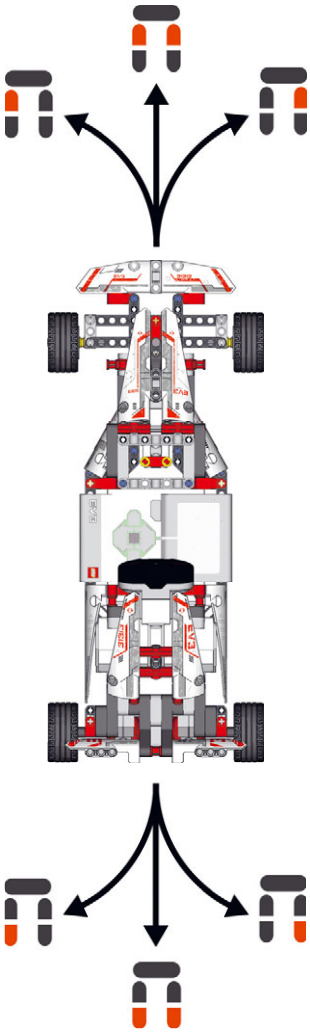


Рис. 12.9. Комбинации кнопок для управления движением гоночного автомобиля FORMULA EV3 в любом направлении

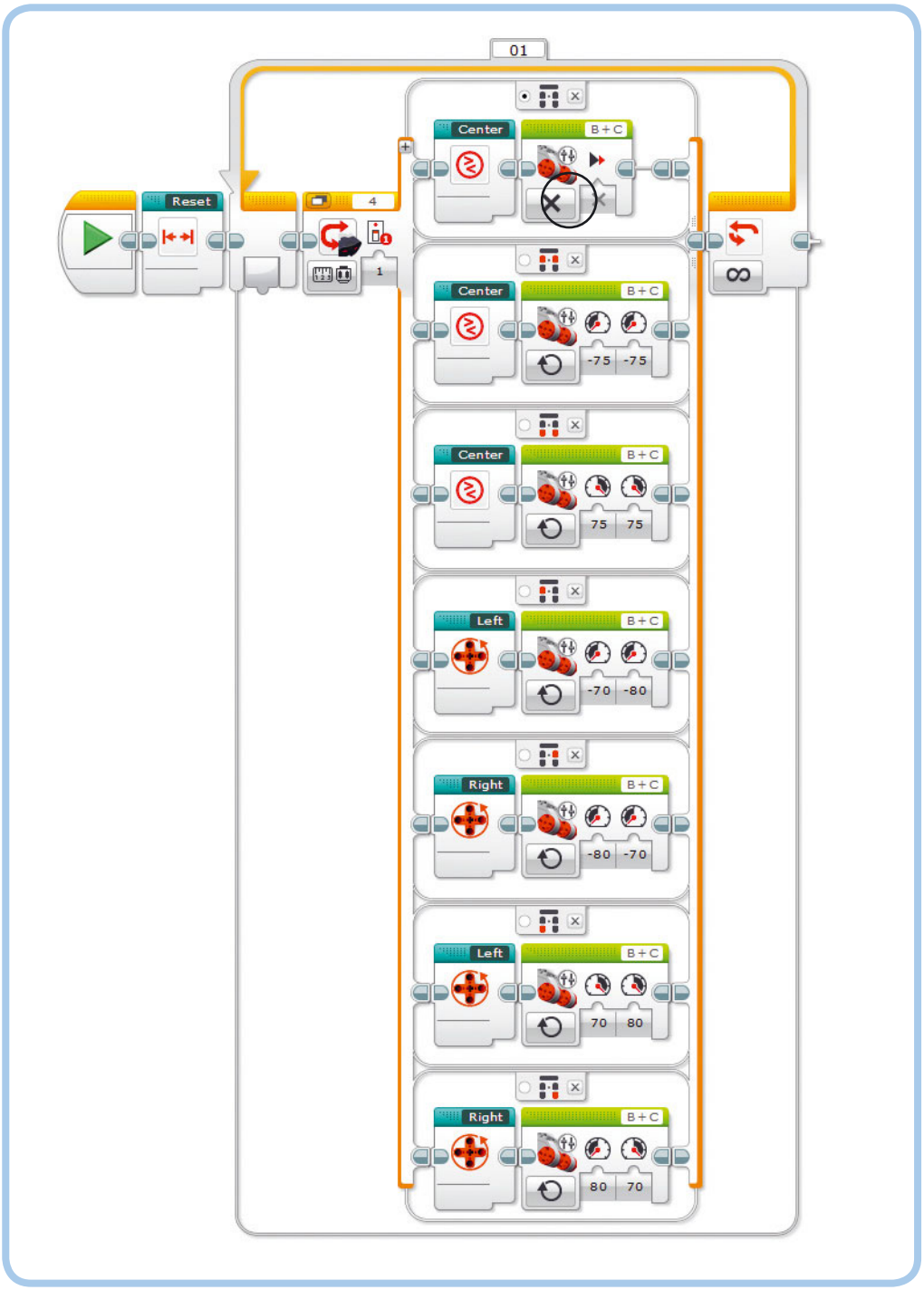


Рис. 12.10. Программа RemoteControl. Не забудьте добавить контейнер «Мой блок» с именем ***Reset*** *в начало программы. Если на инфракрасном маяке не нажата ни одна кнопка (заданный по умолчанию вариант), робот выравнивает передние колеса по центру и отключает питание задних колес*

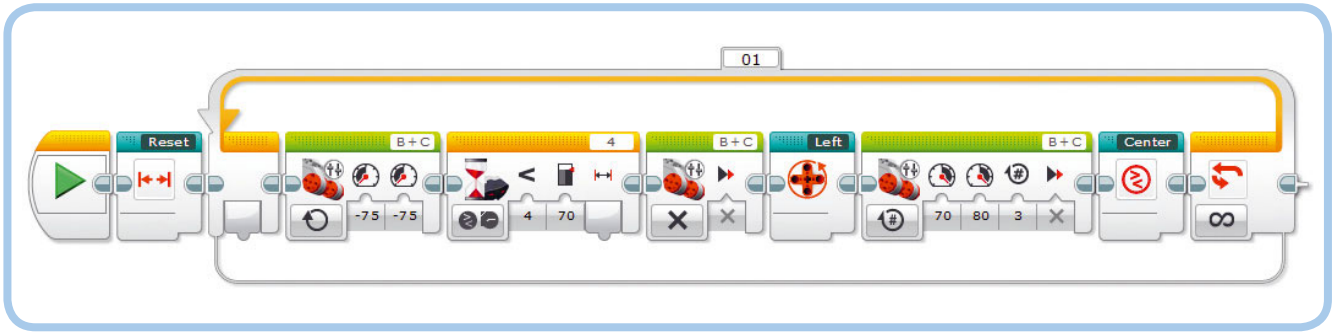


Рис. 12.11. Программа AutonomousDrive

в различных направлениях, а вы будете управлять им, нажимая кнопки на удаленном инфракрасном маяке, как показано на рис. 12.9. При нажатии каждой комбинации кнопок робот будет запускать один из контейнеров «Мой блок» для управления передними колесами и блок **Независимое управление моторами** (Move Tank) для движения задних колес.

Блоки **Независимое управление моторами** (Move Tank) можно использовать для установки скорости каждого мотора в отдельности. При движении робота вперед или назад оба колеса вращаются со скоростью 75%. При повороте внешнее колесо должно вращаться немного быстрее, чем внутреннее. Установите скорость внешнего колеса 80%, а внутреннего — 70%.

Отрицательное значение скорости, например –75, заставит робота двигаться вперед в том направлении, как установлены моторы. При положительном значении, к примеру 75, робот поедет назад. Теперь создайте программу RemoteControl с помощью рис. 12.10.

ПРИМЕЧАНИЕ Если робот при движении вперед не едет прямо, скорректируйте значение в градусах в блоке Средний мотор (Medium Motor) контейнера «Мой блок» с именем Reset, показанного на рис. 12.4. Если робот отклоняется влево, введите значение более 78 градусов, если вправо — введите меньшее значение.

Самостоятельная езда

Теперь создадим программу, которая заставит гоночный автомобиль самостоятельно ездить по всей комнате, объезжая препятствия благодаря работе инфракрасного датчика. Сначала робот едет прямо, пока значение приближения инфракрасного датчика не окажется менее 70%. Тогда робот сдает назад и поворачивает влево, чтобы объехать препятствие. После этого робот выравнивает колеса по центру и может снова ехать прямо вперед в новом направлении. Создайте программу AutonomousDrive, как показано на рис. 12.11.

ПРАКТИКУМ № 65: ЭКСПЕРИМЕНТ НА ТОЧНОСТЬ!

Сложность: **Время:**

В контейнере «Мой блок» с именем **Center** любое положение в диапазоне от –5 до 5 считается достаточно близким к центру. Чтобы понять, зачем указывать роботу такой большой диапазон значений, измените настройки блоков **Ожидание** (Wait) и **Переключатель** (Switch), уменьшив границы центрального диапазона до –1 и 1. Что происходит при запуске программы RemoteControl?

Дальнейшее изучение

В этой главе вы строили и программировали робота по инструкции. Это, конечно, очень интересно, но гораздо увлекательнее сконструировать собственного робота. Например, можно улучшить конструкцию, добавив зубчатые колеса или колеса увеличенного диаметра, чтобы увеличить скорость движения автомобиля или превратить робота в совершенно иное транспортное средство, скажем, в легковой автомобиль или внедорожник.

Не расстраивайтесь, если у вас не все получится с первой попытки. Вы будете приобретать все больше опыта, продолжая пробовать новые конструкции. Для начала постарайтесь выполнить задания практикумов «Сделай сам» в конце главы и обязательно применяйте техники, изученные в главах 10 и 11.

ПРАКТИКУМ № 66: НОЧНЫЕ ГОНКИ

Сложность: **Время:**

Вы пока еще не пользовались датчиком цвета, встроенным у робота сзади. Можно ли с его помощью заставить робота ездить только тогда, когда в комнате выключен свет? Способен ли робот обнаруживать препятствия в темноте?

СОВЕТ **Сначала модифицируйте программу AutonomousDrive, чтобы она позволяла измерять яркость окружающего освещения. Вам понадобится блок Переключатель (Switch) для определения того, включен свет или выключен. Что такое пороговое значение и как разместить другие блоки в последовательности?**

ПРАКТИКУМ № 67: НАЖИМАЕМ НА ПЕДАЛЬ ГАЗА!

Сложность: **Время:**

Можно ли создать программу для управления скоростью гоночного автомобиля с помощью датчика касания, а рулевого управления — с помощью инфракрасного маяка? Создайте в программе две параллельные последовательности: одна должна управлять ориентацией передних колес с помощью маяка, а другая — контролировать скорость вращения задних колес с помощью датчика касания.

Когда все будет готово, добавьте в программу звук работающего автомобильного мотора с помощью блоков **Звук** (Sound), настроенных так, чтобы воспроизводить звук ускорения, холостого хода и замедления.

СОВЕТ **Подключите датчик касания через длинный кабель к входному порту 1 модуля EV3. Помните, что датчик касания может определять только, нажата красная кнопка или нет, но не промежуточные положения.**

ПРАКТИКУМ № 68: МИГАЮЩИЙ ЗАДНИЙ СВЕТ!

Сложность: **Время:**

У гоночных автомобилей «Формулы-1» есть ярко-красные задние фары, которые мигают в плохую погоду, обеспечивая видимость автомобиля для других водителей. Можно ли заставить датчик цвета менять цвет с синего на красный и наоборот, чтобы сымитировать мигание? В конструкторе нет стандартного блока для изменения цвета, поэтому вам придется создать один контейнер «Мой блок» для переключения на синий цвет, а второй — для переключения на красный.

СОВЕТ **Используйте блок Переключатель (Switch) для измерения яркости окружающего света, но не помещайте никакие блоки в последовательности. Робот ничего не будет делать с этими значениями, но будет менять цвет на синий при измерении яркости окружающего освещения.**

ПРАКТИКУМ № 69: ОБНАРУЖЕНИЕ ПРЕПЯТСТВИЙ!

Сложность: **Время:**

При запуске программы AutonomousDrive вы, вероятно, заметили, что инфракрасный датчик хорош при обнаружении стен и других крупных объектов, но не всегда распознает небольшие предметы, такие как ножка стула. Можно ли заставить робота обходить препятствия другими способами?

Используйте блоки **Нерегулируемый мотор** (Unregulated Motor) для движения, а датчики вращения задних моторов — для обнаружения внезапного падения скорости вращения. Запрограммируйте робота сдавать назад и уезжать, когда он распознаёт объект с помощью инфракрасного датчика либо врезается в него.

СОВЕТ **Используйте методы, изученные в практикуме № 53 на с. 124.**

СДЕЛАЙ САМ № 20: МЧАТЬСЯ БЫСТРЕЕ!

Сборка: **Программирование:**

Можно ли улучшить конструкцию гоночного автомобиля FORMULA EV3, чтобы он двигался быстрее? Используйте зубчатые колеса 36T и 12T для ускорения задних колес на коэффициент 3, как показано на рис. 12.12. Кроме того, ускорить автомобиль можно, установив колеса увеличенного диаметра из наборов LEGO Technic, но вам, возможно, придется изменить конструкцию передних и задних крыльев, чтобы освободить место.

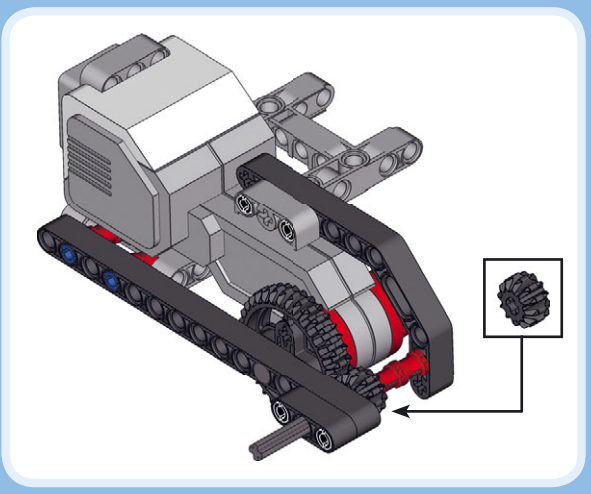


Рис. 12.12. Вы можете запрограммировать гоночный автомобиль ехать быстрее, используя зубчатые колеса из вашего набора MINDSTORMS EV3 (колесо и остальная часть автомобиля не показаны)

СДЕЛАЙ САМ № 21: УСОВЕРШЕНСТВОВАНИЕ АВТОМОБИЛЯ!

Сборка: **Программирование:**

Вы можете разработать собственный автомобиль на основе робота, которого собрали в этой главе? Разберите гоночный автомобиль FORMULA EV3 на части, оставив только механизм рулевого управления с передними колесами (с. 168). Выбрав новое размещение модуля EV3 и задних моторов, вы создадите совершенно другой автомобиль.

Например, чтобы собрать легковой автомобиль, поместите задние колеса ближе к передним, а модуль EV3 — сверху на моторы. Можно сконструировать внедорожник, расположив моторы под углом для увеличения дорожного просвета. Протестируйте свое творение с помощью программы RemoteControl, созданной в этой главе.

13

ANTY: робот-муравей

К этому моменту вы уже научились конструировать модели, движущиеся на колесах. Другой увлекательный, но чуть более сложный тип моделей, которые можно собрать, — робот-животное, передвигающийся на ногах, а не на колесах. В этой главе вы сконструируете и запрограммируете ANTY (рис. 13.1). ANTY — это шестиногое насекомоподобное создание, которое ходит и реагирует на объекты окружающей среды, демонстрируя разные типы поведения в зависимости от того, что «видят» его датчики.

Глазами ANTY служит инфракрасный датчик, позволяющий роботу определять наличие объектов вокруг и находить еду. Датчик цвета в хвосте робота позволяет ANTY распознавать изменения в окружающей среде. Разная среда, т.е. объекты разного цвета, влияет на поведение робота. Зеленые объекты вселяют в него ощущение безопасности, означающее, что он может вздремнуть. Красные объекты обозначают опасность и заставляют робота агрессивно встряхиваться, чтобы отпугнуть врагов. Синие объекты пугают ANTY, вынуждая его спасаться бегством. Наконец, желтые объекты вызывают у робота чувство голода, побуждая его искать еду.

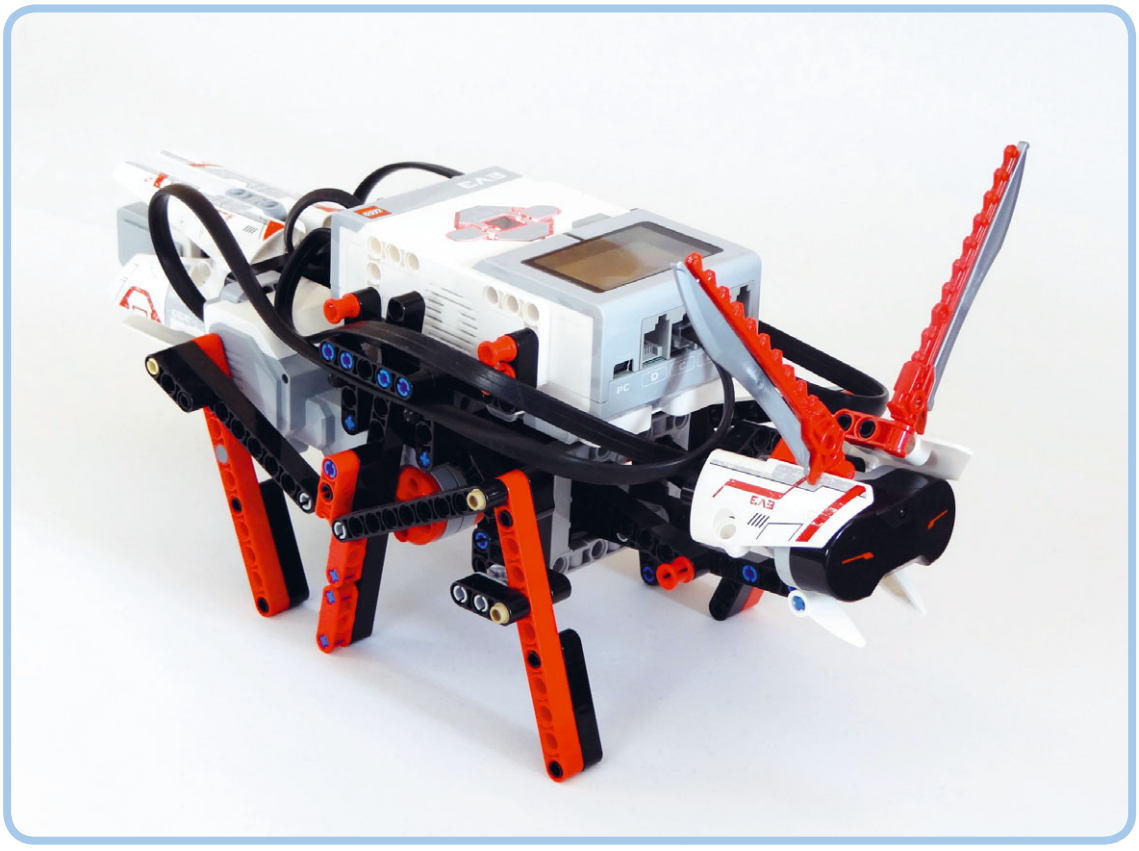


Рис. 13.1. ANTY

Знакомство с механизмом движения

Для ходьбы ANTY использует две конструкции с моторами, каждая управляет тремя ногами. Включение мотора на вращение вперед заставляет три ноги делать шаг вперед, как показано на рис. 13.2. Одновременное включение обоих моторов на вращение вперед заставляет робота шагать вперед.

Это работает, только если левые ноги находятся в положении, противоположном правым ногам. Например, левые ноги должны быть в положении 2, а правые — в положении 4, чтобы обеспечить роботу одновременное касание земли как минимум тремя ногами. Это возможно, если один мотор расположен ровно на 180 градусов впереди другого. (Зеленая точка в положении 4 на 180 градусов впереди зеленой точки в положении 2.)

С помощью датчика касания робот определяет абсолютное положение каждого мотора, чтобы при запуске программы они находились в противоположных положениях. Если датчика касается кулачок, робот знает, что ноги находятся в положении 1, как показано на рис. 13.2.

Завершив сборку робота, добавьте контейнер «Мой блок», который будет помещать ноги по обеим сторонам в нужное положение, после чего робот может начинать шагать с помощью блоков **Независимое управление моторами** (Move Tank). Пока скорость моторов остается одинаковой, разница между моторами будет оставаться 180 градусов.

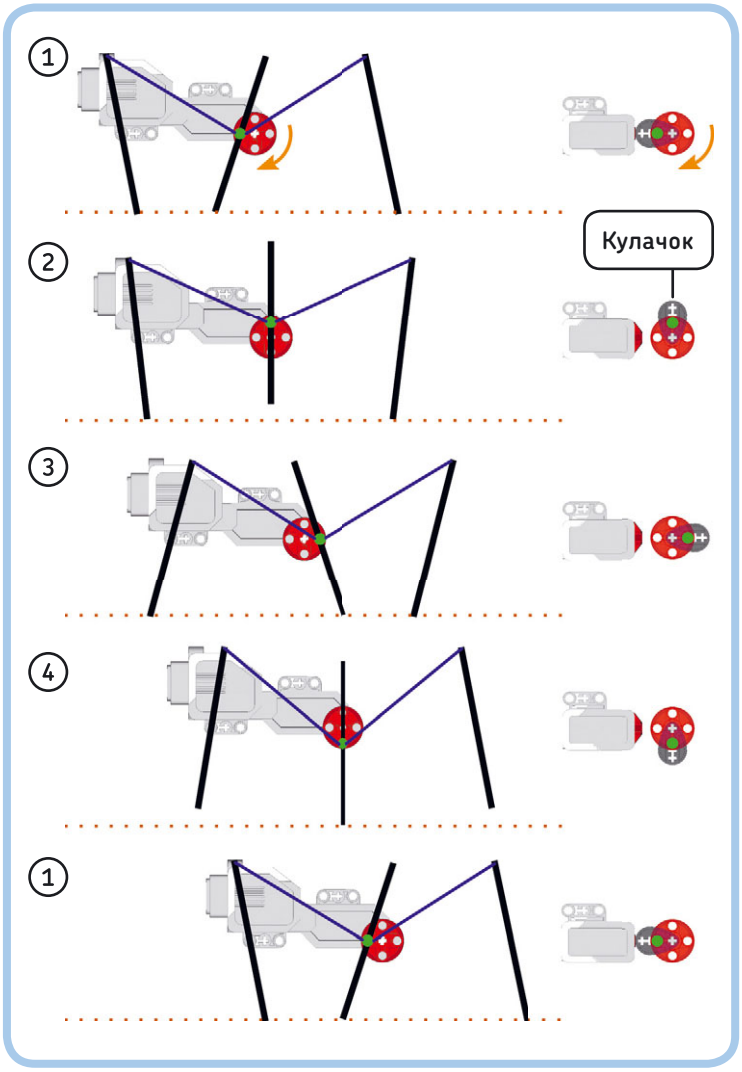


Рис. 13.2. Вращение мотора вперед заставляет три ноги (черные линии) шагать вперед. Робот перемещает среднюю ногу, вращая колесико (зеленая точка) по кругу. Средняя нога, в свою очередь, приводит в движение боковые ноги с помощью перекладин (синие линии). Совершив один полный оборот, ноги снова оказываются в положении 1, и робот перемещается вперед

Сборка робота ANTY

Теперь, когда вы изучили, как работает робот, можно приступать к его сборке. Следуйте приведенным далее инструкциям, но сначала выберите необходимые детали (рис. 13.3).

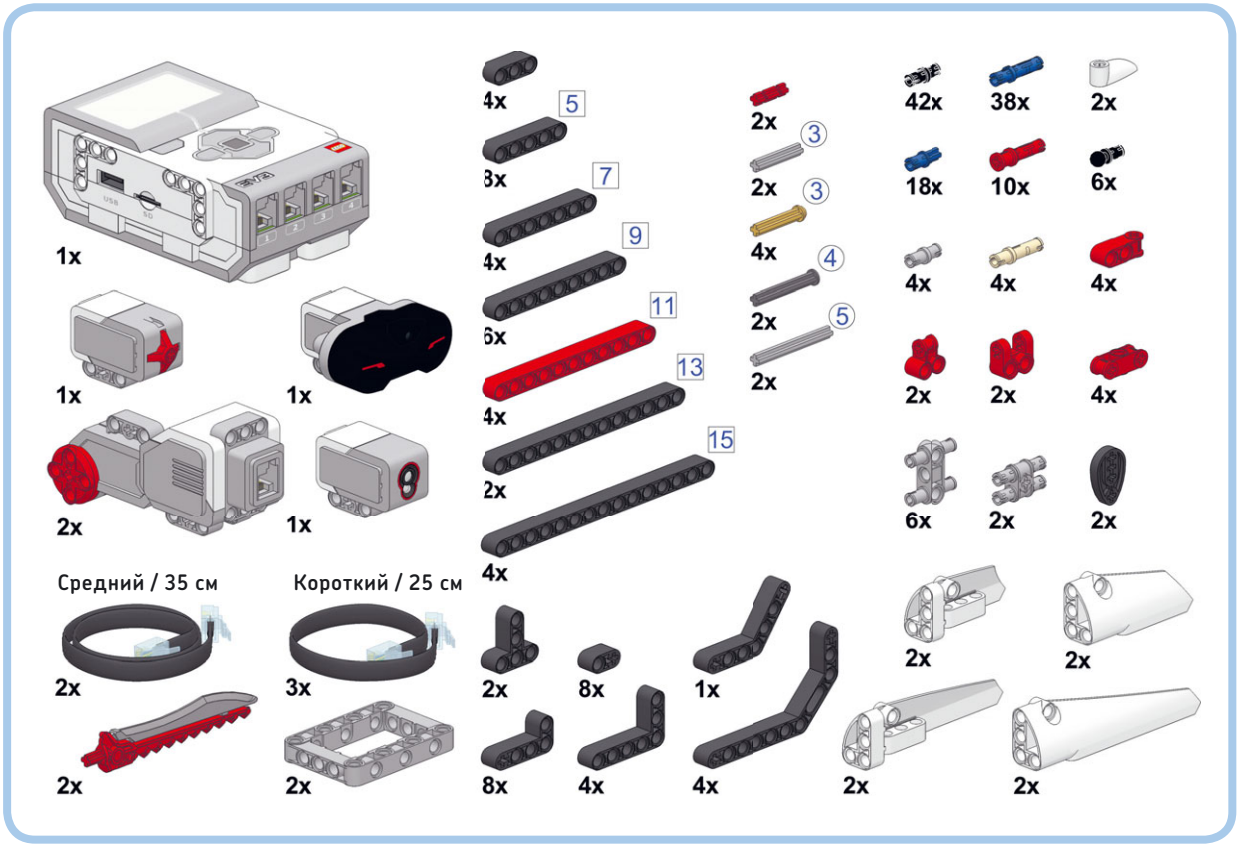
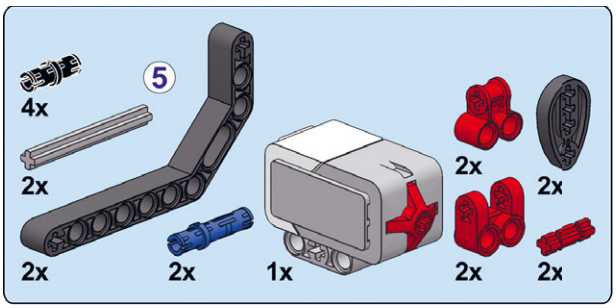
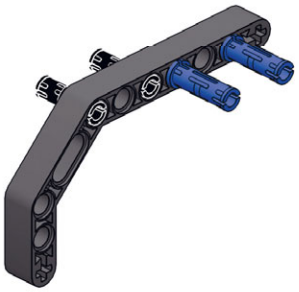


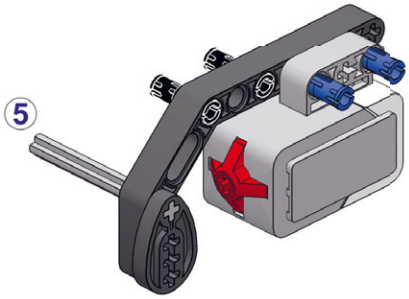
Рис. 13.3. Детали, необходимые для сборки робота ANTY



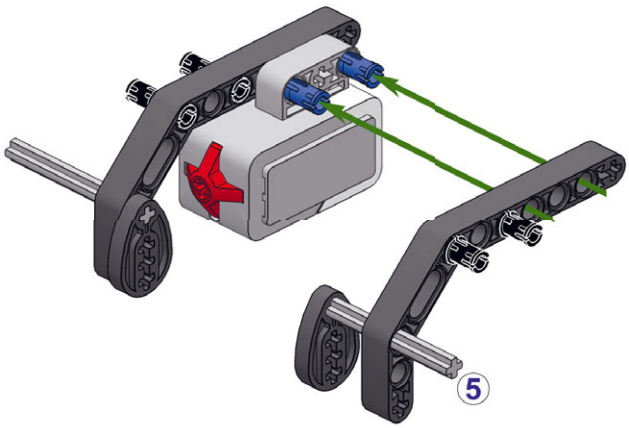
1



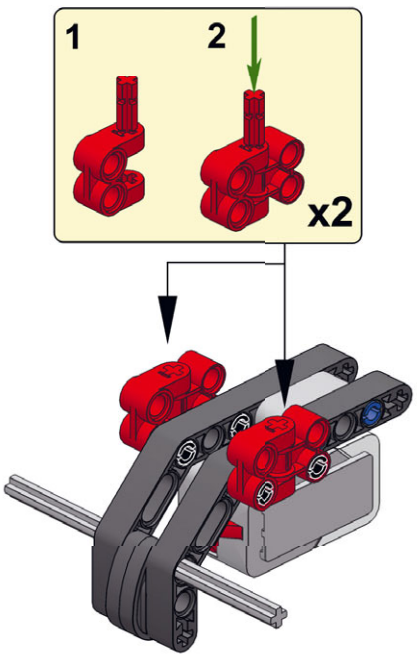
2

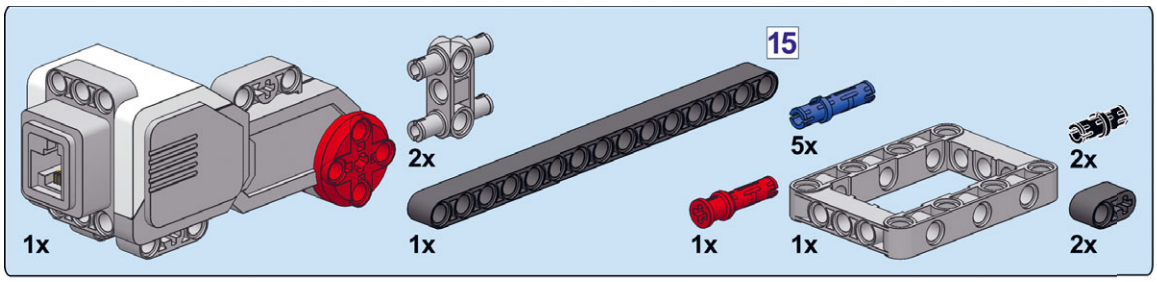


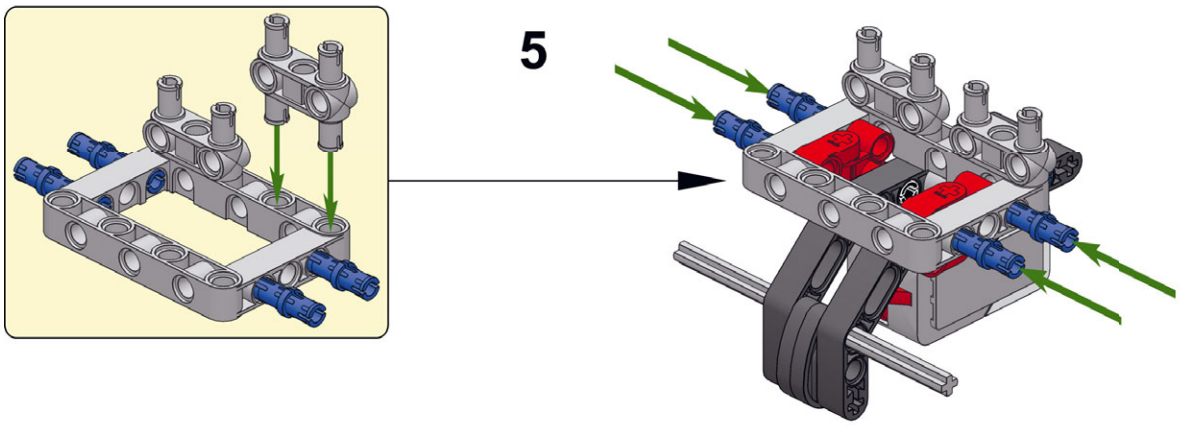
3

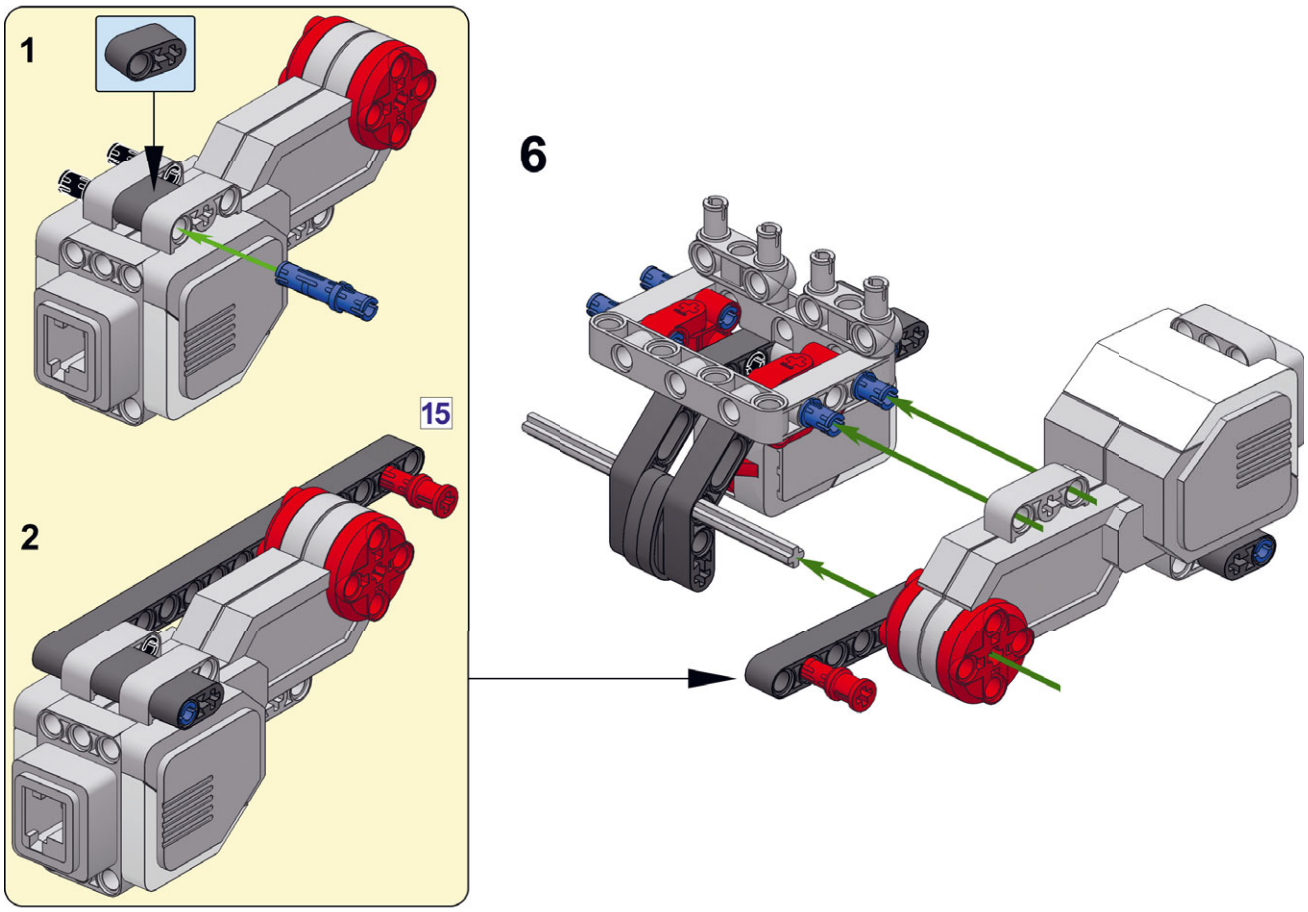


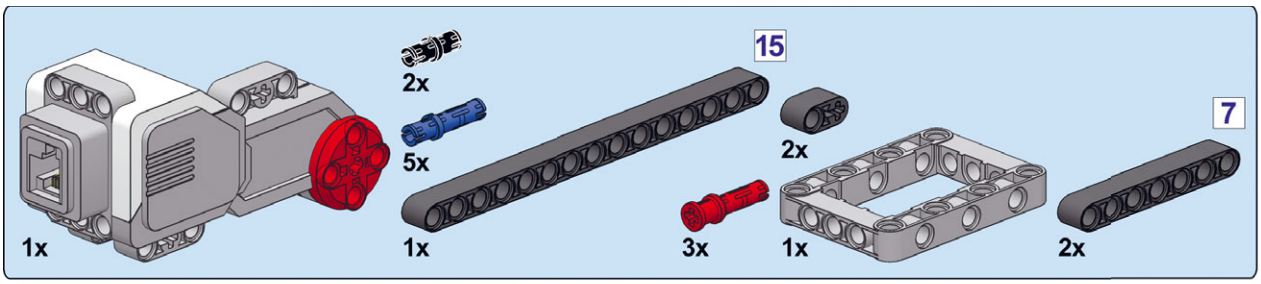
4

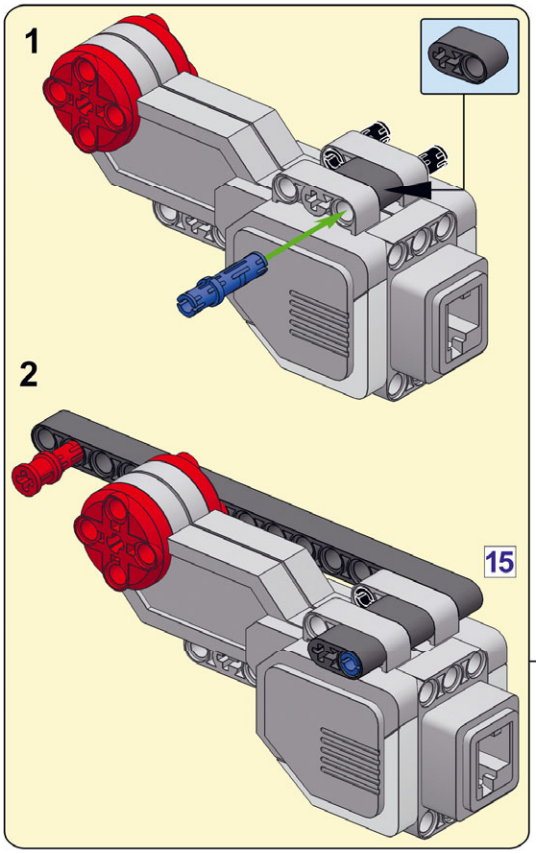


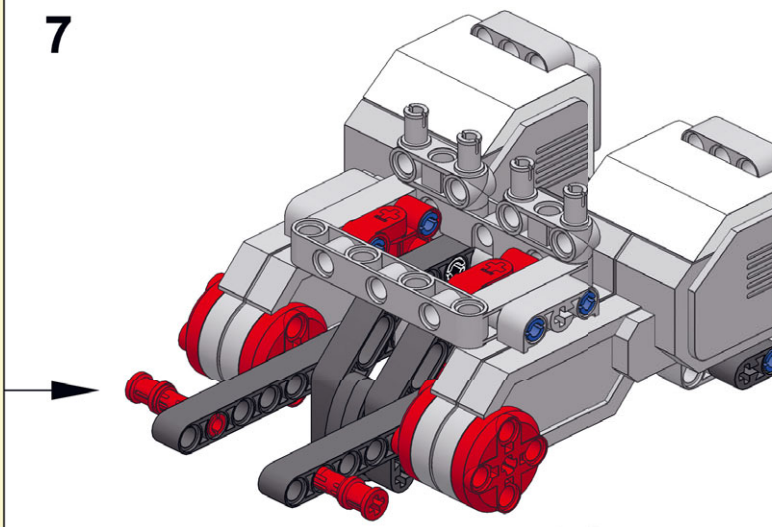

15
1x
2x
1x
1x
5x
1x
2x
2x

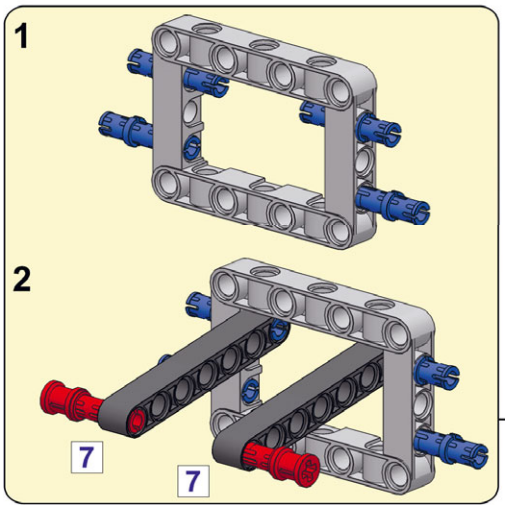

5

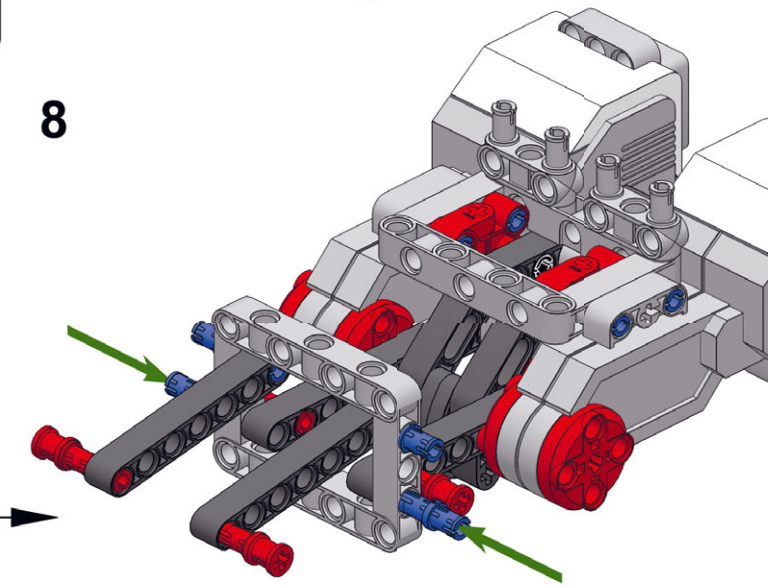

1
15
2
6

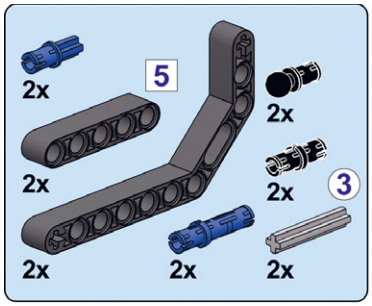

1x
2x
5x
1x
15
3x
2x
1x
7
2x

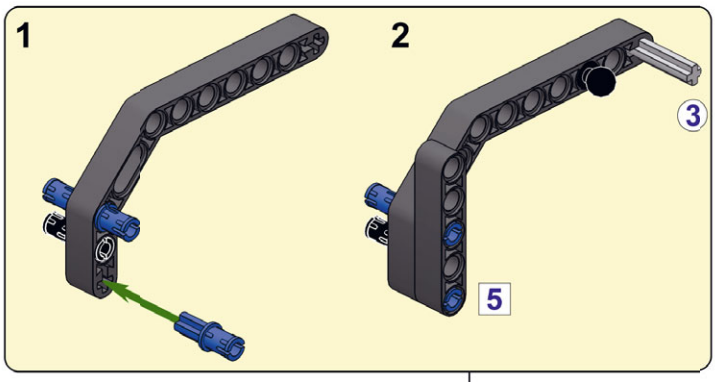

1
2
15


7

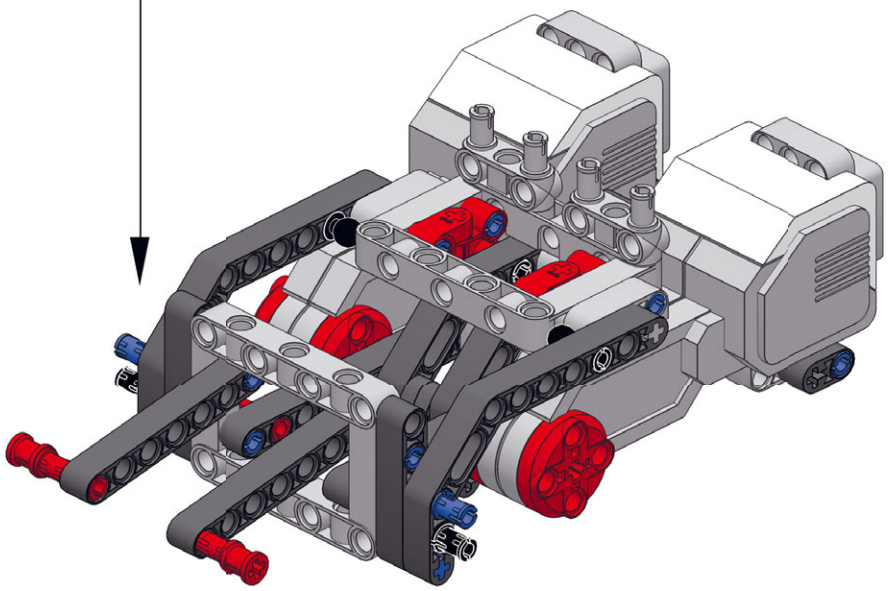

1
2
7
7

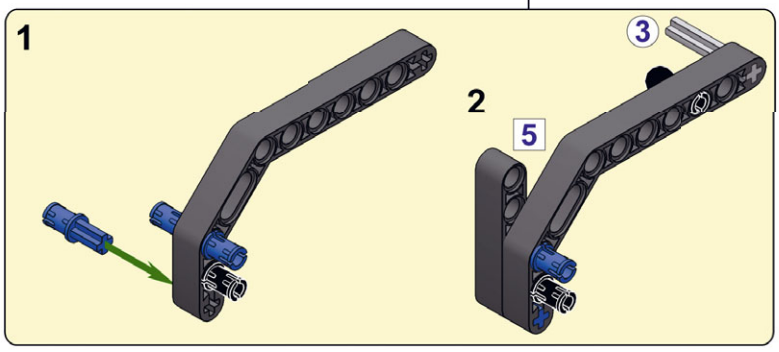

8

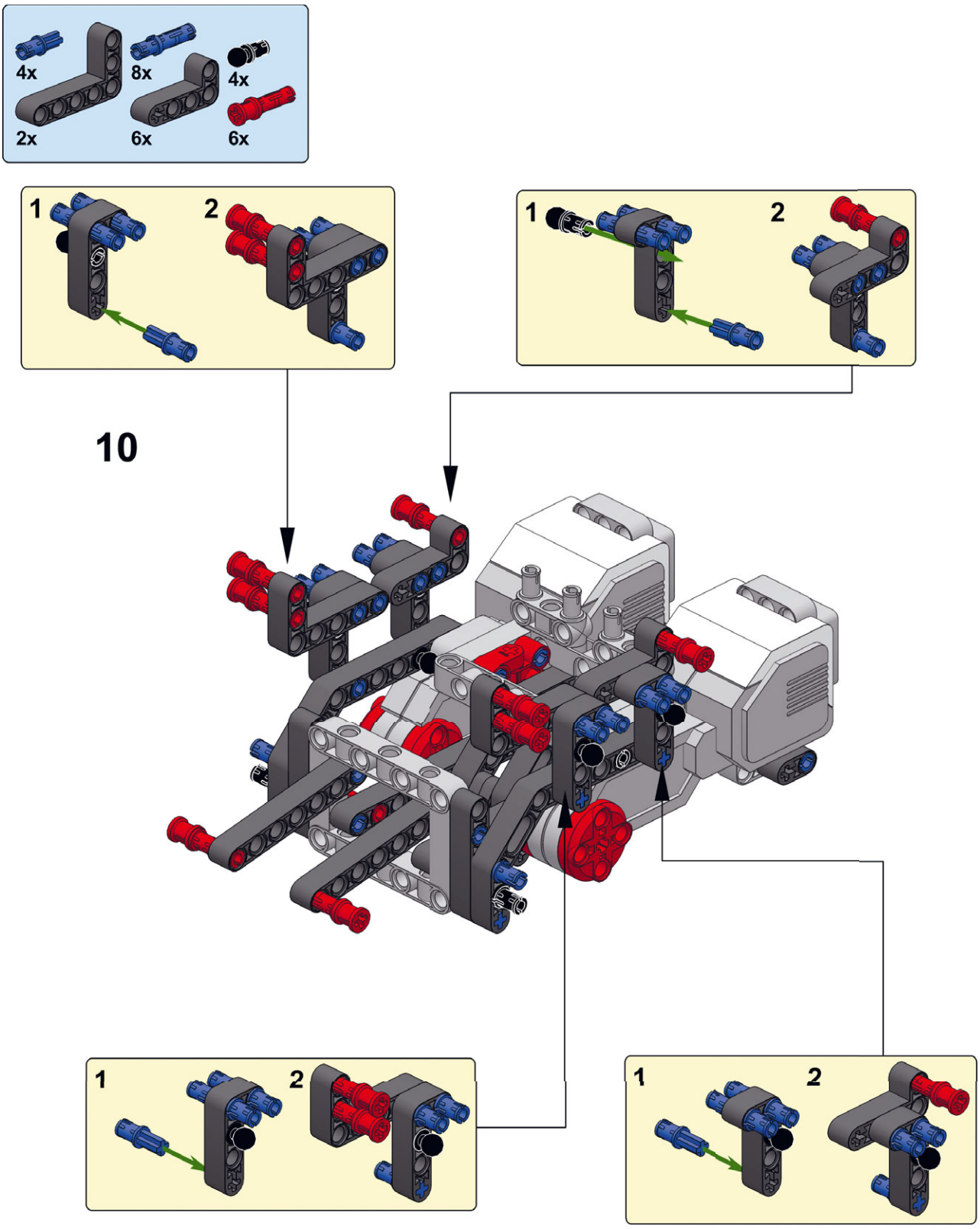


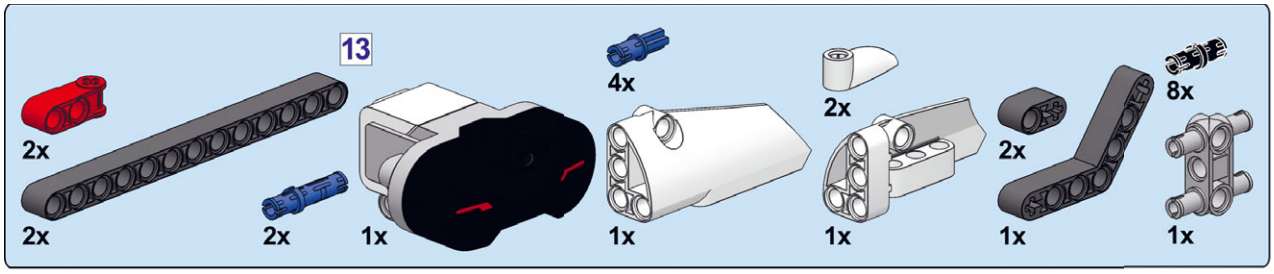


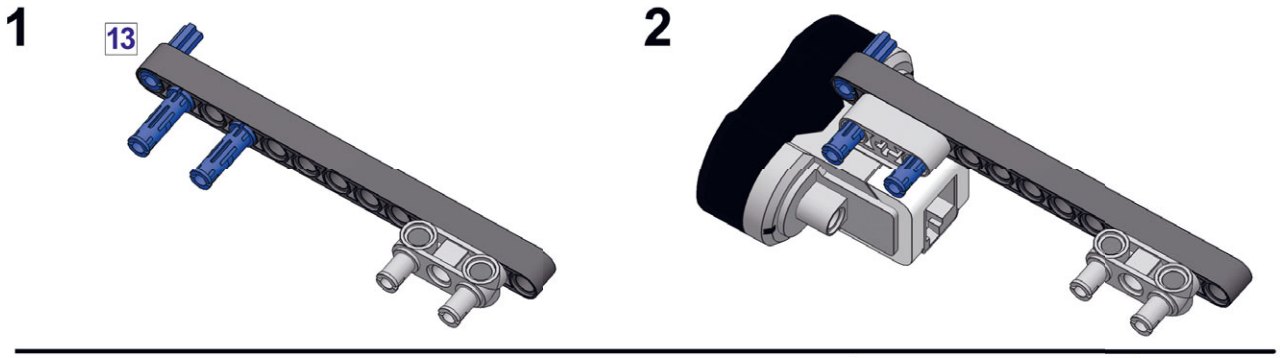
9

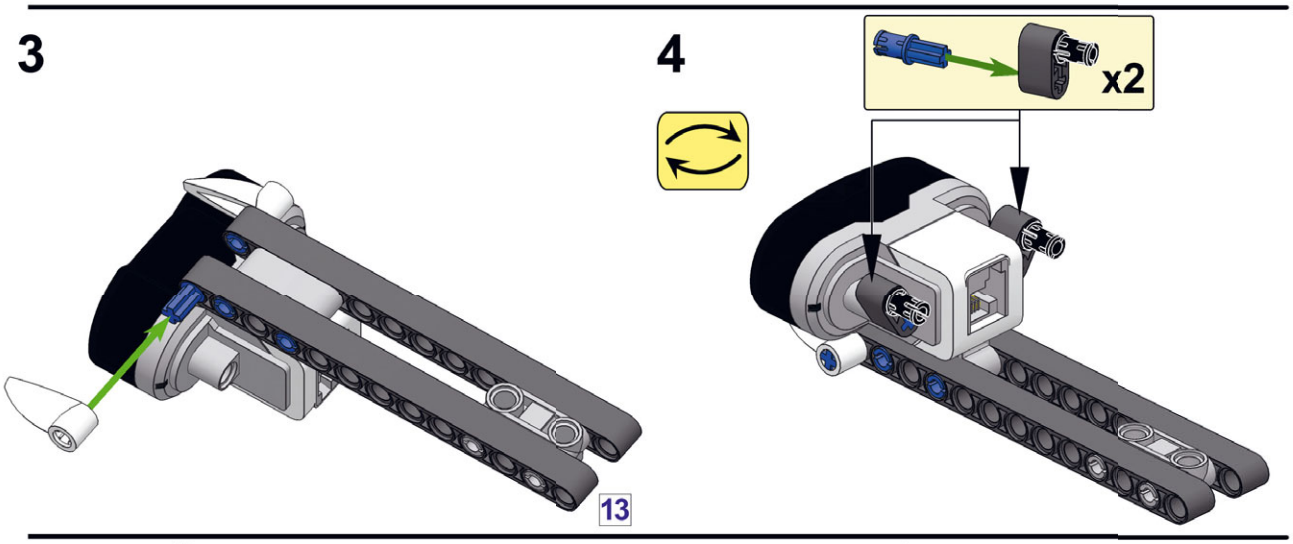


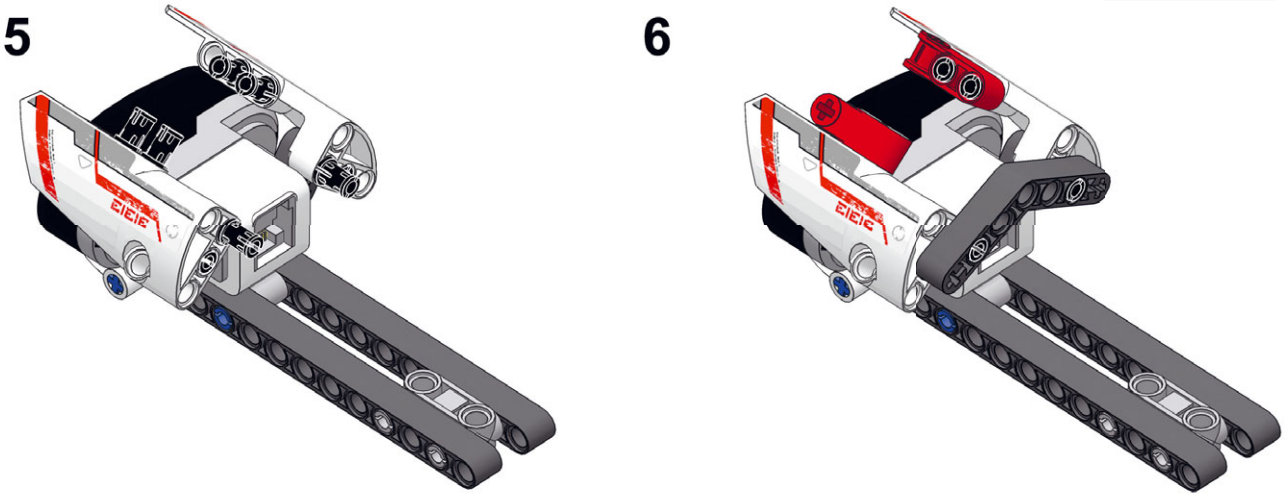


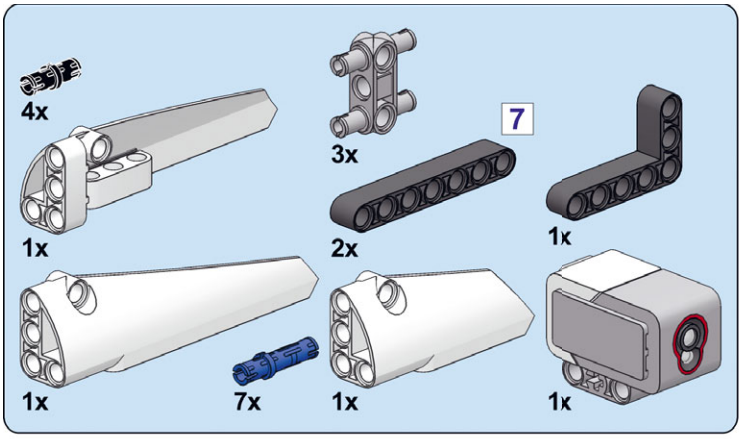

4x
2x
8x
6x
4x
6x
1
2
1
2
10
1
2
1
2

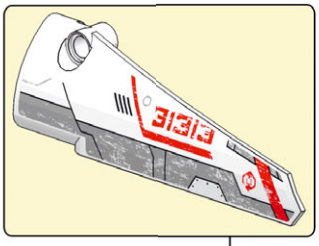

13
2x
2x
2x
1x
4x
1x
2x
1x
2x
1x
8x
1x


1
13
2

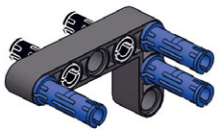

3
13
4
x2


5
6

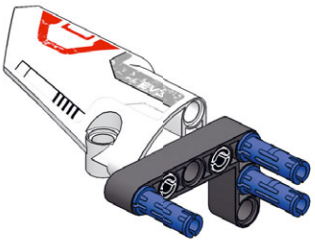




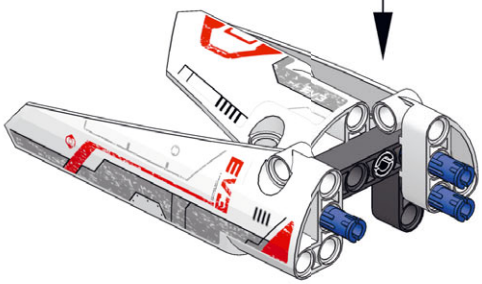
1

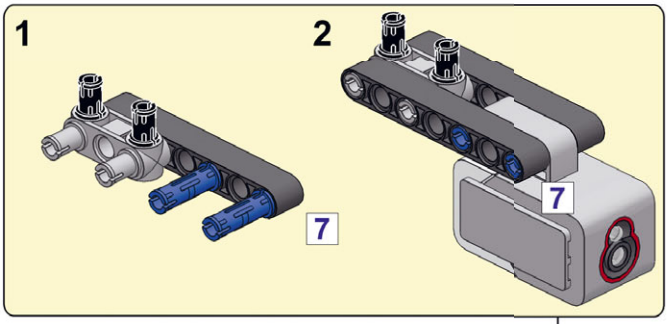


2

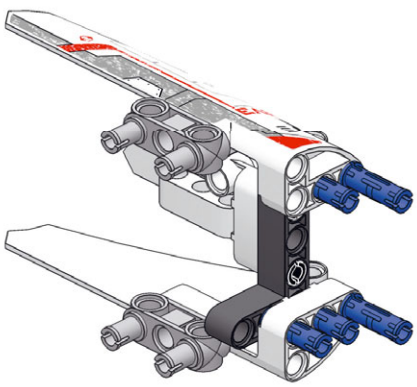


3

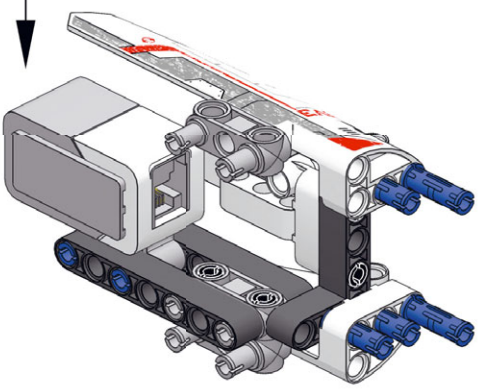


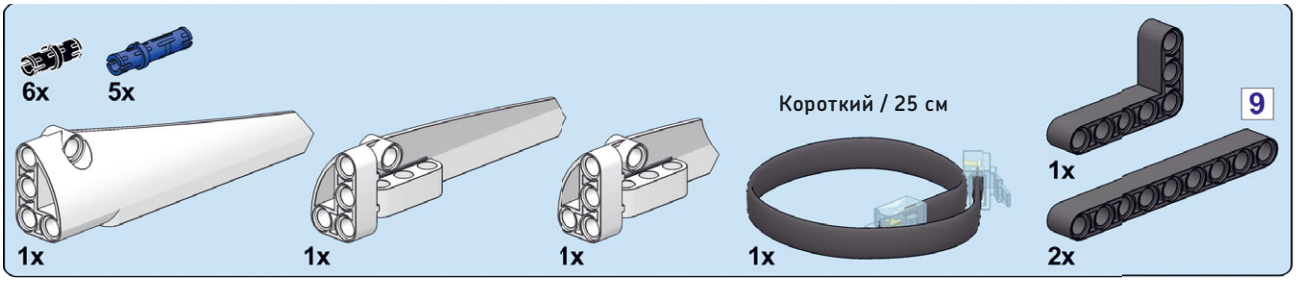


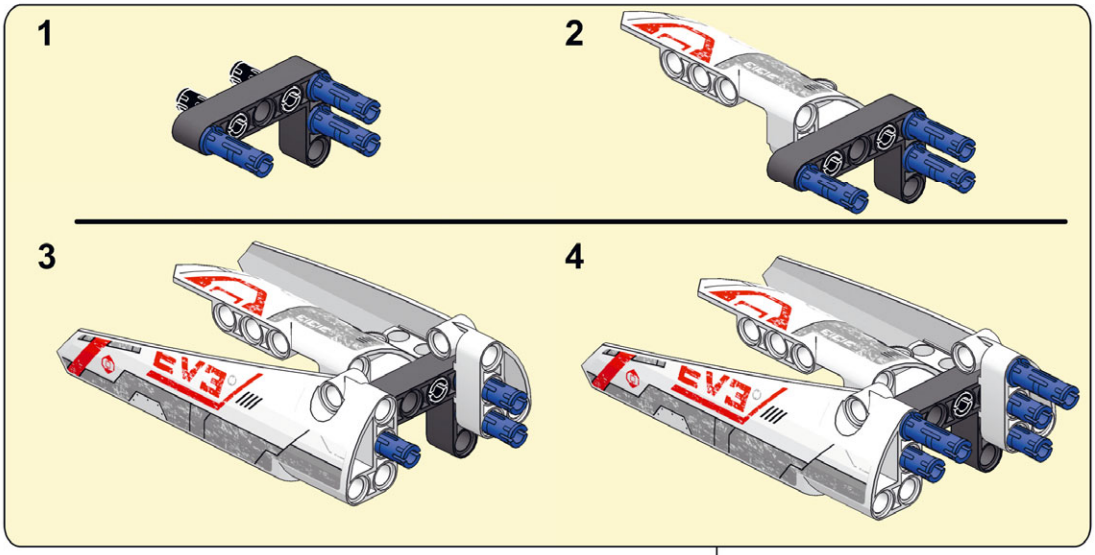
4

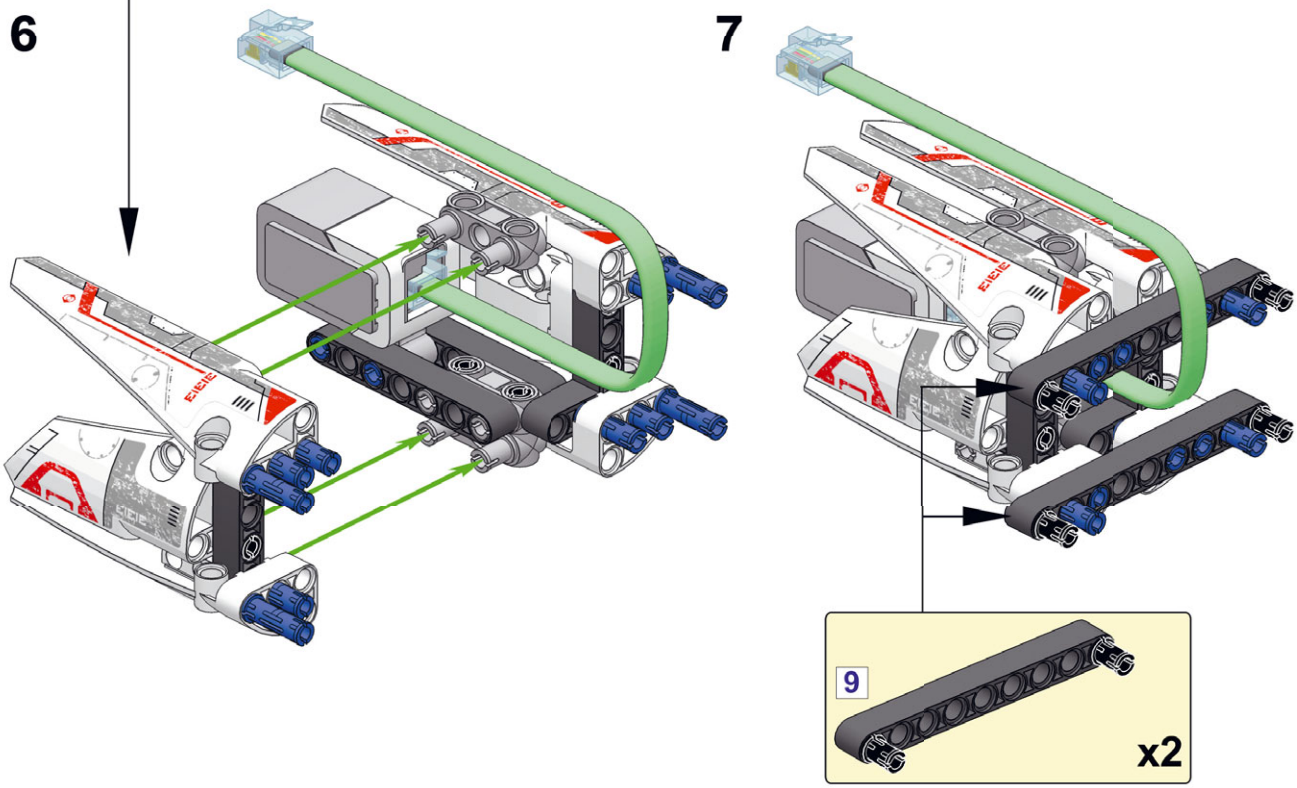


5

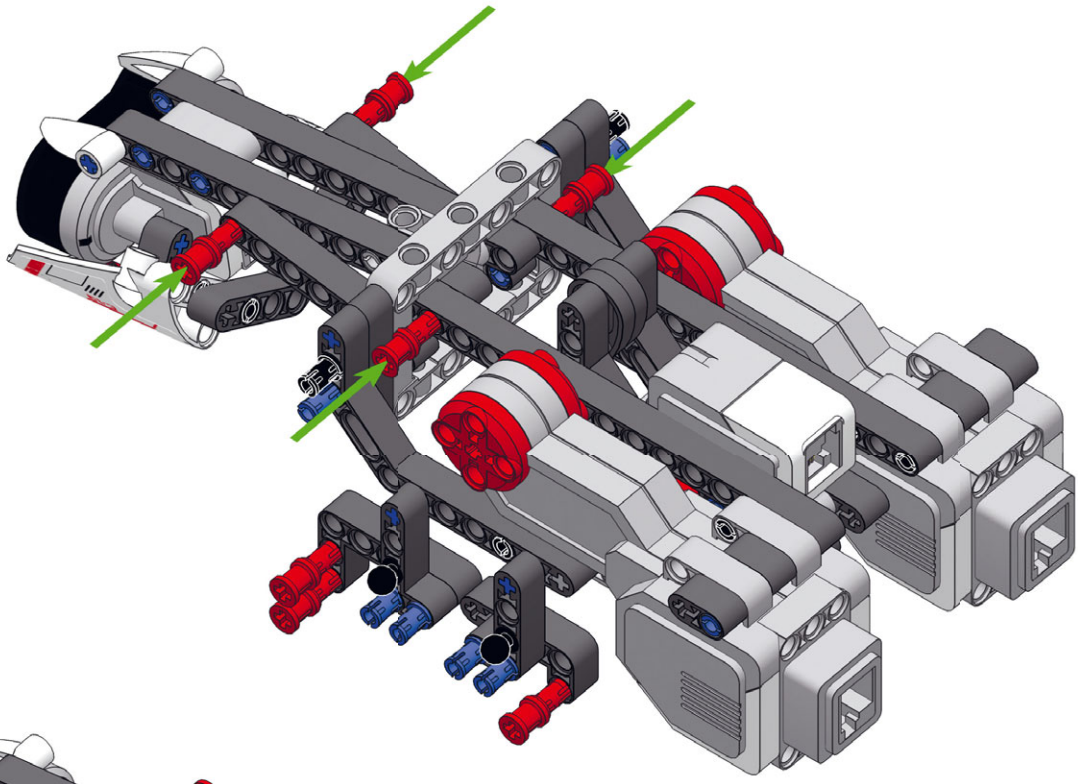



6x
5x
1x
1x
1x
Короткий / 25 см
1x
9
1x
2x

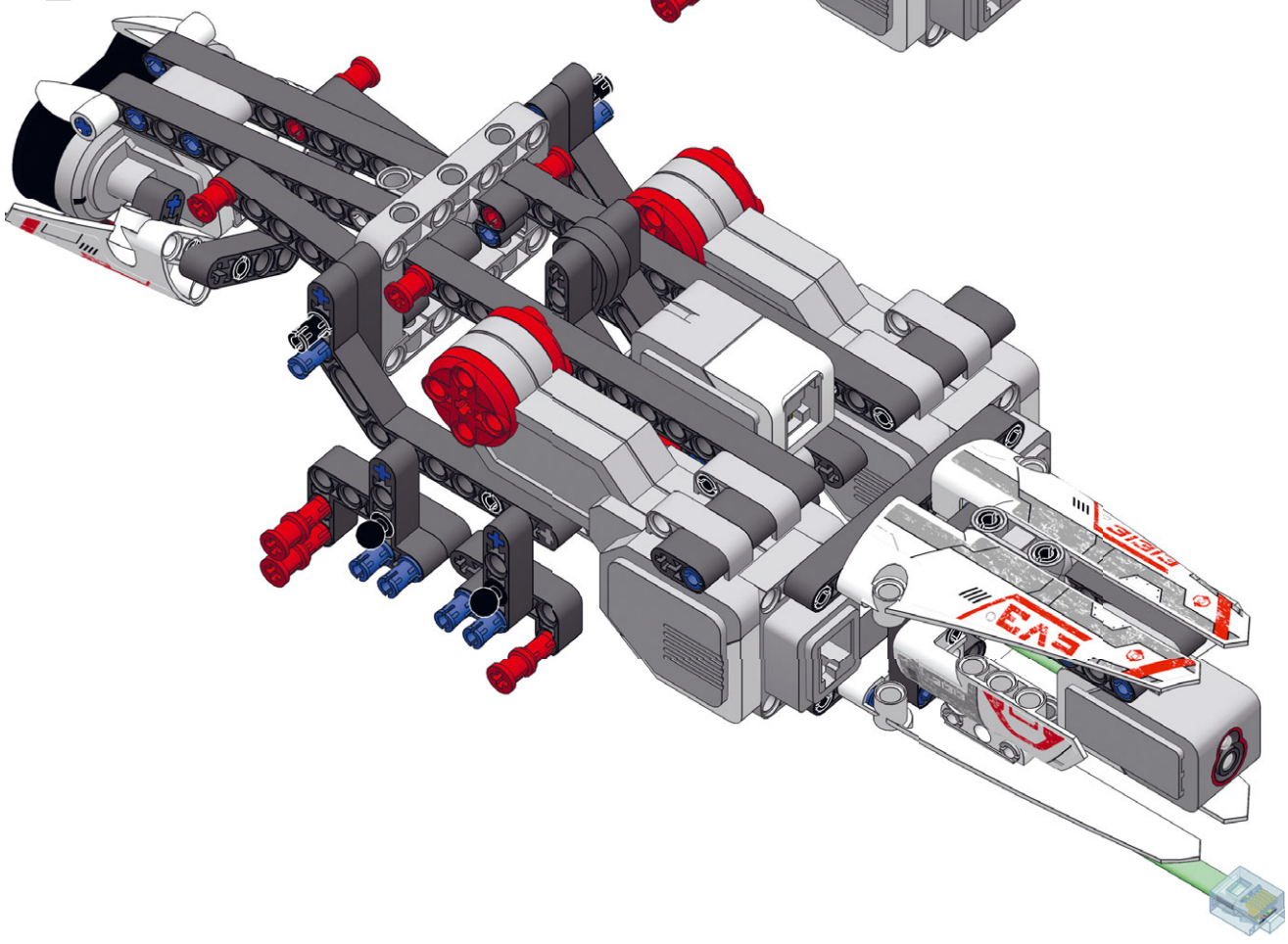

1
2
3
4

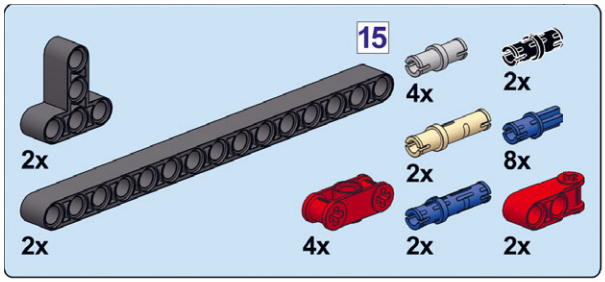

6
7
9
x2

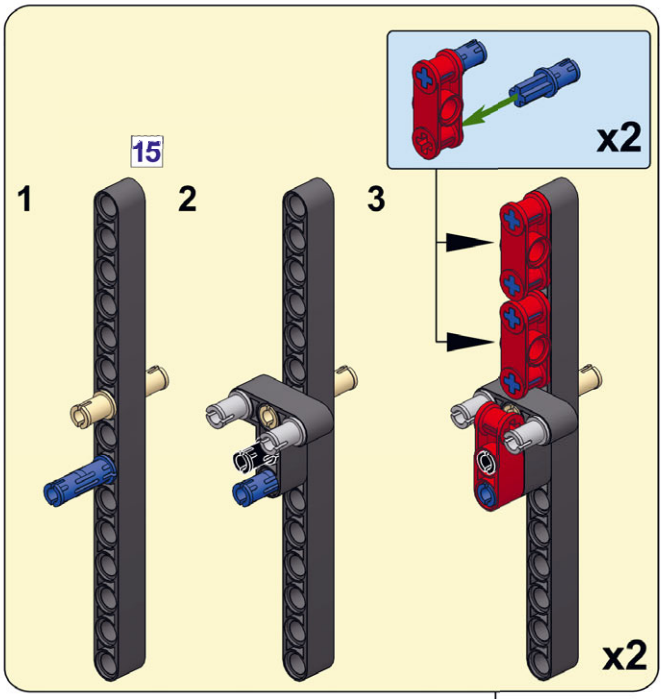
1



2

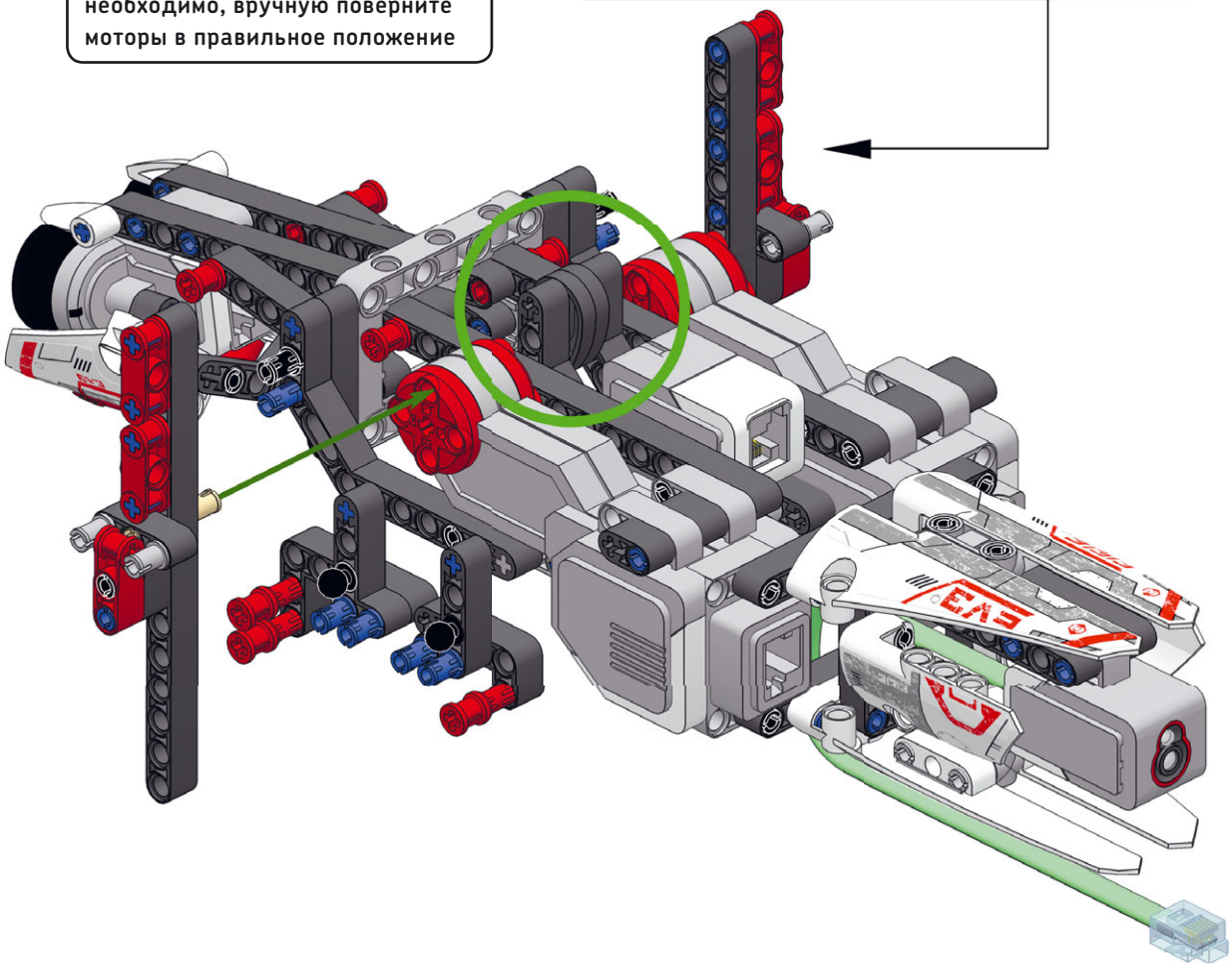


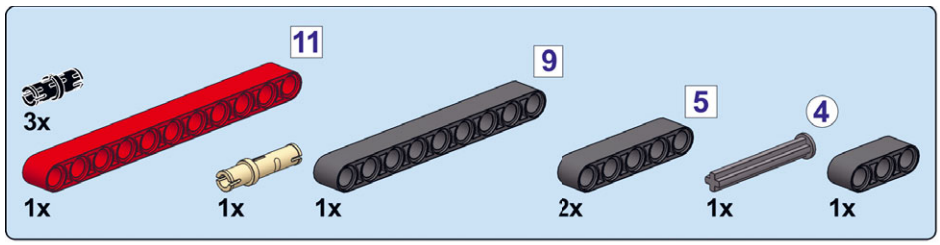


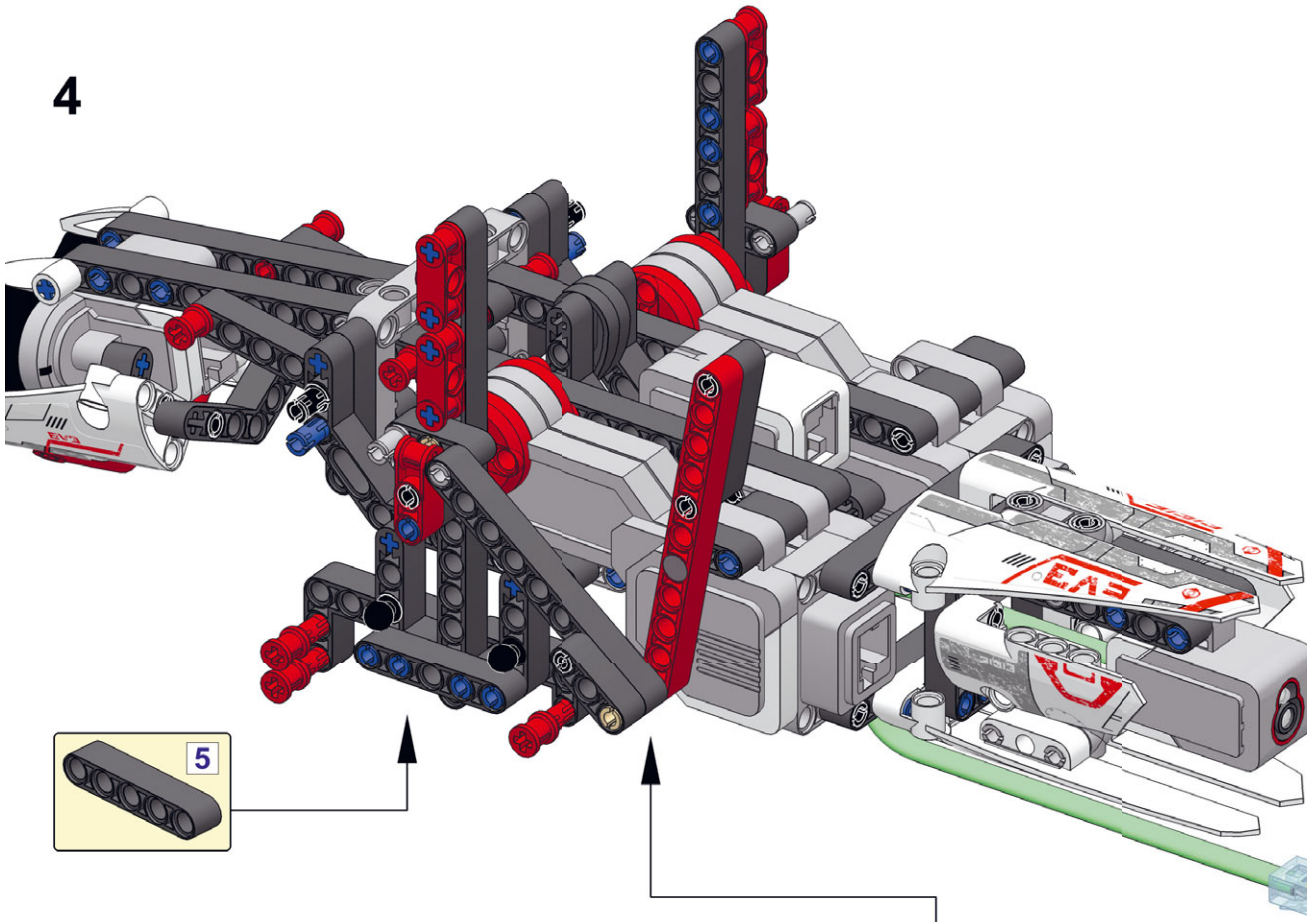


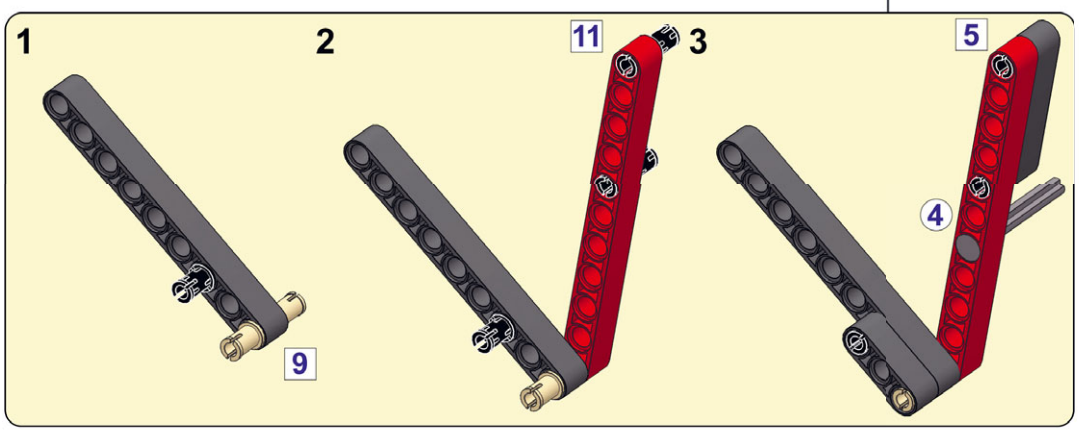
3

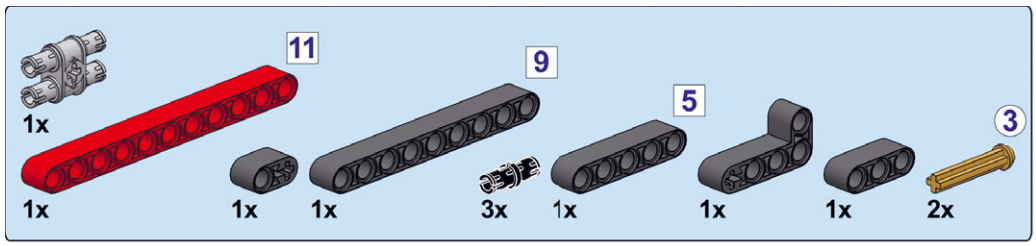
Перед тем как крепить ноги, убедитесь, что кулачки (обведены зеленым цветом) выровнены, как показано на рисунке. Если необходимо, вручную поверните моторы в правильное положение

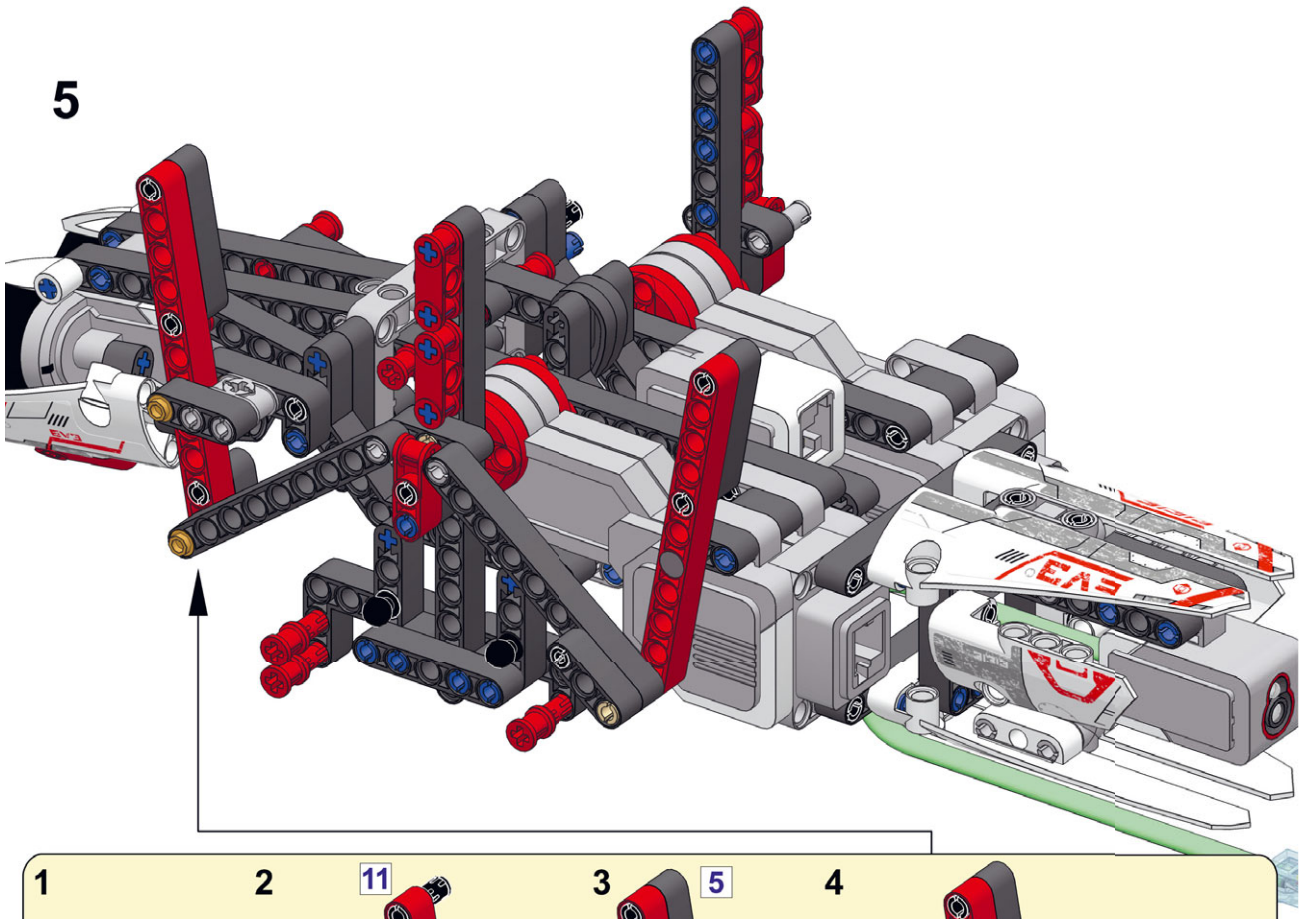


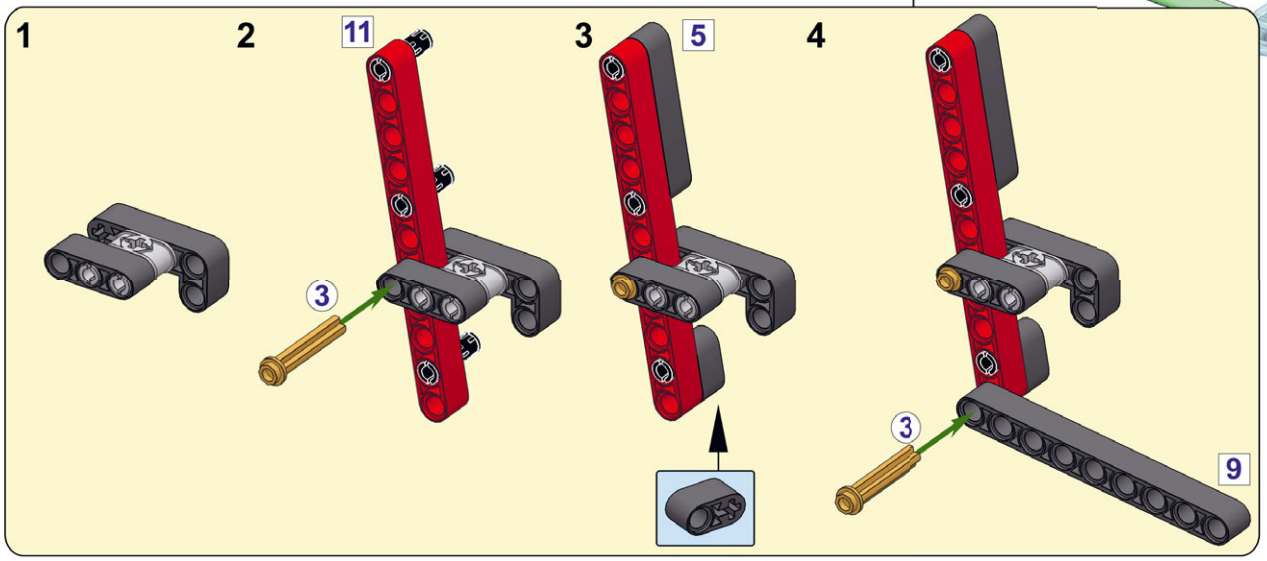

11
9
5
4
3x
1x
1x
1x
2x
1x
1x

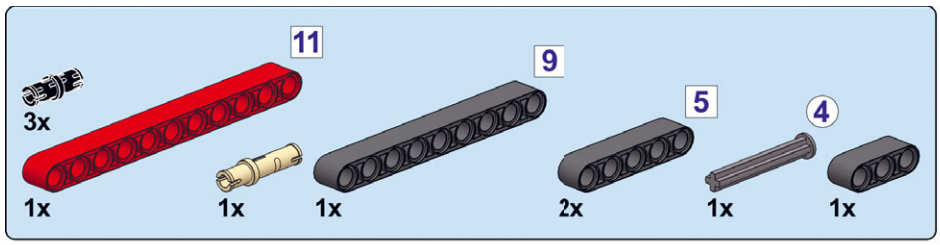

4
5

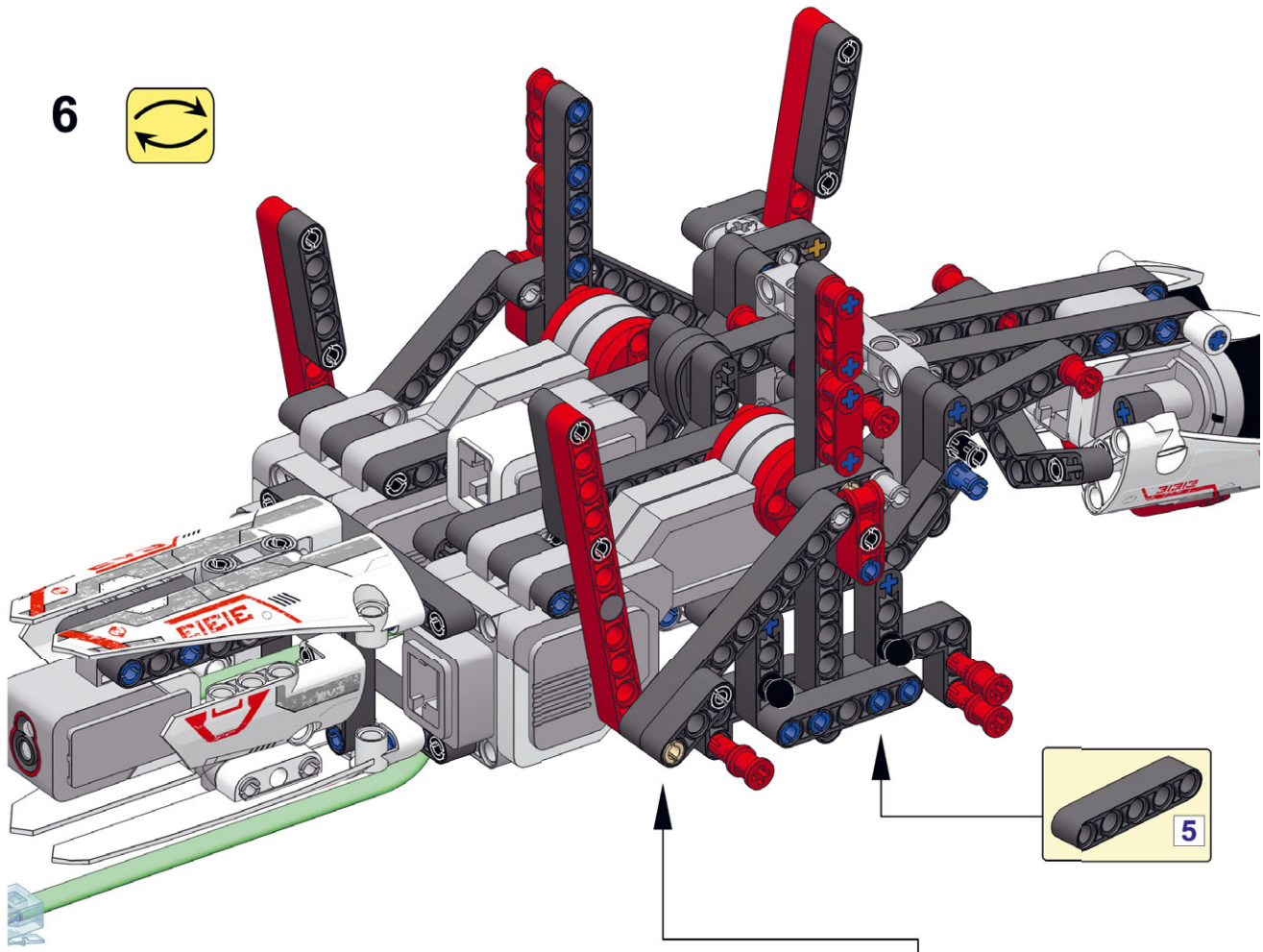

1
2
11
3
5
4
9

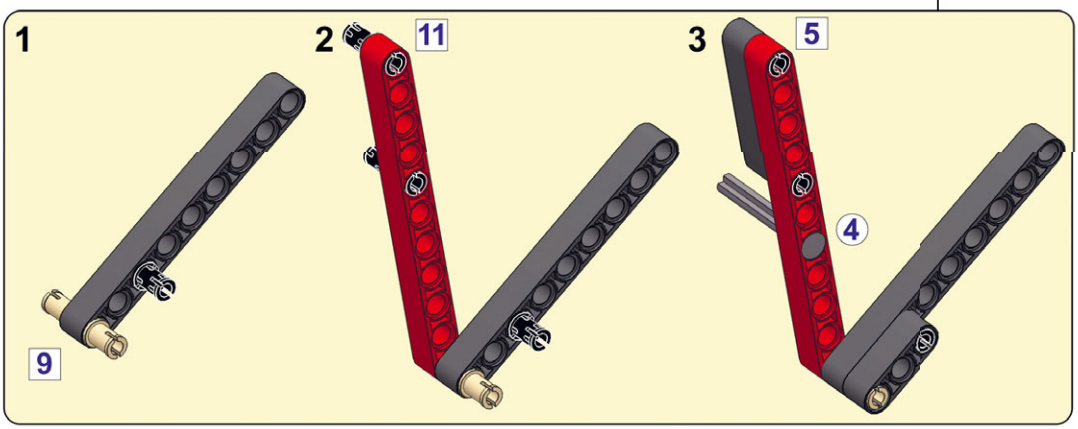

11
9
5
3
1x
1x
1x
1x
3x
1x
1x
1x
2x

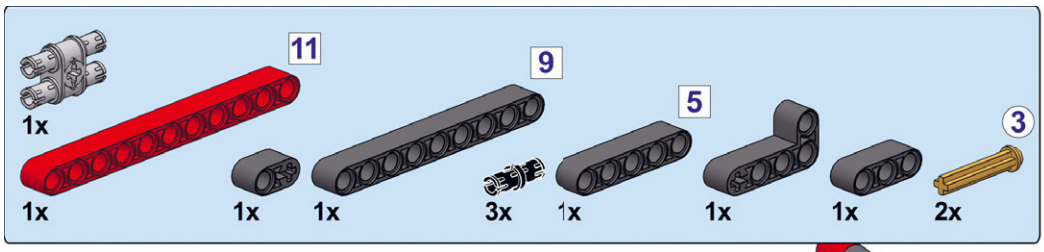

5


1
2
11
3
3
5
4
3
9

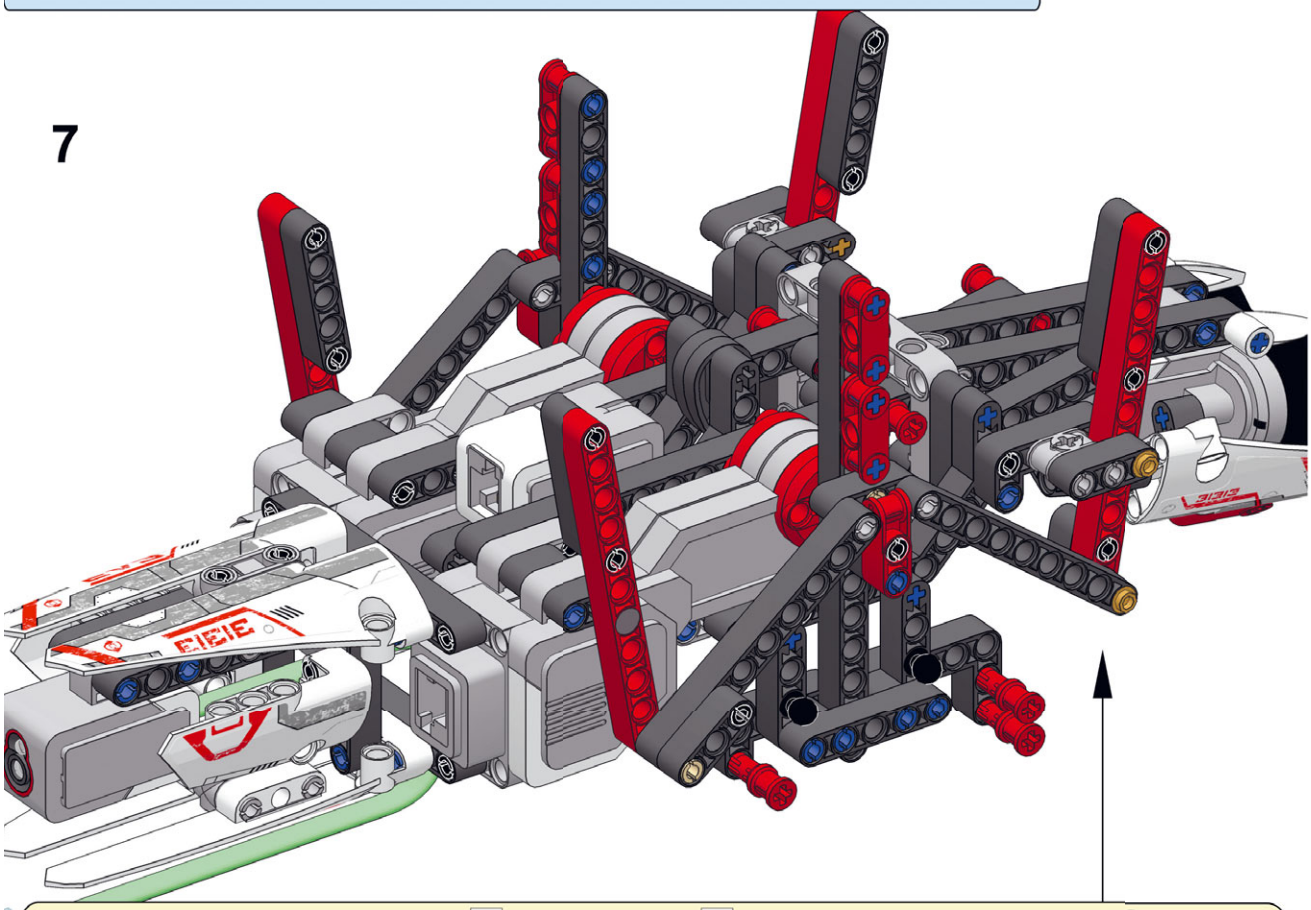

11
9
5
4
3x
1x
1x
1x
2x
1x
1x

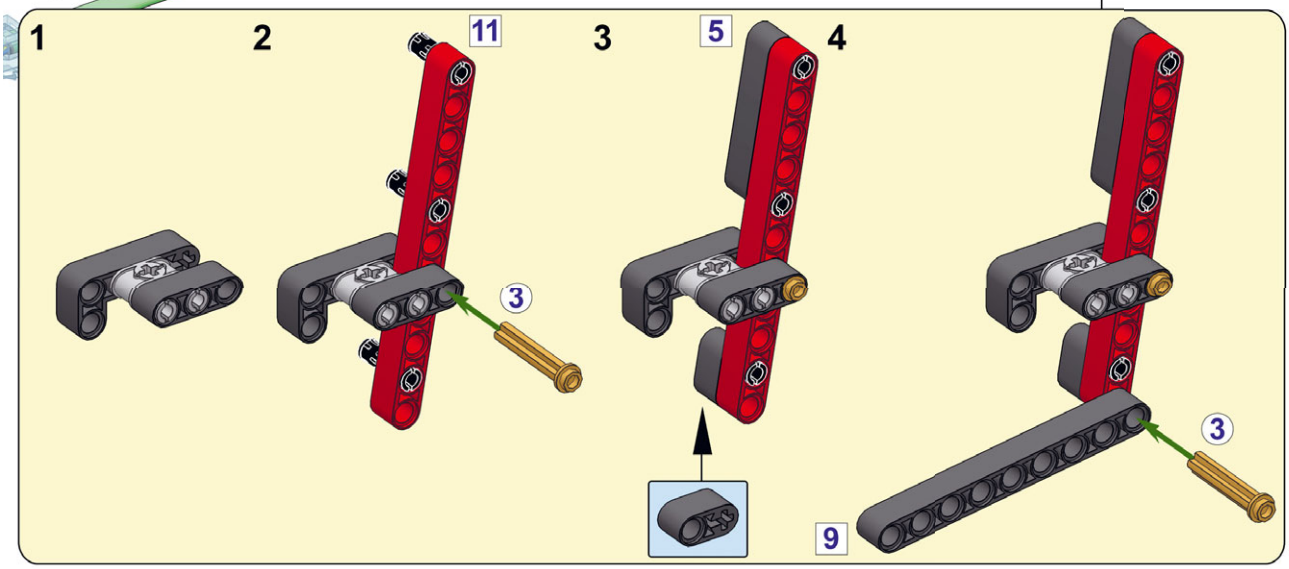

6
5

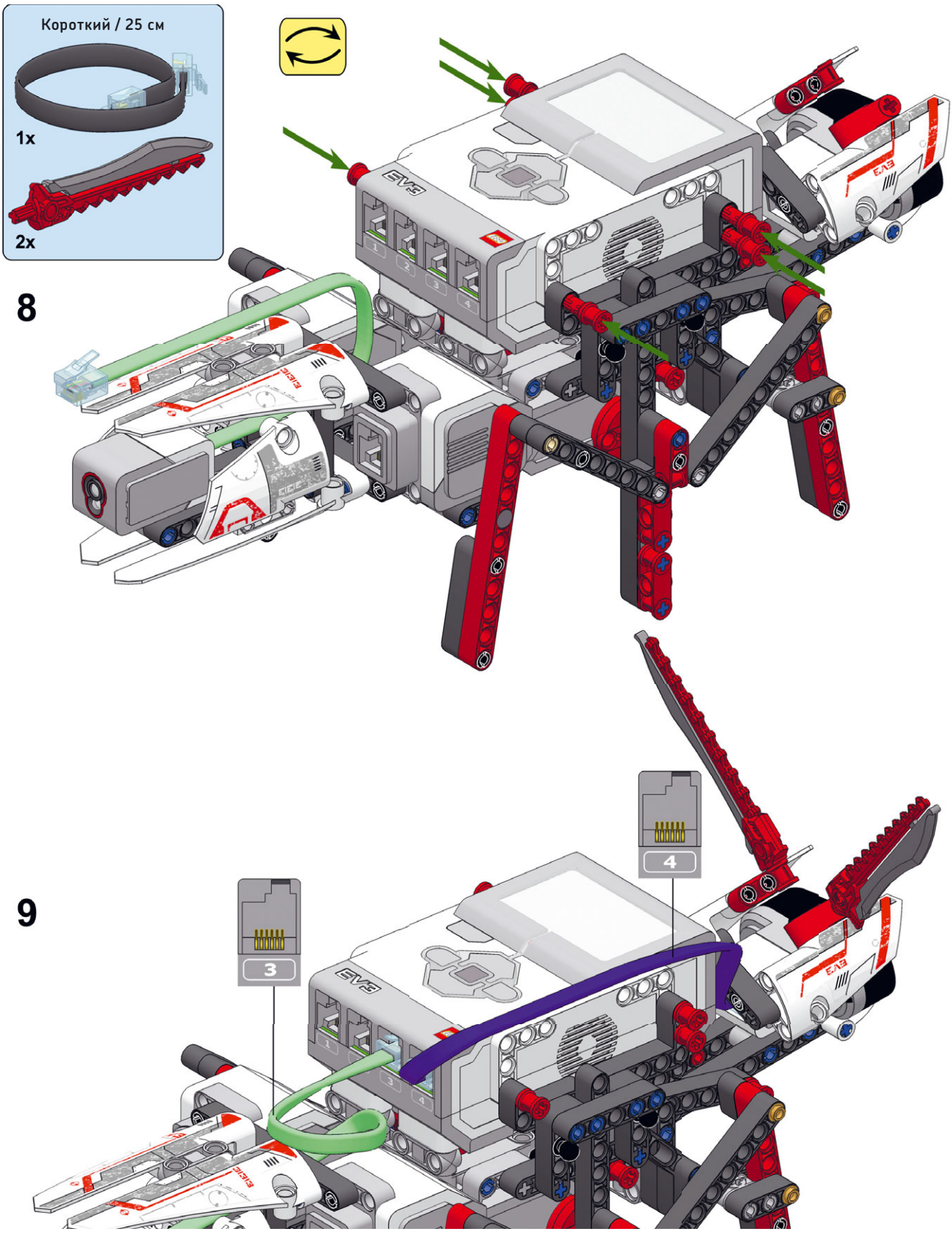

1
2
11
3
5
4
9

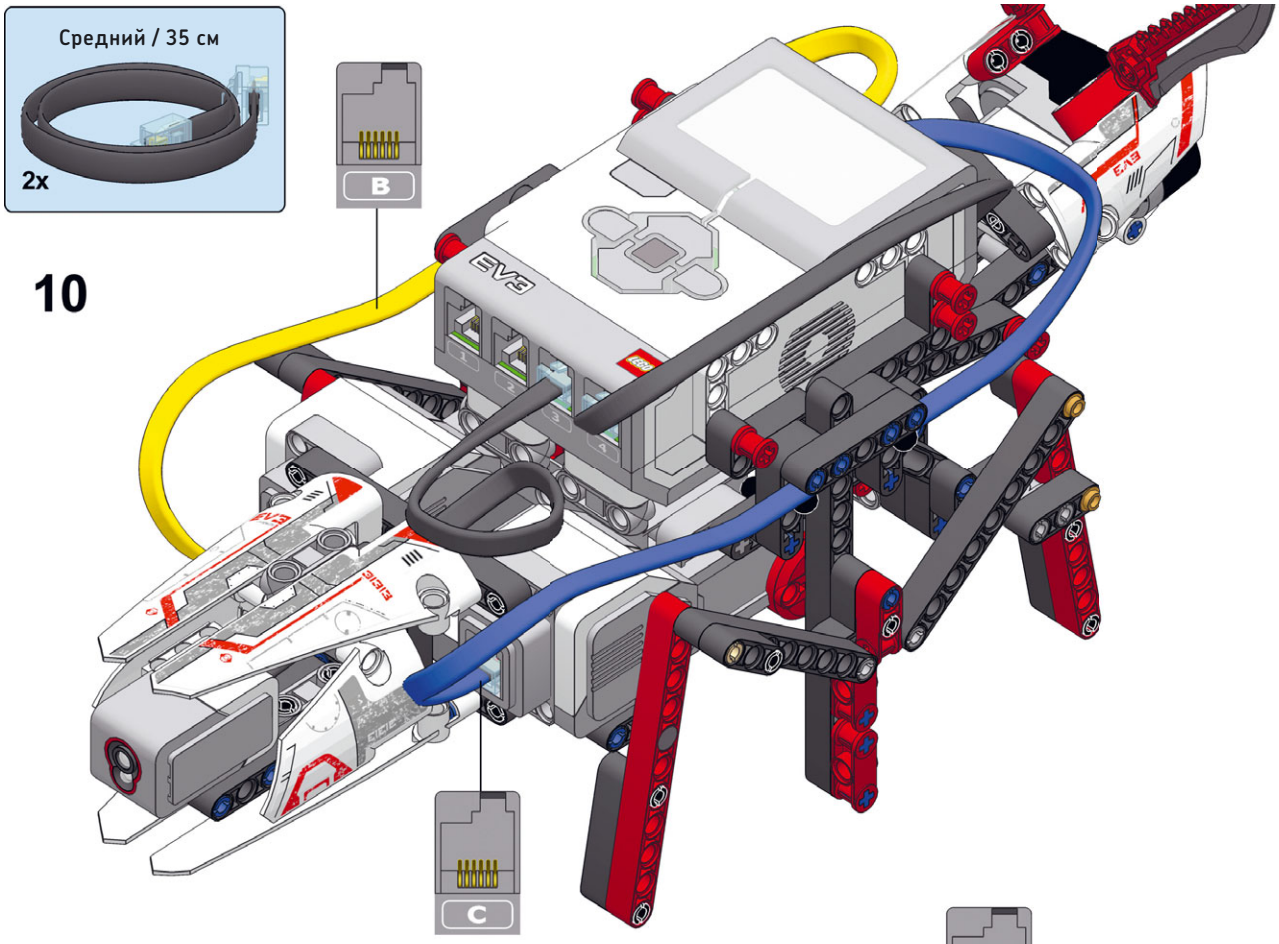


7

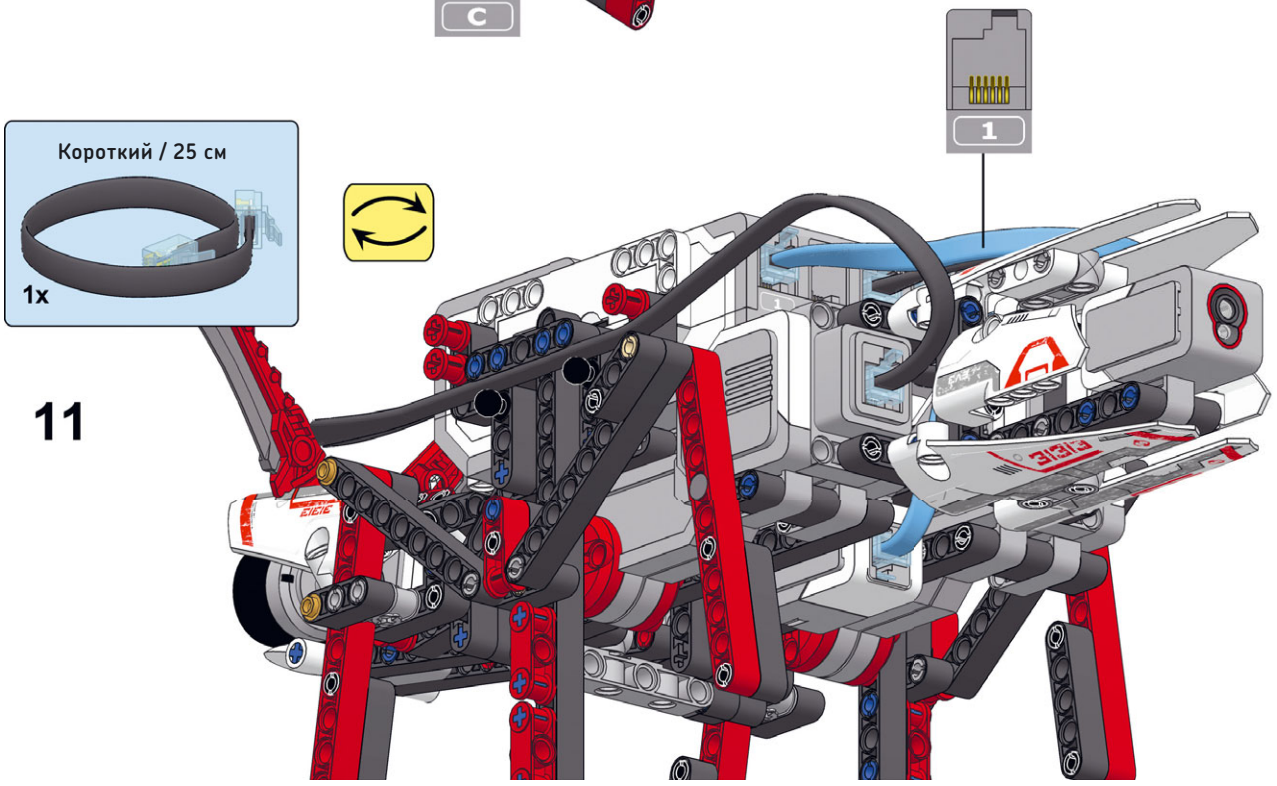





Короткий / 25 см
1x
2x
8
9
3
4



10



11

Ставим ANTY на ноги

Вы уже знаете, что ANTY передвигается вперед благодаря тому, что оба мотора начинают одновременно вращаться вперед, но только после того, как ноги по обеим сторонам его туловища установлены в противоположные позиции (с разницей 180 градусов). Создадим контейнер «Мой блок», чтобы установить правые ноги в положение 1, а левые — в положение 3 (см. рис. 13.2), не забывая о том, что, когда мотор находится в положении 1, он касается кнопки датчика касания. Используйте этот блок в начале всех программ для робота ANTY.

Как только работа блока завершится, блоки **Независимое управление моторами** (Move Tank) заставят робота двигаться, вращая моторы с одинаковой скоростью так, чтобы они все время оставались в противоположных положениях.

Создание противоположных контейнеров «Мой блок»

Так как для определения абсолютного положения обоих моторов используется один и тот же датчик касания, робот не знает, какой мотор касается датчика в данный момент: левый (B), правый (C) или оба. Но положение каждого мотора можно определить, если заставить моторы двигаться поочередно, выполнив по порядку следующие шаги.

1. Одновременно вращайте оба мотора, пока кнопка датчика касания не будет отпущена (робот может сделать это, поочередно вращая моторы вперед на чуть-чуть, пока кнопка датчика касания не будет отпущена).
2. Поворачивайте левый мотор вперед, пока он не коснется кнопки датчика касания. Левые ноги теперь находятся чуть дальше положения 1.
3. Поворачивайте левый мотор на 180 градусов вперед. Левые ноги теперь находятся чуть дальше положения 3.
4. Поворачивайте правый мотор вперед, пока он не коснется кнопки датчика касания. Правые ноги теперь находятся чуть дальше положения 1, точно напротив левых ног.

Создайте новый проект с именем *ANTY*. Разместите и конфигурируйте блоки, выполняющие описанные выше шаги, как показано на рис. 13.4. Затем объедините их в контейнер «Мой блок» с именем **Opposite** и запустите его, чтобы протестировать.

Если контейнер «Мой блок» не устанавливает ноги в противоположные положения, даже несмотря на то что ваша программа выглядит точно так же, как на рисунке, возможно, вы неправильно подсоединили кулачок. Вернитесь на с. 205 и исправьте конструкцию робота, если необходимо. Необязательно разбирать всего робота. Проще оставить кулачок на месте и просто отсоединить средние ноги от моторов, а затем присоединить их, как показано в инструкции.

Избегание препятствий

Прикрепив ноги на место, запрограммируйте робота двигаться вперед с помощью блоков **Независимое управление моторами** (Move Tank), сконфигурированных так, чтобы они вращали оба мотора вперед с одинаковой скоростью.

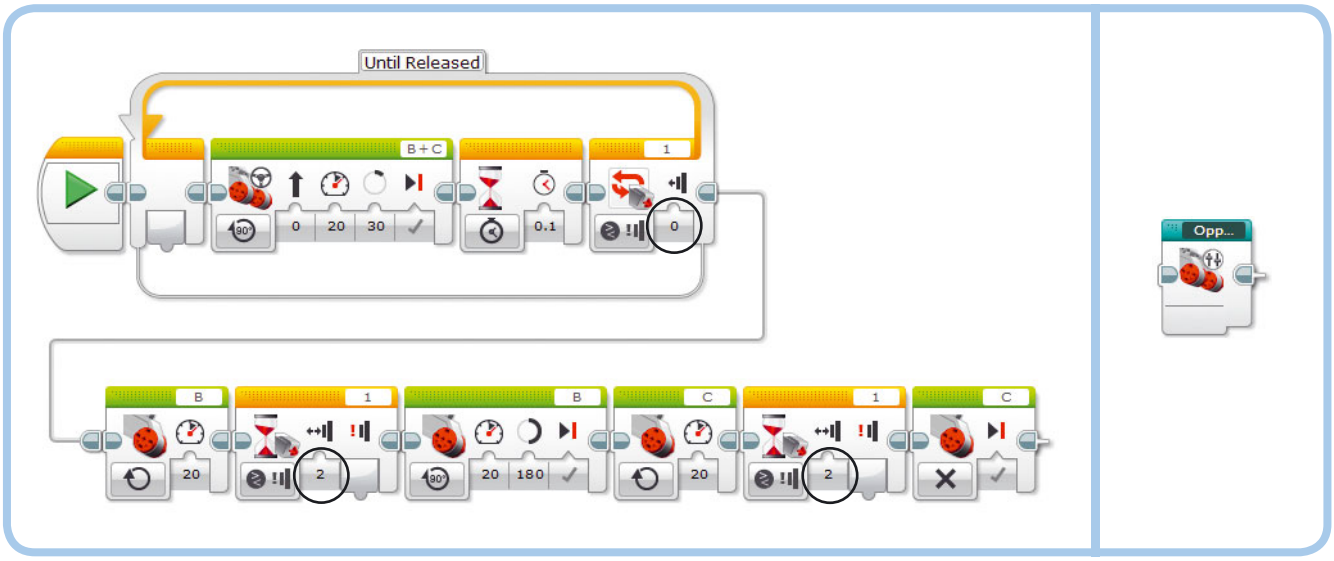


*Рис. 13.4. Конфигурация блоков (слева) и завершенный контейнер «Мой блок» с именем **Opposite** (справа). Я использовал соединитель, чтобы разделить программу на две части для наглядности, но вам этого делать необязательно*

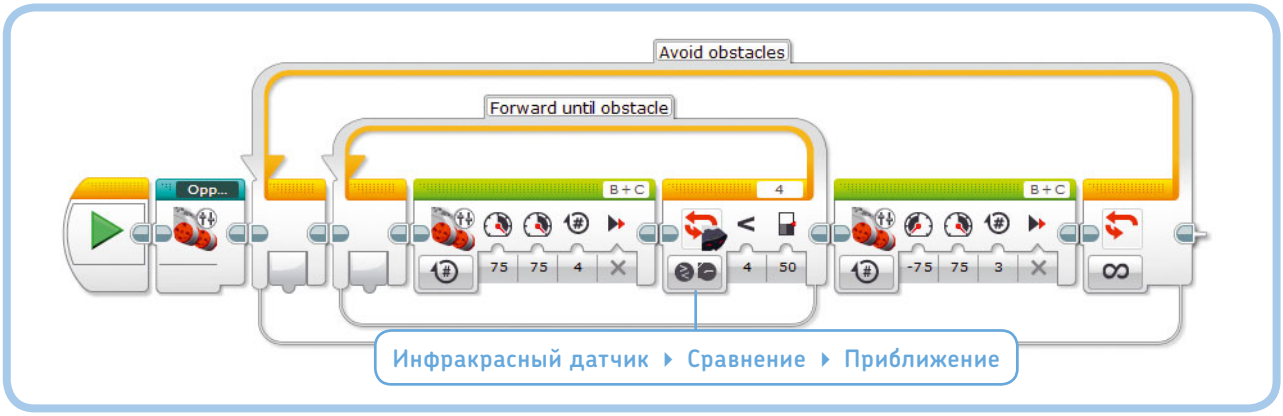


Рис. 13.5. Программа ObstacleAvoid

Если один из моторов будет вращаться назад, робот сможет повернуть направо или налево. Можно использовать эту технику для создания программы избегания препятствий. Вместо того чтобы использовать блоки **Рулевое управление** (Move Steering) в режиме **Включить** (On), как было при программировании робота EXPLOR3R, моторы в этой программе в режиме **Включить на количество оборотов** (On for Rotations) совершают четыре полных оборота. Движение повторяется, пока значение приближения инфракрасного датчика не окажется менее 50%. Затем робот поворачивает налево, включая обратный ход левого мотора и поворачивая правый мотор вперед на три оборота.

Создайте программу ObstacleAvoid, показанную на рис. 13.5. Обратите внимание, что необходимо вращать оба мотора с одинаковой скоростью на фиксированное число полных витков, чтобы ноги всегда оказывались в конечных противоположных положениях, готовые делать новые шаги. Если вы программируете движение, в результате которого моторы окажутся в другом положении, запустите контейнер «Мой блок» с именем **Opposite** еще раз, чтобы вернуть ноги в исходное положение.

ПРИМЕЧАНИЕ Робот ANTY лучше всего передвигается по гладким поверхностям — плиткам или деревянному полу. Если роботу трудно шагать вперед, попробуйте присвоить параметру Мощность (Power) в блоках Независимое управление моторами (Move Tank) значение 40, а не 75.

Программирование поведения

Теперь создайте программу, которая инструктирует робота демонстрировать различное поведение в зависимости от изменений окружающей среды. Начнем с самого сложного — поиска пищи, чтобы можно было протестировать части программы по отдельности. Когда будете готовы, объедините все программы поведения робота.

Поиск пищи

Мы имитируем поиск пищи, заставив робота искать инфракрасный маяк. Используйте методы, изученные в главе 8. Робот ANTY ищет маяк, идет к нему и останавливается, когда подойдет. Для достижения этой цели робот делает два шага влево или вправо, в зависимости от того, с какой стороны он видит маяк, а затем — два шага вперед. Он повторяет это поведение, пока не обнаружит маяк поблизости, и это означает, что он добрался до цели.

Создайте программу FindingFood, показанную на рис. 13.6. Убедившись, что программа работает, добавьте блок **Цикл** (Loop) вместе с содержимым в контейнер «Мой блок» с именем **Find**, чтобы его можно было использовать в следующей программе. Не добавляйте контейнер **Opposite** в контейнер «Мой блок» с именем **Find**, его мы установим в другую позицию в конце программы.

ПРИМЕЧАНИЕ Не забудьте нажать верхнюю кнопку инфракрасного маяка (кнопка с идентификатором 9), чтобы зеленый индикатор оставался включенным. Кроме того, помните, что маяк должен находиться на уровне глаз робота.

Восприятие окружающей среды

В законченном виде программа инструктирует робота ANTY демонстрировать различное поведение в зависимости от того, какой цвет он воспринимает с помощью датчика цвета, расположенного в его хвосте. Например, ANTY вздремнет, если вы будете держать рядом с датчиком зеленый объект, так как зеленый цвет означает для него безопасное поле травы.

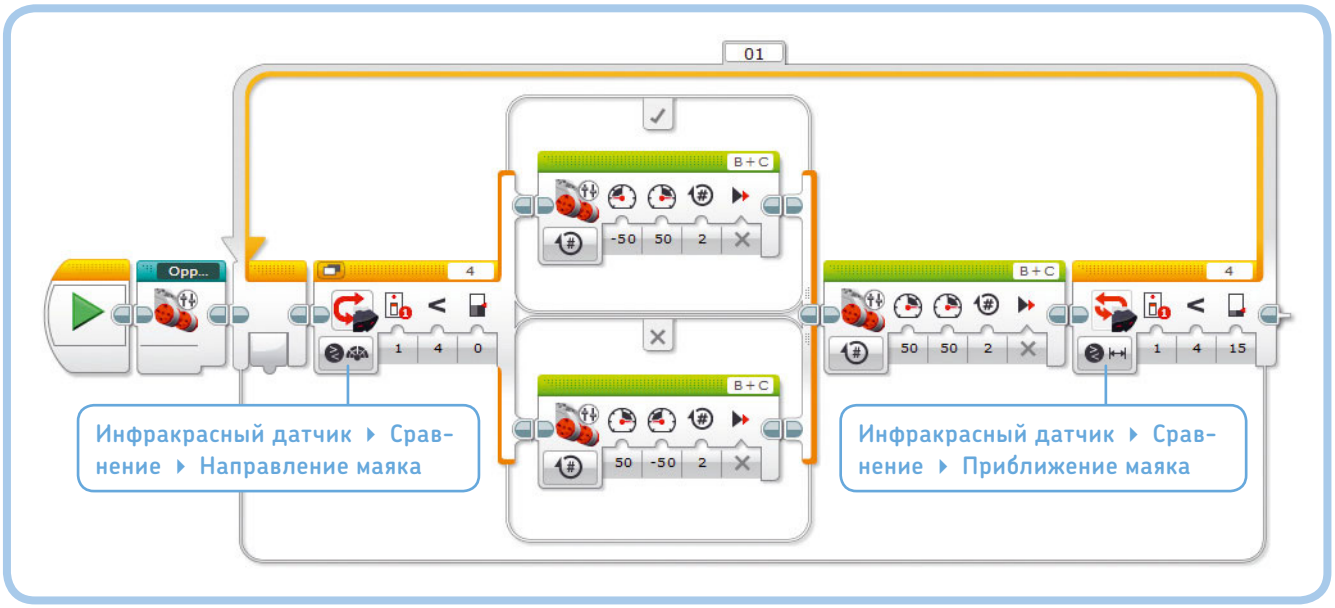


*Рис. 13.6. Программа FindingFood. Блок **Цикл** превратился в контейнер «Мой блок» с именем **Find***

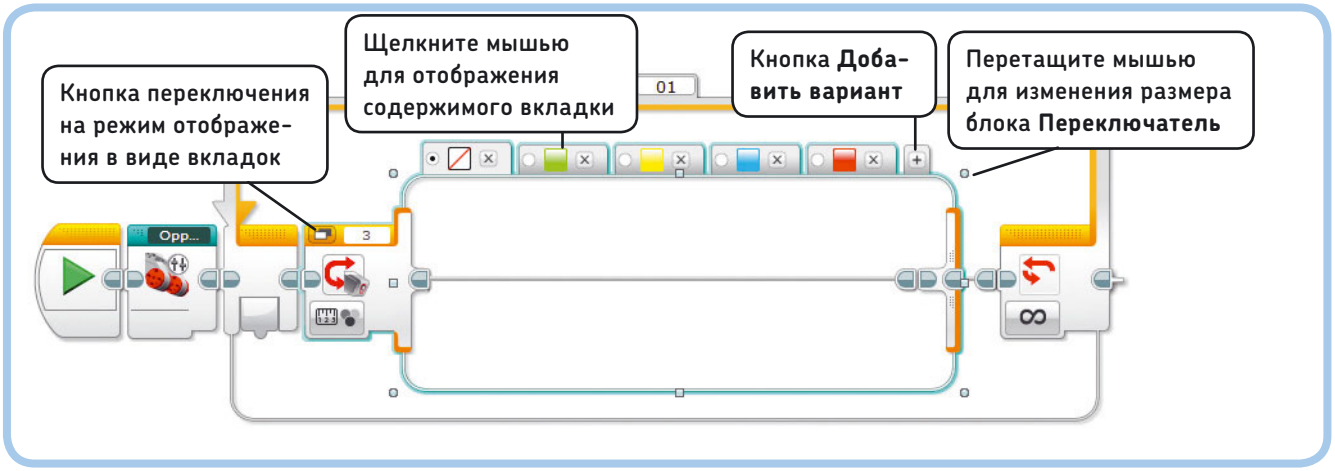


Рис. 13.7. Базовая структура программы ColorBehavior. Блоки будут добавлены на все вкладки

Начнем с создания базовой структуры программы ColorBehavior. Робот должен регулярно проверять, какой цвет он видит, и реагировать на каждый цвет по-разному. Запрограммируйте на это робота с помощью блока **Переключатель** (Switch) в режиме **Датчик цвета ▸ Измерение ▸ Цвет** (Color Sensor — Measure — Color), помещенного в блок **Цикл** (Loop), как показано на рис. 13.7.

Для переключения используется пять вариантов: **Нет цвета** (No Color), **Зеленый** (Green), **Желтый** (Yellow), **Синий** (Blue) и **Красный** (Red). По умолчанию установлен вариант **Нет цвета** (No Color), чтобы блоки на этой вкладке работали, если робот зафиксировал черный, белый или коричневый цвет. Следующий шаг — добавление блоков на каждую вкладку последовательности.

Нет цвета: спокойствие!

При отсутствии зеленого, желтого, синего или красного цветов робот ANTY просто сидит и издает звук, похожий на щебетание. Поместите блок **Звук** (Sound) на вкладку последовательности **Нет цвета** (No Color), как показано на рис. 13.8. Обратите внимание, что файл **Insect chirp** воспроизводит два звука и после воспроизведения второго датчик выполняет повторные измерения.

Зеленый: безопасность!

Когда робот ANTY чувствует зеленую траву на широком поле, он знает, что здесь можно безопасно вздремнуть.

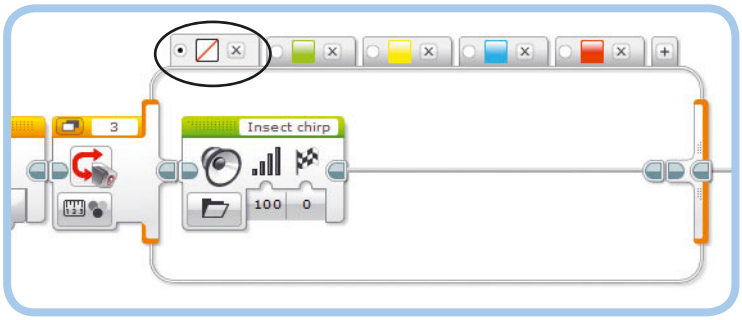


Рис. 13.8. Робот ANTY издает звук, похожий на щебетание, если не обнаруживает объекты зеленого, желтого, синего или красного цвета

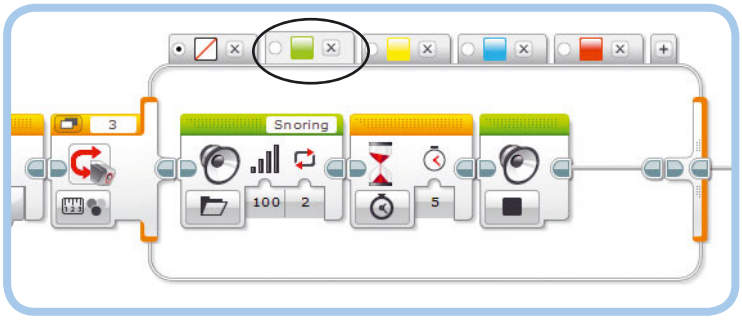


Рис. 13.9. Робот ANTY переходит в спящий режим, когда обнаруживает зеленый объект

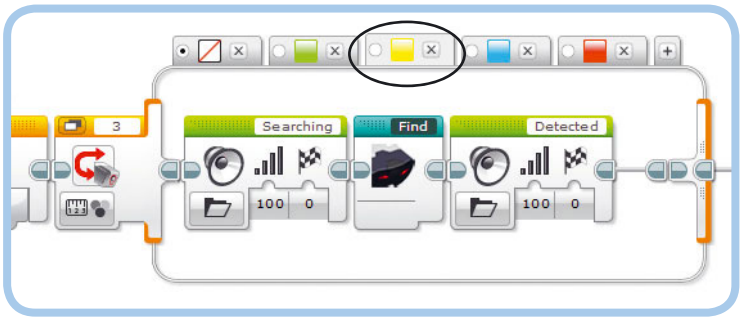


Рис. 13.10. Робот ANTY определяет местонахождение инфракрасного маяка и двигается к нему, если датчик цвета обнаруживает желтый цвет

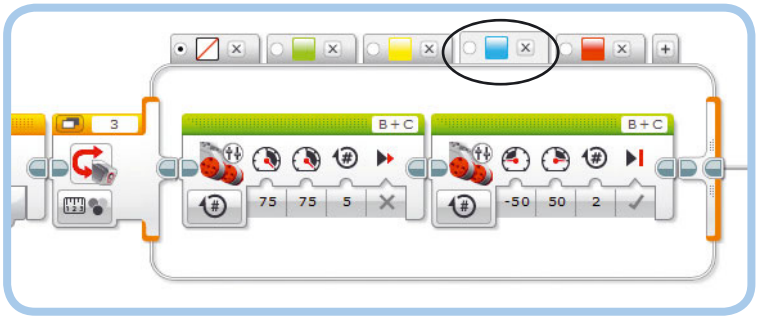


Рис. 13.11. Робот ANTY убегает при виде синего объекта. Обратите внимание, что моторы совершают фиксированное число вращений так, чтобы ноги робота остановились четко в противоположных позициях и робот был готов снова убегать

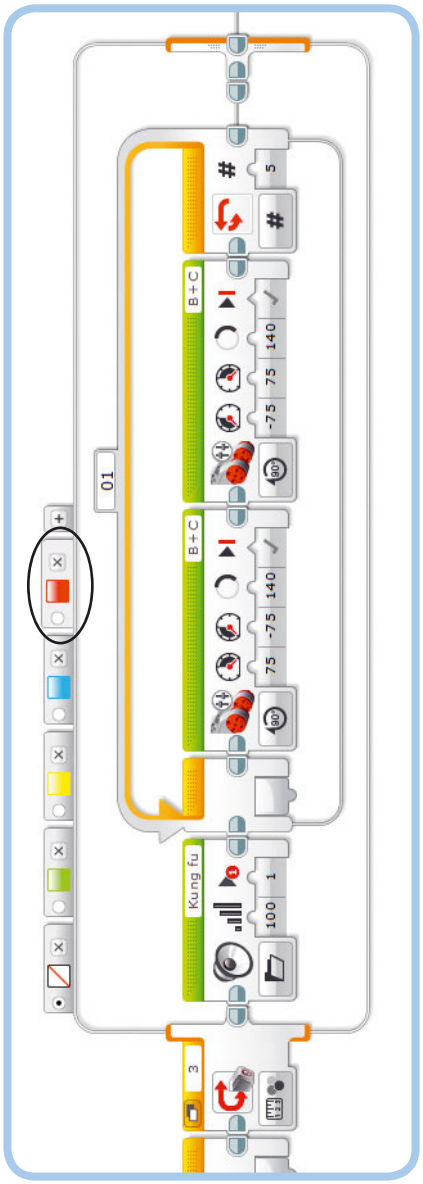


Рис. 13.12. Робот ANTY агрессивно встряхивается, когда «видит» красный предмет

Программа завершена. Запустите ее и протестируйте все варианты поведения робота ANTY, поднося предметы разного цвета к датчику цвета.

Добавьте блоки, которые инструктируют робота перейти в спящий режим, на вкладку **Зеленый** (Green), как показано на рис. 13.9.

Желтый: питание!

Вид желтого объекта вызывает у робота ANTY голод, заставляя искать пищу (маяк) с помощью контейнера **Find**, как показано на рис. 13.10. Если вы еще не создали этот контейнер «Мой блок», прочитайте раздел «Поиск пищи» ранее в этой главе.

ПРАКТИКУМ № 70: ДИСТАНЦИОННОЕ УПРАВЛЕНИЕ!

Сложность: **Время:**

Можно ли создать для робота ANTY программу дистанционного управления? Запрограммируйте робота передвигаться в любом направлении с помощью инфракрасного маяка. Используйте техники, изученные в главе 8.

СОВЕТ **Сделайте так, чтобы при нажатии кнопки в верхней части инфракрасного маяка программа запускала контейнер «Мой блок» с именем Opposite. Нажмите эту кнопку, чтобы изменить положение ног робота, если они расположены неправильно.**

ПРАКТИКУМ № 71: НОЧНОЕ СОЗДАНИЕ!

Сложность: **Время:**

Измените программу ObstacleAvoid, скорректировав скорость передвижения робота ANTY в зависимости от времени суток. Пусть он перемещается ночью с обычной скоростью, на рассвете и в сумерках — крадется, а днем — сидит неподвижно. Используйте режим **Яркость внешнего освещения** (Ambient Light Intensity) для определения времени суток.

СОВЕТ **Протестируйте программу в темной комнате, имитируя ночь. Включите только настольную лампу, чтобы имитировать предрассветное время, а затем включите все освещение, чтобы имитировать дневной свет. Как рассчитать пороговые значения для программы?**

Синий: опасность!

Синие объекты указывают на присутствие хищников. Лучший вариант для робота ANTY — просто убежать с помощью блоков **Независимое управление моторами** (Move Tank), как показано на рис. 13.11. Отображенные на нем блоки заставляют робота сделать пять шагов вперед, а затем два шага влево.

Красный: агрессия!

Красный цвет вызывает у робота ANTY агрессию и заставляет его встряхиваться, чтобы отпугнуть врагов, как показано на рис. 13.12.

Дальнейшее изучение

С помощью EV3 можно конструировать не только транспортные средства и механизмы, но и роботизированных животных и иных существ. В этой главе вы построили шестиногого шагающего робота ANTY. Вы узнали, как запрограммировать его передвигаться с помощью двух больших моторов и датчика касания для определения абсолютного положения каждого мотора.

Вдобавок к конструированию шагающего робота вы создали программу, управляющую его поведением так, что робот ANTY может выполнять различные действия

ПРАКТИКУМ № 72: ГОЛОДНЫЕ РОБОТЫ!

Сложность: **Время:**

Можно ли создать программу, которая заставит робота передвигаться и огибать предметы, пока он не «проголодается»? «Проголодавшись», он должен будет подходить к инфракрасному маяку в поисках «пищи». Повторив эти действия трижды, робот должен перейти в спящий режим.

СОВЕТ **Определяйте, насколько робот ANTY «голоден», по тому, сколько оборотов сделали его моторы с момента последнего «обеда» (50 шагов — хорошее пороговое значение). После того как робот найдет пищу, сбросьте значение датчиков вращения на 0, чтобы восстановить его уровень «энергии».**

в зависимости от сигнала датчика, имитируя поведение настоящего животного.

Теперь попробуйте выполнить задания из практикумов, чтобы еще лучше изучить конструкцию этого робота. Насколько крутое существо вы сможете сконструировать?

СДЕЛАЙ САМ № 22: РОБОТ-ПАУК!

Сборка: **Программирование:**

Сможете ли вы создать робота-паука? Уберите у робота ANTY хвост и переставьте инфракрасные датчики и модуль EV3 так, чтобы тело робота стало более похожим на паучье. Вам нужно будет добавить в конструкцию еще две ноги, ведь у пауков их восемь. Когда все будет готово, прикрепите к телу и ногам робота-паука декоративные элементы, чтобы он выглядел более реалистичным и страшным.

СОВЕТ **Если вы не знаете, как заставить две дополнительные ноги двигаться, просто добавьте неподвижные элементы, похожие на ноги, а робот пусть передвигается на шести рабочих ногах. В этом случае убедитесь, что две новые ноги при ходьбе не касаются земли или других ног.**

СДЕЛАЙ САМ № 23: УСИКИ!

Сборка: **Программирование:**

Насекомые способны ощущать окружающую среду с помощью усиков, которые обычно есть у них на голове. Можно ли приделать роботу ANTY усики, которыми он будет определять наличие препятствий и отходить при приближении к ним вплотную? Датчик касания уже используется, но можете ли вы придумать способ определения контакта с предметом с помощью других датчиков?

СОВЕТ **Сконструируйте усики таким образом, чтобы, когда робот ANTY наталкивается на препятствие, они бы нажимали кнопки на инфракрасном маяке (рис. 13.13), и примените инфракрасный датчик для обнаружения нажатия кнопок.**

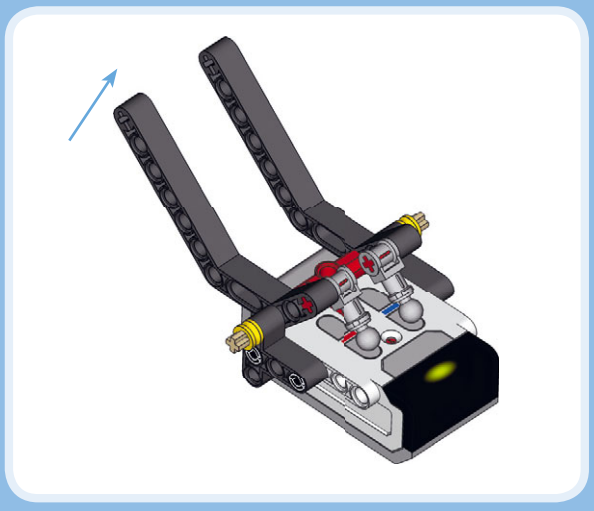


Рис. 13.13. Вы можете использовать удаленный инфракрасный маяк и инфракрасный датчик вместо датчика касания. Касание усика (синяя стрелка) приводит к нажатию кнопки на маяке. С помощью инфракрасного датчика определяйте, какая из кнопок (или обе) нажата

СДЕЛАЙ САМ № 24: ЖУТКИЕ КОГТИ!

Сборка: **Программирование:**

Можно ли улучшить конструкцию робота ANTY, добавив ему когти, чтобы он мог хватать предметы и перетаскивать их свое гнездо? С помощью среднего мотора когти должны раскрываться и сжиматься. В качестве дополнительного задания запрограммируйте робота искать инфракрасный маяк, подходить к нему и хватать его когтями.

СОВЕТ **Удалите или измените голову робота, чтобы освободить место для среднего мотора.**

ЧАСТЬ V

Разработка сложных программ

14

SK3TCHBOT: использование шин данных

В пятой части книги вы научитесь использовать шины данных (в этой главе), блоки операций с данными (глава 15) и переменные (глава 16), чтобы разрабатывать улучшенные программы для роботов. В главе 17 показано, как комбинировать эти методы и средства, чтобы создать большую программу, позволяющую запускать игру наподобие «Волшебного экрана» на модуле EV3. Вы сможете делать все это с помощью робота под названием SK3TCHBOT, который изображен на рис. 14.1.

В предыдущих главах вы выполняли настройку программных блоков путем ввода необходимых параметров вручную. Одним из основных понятий в данной главе является то, что блоки могут настраивать друг друга, передавая данные с одного блока на другой с помощью *шин данных*. Например, один блок может измерять значение параметра **Приближение** (Proximity) инфракрасного датчика и отправлять результат в блок **Большой мотор** (Large Motor). Блок **Большой мотор** (Large Motor) может использовать полученное значение для установки скорости мотора. Как следствие, обороты мотора будут низкими при малых значениях параметра **Приближение** (Proximity) (значение 27% приводит к показателям скорости в 27%) и высокими — при высоких (85% приводят к 85% скорости).

Эта глава научит вас создавать программы, которые используют шины данных. Поначалу это может показаться сложным, но, как только вы проработаете описанные программы и выполните практикумы в конце глав, вы овладеете всеми этими способами программирования!

Рис. 14.1. Вы будете использовать SK3TCHBOT, для того чтобы изучить множество новых способов программирования, и для игры в «Волшебный экран» на экране модуля EV3

Сборка робота SK3TCHBOT

Вы будете использовать робота SK3TCHBOT для тестирования своих программ с помощью шин данных и переменных. Робот состоит из модуля EV3, трех датчиков и двух больших моторов, которые вы будете использовать как вводы и выводы для своих программ, демонстрирующих шины данных в действии. В главе 17 вы создадите программу, которая превращает SK3TCHBOT в устройство, подобное «волшебному экрану», позволяя вам создавать рисунки на экране поворотами стрелок ввода, прикрепленных к моторам.

Соберите робота SK3TCHBOT, следуя инструкциям далее, но сначала выберите необходимые детали, как показано на рис. 14.2.

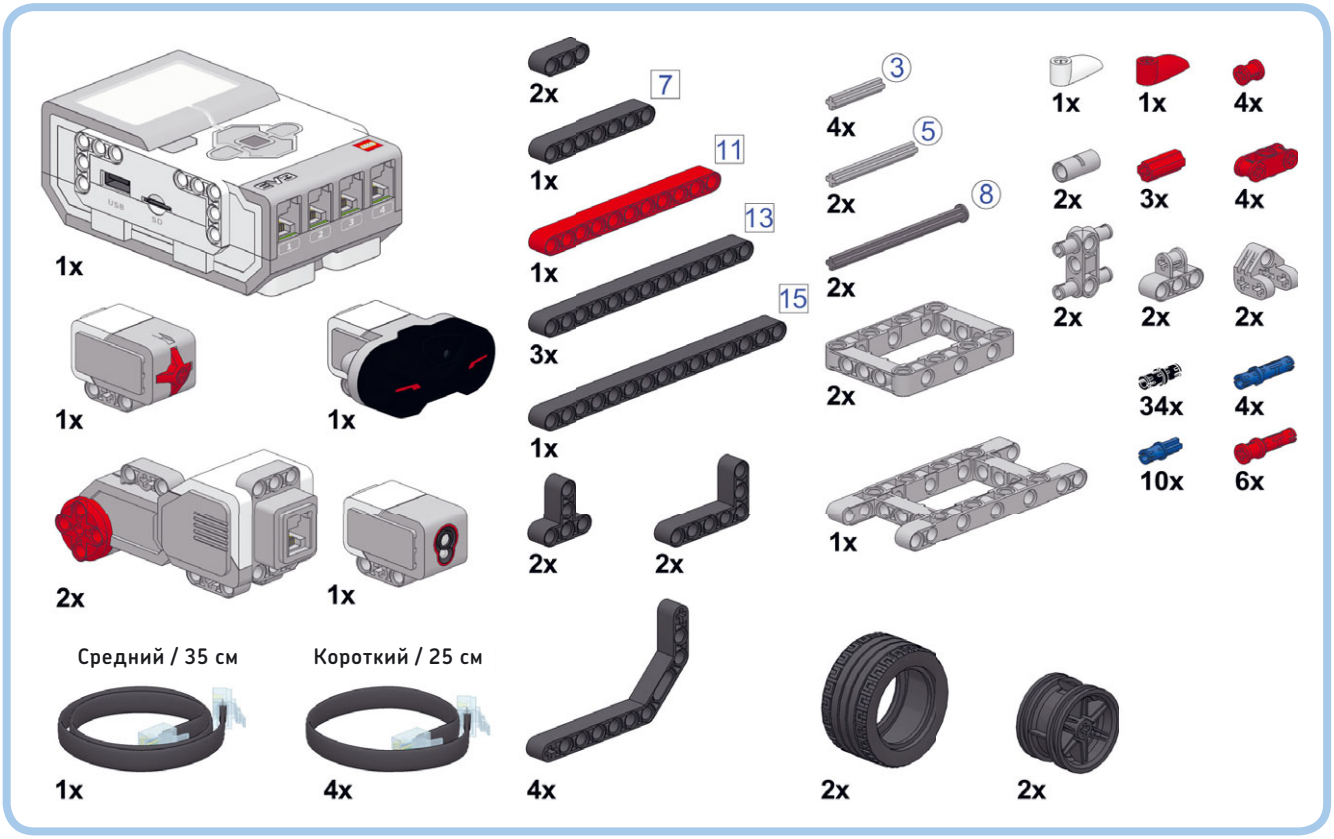
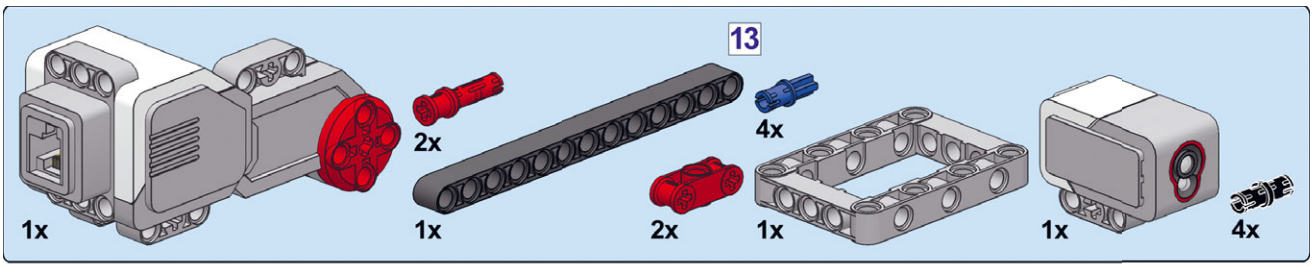
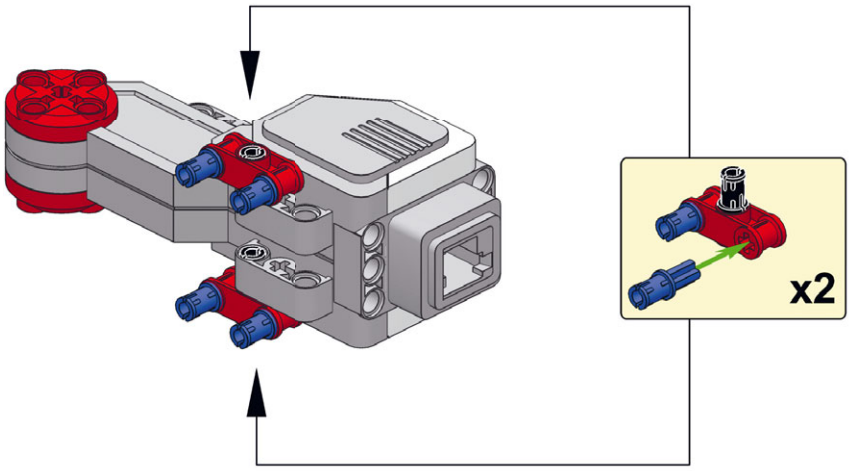


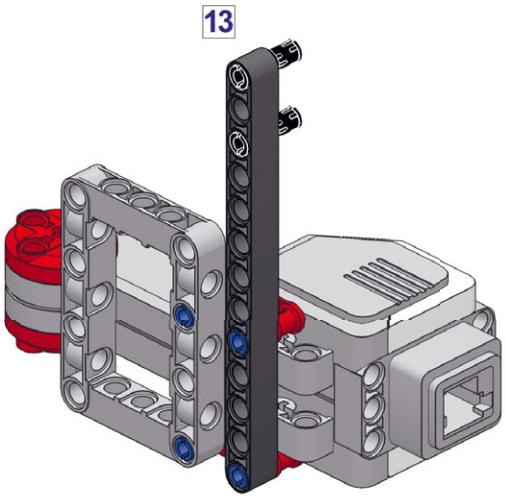
Рис. 14.2. Детали, необходимые для сборки робота SK3TCHBOT



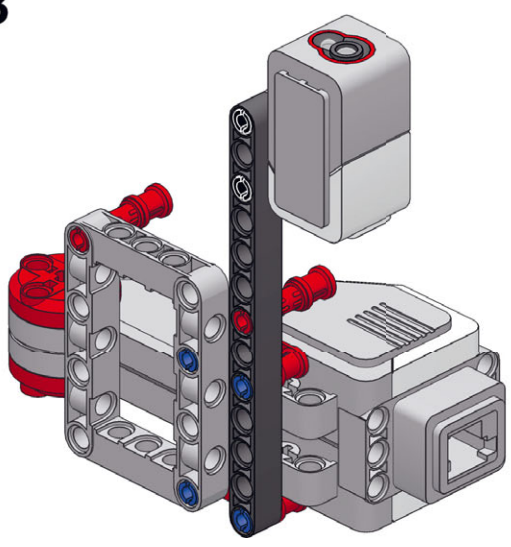
1

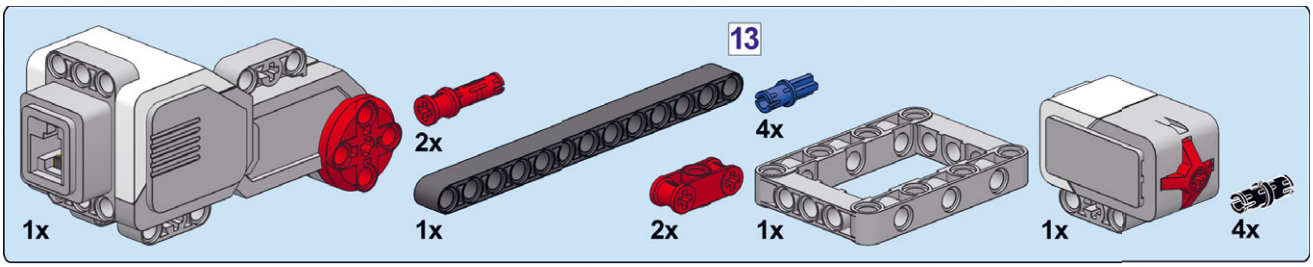


2

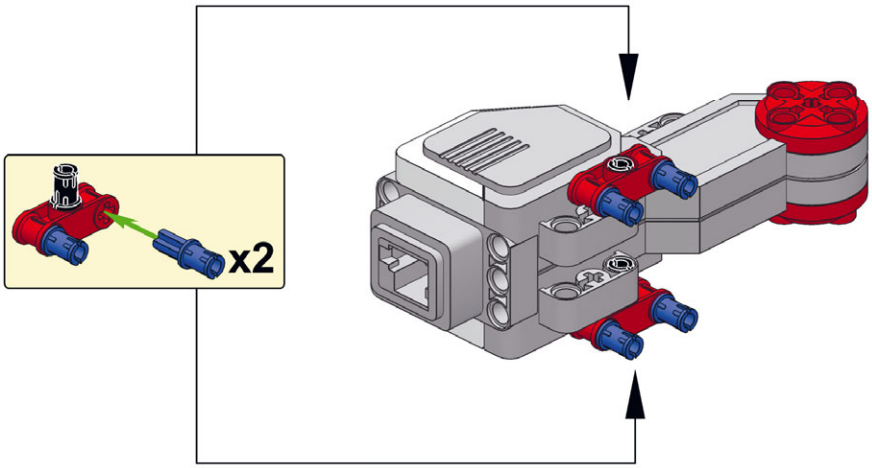


3

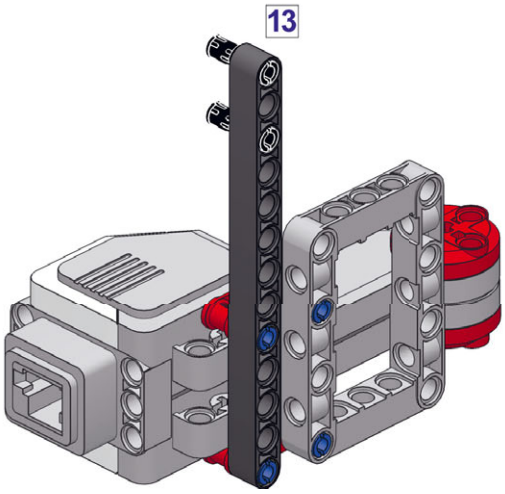




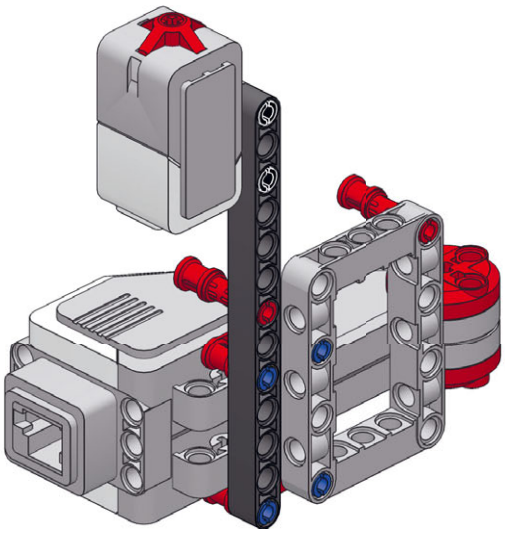
1

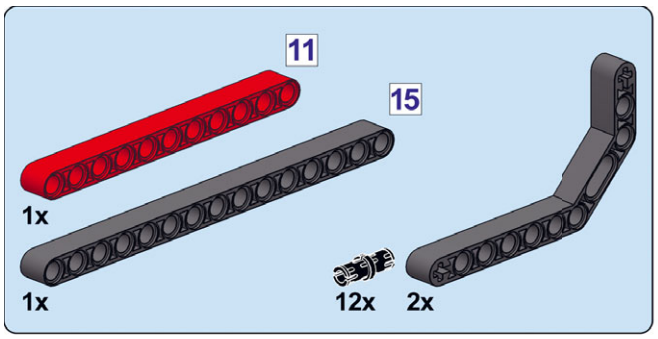


2

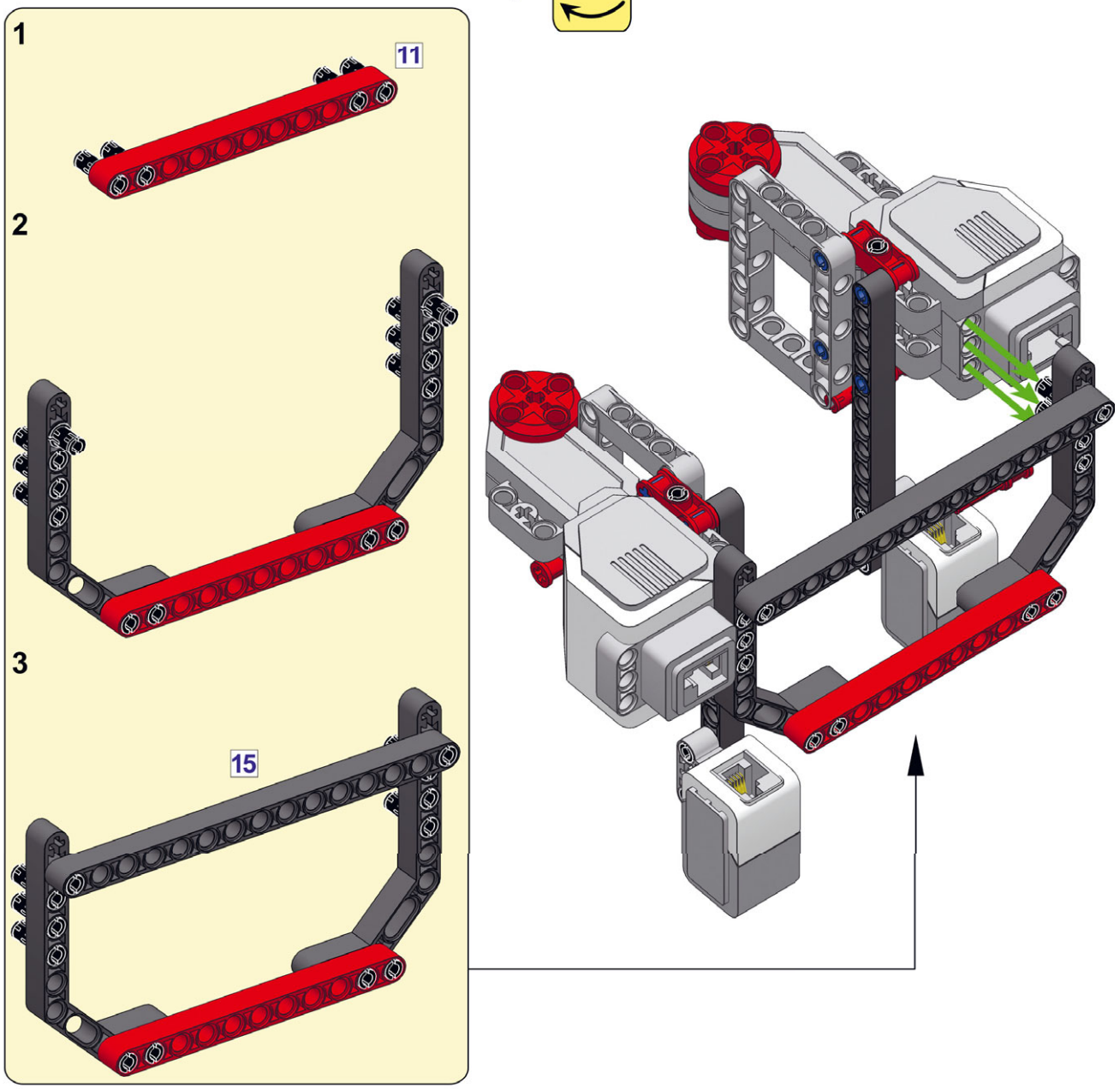


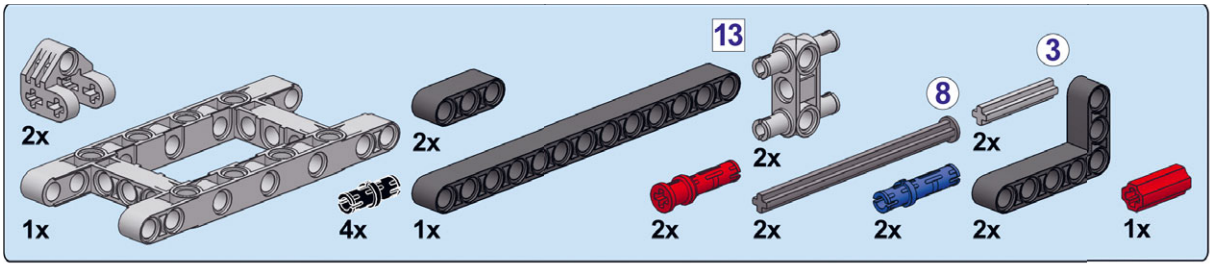
3

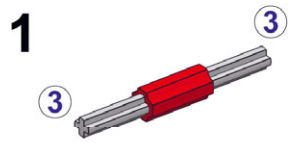


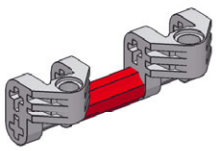


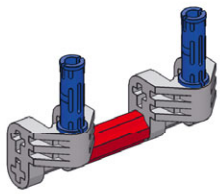
4

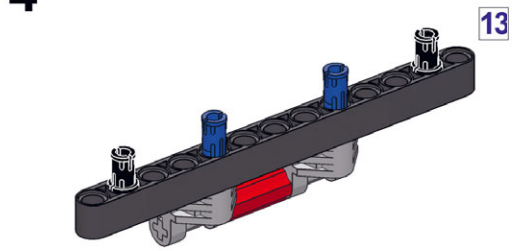


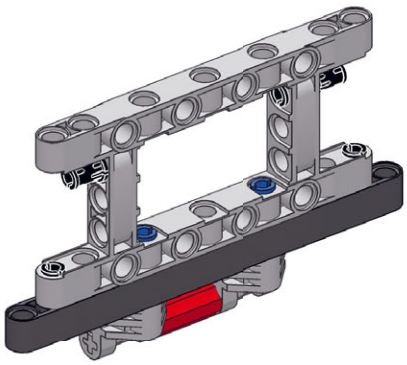

2x
1x
4x
2x
1x
13
2x
2x
8
2x
2x
3
2x
2x
1x

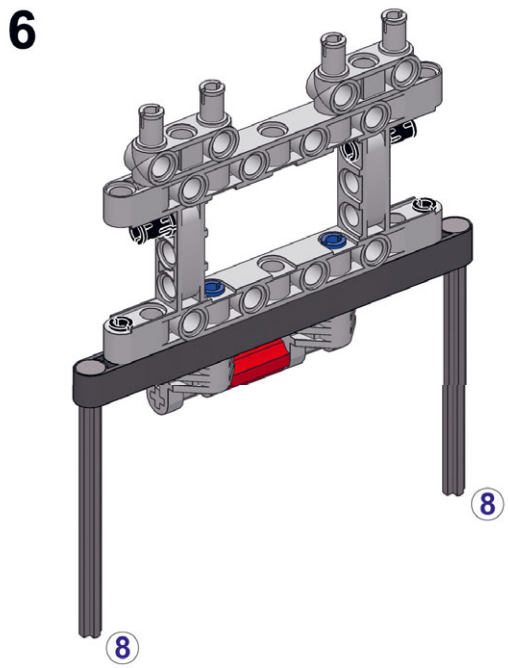

1
3
3

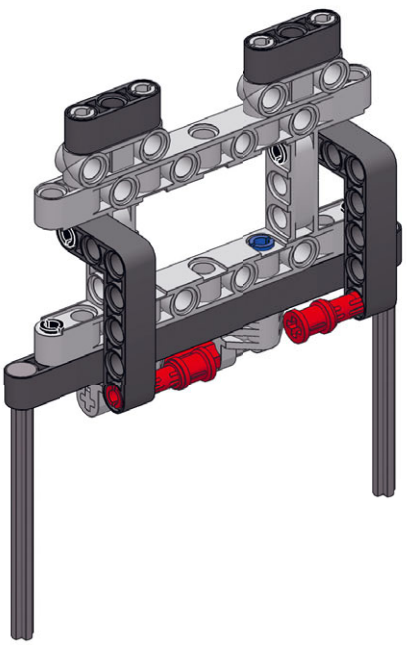

2


3

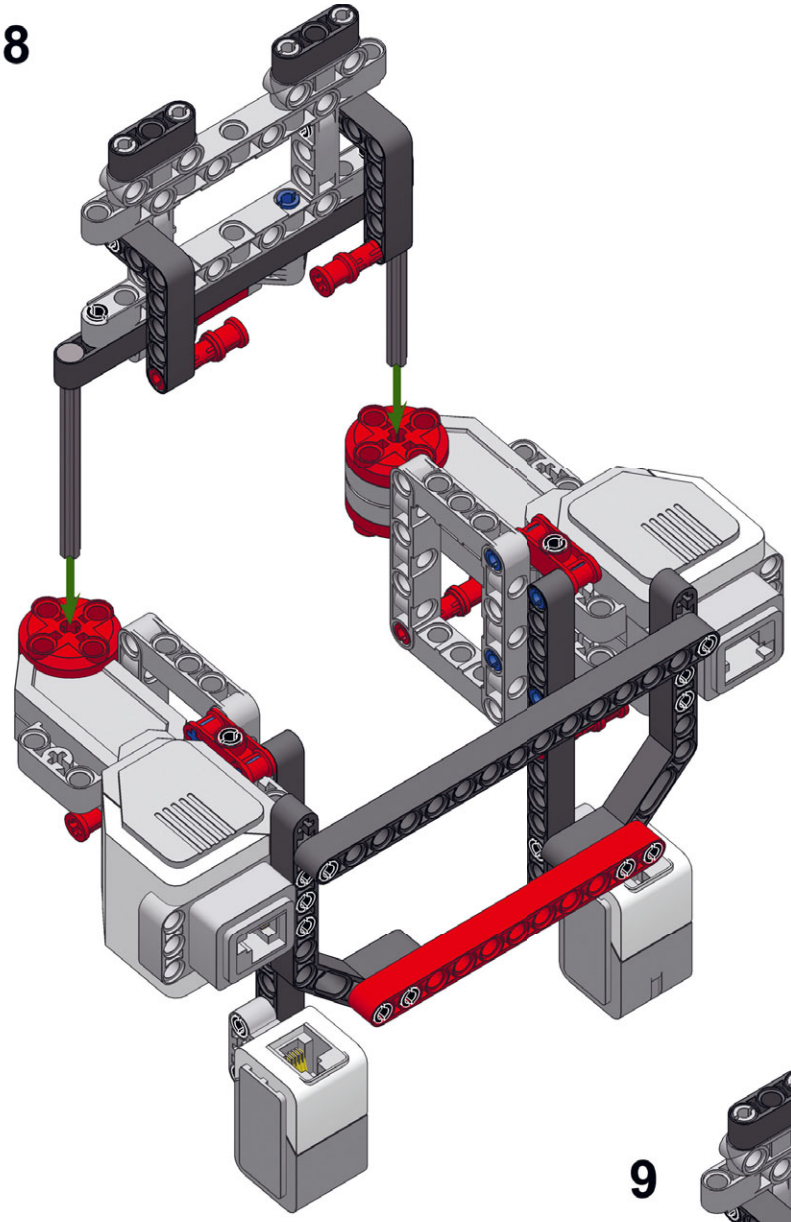

4
13


5

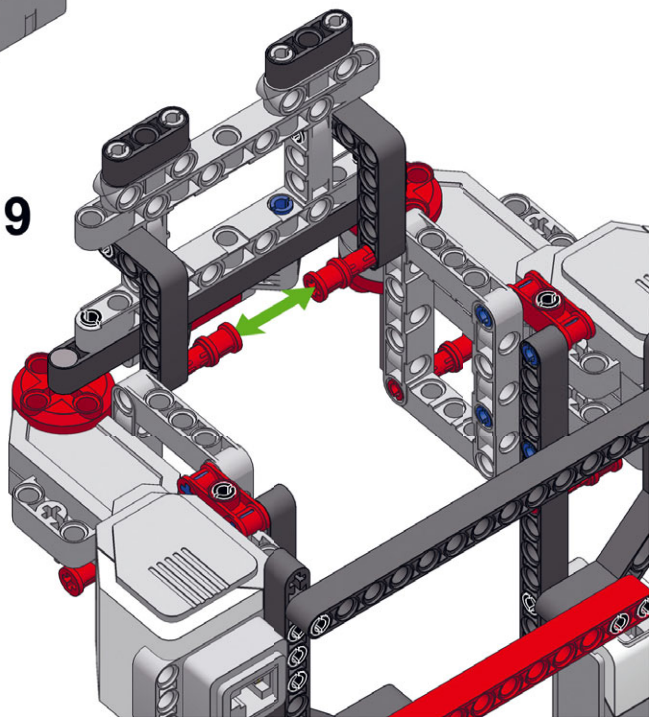

6
8
8

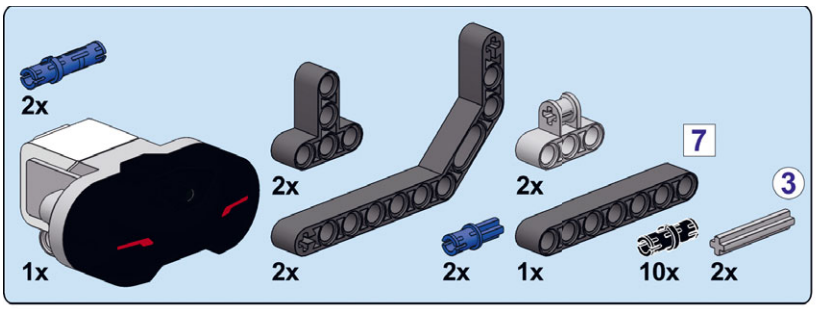

7

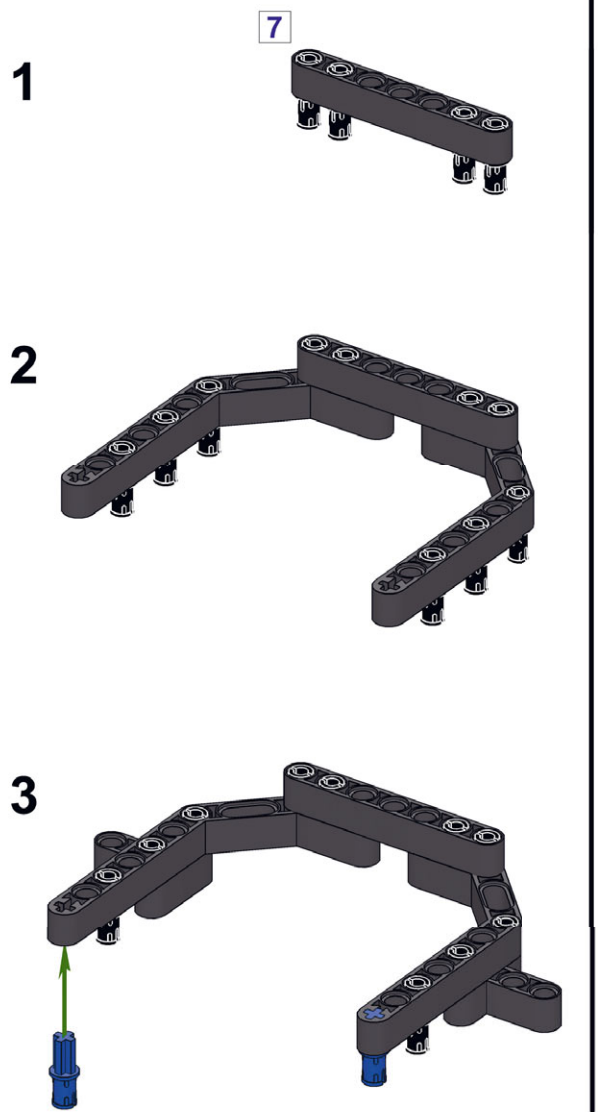
8

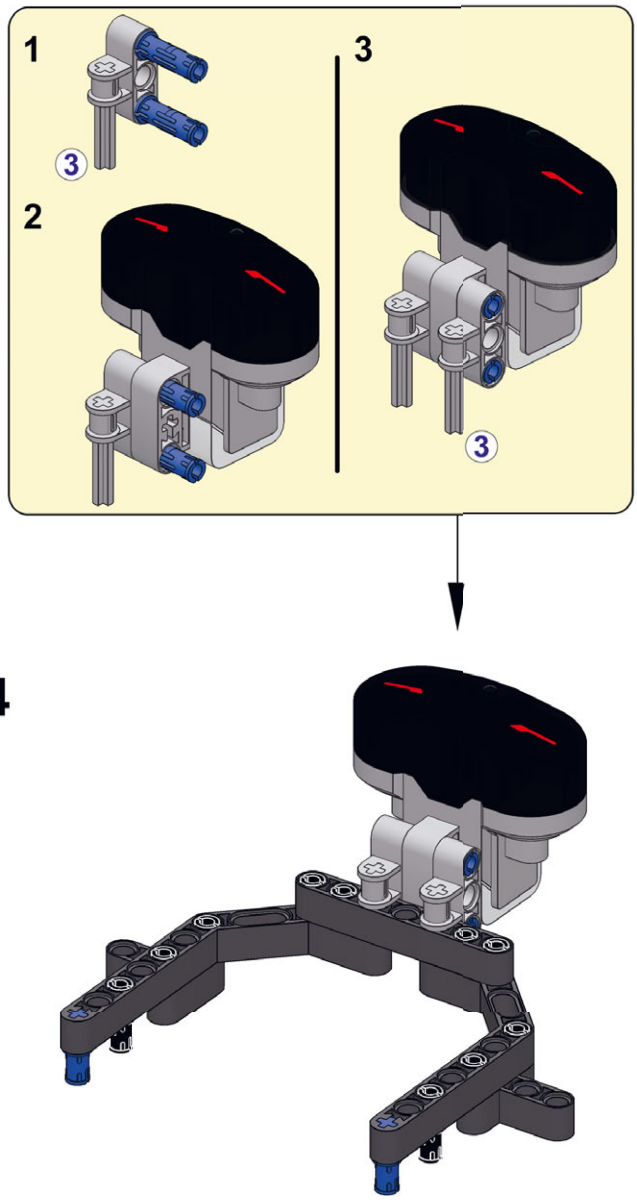


9

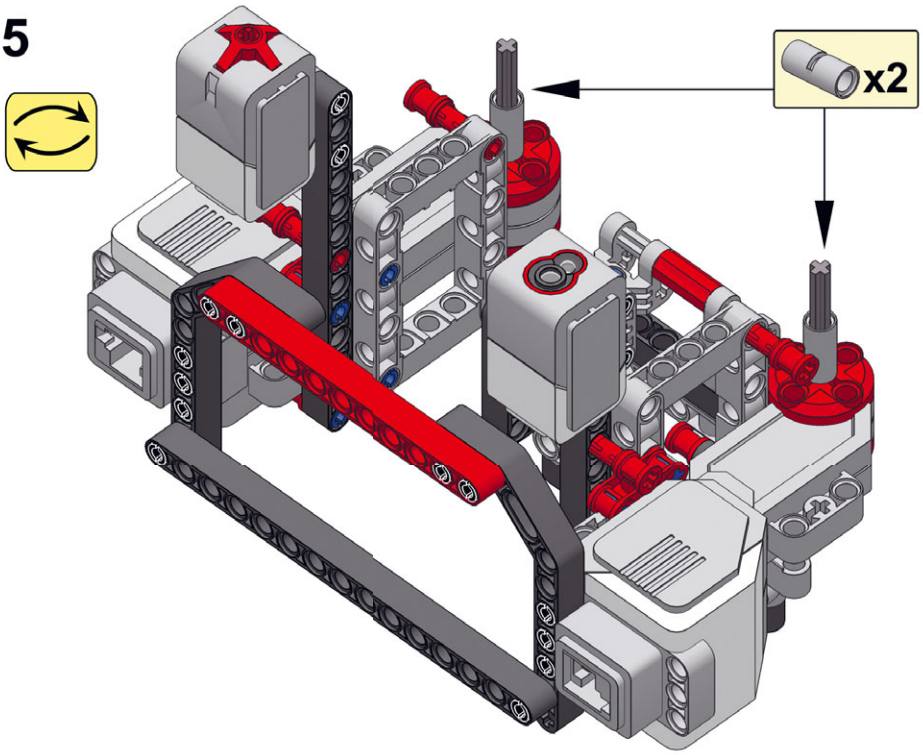



2x
1x
2x
2x
2x
1x
7
10x
3
2x

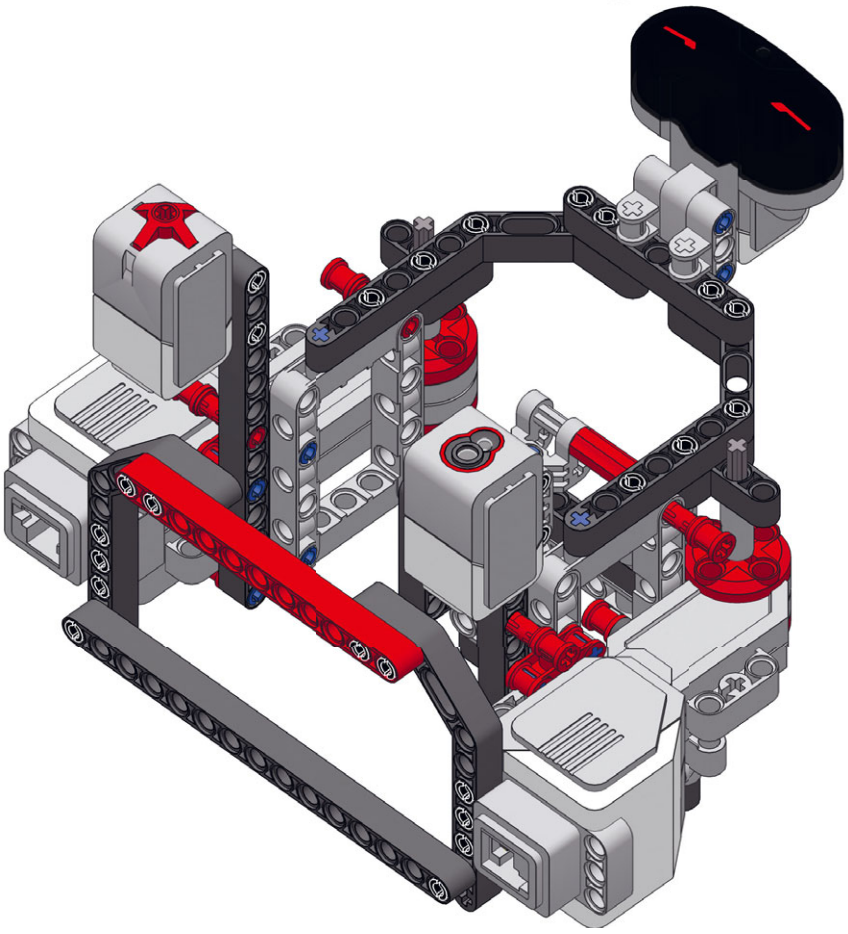

1
7
2
3

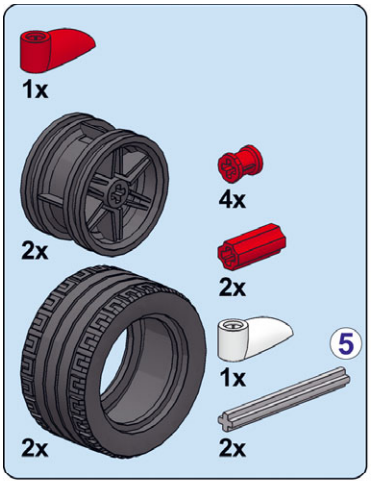

1
3
2
3
3
4

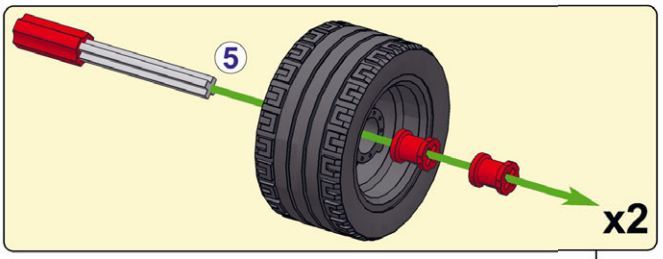
5



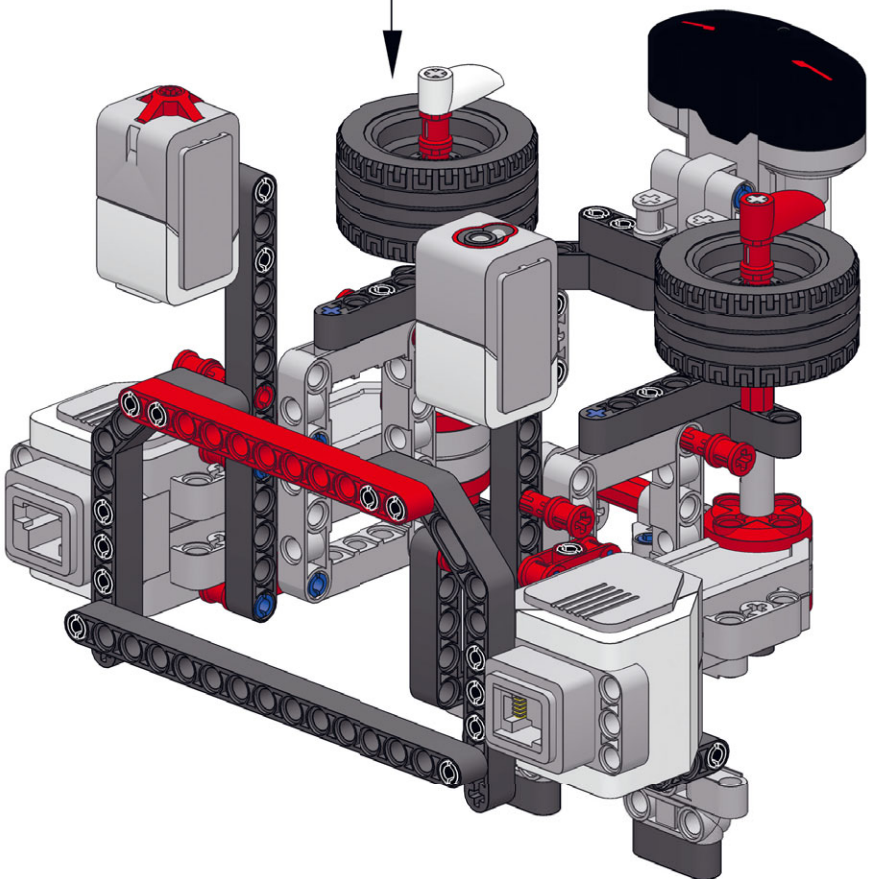
6







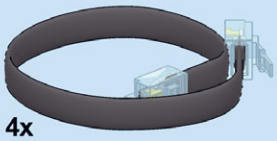
7



Средний / 35 см

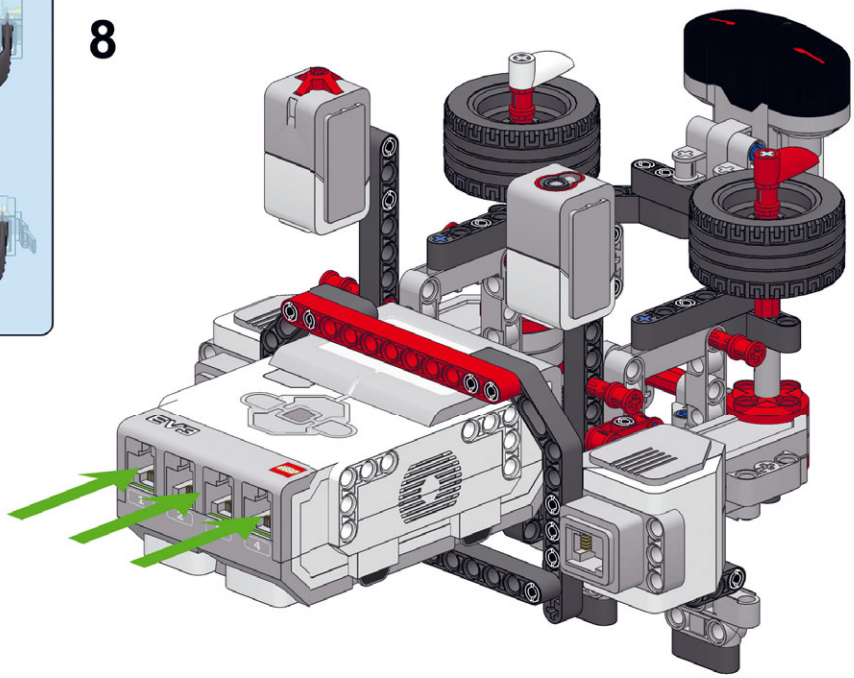
1x

Короткий / 25 см



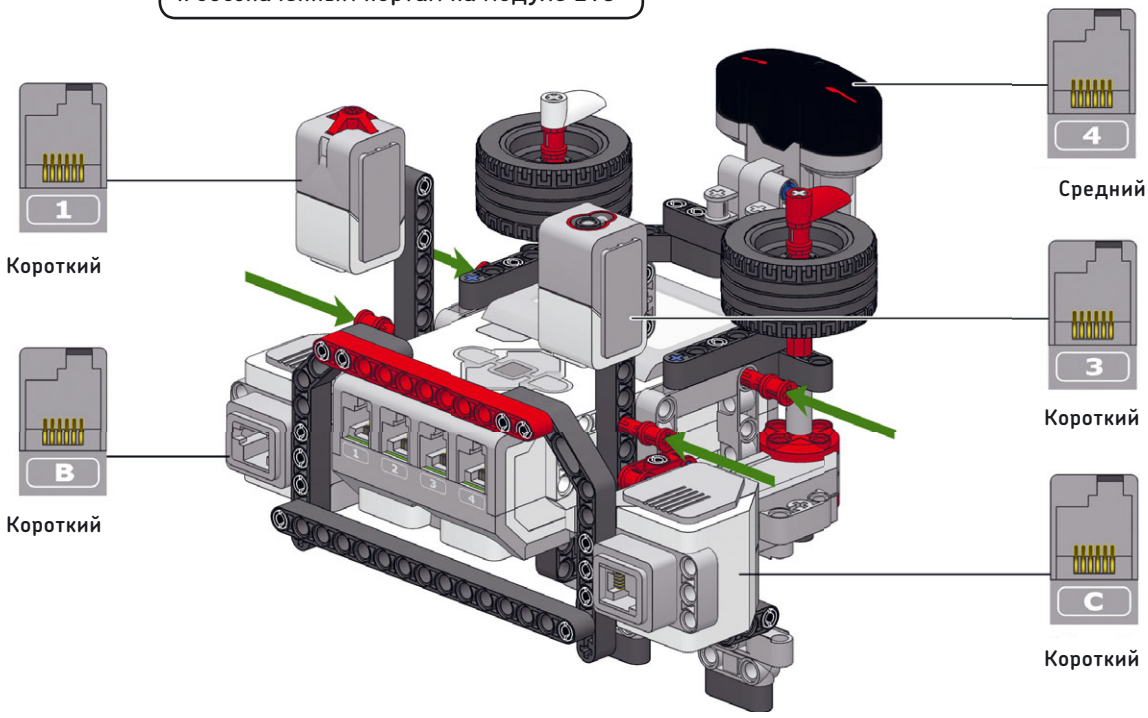
4x

8



9

Вставьте штифты красного цвета в модуль EV3, как показано на рисунке. Затем подключите моторы и датчики к обозначенным портам на модуле EV3



Начало работы с шинами данных

Чтобы увидеть, как работают шины данных, вы создадите небольшую программу, которая инструктирует SK3TCHBOT воспроизводить звук, а затем вращать белую стрелку в течение трех секунд. Параметр **Мощность** (Power) мотора будет настраиваемым, и, следовательно, скорость будет соответствовать значению, фиксируемому инфракрасным датчиком: если значение датчика равно 27%, то и скорость мотора будет 27%; а если датчик считывает 85%, то и скорость мотора будет равна 85%; и так далее.

Вы примените блок **Инфракрасный датчик** (Infrared Sensor), чтобы считывать данные датчика. О блоках датчиков вы узнаете позже. Создайте новый проект под названием SK3TCHBOT-Wire с программой FirstWire, как показано на рис. 14.3 и 14.4, и загрузите ее в своего робота.

Следует отметить, что белая стрелка вращается с различной скоростью каждый раз, когда вы запускаете программу. Например, если вы будете держать руку в непосредственной близости от датчика и запустите программу, колесо будет вращаться медленно. Но оно должно вращаться быстрее, если вы снова запустите программу, а руку отведете на некоторое расстояние.

Поздравляем! Вы только что создали свою первую программу, использующую шины данных. Давайте рассмотрим, как работает каждый блок. Первый блок, **Звук** (Sound), просто проигрывает звук. Как только воспроизведение звука закончено, блок **Инфракрасный датчик** (Infrared Sensor) выполняет одно измерение (скажем, он считывает значение приближения, равное 27%). Желтая шина данных передает измерение датчика в блок **Большой мотор** (Large Motor), который инструктирует мотор B вращаться в течение

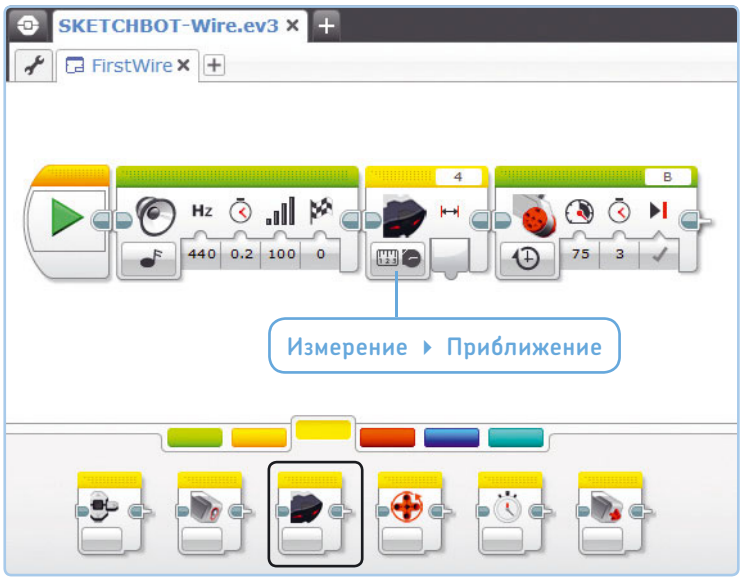


*Рис. 14.3. Шаг 1. Разместите все необходимые блоки в области программирования и скомпонуйте их, как показано на рисунке. Вы найдете блок **Инфракрасный датчик** на желтой вкладке палитры программирования. Убедитесь, что выбран режим **Измерение ▸ Приближение***

ПРАКТИКУМ № 73: ЗВУК НА РАССТОЯНИИ!

Сложность: **Время:**

Удалите блок **Большой мотор** (Large Motor) из программы FirstWire и вставьте вместо него блок **Звук** (Sound), который настроен на произнесение слова «Привет»*. Перетащите шину данных из вывода блока **Инфракрасный датчик** (Infrared Sensor) к вводу **Том**** (Volume) блока **Звук** (Sound). Так вы запрограммируете робота произнести слово «Привет» тихо, когда вы находитесь близко, или громко, когда он определит, что вы далеко.

* Пункт **Hello** в каталоге звуков Lego.

** В русской версии программы слово Volume ошибочно переведено как «Том». Правильно — Громкость.

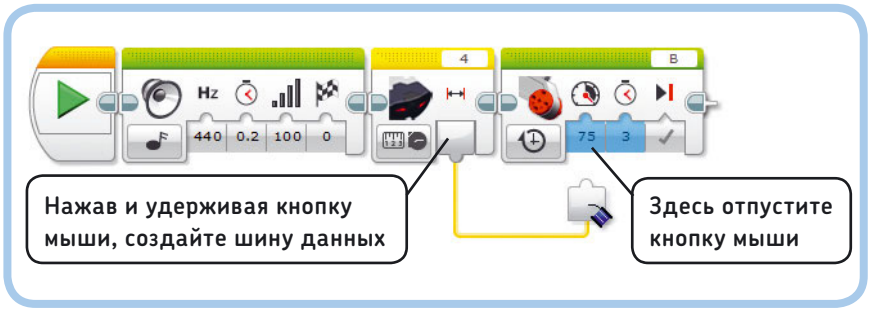


Рис. 14.4. Шаг 2. Нажав и удерживая кнопку мыши, перетащите желтую шину данных с вывода одного блока на ввод другого, как показано на рисунке. Затем запустите готовую программу

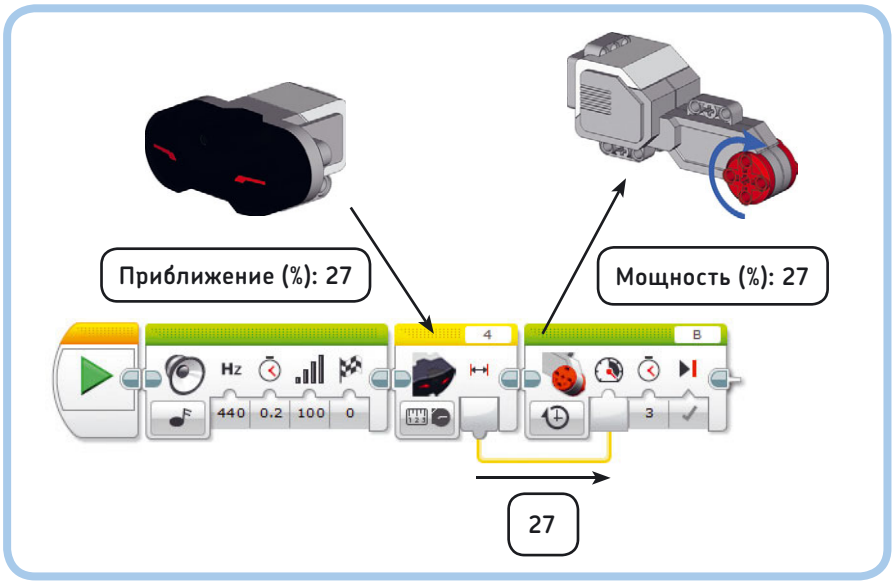


*Рис. 14.5. Блок **Инфракрасный датчик** считывает данные датчика и отправляет полученное значение по шине данных на блок **Большой мотор**, который использует это значение для установки скорости мотора*

3 секунд. Настройка параметра **Мощность** (Power) блока **Большой мотор** (Large Motor) зависит от значения, переданного с помощью шины данных; в данном случае это 27%. На рис. 14.5 можно увидеть, что именно происходит.

Работа с шинами данных

Как вы увидели, шины данных используются для передачи информации между блоками. Данные всегда передаются из *вывода* на *ввод*, как показано на рис. 14.6. В программе, используемой в качестве примера, шина данных передает значение датчика из вывода **Приближение** (Proximity) блока **Инфракрасный датчик** (Infrared Sensor) на ввод **Мощность** (Power) блока **Большой мотор** (Large Motor).

Обратите внимание, что шина данных скрывает значение, которое вы изначально присвоили параметру **Мощность** (Power) (**75**). Программа игнорирует это значение и вместо него использует значение, полученное из шины данных. С другой стороны, так как вы не подключили шины данных к другим вводам (**Секунды** (Seconds) и **Тормозить в конце** (Brake at End)) блока **Большой мотор** (Large Motor), они работают нормально. Так, вы присвоили значение **3** параметру **Секунды** (Seconds), которое заставляет мотор вращаться в течение 3 секунд.

А теперь давайте рассмотрим некоторые другие свойства шин данных.

Просмотр значения в шине данных

Можно просмотреть значение, передаваемое шиной данных, наведя мышь на шину во время работы программы, как

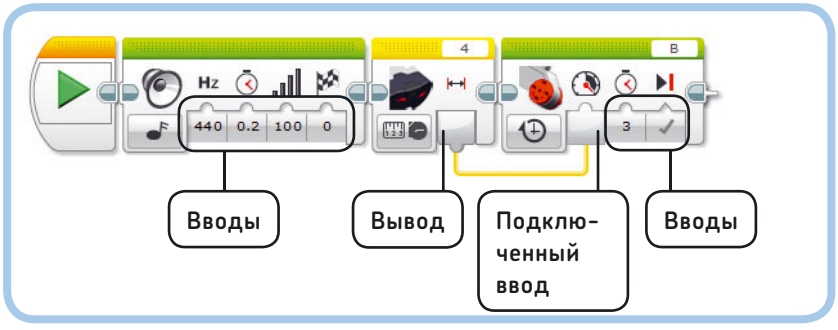


*Рис. 14.6. Шина данных передает данные из вывода (блока **Инфракрасный датчик**) в ввод (блока **Большой мотор**)*

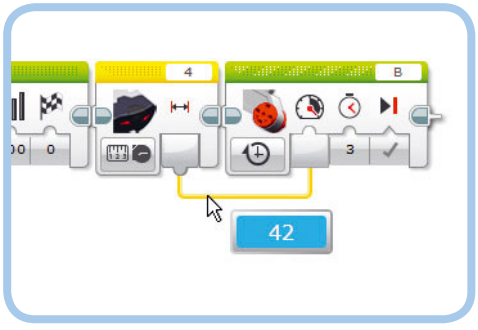


Рис. 14.7. Установите указатель мыши на шину данных, чтобы увидеть передаваемое значение

показано на рис. 14.7. Так вы сможете разобраться, что происходит в вашей программе.

Вы увидите значение в шине данных только в том случае, если робот подключен к компьютеру и вы запустили программу, нажав кнопку **Загрузить и запустить** (Download and Run) в окне программы EV3. Значение не отобразится, если запустить программу вручную, используя кнопки на модуле EV3.

Удаление шины данных

Удаление шины данных производится путем отключения ее правого конца от ввода, как показано на рис. 14.8.

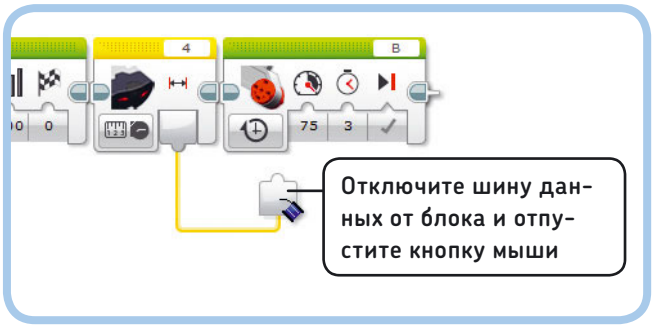


Рис. 14.8. Удаление шины данных

Выбор блока для подключения шины данных

При конфигурации программ с помощью шин данных у вас нет необходимости подключать один блок к другому, располагая их в непосредственной близости; вы можете соединять блоки и в том случае, если между ними установлены другие блоки, как показано на рис. 14.9. Программа WirePause считывает данные датчика, приостанавливается на 5 секунд, затем заставляет мотор вращаться. Скорость мотора основана на измерениях датчика, полученных при запуске программы.

Обратная работа невозможна, как показано на рис. 14.10. Вы не можете заставить блок **Большой мотор**

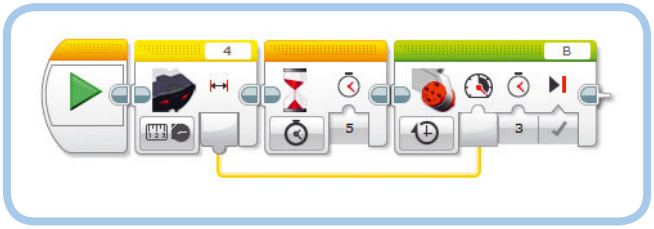


*Рис. 14.9. Программа WirePause выполняет один замер, когда запускается блок **Инфракрасный датчик** (и получает значение, скажем, 34%). Через 5 секунд мотор начинает вращаться на скорости 34%, даже если значение датчика за это время изменилось*

Рис. 14.10. Вы не можете подключить шину данных справа налево. Программное обеспечение EV3 не позволяет этого сделать

(Large Motor) использовать шину данных, потому что она не содержит значение до тех пор, пока не выполнено измерение. Другими словами, блок с вводом (в данном случае **Большой мотор** (Large Motor)) должен располагаться после блока с выводом (**Инфракрасный датчик** (Infrared Sensor)).

Использование нескольких шин данных

Вы можете использовать вывод для подключения нескольких шин данных, как показано на рис. 14.11. Программа MultiWire передает значение параметра **Приближение** (Proximity) на ввод **Мощность** (Power) одного блока **Большой мотор** (Large Motor) (мотора, подключенного к порту B (белая стрелка)) и на ввод **Градусы** (Degrees) второго блока **Большой мотор** (Large Motor) (другого мотора, подключенного к порту C (красная стрелка)). Например, если результат замера датчика составляет 34%, мотор B вращается на скорости 34%, а мотор C вращается на 34 градуса при скорости 75%.

С другой стороны, вы не можете подключить несколько шин данных к одному вводу. Если ввод уже *занят* (см. рис. 14.6), вы не можете подключить к нему другую шину. При возможности подключения более двух шин к одному и тому же вводу блок не знал бы, какое из значений следует использовать.

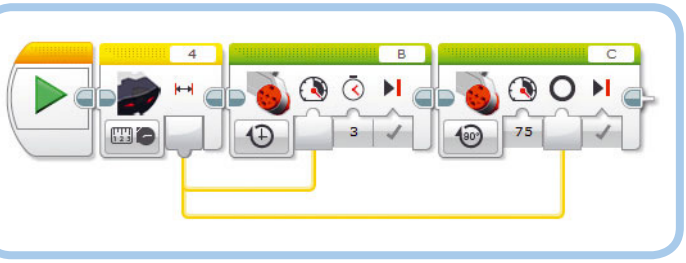


*Рис. 14.11. Две шины данных с одного и того же вывода в программе MultiWire. Инфракрасный датчик выполняет один замер приближения и передает полученное значение на вводы обоих блоков **Большой мотор***

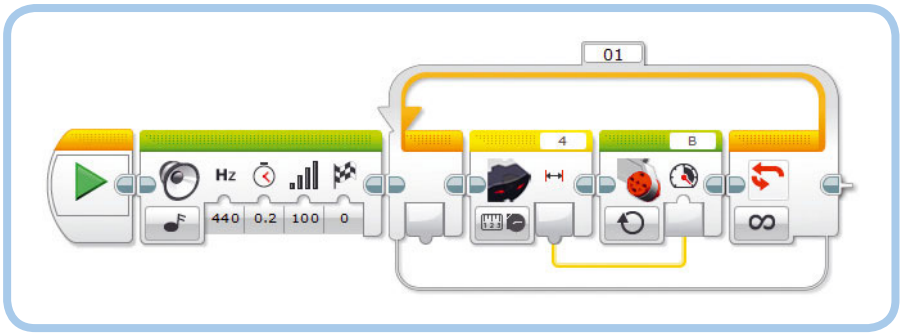


Рис. 14.12. Программа RepeatWire непрерывно подстраивает скорость мотора под измерения инфракрасного датчика

ПРАКТИКУМ № 74: ПОЛОСНАЯ ДИАГРАММА!

Сложность: **Время:**

Программа, изображенная на рис. 14.13, выстраивает полосную диаграмму на экране модуля EV3. Она не закончена, потому что отсутствуют необходимые шины данных. Как их подключить так, чтобы длина полосы представляла собой результат замера инфракрасного датчика?

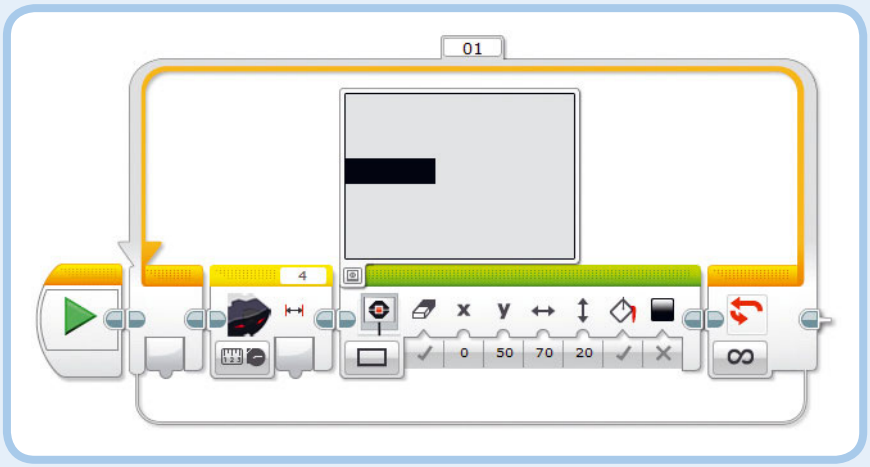


Рис. 14.13. Неполная программа для практикума № 74

ПРАКТИКУМ № 75: РАСШИРЕННАЯ ДИАГРАММА!

Сложность: **Время:**

Расширьте возможности программы, которую вы создали в практикуме № 74, добавив еще две полосные диаграммы, представляющие яркость отраженного света и внешнего освещения, измеряемую с помощью датчика цвета.

СОВЕТ **Используйте два блока Датчик цвета (Color Sensor). Добавьте блок Ожидание (Wait) в цикл, настроенный на ожидание в 0,2 секунды, чтобы предотвратить мерцание на экране. Какой из блоков Экран (Display) должен очищать содержимое экрана, прежде чем на нем появится новое изображение?**

Циклы и шины данных

В программе FirstWire мотор вращался 3 секунды на постоянной скорости на основе значения датчика, которое было получено перед самым началом вращения мотора. Сейчас вы расширите возможности данной программы, чтобы заставлять мотор непрерывно подстраивать свою скорость под значение датчика. Чтобы сделать это, поместите блоки **Инфракрасный датчик** (Infrared Sensor) и **Большой мотор** (Large Motor) в блок **Цикл** (Loop) и выберите для блока **Большой мотор** (Large Motor) режим работы **Включить** (On), как показано на рис. 14.12.

Когда вы запустите программу RepeatWire, скорость мотора должна изменяться постепенно, по мере того как вы медленно отодвигаете свою руку от инфракрасного датчика. Если вы внезапно поднесете руку вплотную к датчику, значение показаний датчика резко уменьшится и мотор быстро остановится.

Скорость постоянно меняется, потому что блокам в цикле почти не требуется времени для запуска. Блок **Инфракрасный датчик** (Infrared Sensor) считывает результаты измерения инфракрасного датчика, а блок **Большой мотор** (Large Motor) переключает мотор на заданную скорость. Затем программа возвращается к началу цикла, чтобы вновь запустить вложенные блоки, постоянно выполняя новые измерения с целью подстройки скорости.

Типы шин данных

До сих пор вы использовали шины данных только для передачи числовых значений, но существует три основных типа данных, которые могут передавать шины: *числовые*, *логические* и *текстовые*. У шин данных каждого типа разный цвет и форма ввода/вывода (круглый/желтый, треугольный/зеленый и квадратный/оранжевый), как показано в табл. 14.1. Форма вывода совпадает с формой ввода, к которому шина подключается, как деталь головоломки. Это сочетание помогает определить, шину данных какого типа можно подключить к конкретному вводу.

Числовые шины данных

Числовые шины данных (желтого цвета) передают числовую информацию, которая может включать целые (как, например, 0, 15 или 1427), десятичные (такие как 0,1 или 73,14) и отрицательные числа (к примеру, –14 или –31,47). Значение приближения инфракрасного датчика представляет собой пример такого числового значения, оно варьируется между 0 и 100.

Логические шины данных

Логические шины данных (зеленого цвета) могут передавать только два значения: **Ложь** (False) или **Истина** (True). Такие шины часто используются, чтобы влиять на параметры, которые могут принимать только два значения, например **Очистить экран** (Clear Screen) блока **Экран** (Display). Датчик касания использует только два возможных значения для кнопки — отпущена и нажата, поэтому для передачи его состояния тоже применяется логическая шина данных. Блок **Датчик касания** (Touch Sensor) в режиме **Измерение** ▸ **Состояние** (Measure ▸ State) обеспечивает передачу через логическую шину данных значения **Истина** (True), если кнопка датчика нажата, и **Ложь** (False), если отпущена.

Создайте программу LogicClear, чтобы опробовать логическую шину данных в действии, как показано на рис. 14.14. Программа начинается с отображения двух сердитых глаз на экране модуля EV3. Двумя секундами позже робот определяет значение датчика касания и передает его в параметр **Очистить экран** (Clear Screen) блока **Экран** (Display) с помощью логической шины данных. Если кнопка датчика нажата (**Истина** (True)), экран очищается, прежде чем отобразится слово MINDSTORMS. Если кнопка не нажата (**Ложь** (False)), экран не очищается, и слово появится поверх предыдущего изображения.

Текстовые шины данных

Текстовые шины данных (оранжевого цвета) передают текст, например, в блок **Экран** (Display), чтобы вывести его на

Таблица 14.1. Основные типы шин данных

Тип		Пример значения
Числовое значение		–5 0 3,75 75
Логическое значение		Истина Ложь
Текст		Hello I'm a robot 5 apples*

* Допускаются только цифры, латинские буквы и специальные символы.

ПРАКТИКУМ № 76: ПЛАВНАЯ ОСТАНОВКА!

Сложность: **Время:**

Разработайте программу, с помощью которой мотор B вращается на максимальной скорости до тех пор, пока инфракрасный датчик не обнаружит поблизости объект. Используйте логическую шину данных, чтобы при нажатии кнопки датчика касания мотор останавливался резко, а при отпущенной кнопке — плавно.

СОВЕТ **Способом остановки мотора можно управлять с помощью параметра Тормозить в конце (Brake at End).**

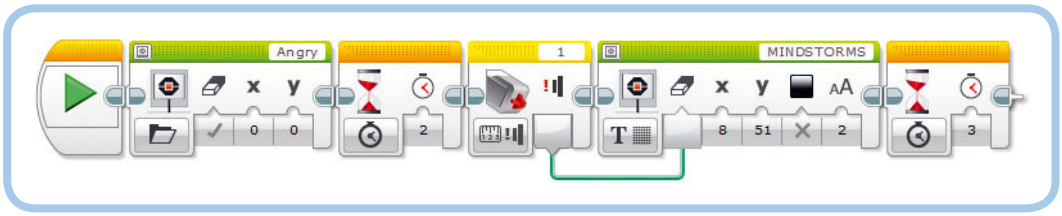


Рис. 14.14. Программа LogicClear. Вы найдете блок ***Датчик касания*** *среди программных блоков датчиков (желтая вкладка палитры программирования)*

экране модуля EV3. Текст может представлять собой слово или фразу, наподобие «привет, я — робот» или «5 яблок»*. Мы вернемся к шинам данных этого типа в главе 15.

Числовой и логический массивы

В дополнение к числовым, логическим и текстовым шинам данных, программное обеспечение EV3 поддерживает такие типы шин данных как *числовой массив* и *логический массив*. Числовой массив фактически является списком числовых значений, используемых для передачи нескольких чисел по одной шине. Например, вы могли бы использовать массив для отправки пяти последних показаний датчика в контейнер «Мой блок», который будет отображать их все на экране модуля EV3.

Аналогичным образом логический массив является списком логических значений. Мы не будем использовать здесь массивы, но вы можете поупражняться с ними в пробной программе на сайте **goo.gl/sgVGyj**, как только закончите чтение этой книги.

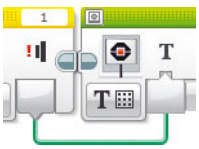
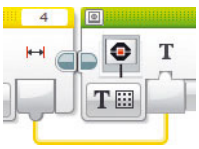
Конвертации шин данных

Как правило, вы должны подключать числовые шины данных к вводам, которые принимают числовые значения (закругленной формы), логические шины данных к логическим вводам (треугольной формы), а текстовые — к текстовым (квадратной формы).

Но программное обеспечение EV3 позволяет выполнить еще три подключения, как показано в табл. 14.2. Можно понять, какие подключения допустимы, просто взглянув на форму вводов. В некоторых случаях формы совпадают не точно, но тем не менее все равно подходят. Например, треугольный ввод вмещается в закругленный, и это означает, что вы можете управлять числовым вводом с помощью логической шины данных. С другой стороны, закругленный ввод не помещается в треугольный, значит, вы не можете

* На самом деле допускаются только цифры, латинские буквы и специальные символы.

Таблица 14.2. Конвертации шин данных

Из типа			В тип	Результат
Логический			Числовой	**Истина** (True) становится равна 1 **Ложь** (False) становится равна 0
Логический			Текстовый	**Истина** (True) становится равна 1 **Ложь** (False) становится равна 0
Числовой			Текстовый	Число конвертируется в формат, поддерживаемый блоком **Экран** (Display), чтобы его можно было отобразить

управлять логическим вводом с помощью числовой шины данных.

В трех случаях, которые показаны в табл. 14.2, данные, передаваемые с помощью шины, конвертируются в подходящий тип, чтобы программа сохраняла работоспособность. Давайте создадим две программы, чтобы разобраться, что это значит.

Конвертация логических значений в числовые

Программа ConvertWire (рис. 14.15) выполняет конвертацию логического значения в числовое. Блок **Ожидание** (Wait) требует числовое значение для параметра **Секунды** (Seconds), но вместо этого он получает логическое значение из блока **Датчик касания** (Touch Sensor). Схема работает, потому что программа конвертирует логическое значение в числовое. Значение **Истина** (True) становится 1, а **Ложь** (False) 0. Следовательно, если вы нажмете кнопку датчика, между звуковыми сигналами вы услышите паузу длиной в 1 секунду, а если не нажмете — паузы не будет.

Отображение чисел на экране модуля EV3

Когда вы используете блок **Экран** (Display) для отображения текста на экране модуля EV3, вы можете либо набрать что-то в текстовом поле, либо выбрать пункт **Проводной** (Wired), как показано на рис. 14.16. Выбор пункта **Проводной** (Wired) создает дополнительный ввод, обеспечивающий поддержку текстовой шины данных.

Программа предполагает, что вы подключите текстовую шину данных к вводу **Текст** (Text), но вы можете подключить и числовую шину данных.

Компьютеры наподобие модуля EV3 хранят числа и текстовые строки двумя разными способами. И поэтому, как правило, вы не можете отправить числовое значение в текстовый ввод. К счастью, программное обеспечение конвертирует числа в текстовый формат, поддерживаемый блоком **Экран** (Display). Эта конвертация позволяет отображать числа на экране модуля EV3, как это демонстрирует программа DisplayNumeric (рис. 14.17).

Программа постоянно обновляет значение датчика и отображает его на экране. Отображение значений — полезный метод тестирования программ. Конечно, вы уже знаете, как отслеживать значения датчика с помощью приложения Port View, но благодаря описанной выше особенности шин данных вы можете, к примеру, выполнять вычисления со значениями датчиков и отображать результаты в режиме реального времени. Вы увидите это в следующей главе.

В дополнение к текстовым и числовым шинам текстовый ввод позволяет подключать логические шины данных. Опять же модуль EV3 хранит логические значения иначе, но программа конвертирует их в текстовый формат, когда это необходимо. Значение **Истина** (True) заставляет блок **Экран** (Display) отображать 1, а **Ложь** (False) — отображать 0. Замените блок **Инфракрасный датчик** (Infrared Sensor) в программе DisplayNumeric блоком **Датчик касания** (Touch Sensor), чтобы опробовать это.

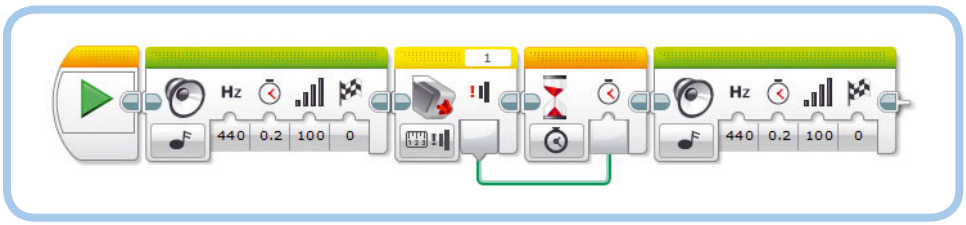


Рис. 14.15. Программа ConvertWire

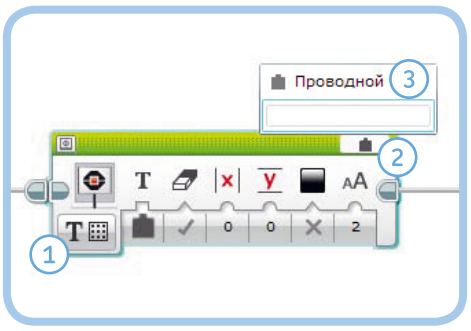


*Рис. 14.16. Перетащите блок **Экран** из палитры, выберите режим **Текст** ▸ **Сетка** (1), щелкните мышью по текстовому полю (2) и выберите пункт **Проводной** (3), чтобы отобразить ввод **Текст***

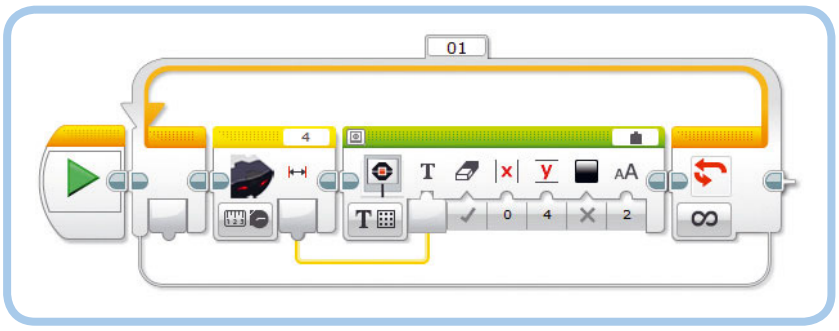


Рис. 14.17. Программа DisplayNumeric

Использование блоков датчиков

Во второй части книги вы научились работать с датчиками, создавая программы с блоками **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch). Обработкой значений датчиков занимаются *программные блоки датчиков*. Эти блоки полезны в том случае, если вы хотите получить значение датчика и передать его в другой блок с помощью шины данных, как вы это делали в программе FirstWire.

Для каждого датчика на вкладке **Датчик** (Sensor) палитры программирования доступен соответствующий блок, как показано на рис. 14.18. Каждый блок можно использовать в режиме **Измерение** (Measure) или **Сравнение** (Compare).

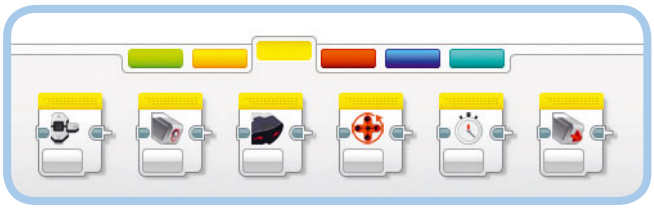


*Рис. 14.18. Слева направо, блоки: **Кнопки управления модулем**, **Датчик цвета**, **Инфракрасный датчик**, **Вращение мотора**, **Таймер** и **Датчик касания***

Режим Измерение

Блок датчика в режиме **Измерение** (Measure) выполняет одно измерение и передает полученное значение в другой блок с помощью шины данных. Вы определяете тип измерения путем выбора одного из режимов работы датчика. Например, вы можете передать значение инфракрасного датчика в блок **Большой мотор** (Large Motor), используя блок **Инфракрасный датчик** (Infrared Sensor) в режиме **Измерение** ▸ **Приближение** (Measure ▸ Proximity).

Все, что вы знаете о режимах работы и значениях датчиков на данный момент, применимо и для блоков датчиков. В табл. 14.3 приводятся сводные значения датчиков для каждого режима работы. Используйте таблицу в качестве справочной информации при разработке собственных программ, будь то блоки **Ожидание** (Wait), **Цикл** (Loop), **Переключатель** (Switch) или блоки датчиков.

Режим Сравнение

Как и в режиме **Измерение** (Measure), блок датчика в режиме **Сравнение** (Compare) выполняет одно измерение и выводит показания датчика с помощью шины данных. Кроме того, он сравнивает полученное значение с пороговым и выводит результат с помощью логической шины данных. Вывод **Результат сравнения** (Compare Result) передает значение **Истина** (True), если условие (например, «значение приближения составляет более 40%») истинно, и значение **Ложь** (False), если условие ложно.

Вы определяете условие посредством присвоения значения параметру **Пороговое значение** (Threshold Value) и выбора режима **Сравнение** (Compare) в настройках блока, как вы это делали для блоков **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch).

Теперь вы разработаете программу, чтобы увидеть, как это работает. Программа SensorCompare (рис. 14.19) содержит блок **Инфракрасный датчик** (Infrared Sensor) в режиме **Сравнение** ▸ **Приближение** (Compare ▸ Proximity), анализирующий значение датчика и проверяющий, выше ли оно 40%. Значение **Приближение** (Proximity) управляет скоростью мотора B, который вращается в течение 5 секунд. Вывод **Результат сравнения** (Compare Result) управляет параметром **Импульсный** (Pulse) блока **Индикатор состояния модуля** (Brick Status Light). Если результат сравнения истинный (значение датчика действительно выше 40%), параметр **Импульсный** (Pulse) приобретает значение **Истина** (True), что заставит индикатор состояния мигать. Если результат сравнения ложный, индикатор будет светиться непрерывно.

Режим Сравнение и значения маяка

Режимы **Приближение маяка** (Beacon Proximity) и **Направление маяка** (Beacon Heading) объединяются в один режим под названием **Маяк** (Beacon), если вы используете блок **Инфракрасный датчик** (Infrared Sensor) в режиме **Измерение** (Measure) (рис. 14.23). Эти же режимы доступны по отдельности, если вы выберете режим **Сравнение** (Compare). В остальном разницы никакой нет. Позже в этой главе вы примените режим **Маяк** (Beacon).

Режим Сравнение и датчик касания

Блок **Датчик касания** (Touch Sensor) в режиме **Сравнение** (Compare) может оповещать о щелчке по кнопке датчика (кнопка была нажата и сразу же отпущена) с последнего момента, когда какой-либо блок был использован для взаимодействия с датчиком касания. То есть, если вы щелкнете по кнопке датчика и позже проверите состояние датчика, блок сообщит вам о щелчке: вывод **Измеренное значение** (Measured value) передаст числовое значение 2. Далее следует некоторая неясность — если вы проверите, отпущена ли кнопка датчика после щелчка, блок выдаст значение **Ложь** (False) (потому что по кнопке щелкнули), даже если кнопка уже отпущена.

Обычно достаточно знать, нажата или отпущена кнопка датчика касания при запуске блока, поэтому лучше использовать блок **Датчик касания** (Touch Sensor) в режиме

Таблица 14.3. Значения датчика для каждого режима работы

Блок датчика	Режим работы	Мин.	Макс.	Значение	Стр.	Заметки
Кнопки управления модулем 	Кнопки управления модулем	0	5	0 = Ничего, 1 = Влево, 2 = По центру, 3 = Вправо, 4 = Вверх, 5 = Вниз	119	Определяет только одну кнопку за один раз
Датчик цвета 	Цвет	0	7	0 = Нет цвета, 1 = Черный, 2 = Синий, 3 = Зеленый, 4 = Желтый, 5 = Красный, 6 = Белый, 7 = Коричневый	99	
	Яркость отраженного света	0	100	0 = Самая низкая яркость 100 = Самая высокая яркость	103	
	Яркость внешнего освещения	0	100	0 = Самая низкая яркость 100 = Самая высокая яркость	107	
Инфракрасный датчик 	Приближение	0	100	0 = Очень близко 100 = Очень далеко	111	
	Маяк (Приближение)	1	100	1 = Очень близко 100 = Очень далеко	115	Значение *не определено*, если не обнаружен сигнал маяка. См. рис. 14.26
	Маяк (Направление)	–25	25	–25 = Слева 0 = По центру 25 = Справа	115	Значение равно 0, если не обнаружен сигнал маяка или если нельзя определить направление сигнала
	Удаленный	0	11	Число представляет собой комбинацию нажатых кнопок на удаленном инфракрасном маяке	114	
Вращение мотора 	Градусы	Н/Д	Н/Д	Количество градусов, на которое повернулся мотор с момента запуска программы	120	Значение может быть сброшено на 0 с помощью режима **Сброс** (Reset)
	Обороты	Н/Д	Н/Д	Количество оборотов, которые выполнил мотор с момента запуска программы	120	Значение представляет собой десятичное число, например 1,5 — для полутора оборотов. Значение можно сбросить на 0 с помощью режима **Сброс** (Reset)
	Текущая мощность	–100	100	Данное число отображает скорость вращения мотора. 100%-ная скорость равна 170 оборотам в минуту для большого мотора и 267 оборотам в минуту для среднего мотора	121	Этот режим измеряет скорость вращения; потребляемая мощность или питание не измеряется. Измеренное значение не зависит от уровня заряда батареи модуля EV3
Таймер 	Время	0	Н/Д	Время в секундах, прошедшее с момента запуска программы	256	Значение представляет собой десятичное число, например 1,5 для полутора секунд. Значение можно сбросить на 0 с помощью режима **Сброс** (Reset)
Датчик касания 	Состояние	**Ложь** (False)	**Истина** (True)	False = Отпущено True = Нажато	88	Значение реализуется с помощью логического канала данных

Измерение (Measure). В этом режиме блок выдает значение **Истина** (True), если кнопка датчика в данный момент нажата, и **Ложь** (False), если отпущена, независимо от того, что произошло ранее во время выполнения программы.

Диапазон значений шин данных

Во время разработки программ, включающих шины данных, важно учитывать, что происходит, когда передаваемое значение шины данных выходит за пределы допустимого диапазона значений. Например, индикатор состояния модуля принимает три значения, которые устанавливают зеленый (0), оранжевый (1) или красный (2) цвет, но что произойдет, если вы будете управлять

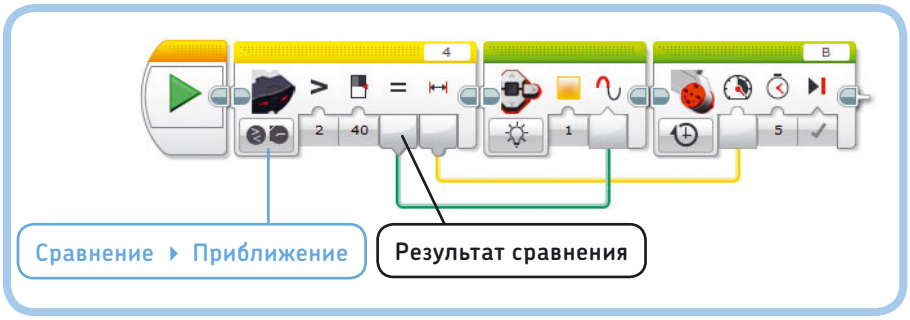


Рис. 14.19. Программа SensorCompare

ПРАКТИКУМ № 77: РЕГУЛЯТОР ДАТЧИКА!

Сложность: **Время:**

Можете ли вы составить программу управления скоростью белой стрелки (мотор B) путем поворота красного диска (мотор C)? Поверните красную стрелку вручную, чтобы протестировать вашу программу.

СОВЕТ **Используйте программу RepeatWire (см. рис. 14.12) в качестве основы и блок Вращение мотора (Motor Rotation) в режиме Измерение ▸ Градусы (Measure ▸ Degrees).**

ПРАКТИКУМ № 78: РАСШИРЕННАЯ ВЕРСИЯ ПРОГРАММЫ MY PORT!

Сложность: **Время:**

Дополните программу DisplayNumeric (см. рис. 14.17), чтобы выводить на экран модуля EV3 следующие значения: датчика цвета (значение яркости отраженного света), датчика касания и датчиков вращения (положения). Эти значения должны обновляться четыре раза в секунду.

Когда будете готовы, поместите блоки внутри цикла в контейнер «Мой блок» с именем **MyPortView**. Вы можете использовать его в своих программах для робота SK3TCHBOT всегда, когда требуется просмотреть данные, получаемые от каждого датчика.

СОВЕТ **Поместите блок Ожидание (Wait) в цикл и приостановите выполнение программы на 0,25 секунд.**

ПРАКТИКУМ № 79: СРАВНИТЕ РАЗМЕР!

Сложность: **Время:**

Можете ли вы отобразить круг в центре экрана модуля EV3 и управлять его размером и цветом с помощью замеров инфракрасного датчика? Используйте параметр **Радиус** (Radius) блока **Экран** (Display), чтобы управлять размером круга; используйте параметр **Заполнить** (Fill), чтобы отобразить закрашенный круг, если значение приближения превышает 30%, и пустой круг в остальных случаях. Поместите блоки **Инфракрасный датчик** (Infrared Sensor) и **Экран** (Display) в блок **Цикл** (Loop), чтобы круг на экране постоянно обновлялся.

СОВЕТ **Используйте блок Инфракрасный датчик (Infrared Sensor) в режиме Сравнение ▸ Приближение (Compare ▸ Proximity).**

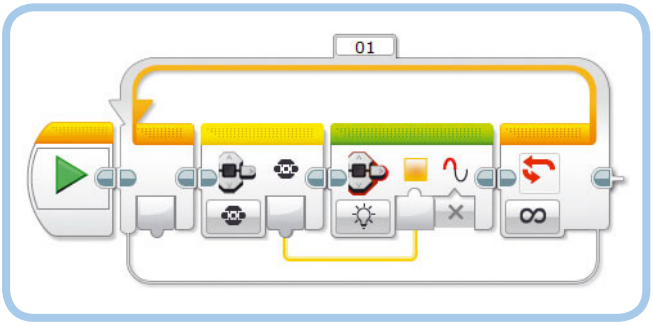


Рис. 14.20. Программа ColorRange

цветом, используя шину данных, передающую значение 4? Чтобы выяснить это, составьте программу ColorRange, показанную на рис. 14.20, и обратитесь к табл. 14.3, чтобы увидеть, какое значение шины данных будет использоваться для каждой кнопки EV3.

После запуска программы индикатор будет подсвечиваться зеленым, если ни одна кнопка не нажата (0), оранжевым, если нажата кнопка «Влево» (1), и красным, если нажата центральная кнопка (2). На все остальные кнопки (3, 4 и 5) индикатор тоже будет реагировать красной подсветкой.

Вы найдете допустимые значения для каждого ввода в справочной системе, выбрав команду меню **Отобразить справку EV3** (Help ▸ Show EV3 Help), но в документации не сказано, что произойдет, если вы выйдете за пределы допустимого диапазона. Как правило, вам достаточно лишь помнить, что программное обеспечение EV3 использует ближайшее допустимое значение, но вам придется провести простой эксперимент, наподобие программы ColorRange, чтобы удостовериться, что данное обобщение применимо к вашей программе. Это правило применимо в следующем случае: значение 2 (красная подсветка) — ближайшее допустимое, если шина данных передает значение 3, 4 или 5.

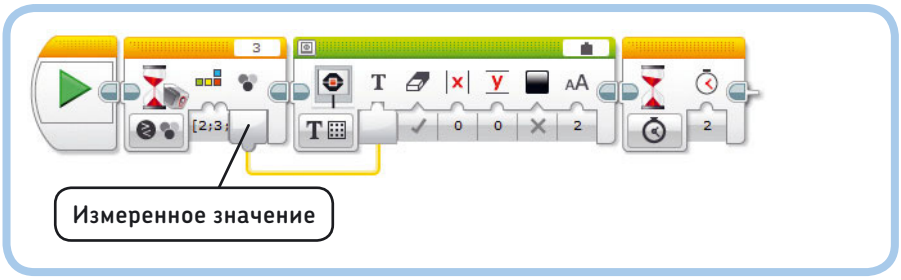


Рис. 14.21. Программа WireWait

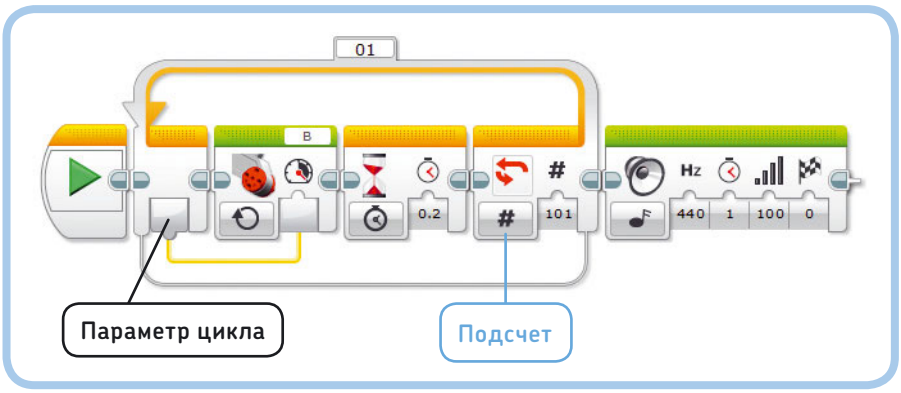


Рис. 14.22. Программа Accelerate

Расширенные функции блоков управления операторами

Теперь, когда вы знаете, как работают шины данных, вы готовы к изучению возможностей блоков **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch), которые требуют использования шин данных. Кроме того, вы научитесь использовать блок **Прерывание цикла** (Loop Interrupt).

Шины данных и блок Ожидание

Напоминаю, что блок **Ожидание** (Wait) приостанавливает программу до тех пор, пока датчик не зафиксирует определенное пороговое значение или не будет достигнуто определенное условие. Вывод **Измеренное значение** (Measured Value) передает значения, согласно которым блок прекращает ожидание. Например, программа WireWait (рис. 14.21) ожидает, пока датчик цвета обнаружит синий (2), зеленый (3), желтый (4) или красный цвет (5), после чего выводит результат последнего измерения на экран модуля.

Шины данных и блок Цикл

Блоки **Цикл** (Loop) поддерживают две функции, которые требуют использования шин данных: **Параметр цикла** (Loop Index) и **Логическое значение** (Logic). Вы попрактикуетесь с каждой функцией в следующей программе.

Использование вывода Параметр цикла

Вывод **Параметр цикла** (Loop Index) (рис. 14.22) предоставляет сведения о количестве завершенных запусков содержимого блока **Цикл** (Loop). Значение параметра начинается с нуля, увеличиваясь на единицу каждый раз после запуска блоков в цикле.

Теперь вы разработаете программу Accelerate, в которой используется параметр **Параметр цикла** (Loop Index) в качестве вводного значения, определяющего скорость мотора. Когда вы запускаете программу, значение параметра будет равно нулю, поэтому мотор будет вращаться на скорости 0 (не вращается) в течение 0,2 секунд. Когда блок **Цикл** (Loop) возвращается к началу, значение параметра **Параметр**

цикла (Loop Index) повышается до 1, после чего вновь запускается блок **Большой мотор** (Large Motor), определяя в этот раз скорость мотора равной 1. Во время следующей итерации цикла скорость мотора возрастет до 2, и так далее.

Блок **Цикл** (Loop) настроен таким образом, что будет запускаться 101 раз, поэтому скорость будет равна 100, когда программа запустится в последний раз. Затем вы услышите сигнал, и программа завершит работу. Вы можете настроить программу так, чтобы цикл запускался 150 раз, но мотор перестанет набирать скорость, когда значение параметра достигнет 100, потому что будет достигнута максимальная скорость.

Завершение цикла в режиме Логическое значение

Ранее вы узнали, что можно прекратить повторение блока **Цикл** (Loop) при достижении определенного количества итераций, секунд или же некоторого значения датчика. В режиме **Логическое значение** (Logic) вы можете остановить цикл путем использования специальной шины данных.

Блок **Цикл** (Loop) проверяет значение в шине данных каждый раз при запуске блоков в цикле. Если значение шины данных **Ложь** (False), цикл запускается вновь. Если значение — **Истина** (True), цикл завершается. Другими словами, цикл повторяется до тех пор, пока значение в шине данных не станет **Истина** (True).

Программа LogicLoop (см. рис. 14.23) демонстрирует данный метод: в ней используется блок **Цикл** (Loop) в режиме **Логическое значение** (Logic) и блок **Инфракрасный датчик** (Infrared Sensor) в режиме **Измерение** ▸ **Маяк** (Measure ▸ Beacon). Блок **Инфракрасный датчик** (Infrared Sensor) предоставляет также выводы **Приближение** (Proximity) и **Направление** (Heading), но здесь вы будете использовать только вывод **Обнаружено** (Detected), который информирует, зафиксировал ли датчик сигнал маяка. Если сигнал обнаружен, шина данных передает значение **Истина** (True), которое приводит к завершению цикла; если сигнал отсутствует, шина данных передает значение **Ложь** (False), и цикл выполняется вновь. Другими словами, программа ожидает, пока датчик не поймает сигнал, а затем воспроизводит звук.

После запуска программы вы увидите, что датчик может успешно обнаружить сигнал на расстоянии до 3 метров.

ПРАКТИКУМ № 80: УДАЛЕННОЕ УСКОРЕНИЕ!

Сложность: **Время:**

Можете ли вы запрограммировать мотор B работать с ускорением до тех пор, пока инфракрасный датчик не обнаружит сигнал от маяка?

СОВЕТ **Объедините программы Accelerate (см. рис. 14.22) и LogicLoop (см. рис. 14.23) в одну.**

Шины данных и блок Переключатель

Как вы помните из главы 6, блоки **Переключатель** (Switch) применяются для того, чтобы робот «принимал решения». Робот использует результаты замеров датчиков для определения, является ли текущее состояние (к примеру, «значение приближения выше 30%») истинным. Если оно истинно, в блоке **Переключатель** (Switch) запускается верхняя последовательность блоков (✓); а если ложно — нижняя (✕). Такое поведение демонстрирует программа SwitchReminder на рис. 14.24.

Режим Логическое значение

Вместо использования результатов замеров датчиков для выбора последовательности вы можете применить логическую шину данных для управления блоком **Переключатель** (Switch), работающим в режиме **Логическое значение** (Logic), как показано на рис. 14.25. Если передается значение **Истина** (True), запускаются блоки в верхней части переключателя; а если **Ложь** (False) — блоки в нижней части.

Программа LogicSwitch1 (рис. 14.25) постоянно проверяет, фиксирует ли инфракрасный датчик сигнал от удаленного маяка. Если это так (**Истина** (True)), робот отображает слово «Успех!»* на экране модуля EV3, а мотор B вращается; если нет (**Ложь** (False)) — слово «Ошибка!»**, и мотор не вращается. Датчик перестает получать сигнал через несколько секунд

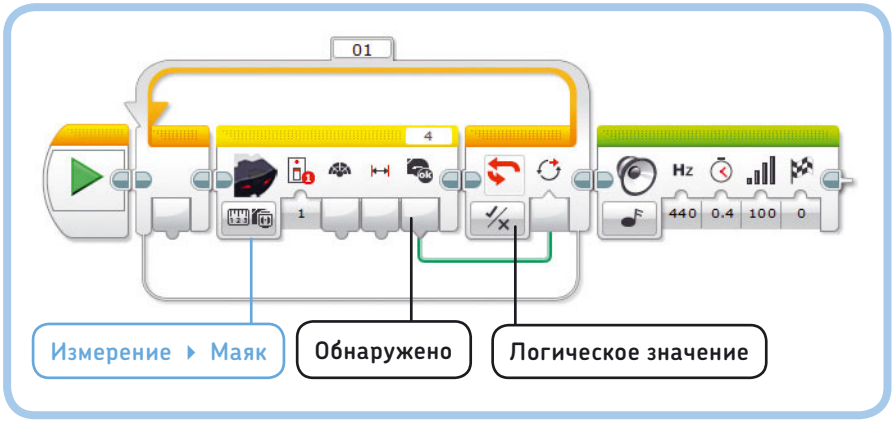


Рис. 14.23. Программа LogicLoop воспроизводит звук, когда инфракрасный датчик фиксирует сигнал удаленного маяка

* Пункт **Success!** в каталоге звуков Lego.
** Пункт **Error!** в каталоге звуков Lego.

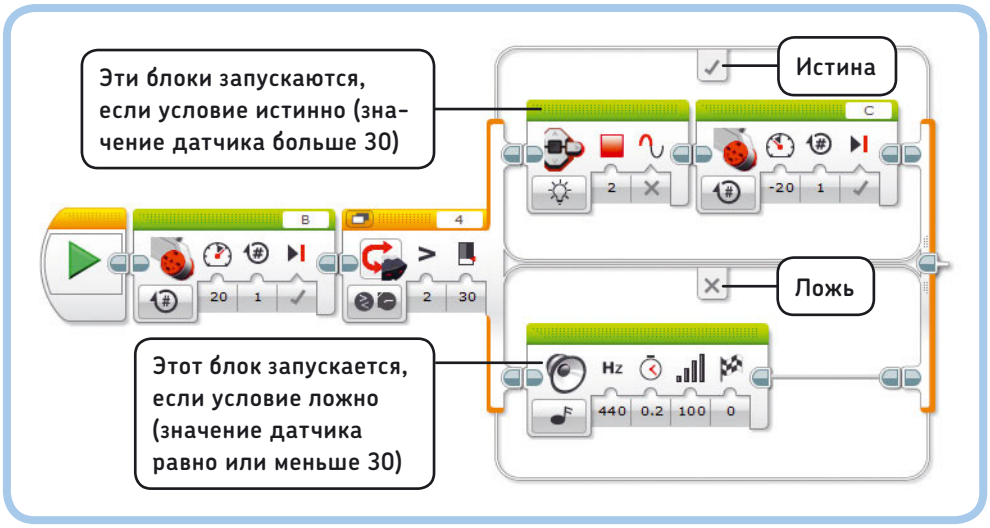


Рис. 14.24. Программа SwitchReminder

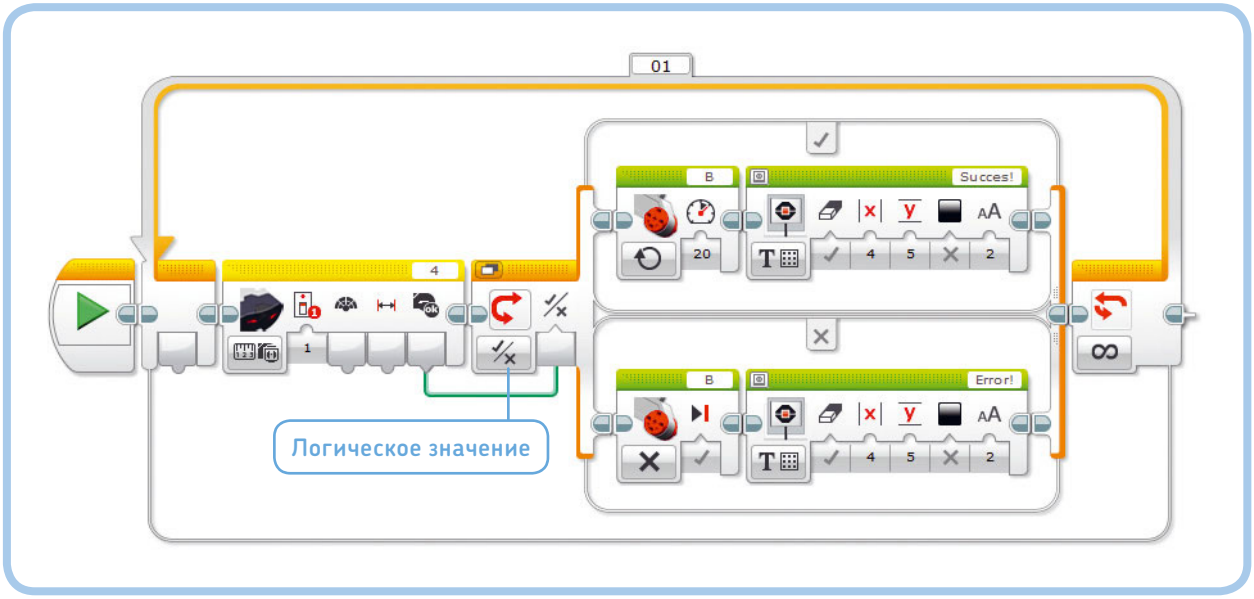


Рис. 14.25. Программа LogicSwitch1

после того, как вы отпустите кнопку, поэтому появление сообщения об ошибке займет около секунды.

Режим Числовое значение

Если вы переведете блок **Переключатель** (Switch) в режим **Числовое значение** (Numeric) и подключите к нему числовую шину данных, то сможете для каждого значения выбрать определенные действия, как это было описано в главе 7 (см. рис. 7.10). Например, вы можете запрограммировать робота произносить «Привет!», если значение в шине данных равно 3, «Доброе утро!», если оно равно 10, и «Нет!» для всех остальных значений (значение по умолчанию). Вы разработаете такую программу в следующей главе.

Подключение шин данных к блокам внутри переключателей

В некоторых случаях может потребоваться подключать шины данных от выводов снаружи блока **Переключатель** (Switch) к вводам блоков внутри переключателя. Например, вы можете изменить свою предыдущую программу, чтобы управлять скоростью мотора на основе расстояния от маяка без необходимости использования другого блока **Инфракрасный датчик** (Infrared Sensor).

Чтобы сделать это, выберите в блоке **Переключатель** (Switch) режим отображения с вкладками и подключите шину данных, как показано на рис. 14.26. Завершенная программа LogicSwitch2 управляет скоростью мотора с помощью

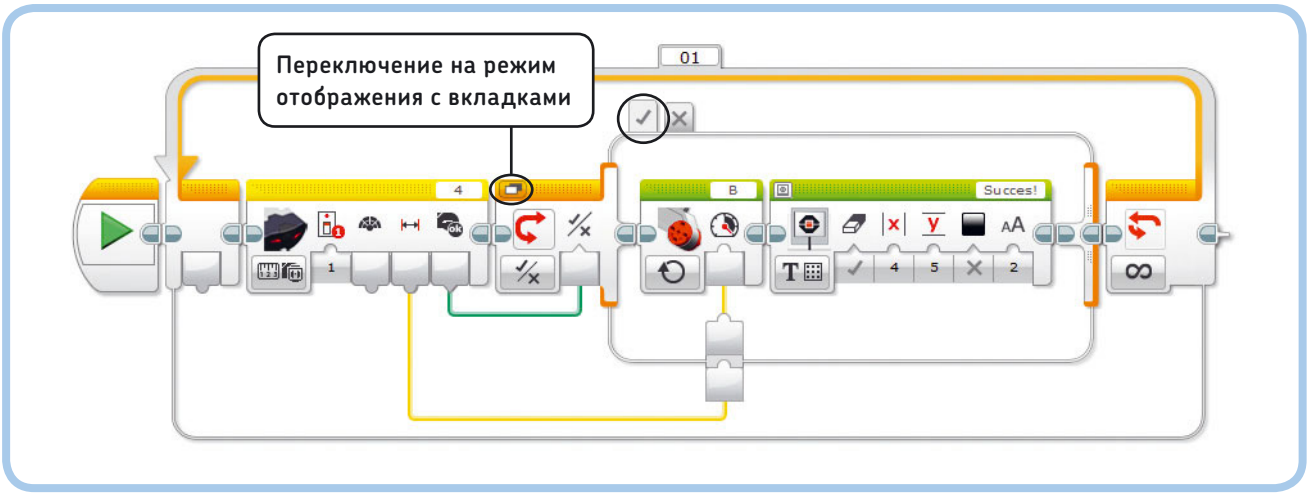


Рис. 14.26. Программа LogicSwitch2. Начните с программы LogicSwitch1, выберите режим отображения с вкладками, перейдите на вкладку ***Истина*** *и подключите числовую шину данных, как показано на рисунке. Пара вводов/выводов на границе блока* ***Переключатель*** *появится автоматически, когда вы попробуете подключить шину. Блоки на вкладке* ***Ложь*** *остаются без изменений*

удаленного маяка, если сигнал обнаружен; в противном случае программа останавливает мотор.

Как показано в табл. 14.3, значение приближения маяка *не определено*, если сигнал не обнаружен. Если вывод **Приближение** (Proximity) подключен к блоку **Большой мотор** (Large Motor), а сигнал не обнаружен, большой мотор не получает значения и вращается неконтролируемо. Вы можете избежать этой проблемы, следя за тем, что значение используется только тогда, когда оно содержит точный результат замеров датчика — только если вывод **Обнаружено** (Detected) передает значение **Истина** (True).

Поэтому вам и нужно использовать блок **Переключатель** (Switch) в программе LogicSwitch2. Значение датчика используется для управления скоростью мотора только в том случае, если обнаружен сигнал от маяка. В противном случае запускаются блоки на вкладке **Ложь** (False), и блок **Большой мотор** (Large Motor) в режиме **Выключить** (Off)) останавливает мотор.

В режиме **Направление маяка** (Beacon Heading) значение на выводе будет 0, даже если маяк находится непосредственно перед датчиком или если сигнал вообще не регистрируется. Можно провести различие между двумя этими случаями, используя такой же метод, как в нашей последней программе. Например, вы можете отобразить значение параметра **Направление маяка** (Beacon Heading) на экране, если сигнал обнаружен, а если он отсутствует, на экране появится слово «Ошибка!».

ПРИМЕЧАНИЕ **Вы можете подключить шины данных к блокам внутри блока Переключатель (Switch) только в том случае, если выбран режим отображения с вкладками. Переключение в режиме без вкладок удаляет шины данных. Подключение шин данных к содержимому блоков Цикл (Loop) осуществляется таким же образом.**

Блок Прерывание цикла

Окончательный способ заставить блок **Цикл** (Loop) остановить повторение заключается в использовании блока **Прерывание цикла** (Loop Interrupt). Блок **Цикл** (Loop), как правило, проверяет состояние датчика или логическое значение каждый раз, когда завершается выполнение вложенных блоков, а блок **Прерывание цикла** (Loop Interrupt) завершает указанный цикл немедленно.

Вы выбираете, какой цикл требуется прервать, выбрав из списка его имя, как показано на рис. 14.27. Вы можете использовать блок **Прерывание цикла** (Loop Interrupt) внутри цикла для его остановки изнутри или завершения цикла на параллельной последовательности. После того как цикл прерван, программа продолжает работу с блоками, размещенными после цикла.

Прерывание цикла изнутри

Блок **Прерывание цикла** (Loop Interrupt) может быть полезен для выхода из блока **Цикл** (Loop) в любой позиции цикла вместо ожидания завершения работы всех блоков в цикле. Рассмотрим программу BreakFromInside, показанную на рис. 14.27, которая систематически вращает мотор B на один поворот, а затем воспроизводит слово «LEGO». Если значение, полученное от инфракрасного датчика, составляет менее 50% после того, как робот произнес слово «LEGO», цикл завершается нормально, и сразу же после этого робот произносит слово «MINDSTORMS».

Благодаря блоку **Прерывание цикла** (Loop Interrupt) вы можете завершить цикл, нажав и удерживая кнопку датчика касания после того, как мотор B совершил вращение. Если вы так поступите, программа перейдет к блоку после цикла, и вы услышите только слово «MINDSTORMS».

Составьте и настройте программу, как показано на рис. 14.27, и запустите ее несколько раз, чтобы определить, какой датчик и когда необходимо привести в действие, чтобы цикл завершил работу.

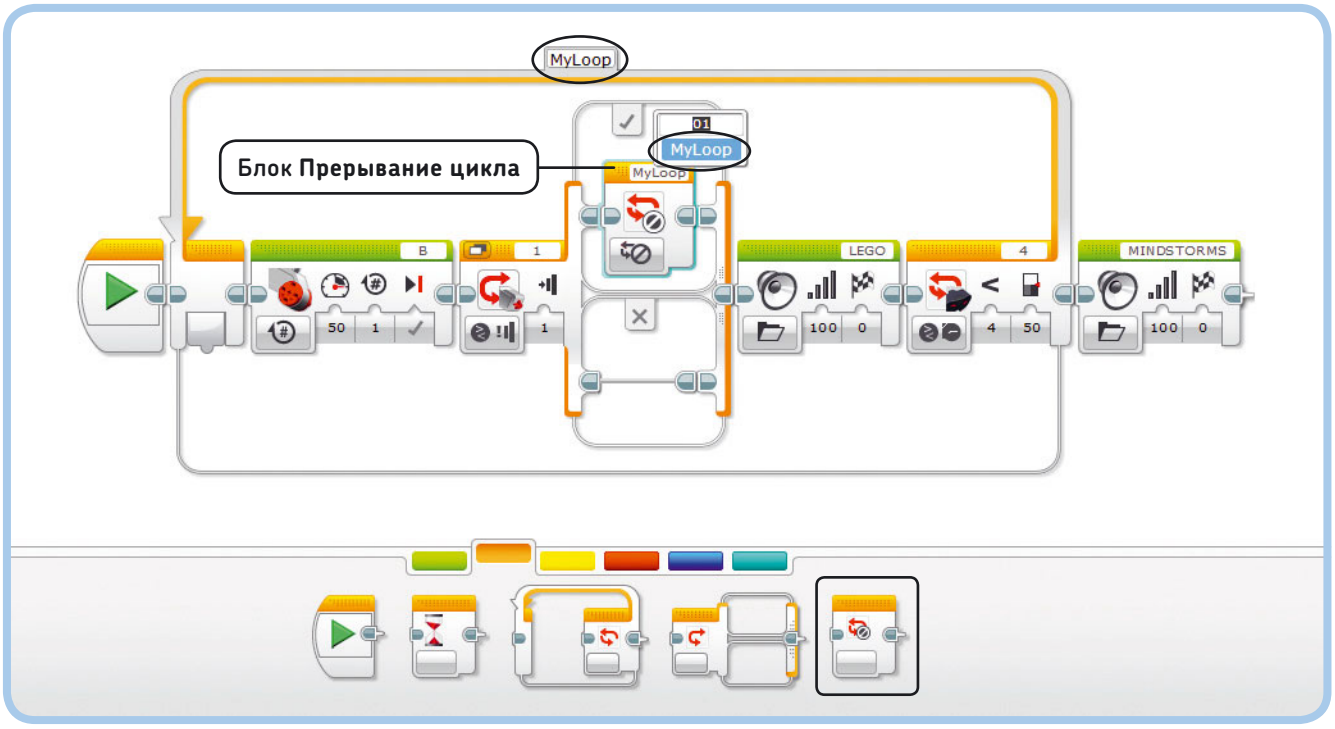


*Рис. 14.27. Программа BreakFromInside содержит цикл с именем **MyLoop**, который завершается, если кнопка датчика касания нажимается сразу же после вращения мотора или если инфракрасный датчик срабатывает сразу после того, как робот произнес слово «LEGO»*

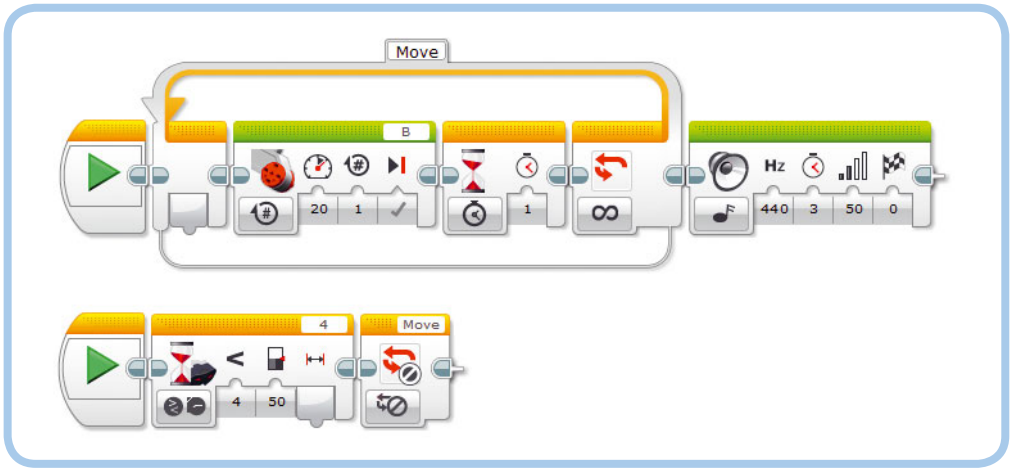


*Рис. 14.28. Программа BreakFromOutside прерывает цикл с именем **Move**, если срабатывает инфракрасный датчик*

Прерывание цикла снаружи

Вы можете прервать выполнение блока **Цикл** (Loop) из последовательности, запущенной параллельно. В этом случае цикл завершит работу и программа запустит блоки после цикла. В то же время программа будет пытаться завершить работу блока, который был запущен при выполнении блока **Прерывание цикла** (Loop Interrupt). Так, программа BreakFromOutside (рис. 14.28) содержит блок **Цикл** (Loop), который систематически поворачивает мотор B на одно вращение. На параллельной последовательности блок **Прерывание цикла** (Loop Interrupt) запускается после того, как срабатывает инфракрасный датчик. Затем цикл завершается, и блок **Звук** (Sound) немедленно воспроизводит звук.

Если вы нажмете кнопку датчика, когда мотор совершил пол-оборота, он закончит вращение во время воспроизведения звука. Блоки **Большой мотор** (Large Motor) и **Звук** (Sound) запустятся одновременно. Теперь присвойте параметру **Продолжительность** (Duration) блока **Звук** (Sound) значение 0,1 секунды и вновь запустите программу. Если вы приведете в действие датчик, когда мотор совершил пол-оборота,

программа завершится почти сразу же, останавливая мотор прежде, чем он закончит оборот.

Рекомендую вам использовать этот способ с осторожностью. Прерывание цикла, который выполняется параллельно, может привести к тому, что робот будет вести себя непредсказуемо. (Что произойдет, если, к примеру, вы запрограммируете мотор совершить еще один оборот сразу же после цикла? Будет ли продолжать выполняться предыдущий блок или же запустится новый?)

Обратите внимание, что прерывание цикла изнутри не приводит к неоднозначности поведения: программа BreakFromInside прерывает цикл, но не завершает работу другого запущенного блока.

Дальнейшее изучение

В этой главе вы научились использовать шины данных для переноса информации из одного блока в другой. Кроме того, вы научились считывать значения датчиков с помощью блоков датчиков и работать с расширенными функциями блоков **Ожидание** (Wait), **Цикл** (Loop) и **Переключатель** (Switch).

Большинство программ, которые вы создали на данный момент с помощью шин данных, относительно просты, и шины данных пока не представляются очень полезными. Но они необходимы для программирования усовершенствованных роботов, которые рассматриваются в части VI книги. Следующие практикумы предоставят вам возможность на практике реализовать навыки программирования, полученные в этой главе.

ПРАКТИКУМ № 81: ПРЕРЫВАНИЕ ПРЕРЫВАНИЙ!

Сложность: **Время:**

Можете ли вы создать программу наподобие BreakFromInside, которая постоянно вращает мотор и воспроизводит звук, пока нажата кнопка датчика касания, а датчик цвета фиксирует зеленый цвет? Одновременное срабатывание этих датчиков должно завершить цикл после вращения мотора. В следующей главе вы узнаете еще об одном способе завершения цикла, когда срабатывает несколько датчиков, а пока просто используйте блок **Прерывание цикла** (Loop Interrupt).

СОВЕТ **Добавьте блок Переключатель (Switch) в программу BreakFromInside (см. рис. 14.27).**

ПРАКТИКУМ № 82: РЕЖИМ РАБОТЫ ДАТЧИКА!

Сложность: **Время:**

Создайте программу, которая будет вращать белую стрелку на скорости, зависимой от состояния инфракрасного датчика, но только когда вы одновременно нажимаете кнопку датчика касания и касаетесь датчика цвета. Если любой из этих датчиков не сработал, мотор не должен вращаться.

СОВЕТ **В этой главе вы узнали о шинах данных и блоках датчиков, но иногда вам все равно придется использовать блоки Ожидание (Wait), Цикл (Loop) и Переключатель (Switch) для работы с датчиками.**

ПРАКТИКУМ № 83: ПИТАНИЕ VS СКОРОСТЬ!

Сложность: **Время:**

Разработайте программу, которая вращает мотор B на скорости 30% (51 оборот в минуту) с помощью блока **Большой мотор** (Large Motor), и мотор C на скорости 30% с помощью блока **Нерегулируемый мотор** (Unregulated Motor). Затем непрерывно отображайте скорость обоих моторов, используя блоки **Вращение мотора** (Motor Rotation) в режиме **Текущая мощность** (Current Power). Напомним, что этот режим показывает *скорость вращения мотора*, как описано в главе 9.

Теперь посмотрим, что произойдет, когда вы попытаетесь замедлить моторы руками. Вы должны увидеть, что скорость мотора C быстро падает: 30% мощности недостаточно для преодоления торможения. Но мотор B продолжает вращаться на почти 30% скорости, потому что блок программирует модуль EV3 использовать дополнительное питание для мотора при его замедлении.

ПРАКТИКУМ № 84: РЕАЛЬНОЕ НАПРАВЛЕНИЕ!

Сложность: **Время:**

Можете ли вы создать программу, которая будет отображать значение параметра Направление маяка (Beacon Heading), если зафиксирован сигнал маяка, и слово «Ошибка!»*? если сигнал не обнаружен?

СОВЕТ **В качестве основы возьмите программу LogicSwitch2 (см. рис. 14.26).**

ПРАКТИКУМ № 85: SK3TCHBOT НАБЛЮДАЕТ ЗА ВАМИ!

Сложность: **Время:**

Создайте программу, которая будет подсчитывать количество людей, проходящих рядом с вашим роботом. Разместите робота таким образом, чтобы инфракрасный датчик мог фиксировать людей, находящихся перед ним. Вложите два блока **Ожидание** (Wait) внутрь блока **Цикл** (Loop) и настройте первый блок так, чтобы он ожидал до тех пор, пока кто-то не пройдет мимо; используйте второй блок, чтобы подождать, пока человек не окажется вне поля зрения. В конечном счете, цикл должен выполняться однократно каждый раз, когда человек проходит мимо. Теперь при отображении значения параметра **Параметр цикла** (Loop Index) на экране модуля EV3 вы будете видеть, сколько людей прошло мимо. Чтобы убедиться, что ваша программа работает правильно, поместите блок **Звук** (Sound) внутрь цикла, чтобы слышать звуковой сигнал каждый раз, когда кто-то идет мимо.

СДЕЛАЙ САМ № 25: БИОНИЧЕСКАЯ РУКА!

Сборка: **Программирование:**

Можете ли вы спроектировать захватное устройство, которое затем можно будет прикрепить к вашей руке? Используйте датчики и кнопки модуля EV3 для управления движениями захватного устройства. Добавьте блок **Инфракрасный датчик** (Infrared Sensor) и запрограммируйте робота таким образом, чтобы рука предупреждала вас, когда датчик фиксирует приближение к стене. Используйте методы программирования, которые вы изучили в этой главе, чтобы выводить результаты замеров датчика на экран или настроить звуковое оповещение на основе замеров датчика.

ПРАКТИКУМ № 86: ОСЦИЛЛОГРАФ!

Сложность: **Время:**

Можете ли вы превратить ваш модуль EV3 в измерительное устройство, которое записывает показания датчика на экране, как показано на рис. 14.29? Отобразите значение параметра **Приближение** (Proximity) в виде небольших окружностей, чьи координаты *x* определяются с помощью значения параметра **Параметр цикла** (Loop Index), а координаты *y* — с помощью значения параметра **Приближение** (Proximity). В процессе работы цикла вы должны видеть, как новые результаты измерений отображаются один за другим и образуют диаграмму, подобную показанной на рисунке.

Так как экран содержит 178 пикселов по горизонтали, блок **Цикл** (Loop) должен сработать 178 раз, чтобы отобразить 178 кругов, и использовать паузу в 0,05 секунды между замерами. Затем программа должна очистить экран и начать все сначала.

СОВЕТ **Чтобы блок Экран (Display) очищал экран после 178 измерений, сделайте так, чтобы на экран выводился белый прямоугольник, присвоив параметру Цвет (Color) значение Истина (True).**

Рис. 14.29. Диаграмма из практикума № 86. Подобная диаграмма может быть получена в результате постоянного перемещения объекта вперед и назад перед датчиком

* Пункт **Error!** в каталоге звуков Lego.

15

Использование блоков операций с данными и контейнеров с шинами данных

Теперь, когда вы знаете, как работают шины данных, можно сделать много интересного, используя большее количество программных блоков. Например, вы можете сделать так, чтобы модуль EV3 объединял и обрабатывал данные датчиков, чтобы их можно было использовать как входные значения для выполнения других действий. С помощью приемов, которые вы изучите в этой главе, вы сможете запрограммировать своего робота на выбор случайного действия или же на то, чтобы он выполнял некие действия, когда два датчика срабатывают одновременно, вместо выполнения заранее запрограммированных действий.

В этой главе вы узнаете, как применить блок **Математика** (Math), чтобы роботы выполняли расчеты, которые можно использовать в программах. Например, робот может вычислить расстояние, которое он должен пройти, основываясь на показаниях датчика. Еще я представлю вам несколько новых блоков программирования, таких как **Случайное значение** (Random), **Сравнение** (Compare) и **Логические операции** (Logic Operations), и покажу, как создавать контейнеры «Мой блок» с вводами и выводами.

Эти приемы и программные блоки необходимы для разработки сложных программ для роботов. Как и ранее, в этой главе вы будете использовать робота SK3TCHBOT для проверки создаваемых программ.

Практикумы в этой главе сначала могут показаться немного сложными, но они помогут вам овладеть необходимыми навыками программирования, что позволит разрабатывать гораздо более интересные программы и собирать действительно умных роботов!

Применение блоков операций с данными

Палитра программирования содержит ряд блоков, которые вы еще не использовали, например блоки операций с данными (рис. 15.1). Они включают в себя блоки **Математика** (Math), **Случайное значение** (Random), **Сравнение** (Compare) и **Логические операции** (Logic Operations). Каждый блок обладает собственным функционалом, но все они обрабатывают значения, передаваемые шинами данных, и генерируют новые значения на их основе. Данный раздел объясняет, как использовать эти блоки в программах.

Рис. 15.1. Блоки операций с данными

Блок Математика

Блок **Математика** (Math) (рис. 5.2) позволяет модулю EV3 выполнять арифметические операции, такие как сложение, вычитание, умножение и деление. Вы вводите два числа в поля **a** и **b** блока и используете раскрывающийся список выбора режима, чтобы выбрать, какую операцию (например, деление) следует применить к ним (в случае с делением значение **a** будет делиться на **b**). Шина данных выводит результат. Чтобы вручную не присваивать значения параметрам **a** и **b**, вы можете передавать их через шину данных.

Блок Математика в действии

Программа на рис. 15.3 демонстрирует блок **Математика** (Math), который выполняет операцию по умножению результата замеров датчика цвета для управления скоростью мотора B. Значение датчика цвета (число в диапазоне от 0 до 7) не подходит для управления скоростью, потому что с этим значением мотор будет вращаться очень медленно (в лучшем случае 7% скорости). Блок **Математика** (Math) увеличивает значение датчика в десять раз и передает результат (число в диапазоне от 0 до 70) во ввод **Мощность** (Power) блока **Большой мотор** (Large Motor). Как следствие, мотор вращается на скорости 10% при обнаружении черного цвета, 20% — синего и так далее, вплоть до 70% скорости при обнаружении коричневого цвета.

Создайте новый проект под названием SK3TCHBOT-Data и в нем программу MathSpeed, как показано на рис. 15.3.

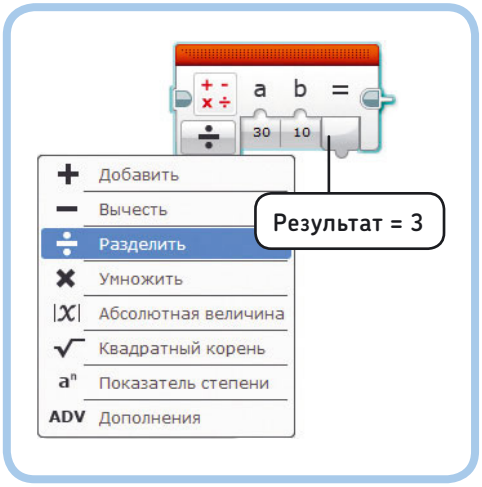


*Рис. 15.2. Блок **Математика**. Выходное значение этого блока равно 30 ÷ 10 = 3*

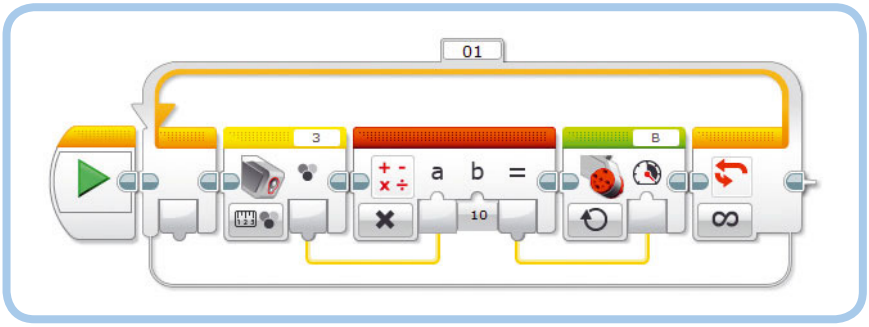


Рис. 15.3. Программа MathSpeed

ПРАКТИКУМ № 87: 100%-НАЯ МАТЕМАТИКА!

Сложность: **Время:**

Благодаря блоку **Математика** (Math) в программе MathSpeed мы можем использовать определенный цвет для управления скоростью мотора, но скорость доходит только до 70%. Можно ли изменить программу так, чтобы использовать полный диапазон значений скорости? Добавьте блок **Экран** (Display) в свою программу, чтобы убедиться, что результирующий вывод блока **Математика** (Math) достигает 100% при обнаружении коричневого цвета.

СОВЕТ **Вы можете использовать десятичные числа в полях a и b блока Математика (Math).**

вычитания, второй для умножения, но в режиме **Дополнения** (Advanced) вы сможете проводить эти расчеты, используя всего лишь один блок **Математика** (Math), как показано на рис. 15.4.

В поле ввода уравнения вы вводите расчеты, которые блок **Математика** (Math) должен выполнить, таким же образом, как и на калькуляторе. Например, ввод (7–3)*1,5 в результате даст ответ 6. Так же как и на калькуляторе, вы используете круглые скобки, чтобы вычитание выполнялось прежде умножения.

Вы можете использовать в уравнении символы переменных (**a**, **b**, **c** и **d**), и присвоить значение каждой переменной с помощью параметров **a**, **b**, **c** и **d** соответственно. Например, введя формулу (b–c)*a в качестве уравнения и присвоив значение 7 параметру **b**, 3 — параметру **c** и 1,5 — параметру **a**, мы получим такой же результат: 6. Наконец, вы можете присваивать значения каждой переменной, используя числовые шины данных.

Режим Дополнения

Некоторые вычисления требуют выполнения более одного арифметического действия. Например, вы можете вычесть два числа и умножить результат на третье число. Можно было бы использовать два блока **Математика** (Math), один для

Чтобы увидеть режим **Дополнения** (Advanced) в действии, создайте программу, которая заставляет мотор C (красная стрелка) следовать за движениями мотора B (белая стрелка) — его вы будете вращать вручную. Так как мотор C установлен в вашем роботе вверх ногами, стрелка на практике будет вращаться в противоположном направлении, но на такое же количество оборотов. Чтобы решить эту задачу, задайте скорость мотора C следующим образом:

Скорость мотора C = (положение в градусах мотора B – положение в градусах мотора C) × 1,5

Чтобы разобраться, почему такая формула работает, рассмотрим, что происходит, когда вы поворачиваете стрелку мотора B вперед на 70 градусов и мотор C поворачивается на 60 градусов. Скорость мотора C будет равна 15 по формуле (70 – 60) × 1,5. Согласно этому значению, мотор C будет поворачиваться вперед, чтобы догнать мотор B. Если C обгоняет B, результат будет отрицательным, и мотор C повернется назад. Чем больше разница между положениями моторов, тем быстрее будет двигаться мотор C. Как только оба мотора окажутся в одинаковом положении, результат будет равен 0 и мотор остановится.

Составьте программу PositionControl, которая показана на рис. 15.5. Поворачивайте белую стрелку вручную и наблюдайте, как красная стрелка двигается так же, но в обратном направлении.

Практическое применение блока Математика

Поскольку блок **Математика** (Math) является неотъемлемым компонентом многих программ, в которых используются шины данных, будет неплохо, если вы попрактикуетесь в его использовании, прежде чем продолжите чтение книги. Следующие практикумы помогут вам в этом.

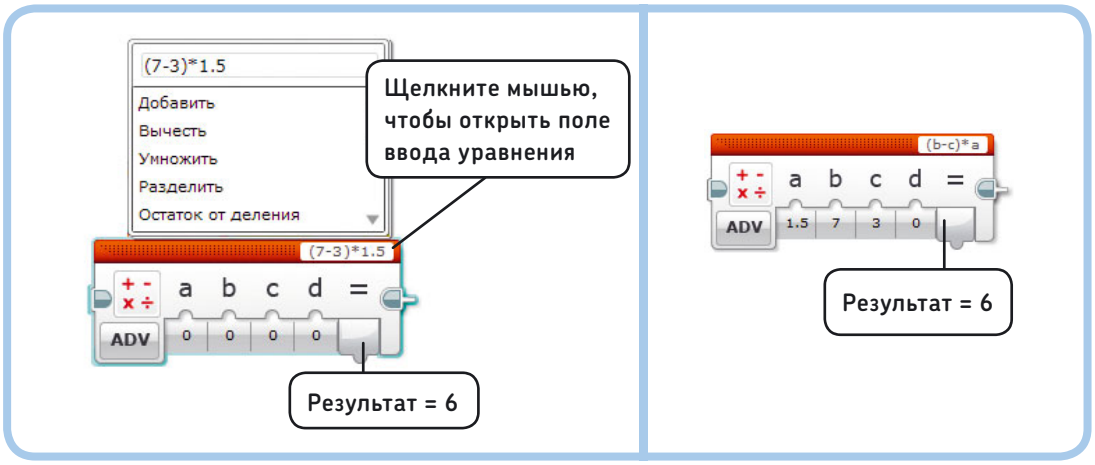


*Рис. 15.4. Блок **Math** в режиме **Дополнения**. Вы можете ввести формулу с числами (слева), с переменными (справа) или комбинацию из переменных и чисел. Операторы указываются вручную, например * для умножения и / для деления, или выбираются из списка*

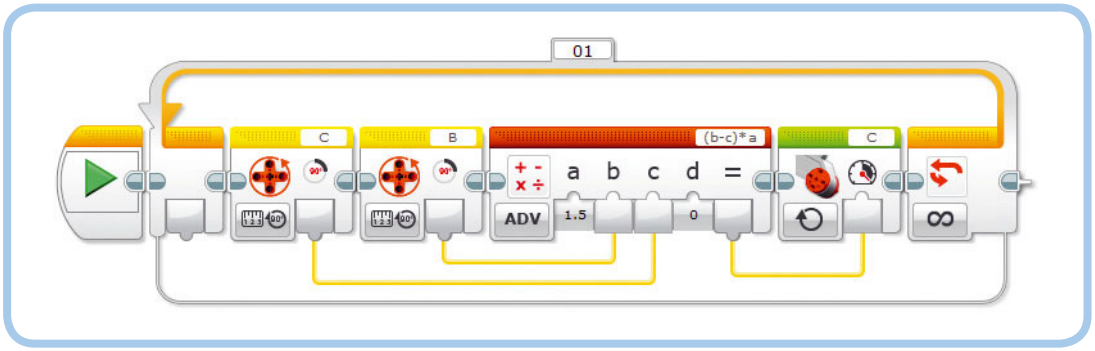


*Рис. 15.5. Программа PositionControl. Вывод **Градусы** мотора B подключен к вводу **b** блока **Математика**; вывод **Градусы** мотора C подключен к вводу **c***

ПРАКТИКУМ № 88: ДОБАВЛЕННОЕ ЗНАЧЕНИЕ!

Сложность: **Время:**

Можете ли вы разработать программу, которая будет постоянно отображать на экране результаты замера приближения инфракрасным датчиком и яркости отраженного света датчиком цвета, а также сумму результатов двух этих измерений?

ПРАКТИКУМ № 89: СКОРОСТЬ НА ОСНОВЕ ЗАМЕРОВ ИНФРАКРАСНОГО ДАТЧИКА!

Сложность: **Время:**

Разработайте программу, похожую на MathSpeed, для управления скоростью и направлением вращения мотора B на основе результатов замера приближения инфракрасным датчиком. Мотор должен вращаться на скорости 50% при 100%-ном значении параметра **Приближение** (Proximity), останавливаться при 50%-ном значении и вращаться на скорости –50% (обратное направление) при значении 0%.

СОВЕТ **Используйте формулу: Скорость = Приближение – 50. Как вы настроите блок Математика (Math), чтобы выполнить эту операцию?**

ПРАКТИКУМ № 90: УДВОЕННАЯ СКОРОСТЬ НА ОСНОВЕ ЗАМЕРОВ ИНФРАКРАСНОГО ДАТЧИКА!

Сложность: **Время:**

Можно ли доработать программу из практикума № 89, чтобы скорость мотора варьировалась в диапазоне от –100 до 100?

СОВЕТ **Используйте следующую формулу: Скорость = (Приближение – 50) × 2.**

ПРАКТИКУМ № 91: РЕГУЛЯТОР УСИЛЕНИЯ!

Сложность: **Время:**

Каким будет результат использования значения 1,5 в программе PositionControl? Попробуйте поэкспериментировать, изменяя значение до минимума (0,1) и максимума (5) и наблюдая за тем, насколько быстро мотор C сможет следовать за вашими движениями.

ПРАКТИКУМ № 92: УПРАВЛЕНИЕ НАПРАВЛЕНИЕМ!

Сложность: **Время:**

Можно ли доработать программу PositionControl таким образом, чтобы красная стрелка вращалась в том же направлении, что и белая?

СОВЕТ **Умножьте значение Градусы (Degrees) мотора C на –1, чтобы поступательное движение измерялось как реверсивное, и наоборот. Аналогично, умножьте результирующую скорость на –1, прежде чем передать ее в блок Большой мотор (Large Motor), чтобы изменить направление.**

Блок Случайное значение

Блок **Случайное значение** (Random), как можно понять из названия, позволяет генерировать случайное значение для использования в программе. В режиме **Логическое значение** (Logic) к выводу блока подключается логическая шина данных, передающая значение либо **Истина** (True), либо **Ложь** (False). Использование блока в данном режиме — это как подбрасывание монетки, результатом которого является **Истина** (True) (орел) или **Ложь** (False) (решка). Когда мы подбрасываем монетку, вероятность того, что выпадет **Истина** (True), составляет 50%, в результате значение **Истина** (True) выпадает приблизительно в половине раз. Блок **Случайное значение** (Random) позволяет вам определить вероятность выпадения значения **Истина** (True) настройкой параметра **Вероятность значения «истина»** (Probability of True) в процентах. Например, ввод числа 33 приведет к передаче значения **Истина** (True) приблизительно в одной трети раз, в то время как в двух третях раз будет выпадать **Ложь** (False).

В режиме **Числовое значение** (Numeric) к выводу блока подключается числовая шина данных, передающая случайное

целое число в диапазоне, определяемом параметрами **Нижняя граница** (Lower Bound) и **Верхняя граница** (Upper Bound). Например, присвоение параметру **Нижняя граница** (Lower Bound) значения 1, а параметру **Верхняя граница** (Upper Bound) значения 6 в результате приведет к 1, 6 или любому другому целому числу между ними. Выпасть может в равной вероятности любое число, как если бы вы бросали игральный кубик.

Блок **Случайное значение** (Random) полезен в том случае, если вы хотите, чтобы ваш робот сделал что-то неожиданное. Например, вы можете использовать блок **Случайное значение** (Random), чтобы робот в произвольном порядке произносил слова «Налево» или «Направо» или чтобы белая стрелка вращалась на произвольной скорости мотора, как показано в программе RandomMotor на рис. 15.6.

Блок **Переключатель** (Switch) в программе RandomMotor запускает либо блоки в верхней части (**Истина** (True)), либо блоки в нижней части (**Ложь** (False)). Вы можете сделать так, чтобы программа в произвольном порядке выбирала тот или иной вариант, переведя блоки **Случайное значение** (Random) и **Переключатель** (Switch) в режим **Числовое значение** (Numeric). Тогда вы сможете определить действия для каждого возможного случайного значения, как продемонстрировано в программе RandomCase на рис. 15.7.

ПРАКТИКУМ № 93: СЛУЧАЙНАЯ ЧАСТОТА!

Сложность: **Время:**

Можно ли сделать так, чтобы модуль EV3 воспроизводил произвольный тон длительностью 0,5 секунды каждый раз, когда вы нажимаете кнопку датчика касания? Сгенерируйте случайное значение и передайте его на ввод **Частота** (Frequency) блока **Звук** (Sound). Наконец, дополните программу, чтобы она выводила частоту на экран модуля EV3.

СОВЕТ **Какие значения принимает ввод Частота (Frequency) блока Звук (Sound)?**

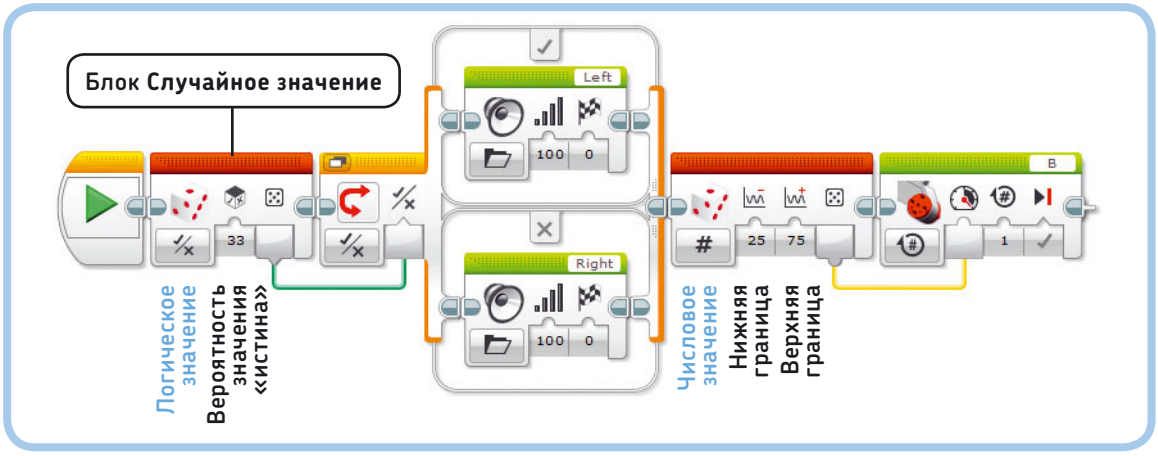


Рис. 15.6. Программа RandomMotor. Если вы запустите программу несколько раз, вы обнаружите, что робот произносит «Налево» примерно в трети случаев. После этого мотор B вращается на один оборот на произвольной скорости между 25 и 75%

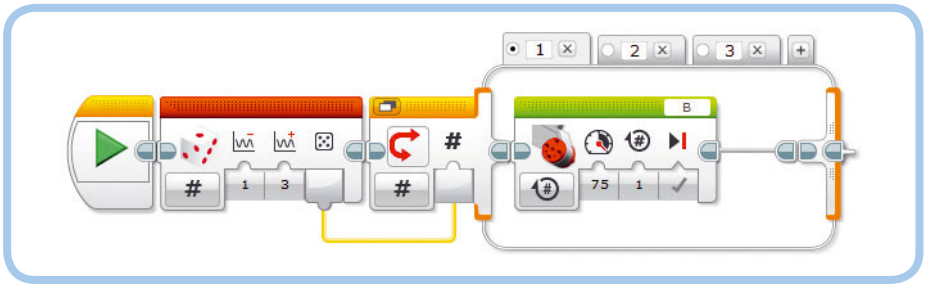


Рис. 15.7. Программа RandomCase в произвольном порядке выбирает, на какой вкладке выполнить блоки. Здесь показан блок только на первой вкладке; вы можете добавить любые блоки, какие вам необходимы

ПРАКТИКУМ № 94: ПРОИЗВОЛЬНО ВЫБРАННЫЙ МОТОР И СКОРОСТЬ!

Сложность: **Время:**

Разработайте программу, которая генерирует случайное число между 10 и 100, чтобы управлять скоростью мотора B или C. Если случайное значение меньше 50, мотор B должен повернуться на один оборот со случайным значением параметра **Мощность** (Power); если значение выше 50, на один оборот должен повернуться на произвольной скорости мотор C.

СОВЕТ **Вам понадобятся блоки Случайное значение (Random), Сравнение (Compare), Переключатель (Switch) (выберите режим просмотра в виде вкладок) и два блока Большой мотор (Large Motor).**

Блок Сравнение

Блок **Сравнение** (Compare) выполняет проверку, равно (=), не равно (≠), больше (>), больше или равно (≥), меньше (<), меньше или равно (≤) одно числовое значение в сравнении с другим. Вы можете присвоить любые значения, которые хотите сравнить, параметрам (**a** и **b**) блока или же передать данные в блок с помощью шин данных.

Блок **Сравнение** (Compare) позволяет подключить одну логическую шину данных, передающую значение **Истина** (True) или **Ложь** (False), на основе результатов сравнения значений **a** и **b**. Так, если вы выберете режим **Равно** (Equal to) (=), блок передаст значение **Истина** (True), если **a** равно **b**.

Программа CompareValues, показанная на рис. 15.8, демонстрирует блок **Сравнение** (Compare) в действии.

С помощью данной программы робот ожидает, когда значение параметра **Приближение** (Proximity) упадет ниже 80. Значение, которое заставило блок **Ожидание** (Wait) прекратить ожидание, переносится в блок **Сравнение** (Compare), и он определяет, меньше ли полученное значение 40. Если это так (**Истина** (True)), SK3TCHBOT говорит «Вниз»; если нет (**Ложь** (False)) — «Вверх».

Блок Логические операции

Блок **Логические операции** (Logic Operations) сравнивает два значения, полученные из логических шин данных, и выдает результат тоже с помощью логической шины данных. В режиме **И** (AND) он выполняет проверку, истинны ли оба входных значения (**a** и **b**). Если это так, выходной результат будет **Истина** (True). Если один или оба входных значения ложны, выходной результат будет **Ложь** (False).

В этом режиме вы можете использовать блок **Логические операции** (Logic Operations), чтобы создать программу, рисующую на экране закрашенный круг, если нажата кнопка датчика касания и результат замера составляет меньше 50. Если не выполняется хотя бы одно из этих условий (или оба), на экране отобразится пустой круг, потому что на выводе блока **Логические операции** (Logic Operations) окажется значение **Ложь** (False). Рис. 15.9 демонстрирует программу LogicAnd.

Логические операции

При настройке блока **Логические операции** (Logic Operations) вы можете выбрать один из четырех режимов: **И** (AND), **ИЛИ** (OR), **Исключающее ИЛИ** (XOR) и **Исключение** (NOT). В каждом режиме блок по-разному сравнивает логические входные значения. Подходящий режим зависит от того, что должна выполнить ваша программа. Табл. 15.1 перечисляет доступные режимы, а также входные значения, при которых вывод будет равен **Истина** (True).

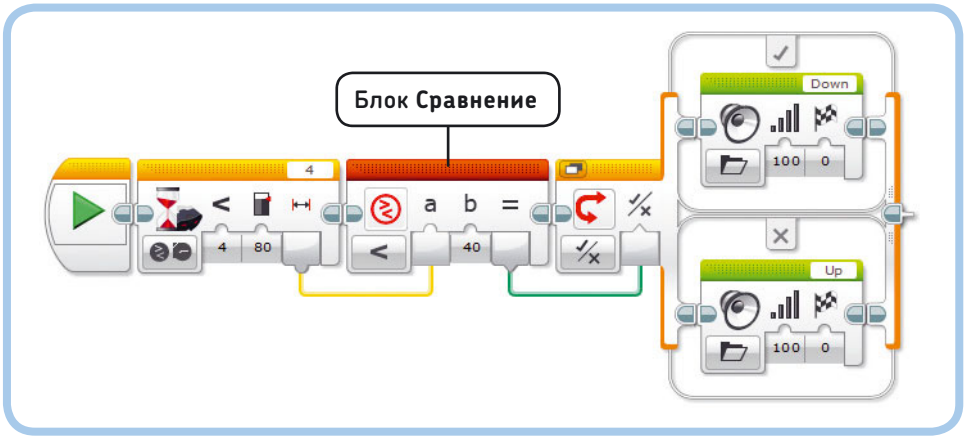


Рис. 15.8. Программа CompareValues

Таблица 15.1. Режимы блока Логические операции и соответствующие выходные значения

Режим		Результат Истина (True), если...
И (AND)		Оба входных значения истинны
ИЛИ (OR)		Одно или оба входных значения истинны
Исключающее ИЛИ (XOR)		Одно значение истинно, а другое — ложно
Исключение (NOT)		Входное значение ложно

В режиме **ИЛИ** (OR) вывод блока передает значение **Истина** (True), если одно или оба входных значения истинны, как это продемонстрировано в программе LogicOr на рис. 15.10. Программа постоянно проверяет, срабатывает ли датчик касания или инфракрасный датчик. Если срабатывают один или оба, выходным результатом будет **Истина** (True), что приведет к завершению цикла. Как только цикл завершается, блок **Звук** (Sound) воспроизводит звуковой сигнал. Этот метод полезен, поскольку позволяет программе ждать до тех пор, пока по крайней мере один из нескольких датчиков не сработает.

Теперь измените режим блока **Логические операции** (Logic Operations) на **Исключающее ИЛИ** (XOR). Когда запустите программу, вы будете слышать звук, когда срабатывает либо датчик касания, либо инфракрасный датчик, но звук не будет воспроизводиться, если оба датчика срабатывают одновременно.

Режим Исключение (NOT)

Если выбран режим **Исключение** (NOT), блок **Логические операции** (Logic Operations) содержит только один ввод. Этот режим попросту инвертирует входящее значение (**a**): если оно истинно, на вывод будет передано значение **Ложь** (False), а если ложно — то **Истина** (True).

Чтобы увидеть этот режим в действии, измените программу LogicOr, которую вы только что создали, удалив из нее блок **Инфракрасный датчик** (Infrared Sensor) и изменив режим блока **Логические операции** (Logic Operations) на **Исключение** (NOT). Вывод блока **Логические операции** (Logic Operations) будет передавать значение **Ложь** (False), если нажата кнопка датчика касания, и **Истина** (True), если кнопка отпущена. Таким образом, цикл будет выполняться, пока вы не отпустите кнопку датчика.

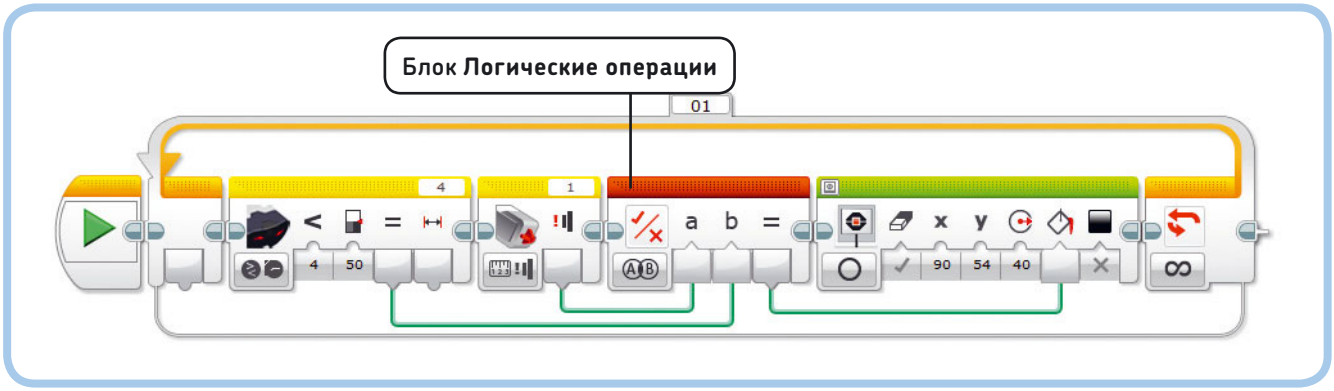


Рис. 15.9. Программа LogicAnd

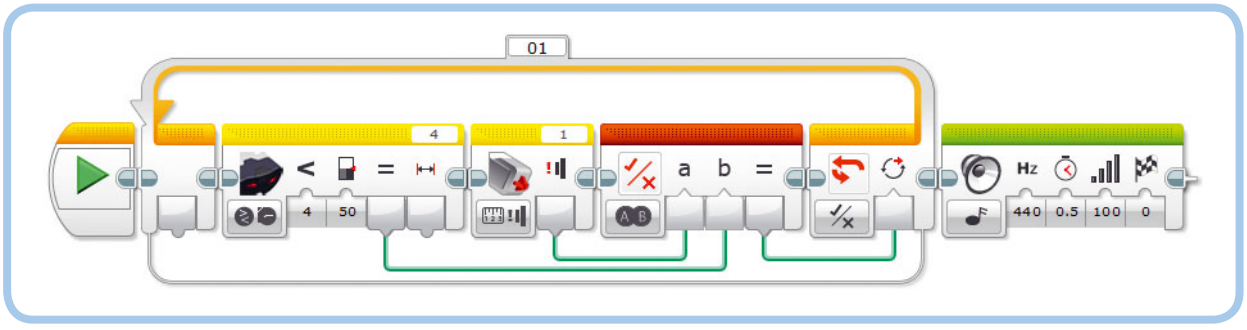


Рис. 15.10. Программа LogicOr

ПРАКТИКУМ № 95: ЛОГИЧЕСКИЕ ДАТЧИКИ!

Сложность: **Время:**

Программа LogicOr воспроизводит звук, когда срабатывает датчик касания или инфракрасный датчик, но она не сообщает, какой датчик стал причиной завершения цикла. Можно ли дополнить программу, чтобы робот произносил слово «Касание», если цикл завершил датчик касания, и «Обнаружено», если был задействован инфракрасный датчик?*

СОВЕТ **Поместите блок Переключатель (Switch) в режиме Логические операции (Logic) сразу же после блока Цикл (Loop) и используйте вывод имеющегося блока Датчик касания (Touch Sensor) для управления переключением.**

* Пункты **Touch** и **Detected** в каталоге звуков Lego.

ПРАКТИКУМ № 96: ДЛЯ ТРЕХ ДАТЧИКОВ!

Сложность: **Время:**

Можно ли разработать программу, которая воспроизводит звук при срабатывании одного из трех датчиков? Программа должна ожидать нажатия кнопки датчика касания, или значения приближения инфракрасного датчика ниже 50%, или яркости отраженного света выше 15%.

СОВЕТ **Взяв за основу программу LogicOr, добавьте блок Датчик цвета (Color Sensor) и еще один блок Логические операции (Logic Operations). Каким образом следует подключать шины данных?**

Блок Интервал

Блок **Интервал** (Range) определяет, находится ли передаваемое через числовую шину данных значение в диапазоне, ограниченном нижним и верхним пределами. Сами граничные значения должны находиться в рамках диапазона.

Если выбран режим **В пределах** (Inside), выходной результат блока будет равен **Истина** (True), если значение параметра **Тестовое значение** (Test Value) находится в пределах диапазона; вывод выдаст **Ложь** (False), если тестовое значение вне этих пределов.

Если выбран режим **За пределами** (Outside), происходит противоположное: выходной результат будет равен **Истина** (True), если тестовое значение находится за пределами указанного диапазона, и **Ложь** (False), если значение в рамках диапазона.

Программа SensorRange (рис. 15.11) содержит блоки **Интервал** (Range) (режим **В пределах** (Inside)) и **Цикл** (Loop) (режим **Логическое значение** (Logic)) и ожидает, пока показание приближения инфракрасного датчика не достигнет значения в диапазоне от 40 до 60, включая эти числа.

Блок Округление

Блок **Округление** (Round) позволяет превратить входное десятичное значение в целое число путем его округления. Вы можете выбрать режим **Округлить к большему** (Round Up) (к примеру, 1,2 или 1,8 становятся 2), **Округлить к меньшему** (Round Down) (1,2 или 1,8 становятся 1) или **До ближайшего** (Nearest) (1,2 становится 1, тогда как 1,5 или 1,8 становятся 2). Также вы можете выбрать режим **Отбросить дробную часть** (Truncate), чтобы отбросить десятичные знаки от числа без округления (1,877 становится 1,8, если вы оставляете только один десятичный знак).

Программа RoundTime (рис. 15.12) отображает время, прошедшее с момента запуска, на экране модуля EV3 путем округления десятичного значения таймера (к примеру, 3,508) до целого числа в секундах (3).

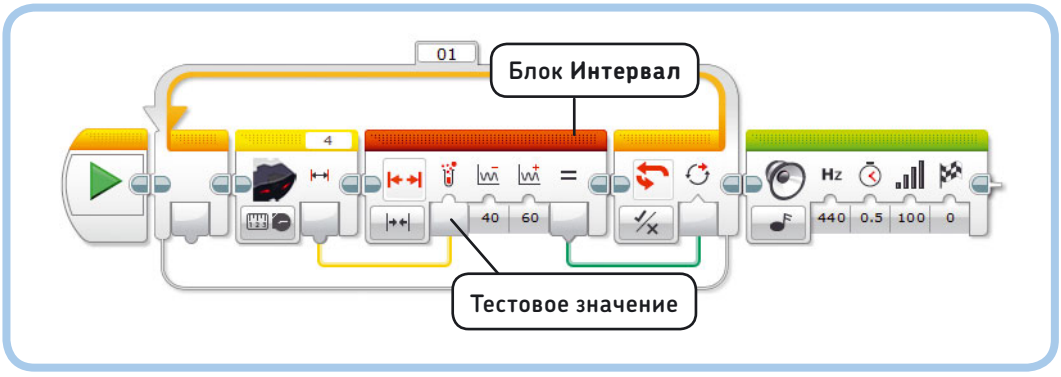


Рис. 15.11. Программа SensorRange

В предыдущих программах мы не использовали блок **Таймер** (Timer), но применять его предельно просто. Блок измеряет время, подобно секундомеру, начиная отсчет с нуля с момента запуска программы. Блок **Таймер** (Timer) в режиме **Измерение** (Measure) выдает текущее время в виде десятичного значения, например 1,500 для полутора секунд. Вы можете сбросить таймер до 0, выбрав режим **Сброс** (Reset) блока.

В этой программе вы можете использовать до восьми различных таймеров. Например, первый таймер для измерения длительности работы программы, второй — для подсчета времени, прошедшего с того момента, как была нажата кнопка датчика касания, обнуляя его при каждом нажатии кнопки. Чтобы указать, значение какого таймера следует считать или сбросить, вы должны выбрать один из восьми *идентификаторов таймера*. В программе RoundTime параметру **Идентификатор таймера** (Timer ID) присвоено значение 1, т.е. используется первый таймер.

Блок Текст

Блок **Текст** (Text) объединяет до трех текстовых вводов шин данных в текстовую строку. Например, если входные текстовые строки содержат значения «EV3 », «это » и «весело», вывод **Результат** (Result) выдаст текст «EV3 это весело»*. Необходимо добавлять пробелы после слов «EV3 » и «это », иначе вывод даст текст «EV3этовесело». Если ввод не содержит значения, он будет проигнорирован.

Можно использовать блок **Текст** (Text) для сочетания текста и чисел и вывода их на экран модуля EV3. Например, вы можете доработать предыдущую программу таким образом, чтобы она выводила текст «Time: 41 s», а не просто число. Чтобы сделать это, вы должны объединить слова «Time:», «s» и числовое значение времени и отобразить результат на экране, как показано в программе TextTime на рис. 15.13.

* Как уже упоминалось ранее в книге, поддерживаются только латинские символы.

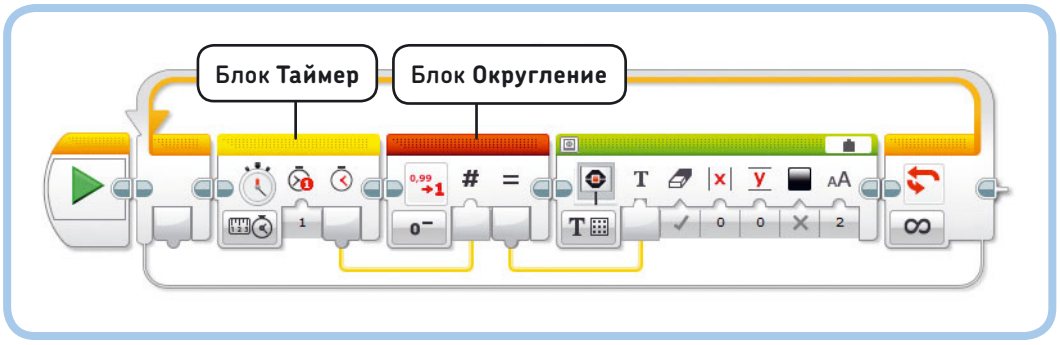


Рис. 15.12. Программа RoundTime

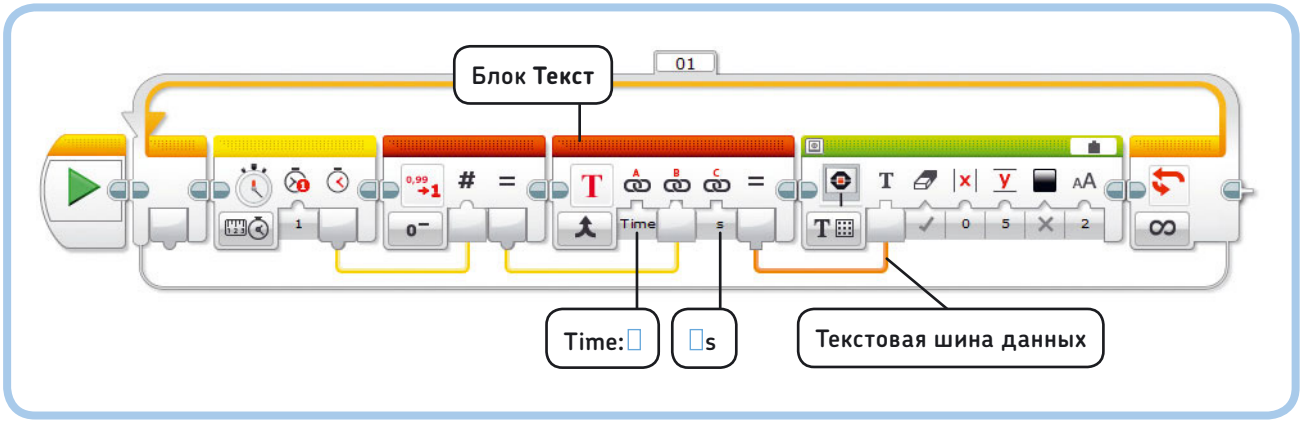


Рис. 15.13. Программа TextTime. Не забудьте о пробелах в словах «Time: », « s» (обозначены на рисунке синими прямоугольниками). При достижении 41 секунды работы программы на экране должен отобразиться текст «Time: 41 s»

ПРАКТИКУМ № 97: ОТСЧЕТ!

Сложность: **Время:**

Можно ли отобразить таймер обратного отсчета на экране модуля EV3? Пусть программа выполняет обратный отсчет от 60 до 0, показывая оставшееся время («46 s to go!»), и пусть воспроизведется оповещение, когда останется 5 секунд. Когда время истечет, робот должен произнести «Игра окончена»*.

СОВЕТ **Используйте программу TextTime в качестве основы. Вместо того чтобы отображать истекшее время, выведите результат этого вычисления: Результат = 60 – Истекшее время. Используйте блоки Сравнение (Compare) для определения, когда оставшееся время составит 5 секунд, а когда оно истечет совсем.**

Создание контейнеров «Мой блок» с шинами данных

До сих пор вы использовали шины данных для передачи информации между существующими программными блоками, например блоками датчиков, действий и операций с данными. Вы можете использовать шины данных со специфическими контейнерами «Мой блок». Благодаря этому вы сможете разрабатывать контейнеры «Мой блок» с *параметрами* (вводами и выводами). Таким образом, контейнер «Мой блок» может обрабатывать значения, переданные в него с помощью шин данных.

В главе 5 вы узнали, как создать базовую версию контейнера «Мой блок». В этом разделе вы научитесь создавать контейнеры «Мой блок» с вводами и выводами и получите некоторые рекомендации по созданию контейнеров «Мой блок».

Контейнеры «Мой блок» с вводами

Для начала вы создадите контейнер «Мой блок» с двумя вводами и именем **DisplayNumber**, как показано на рис. 15.14. Задача этого блока состоит в том, чтобы

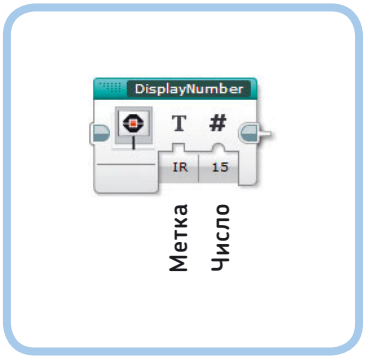


*Рис. 15.14. Контейнер **DisplayNumber**. В этой конфигурации он выводит текст «IR: 15»*

объединять данные, получаемые из текстового ввода **Метка** и числового ввода **Число**, и отображать результат на экране модуля EV3.

Этот блок с легкостью позволит отобразить на экране число с текстовой меткой, которая будет его описывать. Например, если вы присвоите значение IR параметру **Метка** и значение 15 параметру **Число**, блок отобразит на экране текст «IR: 15».

Создайте контейнер «Мой блок», выполнив следующие шаги.

1. Создайте новую программу под названием NumberTest. Она понадобится вам для проверки работы контейнера «Мой блок», когда он будет готов.
2. Установите и настройте блок **Текст** (Text) и блок **Экран** (Display) в области программирования, как показано на рис. 15.15. Вы будете использовать блок **Текст** (Text) для объединения значения ввода **Метка**, двоеточия, за которым следует пробел, и значения ввода **Число** в одну текстовую строку, и блок **Экран** (Display) для дальнейшего вывода ее на экран. Затем с помощью мыши выделите оба блока, заключив их в рамку выделения.

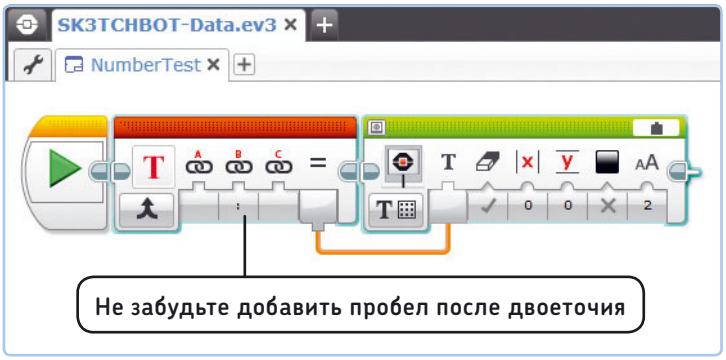


*Рис. 15.15. Настройте блоки **Текст** и **Экран**, как показано на рисунке, а затем выделите оба блока*

* Пункт **Game over** в каталоге звуков Lego.

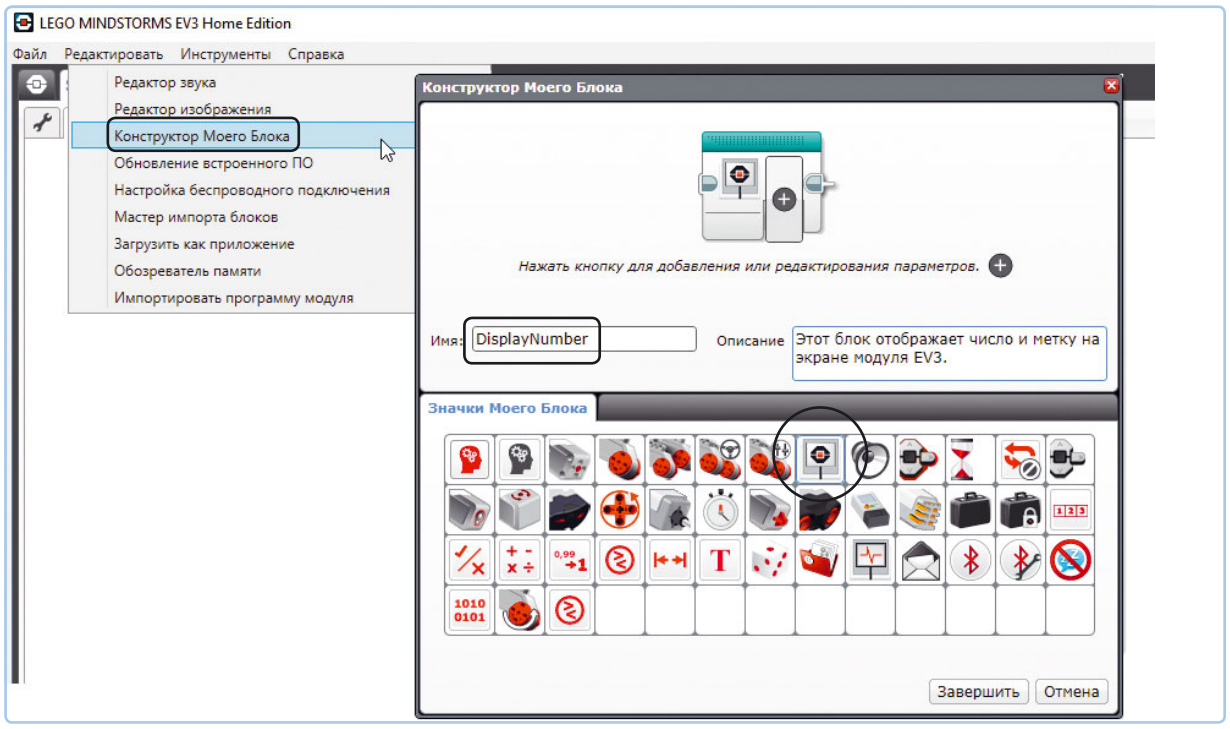


*Рис. 15.16. Откройте окно **Конструктор Моего Блока**, укажите имя и описание контейнера «Мой блок», а затем выберите для него значок. Так как этот блок будет выводить данные на экран модуля EV3, рекомендую выбрать выделенный значок*

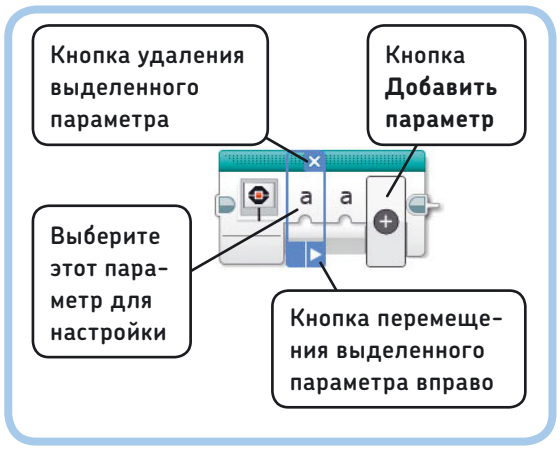


Рис. 15.17. Добавление, удаление и изменение порядка следования параметров

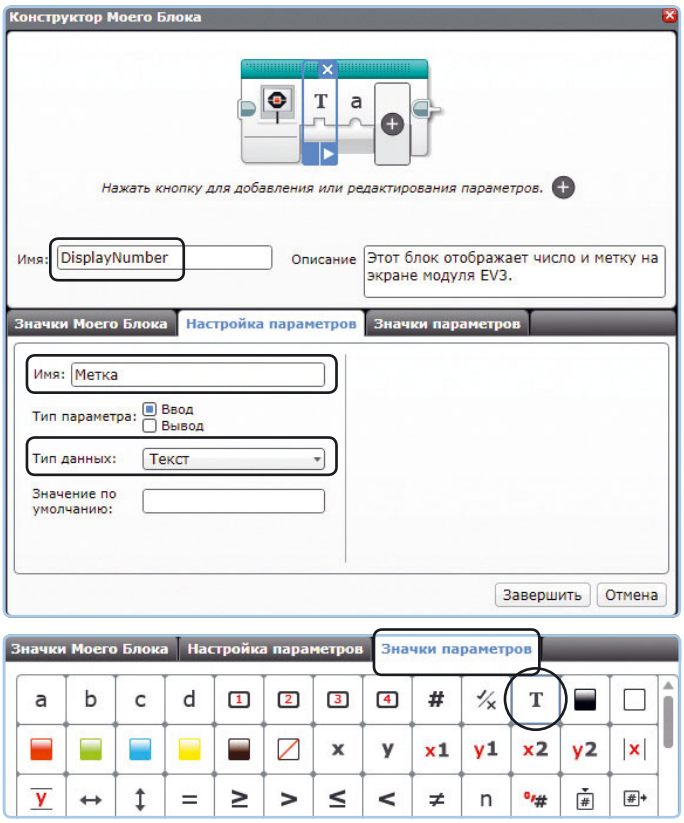


*Рис. 15.18. Настройка ввода **Метка***

3. Создайте новый контейнер «Мой блок», выбрав команду меню **Инструменты** ▸ **Конструктор Моего Блока** (Tools ▸ My Block Builder).

4. Введите текст **DisplayNumber** в качестве имени контейнера «Мой блок», укажите описание и выберите значок для своего блока, как показано на рис. 15.16.

5. Теперь мы настроим параметры блока. Добавьте два ввода, дважды нажав кнопку **Добавить параметр** (Add Parameter), как показано на рис. 15.17.

6. Перейдите на вкладку **Настройка параметров** (Parameter Setup), выберите первый параметр и настройте его, указав тип параметра **Ввод** (Input), тип данных **Текст** (Text) и имя **Метка**, как показано на рис. 15.18. Выберите информативный значок, например в виде буквы «T», для этого ввода на вкладке **Значки параметров** (Parameter Icons).

7. Теперь выберите второй параметр и настройте его, указав тип параметра **Ввод** (Input), тип данных **Число** (Number) и имя **Число**, как показано на рис. 15.19. В поле **Значение по умолчанию** (Default Value) указано число 0, и это означает, что параметру **Число** по умолчанию присвоится значение 0, когда позже вы выберете контейнер на палитре программирования. Выберите стандартный тип ввода в группе элементов управления **Стиль параметра** (Parameter Style) (выделен на рисунке). В этом случае будет создан стандартный числовой ввод, который может принимать любое значение (как вводы блока **Математика** (Math)), а не ползунковый регулятор, который принимает значения только в определенном диапазоне (как, например, параметры **Мощность** (Power) и **Рулевое управление** (Move Steering) блока **Рулевое управление** (Move Steering)).

8. Нажмите кнопку **Завершить** (Finish). Теперь вы можете просматривать содержимое созданного контейнера **DisplayNumber** на его собственной вкладке внутри проекта, как показано на рис. 15.20. Вы должны увидеть параметры **Метка** и **Число**, хотя шины еще не подключены. Эти параметры переносят значения, *переданные в* контейнер «Мой блок» *из основной программы*. Таким образом, если указать значения IR и 15, как показано на рис. 15.14, параметр **Метка** передает значение IR, а **Число** — значение 15.

9. Завершите создание контейнера «Мой блок» путем подключения выводов **Метка** и **Число** к блоку **Текст** (Text), как показано на рис. 15.20, и сохраните проект. Помните, что блок **Текст** (Text) объединяет данные с вводов в одну текстовую строку, а блок **Экран** (Display) отображает ее на экране.

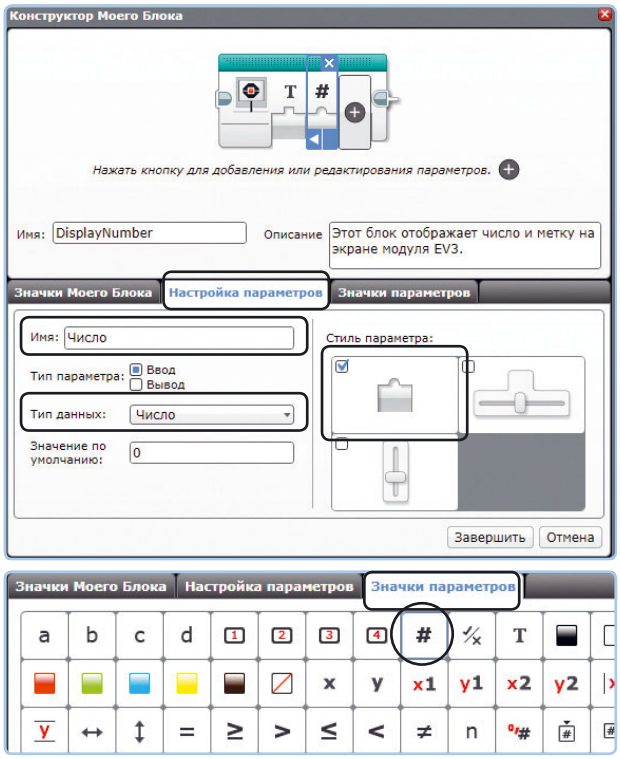


*Рис. 15.19. Настройка ввода **Число***

Поздравляем, вы создали контейнер «Мой блок» с вводами! Он размещается на голубой вкладке палитры программирования. Теперь завершите программу NumberTest, чтобы протестировать новый контейнер, как показано на рис. 15.21. В этом примере основная программа (NumberTest) передает два значения в контейнер **DisplayNumber**. Блоки внутри

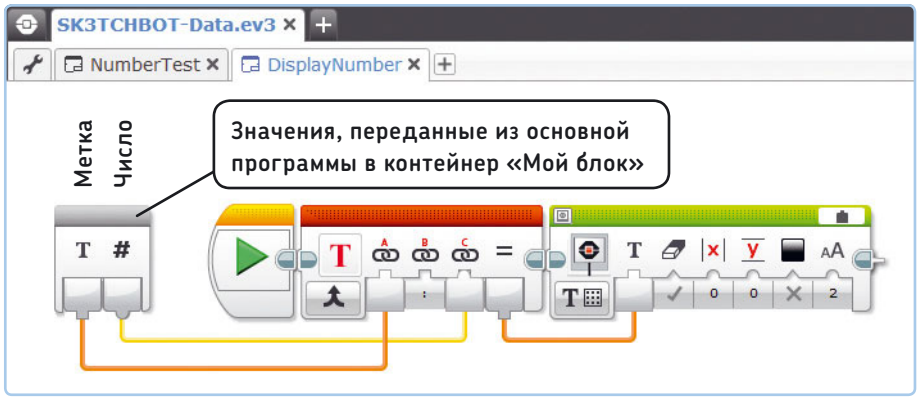


*Рис. 15.20. После нажатия кнопки **Завершить** в окне **Конструктор Моего Блока** вы увидите содержимое контейнера **DisplayNumber** на отдельной вкладке в проекте SK3TCHBOT-Data. Завершите работу с блоком посредством подключения шин данных так, как показано на рисунке*

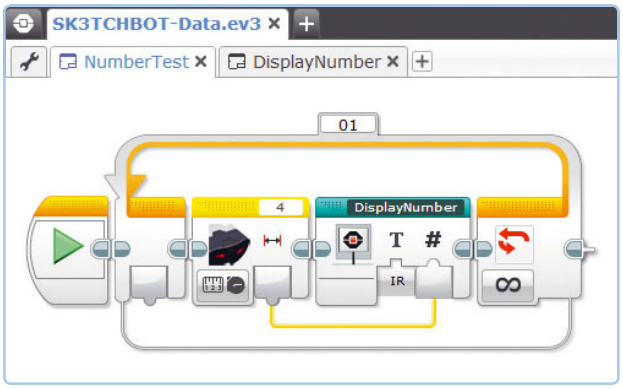


*Рис. 15.21. Программа NumberTest. Блок **Инфракрасный датчик** отправляет значение параметра **Приближение** на числовой ввод **Число** контейнера «Мой блок». Значение параметра **Метка** введено вручную (**IR**). Контейнер «Мой блок» объединяет значение параметра **Метка** с двоеточием, пробелом и значением инфракрасного датчика (параметра **Приближение**) и выводит результат на экран модуля EV3. При получении результата приближения, равного, к примеру, 65 на экране отобразится строка «IR: 65»*

контейнера «Мой блок» извлекают эти значения и отображают их на экране модуля EV3.

Изменение контейнеров «Мой блок»

После того как контейнер «Мой блок» создан, вы можете настроить его функции, дважды щелкнув по нему мышью и изменив вложенные блоки. Например, можно добавить блок **Звук** (Sound) в контейнер **DisplayNumber**, чтобы робот воспроизводил сигнал каждый раз, когда на экране модуля отображается значение.

На момент написания этой книги программное обеспечение LEGO MINDSTORMS EV3 (версия 1.2) не позволяло изменять входные и выходные параметры контейнеров «Мой блок» после того, как завершено его создание в окне **Конструктор Моего Блока** (My Block Builder). Если вы хотите изменить параметры или добавить новые, вам придется создать новый контейнер «Мой блок» с нуля. После того как блок с нужными параметрами создан, скопируйте содержимое старого блока и вставьте его в новый, чтобы сэкономить время.

Контейнеры «Мой блок» с выводами

Кроме контейнеров «Мой блок» с вводами, вы можете создавать пользовательские блоки с выводами. В этом разделе вы соберете контейнер «Мой блок», определяющий направление движущихся объектов с помощью

* Мотор B: 375 оборотов.

ПРАКТИКУМ № 98: МОЙ МОДУЛЬ!

Сложность: **Время:**

Создайте новый контейнер «Мой блок» на основе контейнера **DisplayNumber** с дополнительным текстовым вводом с именем **Модуль** (см. рис. 15.22). Затем добавьте метку и единицу измерения значения. Например, отображение значения, получаемого от датчика вращения, можно сопроводить текстовой меткой «MB» и единицей измерения «Deg»: «MB: 375 Deg» (сокращение для строки «Motor B: 375 Degrees»*).

СОВЕТ **Вам понадобится дополнительный блок Текст (Text).**

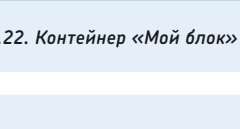


Рис. 15.22. Контейнер «Мой блок» из практикума № 98

ПРАКТИКУМ № 99: ПРОДВИНУТЫЙ ИНТЕРФЕЙС!

Сложность: **Время:**

Контейнер **DisplayNumber**, который вы только что создали, применим для отображения одного значения в верхней части экрана. Можно ли создать продвинутую версию этого контейнера «Мой блок» с возможностями выбора номера строки и удаления содержимого экрана, как показано на рис. 15.23?

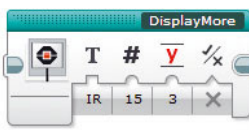


Рис. 15.23. Контейнер «Мой блок» из практикума № 99

СОВЕТ **На экране модуля EV3 пространства достаточно для размещения шести текстовых строк, если вы используете самый большой размер шрифта. Используйте блок Математика (Math), чтобы преобразовать значение на вводе КоличествоСтрок контейнера «Мой блок» в значение для параметра Строка (Row) блока Экран (Display): Строка (Row) = КоличествоСтрок × 2.**

инфракрасного датчика. Контейнер **Direction** будет содержать один логический вывод с именем **Приближение**, как показано на рис. 15.24.

Вывод будет отправлять значение **Истина** (True), если объект приближается к датчику (расстояние до объекта сокращается); в противном случае вывод передаст значение **Ложь** (False) (если расстояние до объекта увеличивается или остается неизменным). Создайте контейнер **Direction** и тестовую программу DirectionSound, следуя приведенным ниже инструкциям.

1. Создайте новую программу с именем DirectionSound и поместите в нее два блока **Инфракрасный датчик** (Infrared Sensor), блок **Ожидание** (Wait) и блок **Сравнение** (Compare), как показано на рис. 15.25. Эти блоки выполняют два замера приближения с разницей в 0,2 секунды и сравнивают их, определяя, является ли второе значение (**a**) меньше, чем первое (**b**). Если это так, объект приближается к датчику, и вывод блока **Сравнение** (Compare) выдает значение **Истина** (True).

2. Выделите все четыре блока и выберите команду меню **Инструменты** ▸ **Конструктор Моего Блока** (Tools ▸ My Block Builder). Укажите слово **Direction** в качестве имени блока и выберите значок в виде инфракрасного датчика, как показано на рис. 15.26.

3. Добавьте один параметр и настройте его, указав тип параметра **Вывод** (Output), тип данных **Логическое значение** (Logic) и имя **Приближение**. Затем выберите подходящий значок, как показано на рис. 15.26. Нажмите кнопку **Завершить** (Finish).

4. Чтобы завершить работу с контейнером «Мой блок», подключите вывод блока **Сравнение** (Compare) к вводу **Приближение**, как показано на рис. 15.27. Затем это значение передается в основную программу, которая содержит контейнер **Direction**.

Теперь вернемся к программе DirectionSound и протестируем наш контейнер «Мой блок», заставив робота воспроизвести высокий тон, когда объект приближается (**Истина** (True)), и низкий, если объект отдаляется или стоит на месте (**Ложь** (False)), как показано на рис. 15.28. В этом примере блоки внутри контейнера **Direction** вычисляют направление движущегося объекта и передают результат в основную программу (DirectionSound) через вывод **Приближение**.

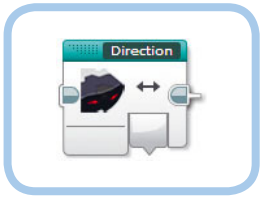


Рис. 15.24. Контейнер Direction

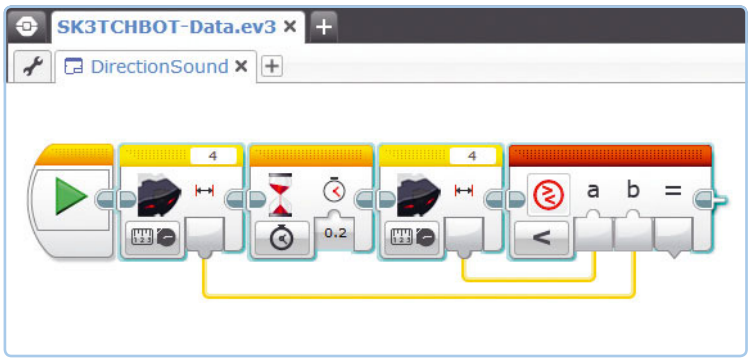


*Рис. 15.25. Разместите и настройте блоки в контейнере **Direction**, как показано на рисунке. Когда все будет готово, выделите блоки мышью и выберите команду меню **Инструменты** ▸ **Конструктор Моего Блока***

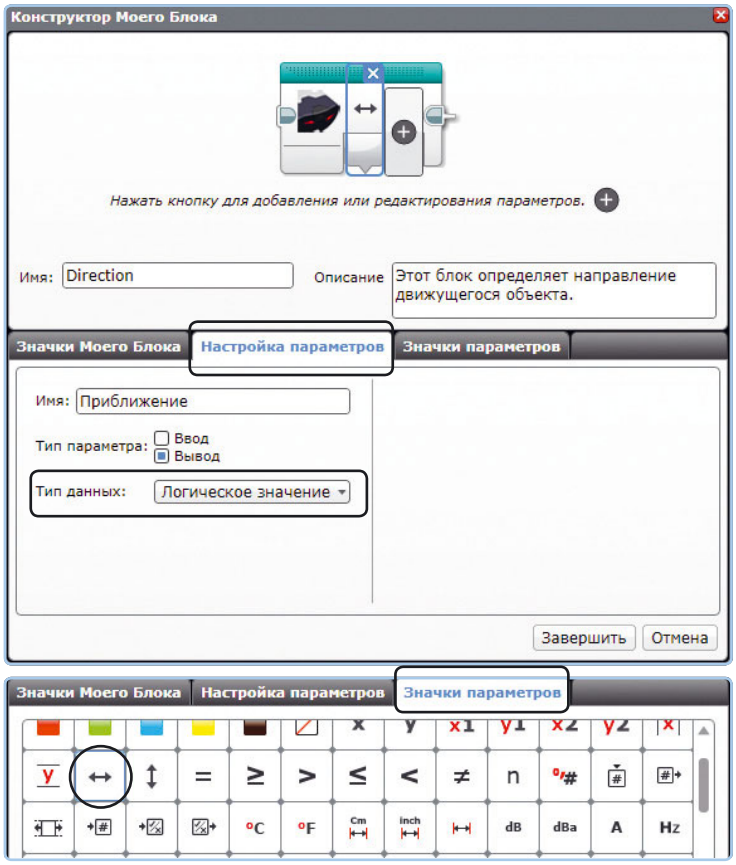


Рис. 15.26. Выполните настройку контейнера «Мой блок» и его логического вывода, как показано на рисунке

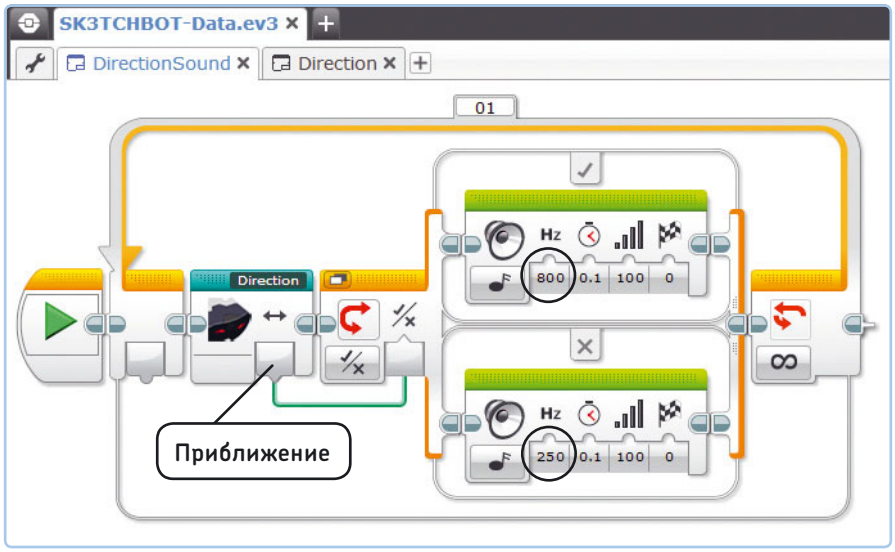


*Рис. 15.27. Программа DirectionSound использует логическое значение, передаваемое на вывод **Приближение**, для воспроизведения высокого тона, если какой-либо объект приближается, и низкого тона, если объект не приближается*

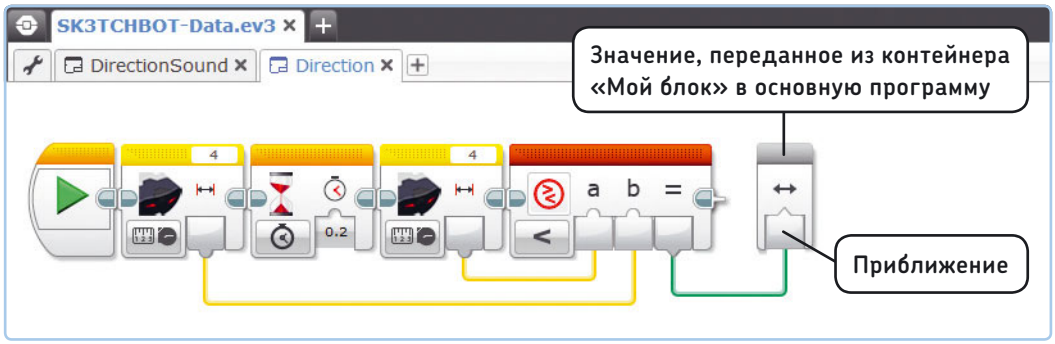


*Рис. 15.28. Подключите вывод блока **Сравнение** к вводу **Приближение***

ПРАКТИКУМ № 100: СРЕДНЕЕ ЗНАЧЕНИЕ ПРИБЛИЖЕНИЯ!

Сложность:

Время:

Создайте контейнер «Мой блок», который вычисляет среднее значение из двух замеров приближения. Затем он должен выдавать результат с помощью числовой шины данных. Как и в контейнере **Direction**, измерения следует делать с интервалом в 0,2 секунды.

ПРАКТИКУМ № 101: СКОРОСТЬ ПРИБЛИЖЕНИЯ!

Сложность: **Время:**

Создайте контейнер «Мой блок» с одним числовым выводом, который определяет скорость движущегося объекта путем вычисления скорости изменения значений приближения. Для этого блок должен выполнить два замера, как и в контейнере **Direction**, а затем вычислить скорость изменений по следующей формуле:

$$\text{Скорость} = \frac{\text{Второе измерение} - \text{Первое измерение}}{\text{Время между измерениями}}$$

Чтобы выполнить проверку блока, отобразите значение скорости на экране и измените подсветку модуля на зеленый цвет, если объект приближается, оранжевый, если объект не двигается, и красный, если объект удаляется от датчика. Как можно определить направление, используя значение скорости?

Контейнеры «Мой блок» с вводами и выводами

В следующем примере вы создадите контейнер «Мой блок» с вводами и выводами, показанный на рис. 15.29. Контейнер **IsEven** определяет, является ли четным числовое значение на вводе **Число**. Если это так, логический вывод **Четность** передаст значение **Истина** (True); в противном случае — значение **Ложь** (False). Создайте блок, выполнив следующие шаги.

1. Создайте новую программу с именем EvenSound и разместите блоки **Математика** (Math) и **Сравнение** (Compare) в области программирования. Пока оставьте их настройки без изменений.

2. Выделите оба размещенных блока и выберите команду меню **Инструменты** ▸ **Конструктор Моего Блока** (Tools ▸ My Block Builder). Введите текст **IsEven** в качестве имени блока и выберите соответствующий значок, как показано на рис. 15.29.

3. Добавьте один числовой ввод и присвойте ему имя **Число** и один логический вывод с именем **Четность**. Выберите значки, как показано на рис. 15.29. Нажмите кнопку **Завершить** (Finish).

4. Теперь завершите работу с контейнером «Мой блок», настроив оба блока и подключив шины данных, как показано на рис. 15.30. В блоке **Математика** (Math) используется оператор деления по модулю (%), который вычисляет остаток от деления двух чисел. В этом примере он сообщает остаток от деления входного значения **a** на 2. Четные числа делятся на 2, поэтому остаток будет равен 0, и блок **Сравнение** (Compare) выдаст значение **Истина** (True). Для нечетных чисел остаток не будет равен 0, поэтому блок **Сравнение** (Compare) выдаст значение **Ложь** (False). Например, выражение 7/2 даст в остатке 1, и блок **Сравнение** (Compare) выдаст значение **Ложь** (False), сообщающее, что на ввод передано нечетное число.

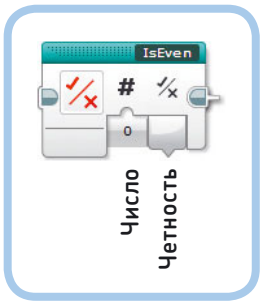


Рис. 15.29. Контейнер IsEven

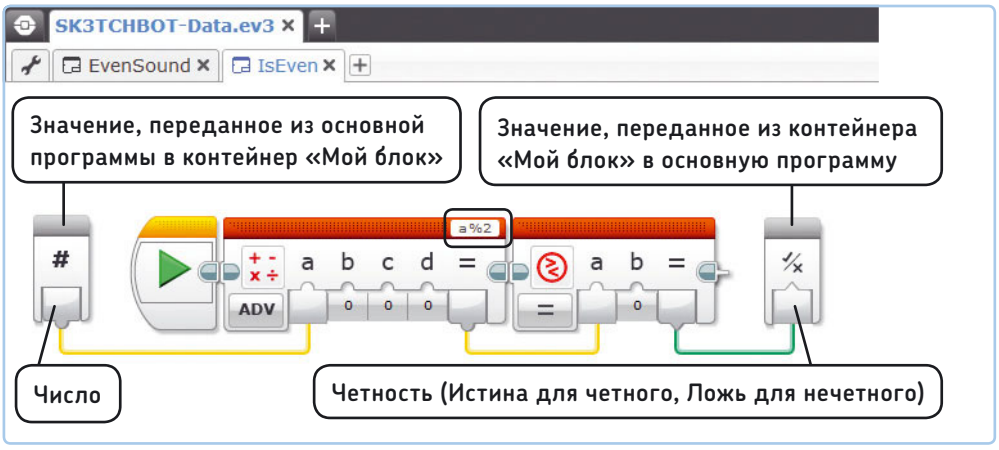


*Рис. 15.30. Настройка блоков в контейнере **IsEven***

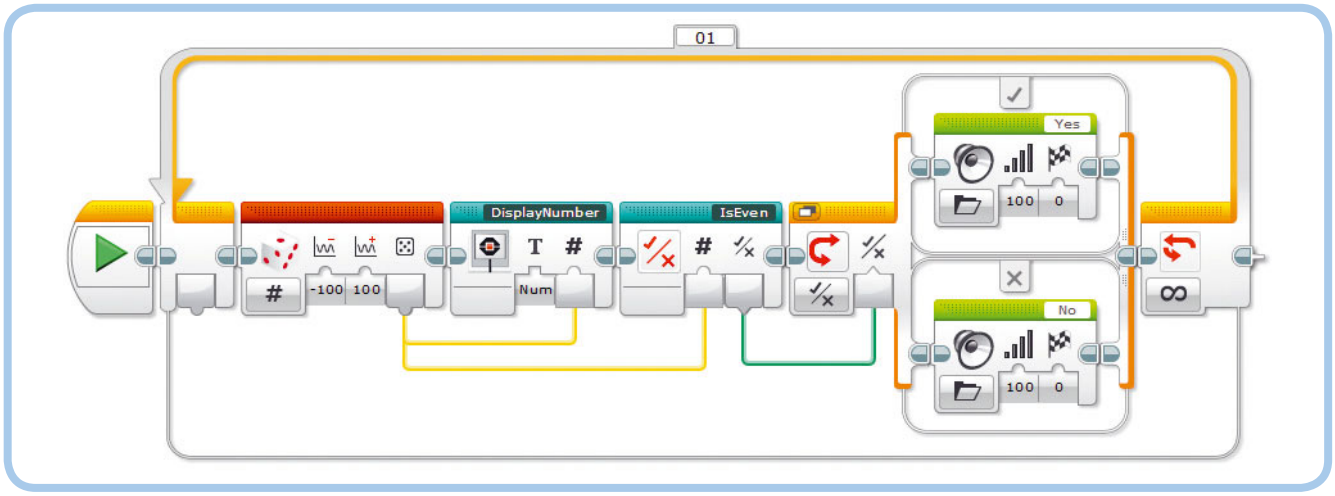


Рис. 15.31. Программа EvenSound

Теперь, когда вы закончили работу с контейнером «Мой блок», протестируйте его, завершив программу EvenSound, показанную на рис. 15.31. Программа отображает случайное число в диапазоне от –100 до 100 на экране модуля EV3 и определяет, четное оно или нет, используя контейнер **IsEven**. Робот произносит «Да» при обнаружении четных чисел и «Нет» — при нечетных.

ПРАКТИКУМ № 102: МАТЕМАТИКА КРУГА!

Сложность: **Время:**

Можете ли вы создать контейнер «Мой блок», который будет вычислять длину окружности и площадь круга, используя радиус круга в качестве входного значения? Создайте контейнер «Мой блок» с одним числовым вводом с именем **Радиус** и двумя числовыми выводами **Окружность** и **Площадь**.

СОВЕТ **Используйте следующие формулы, учитывая, что число π (pi) приблизительно равняется 3,14:**

Окружность = 2 × π × Радиус
Площадь = Радиус^2 × π

Рекомендации по использованию контейнеров «Мой блок»

Контейнеры «Мой блок» с вводами и выводами позволяют выполнять множество различных задач. Их можно использовать в следующих случаях.

Повторяющиеся задачи: некоторые действия одинаковы для многих программ, как, например, вывод значений на экран модуля. Если вы один раз создадите контейнер «Мой блок» для выполнения подобной задачи, в долгосрочной перспективе вы сэкономите много времени.

Организация блоков: вы можете собрать блоки сложной программы в контейнеры «Мой блок», чтобы было проще разобраться в ней и по отдельности протестировать ее части. Например, если требуется создать программу для автономного робота-манипулятора, можно собрать один контейнер «Мой блок» для определения местоположения объекта, второй — для перемещения к нему и третий для подъема и переноса объекта.

Обработка информации: поскольку вы создаете программы, которые обрабатывают значения с помощью блоков операций с данными, будет сложно понять, как на самом деле работает программа, когда вы взглянете на нее спустя некоторое время. Чтобы решить эту проблему, вы можете раздробить сложные вычисления на более мелкие части и сгруппировать их в контейнеры «Мой блок». Например, в программе EvenSound (см. рис. 15.31) сокрытие операции деления по модулю в контейнере **IsEven** упрощает понимание, как работает программа.

Способы создания контейнеров «Мой блок»

Вы можете создавать контейнеры «Мой блок» двумя способами.

Преобразовать группу блоков в контейнер «Мой блок»: используйте этот способ, если вы уже настроили и протестировали блоки в вашей программе. Дробление большой программы на подразделы, каждый из которых обладает собственной функциональностью, — весьма полезный метод. Если группа блоков подключена к другим блокам в вашей программе с помощью шин данных, вам понадобится настроить параметры контейнера «Мой блок» для каждой шины данных. На самом деле, инструмент **Конструктор Моего Блока** (My Block Builder) автоматически настраивает такие параметры, но вам все-таки придется добавить имя и значок для каждого из них.

Создать контейнер «Мой блок» с нуля: используйте этот способ, если знаете, что должен делать ваш контейнер «Мой блок», но пока не уверены в том, как настроить блоки внутри него, чтобы выполнить задачу. Например, если вы хотите создать блок, который будет определять, является число четным или нечетным, но не знаете, как точно он будет работать, вы можете начать с создания контейнера «Мой блок» с одним числовым вводом и одним логическим выводом. Затем поэкспериментируйте с блоками внутри контейнера, пока не решите поставленную задачу. Обратите внимание, что вы не можете запустить инструмент **Конструктор Моего Блока** (My Block Builder), не выделив хотя бы один блок, поэтому вам придется начать с временного блока, например **Ожидание** (Wait), и позже удалить его.

Использование блоков в разных проектах

Когда вы создаете контейнер «Мой блок», он становится доступен для использования только внутри текущего проекта. Чтобы использовать его в рамках другого проекта, вам придется скопировать его или экспортировать из вашего текущего проекта и вставить или импортировать его в другой проект. (См. раздел «Управление контейнерами “Мой блок” в проектах» главы 5.)

Дальнейшее изучение

В этой главе вы узнали о блоках операций с данными, а также о том, как создавать контейнеры «Мой блок» с вводами и выводами для подключения шин данных. Эти методы позволят вашему роботу обрабатывать и объединять значения, получаемые от датчиков, чтобы использовать их в качестве входных данных для действий, таких как перемещение и звуковое оповещение. Вы увидите множество практических примеров в оставшихся главах этой книги. А теперь укрепите приобретенные навыки, выполняя приведенные ниже практикумы.

ПРИМЕЧАНИЕ **Помните, что вы можете найти решения для многих практикумов по ссылке eksmo.ru/files/Lego_Mindstorms_Primers.zip.**

ПРАКТИКУМ № 103: ЭТО ЦЕЛОЕ ЧИСЛО?

Сложность: **Время:**

Можно ли создать контейнер «Мой блок», который определяет, каким числом представлено входящее числовое значение — целым или десятичным? Создайте контейнер «Мой блок» с именем **IsInteger** и с параметрами, аналогичными контейнеру **IsEven**, но используйте другие вложенные блоки.

СОВЕТ **Округлите входящее значение и сравните его с исходным. Почему в некоторых случаях входящее значение идентично округленному?**

ПРАКТИКУМ № 104: ДВОЙНАЯ ПОТЕРЯ СКОРОСТИ!

Сложность: **Время:**

Можно ли создать программу, с помощью которой оба мотора, B и C, будут вращаться до тех пор, пока один из них не заглохнет? Когда программа будет завершена, вложите блоки, отвечающие за ожидание остановки мотора, в контейнер «Мой блок» с именем **WaitForStall**, чтобы вы могли использовать его в любой из ваших программ. Он будет особенно полезен для таких роботов, как EXPLOR3R и FORMULA EV3.

СОВЕТ **Часть этой программы похожа на программу LogicOr, показанную на рис. 10.15. Используйте два блока Вращение мотора (Motor Rotation) в режиме Сравнение ▸ Текущая мощность (Compare ▸ Current Power).**

ПРАКТИКУМ № 105: ТЕСТ НА РЕФЛЕКС!

Сложность: **Время:**

Можно ли создать программу для проверки вашей реакции? Сделайте так, чтобы цвет индикатора состояния оставался зеленым в течение произвольного времени, а затем неожиданно загорался красным. Как только вы увидите красный свет, вы должны нажать кнопку датчика касания. Затем программа должна отобразить, сколько времени прошло с момента, когда была включена красная подсветка, до нажатия кнопки. Вложите программу в цикл, чтобы увидеть, сможете ли вы улучшить время вашей реакции! Когда будете готовы, дополните программу таким образом, чтобы избегать жульничества с преждевременным нажатием кнопки датчика касания.

СОВЕТ **Работа программы заключается в следующем: установить зеленый цвет подсветки. Подождать некоторое время. Установить красный цвет подсветки. Сбросить таймер. Подождать нажатия кнопки датчика касания. Отобразить значение таймера на экране. Подождать 3 секунды, чтобы вы могли прочитать результат.**

СДЕЛАЙ САМ № 26: РОБОТИЗИРОВАННЫЕ ЧАСЫ!

Сборка: **Программирование:**

Можно ли сконструировать работающие часы EV3? Используйте три мотора из набора MINDSTORMS EV3, чтобы управлять часовой, минутной и секундной стрелками на ваших часах. Используйте блок **Таймер** (Timer) и вычисляйте положение каждой стрелки, основываясь на количестве секунд, которые прошли с момента запуска программы.

СОВЕТ **Во время тестирования робота увеличьте значение таймера в 10 или более раз, чтобы упростить проверку движения часовой стрелки.**

16

Использование констант и переменных

Если вы добрались до этой главы, значит, вы уже почти овладели всеми навыками программирования, описанными в этой книге. Вы научились применять множество различных программных блоков и освоили работу с основными инструментами, такими как шины данных. Эта глава завершает раздел, посвященный программированию, и из нее вы узнаете, как использовать константы и память EV3 для работы с переменными.

Использование констант

Блок **Константа** (Constant) представляет собой начальную точку шины данных, значение которой вы можете указать вручную. Тип шины данных определяется выбором режима (**Текст** (Text), **Числовое значение** (Numeric) или **Логическое значение** (Logic)), а значение вводится в поле **Значение** (Value), как показано на рис. 16.1.

Блок **Константа** (Constant) удобно использовать, если вам необходимо сконфигурировать параметры нескольких блоков с одинаковым значением. Например, программа ConstantDemo (рис. 16.1) заставляет моторы робота SK3TCHBOT вращаться с одинаковой скоростью, но в разных направлениях. В довершение всего блок **Константа** (Constant) отправляет значение 50 первому блоку **Большой мотор** (Large Motor) и блоку **Математика** (Math), который умножает значение на –1 перед тем, как передать его второму блоку **Большой мотор** (Large Motor). Чтобы одновременно изменить скорость вращения обоих моторов, достаточно просто указать другое значение в блоке **Константа** (Constant). Если бы этого блока не было, пришлось бы менять значение дважды.

Использование переменных

Представьте, что переменная — это чемодан, в котором можно хранить информацию. Когда программе нужно запомнить

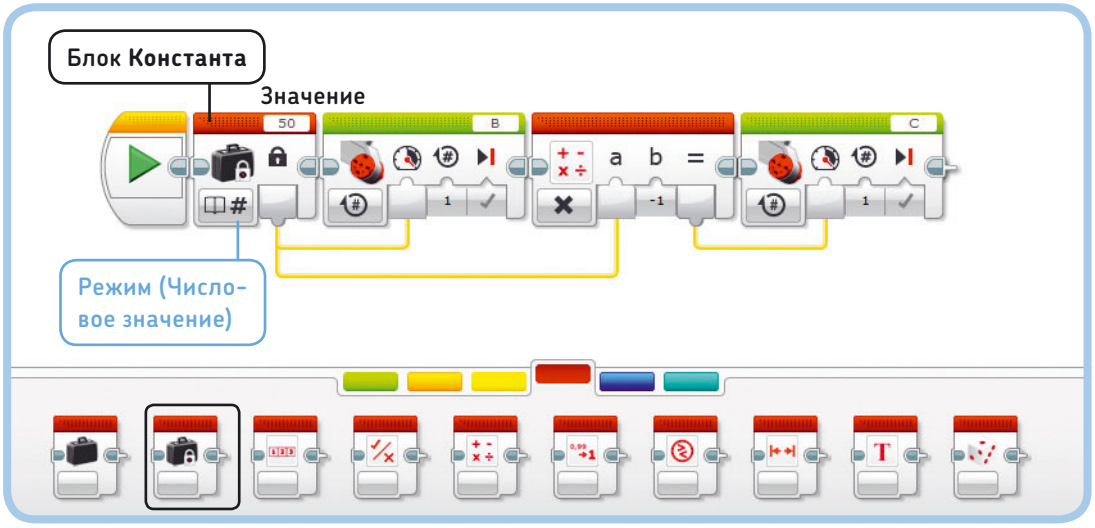


Рис. 16.1. Программа ConstantDemo. Для работы с программами этой главы создайте новый проект с именем SK3TCHBOT-Variable

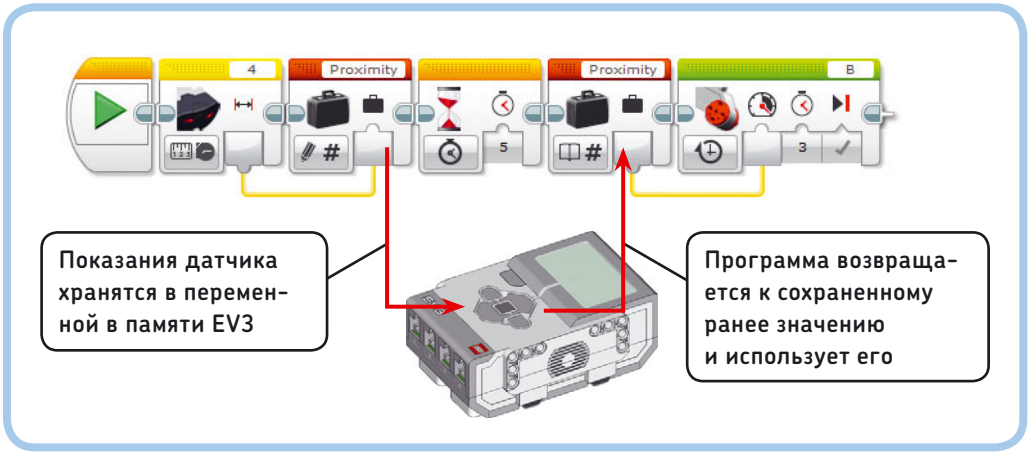


Рис. 16.2. Значение, к примеру расстояние, показанное на рисунке, можно хранить в переменных в памяти модуля EV3. Далее вы узнаете, на что способна эта программа и как ее написать

значение (к примеру, показания датчика), чтобы обратиться к нему в дальнейшем, она помещает значение в этот чемодан и убирает его в сторонку. Когда программе необходимо применить это значение, она открывает «чемодан» и использует сохраненное значение. Переменная хранится в памяти модуля EV3, пока не понадобится.

К хранящейся в переменной информации можно получить доступ из других частей программы. К примеру, можно поместить показания приближения инфракрасного датчика в этот «чемодан» и использовать их для управления скоростью вращения мотора через 5 секунд.

Программа может получить доступ к сохраненной информации в любой момент во время работы, но, как только она завершит работу, данные будут утеряны. Для сохранения и применения информации из переменных используйте блок **Переменная** (Variable), который можно узнать по значку в виде чемодана. На рис. 16.2 вкратце показано, что происходит при использовании переменных.

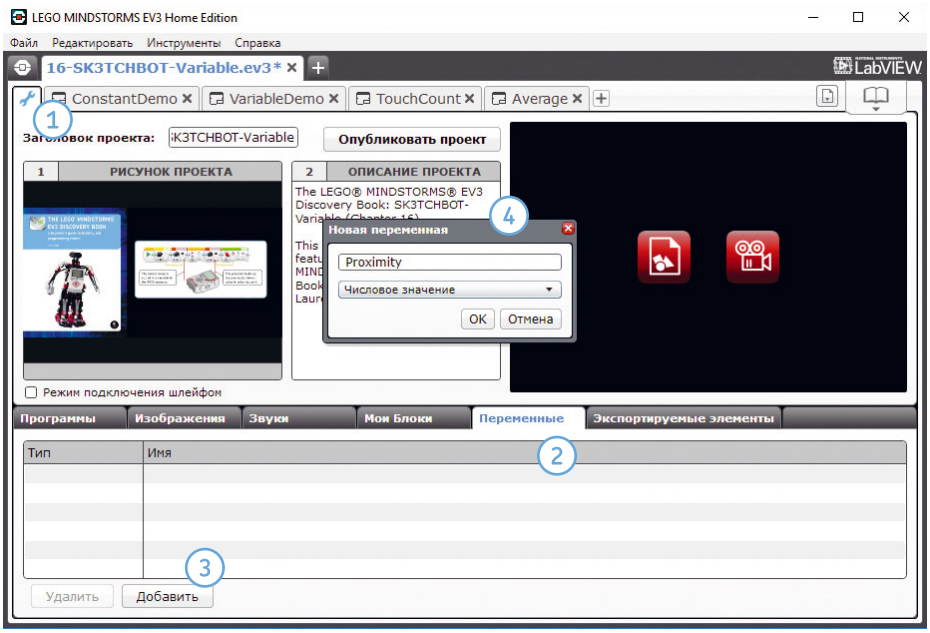


*Рис. 16.3. Определение переменной на странице **Свойства проекта**. Шаг 1: откройте страницу **Свойства проекта**. Шаг 2: перейдите на вкладку **Переменные**. Шаг 3: чтобы создать новую переменную, нажмите кнопку **Добавить**. Шаг 4: введите имя новой переменной — **Proximity** и выберите тип данных **Числовое значение**. Затем нажмите кнопку **OK***

Определение переменной

У каждой переменной есть имя, тип данных и содержащееся в ней значение. Например, переменной с числовым значением можно присвоить имя **Proximity** и значение **56**. Кроме числовых переменных существуют логические (содержат значения **Истина** (True) или **Ложь** (False)) и текстовые (содержат строку текста, например «Hello»*).

* Обратите внимание, что как в именах, так и в значениях переменных можно использовать только латинские буквы, цифры и некоторые специальные символы.

ПРИМЕЧАНИЕ **В списке Название переменной (Name) содержатся только те переменные, тип которых подходит к выбранному режиму блока. Например, чтобы просмотреть только что созданную числовую переменную Proximity, блок должен находиться в режиме Считывание ▸ Числовое значение (Read ▸ Numeric) или Записать ▸ Числовое значение (Write ▸ Numeric).**

Перед использованием переменной в программе необходимо *определить* ее, указав имя и тип. Это можно сделать на странице **Свойства проекта** (Project Properties), как показано на рис. 16.3, или использовать блок **Переменная** (Variable), как вы скоро увидите.

Определив переменную в проекте, вы сможете использовать ее в любой программе в рамках данного проекта.

Для удаления переменной откройте вкладку **Переменные** (Variables) на странице **Свойства проекта** (Project Properties), выберите переменную, которую хотите удалить, и нажмите кнопку **Удалить** (Delete).

Использование блока Переменная

Определив переменную, в дальнейшем ее можно использовать в программе с помощью блока **Переменная** (Variable). Блок **Переменная** (Variable) позволяет считывать данные из памяти EV3 или записывать (сохранять) данные в память, как показано на рис. 16.4.

Для настройки блока **Переменная** (Variable) сначала перейдите в меню выбора режима и укажите, следует ли считывать данные переменной (**Считывание** (Read), значок в виде книги) или записывать данные в переменную (**Записать** (Write), значок в виде карандаша). Затем выберите тип переменной, которую необходимо считать или записать (**Числовое значение** (Numeric), **Логическое значение** (Logic) или **Текст** (Text)). Наконец, в списке **Название переменной** (Name) выберите переменную, которую следует использовать.

У блока **Переменная** (Variable) есть параметр **Значение** (Value). В режиме **Записать** (Write) вы можете присвоить этому параметру значение, которое нужно сохранить. Если вместо ввода значения подключить шину данных, в параметре будет сохранено переданное через нее значение. Если в переменной уже сохранено какое-либо значение, оно будет стерто и вместо него будет записано новое.

В режиме **Считывание** (Read) блок **Переменная** (Variable) извлекает информацию из памяти модуля EV3 и присваивает ее в качестве значения параметру **Значение** (Value) так, что эти данные можно передать другим блокам через шину данных. При считывании значение не меняется, поэтому, если считать его повторно с помощью другого блока **Переменная** (Variable), оно останется неизменным.

Определение переменных с помощью блока Переменная

Другой способ задать переменную — выбрать команду **Add Variable** (добавить переменную) в раскрывающемся списке **Название переменной** (Name) блока **Переменная** (Variable), как показано на рис. 16.5. При этом создается новая переменная того же типа, что и выбранный режим блока. Например, когда блок находится в режиме **Считывание** ▸ **Числовое значение** (Read ▸ Numeric), новая переменная будет числовой. Для создания таким способом логической переменной сначала следует изменить режим на **Считывание** ▸ **Логическое значение** (Read ▸ Logic) или **Записать** ▸ **Логическое значение** (Write ▸ Logic).

*Рис. 16.4. Хранение и считывание значений с помощью блока **Переменная***

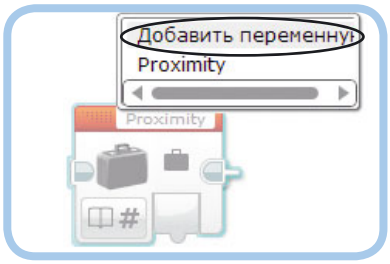


*Рис. 16.5. Для определения переменной непосредственно в блоке **Переменная** выберите команду **Добавить переменную**. Введите имя переменной в появившемся диалоговом окне и нажмите кнопку **OK***

Создание программ с переменными

Теперь, когда вы познакомились с принципами определения и использования переменных, можно приступать к разработке программы VariableDemo, показанной на рис. 16.6. Эта программа сохраняет значение приближения инфракрасного датчика в переменную с именем **Proximity**. Через 5 секунд она извлекает значение из переменной и использует его для управления скоростью мотора B, а это значит, что скорость мотора зависит от показаний

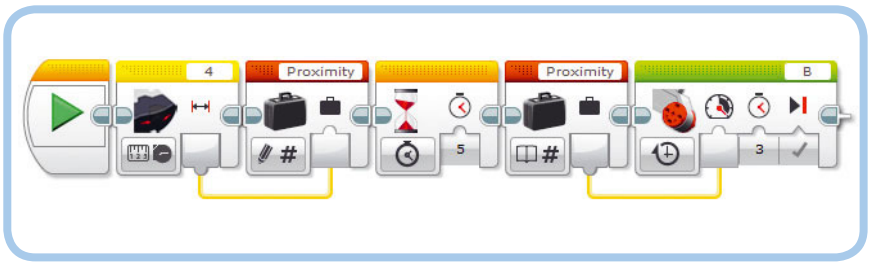


Рис. 16.6. Программа VariableDemo

ПРАКТИКУМ № 106: СТАРОЕ ИЛИ НОВОЕ!

Сложность: **Время:**

Можете ли вы разработать программу, которая многократно сравнивала бы новые показания датчика с показаниями, сохраненными в переменной при запуске программы? Если новые показания окажутся меньше, робот должен сказать «Да», а если больше — «Нет»*. Незаконченная версия программы показана на рис. 16.7.

СОВЕТ **С чего вы начнете при разработке программы с переменными? Какой блок Переменная (Variable) должен считывать или записывать значение? Как подключить шины данных?**

ПРАКТИКУМ № 107: ПРЕДЫДУЩЕЕ ИЛИ НОВОЕ!

Сложность: **Время:**

Программа, описанная в практикуме № 106, сравнивает новые показания датчика со значением, измеренным при запуске программы. Можете ли вы разработать программу, которая сравнивала бы каждое новое значение с предыдущим? Если новое значение окажется меньше, робот должен сказать «Да», а если больше — «Нет»**. Следовательно, при приближении объекта к инфракрасному датчику спереди робот будет говорить «Да»***.

СОВЕТ **Напишите программу, похожую на показанную на рис. 16.7, с одной переменной Proximity. Каждый раз в цикле роботу нужно будет сравнивать значение переменной Proximity с новыми показаниями датчика. Последним блоком цикла должен быть блок Переменная (Variable), запрограммированный сохранять новые показания датчика в качестве значения переменной Previous, тогда при следующем цикле в переменной Previous будет храниться значение, полученное при предыдущем повторе цикла.**

* Пункты **Yes** и **No** в каталоге звуков Lego.
** Пункты **Yes** и **No** в каталоге звуков Lego.
*** Пункт **Yes** в каталоге звуков Lego.

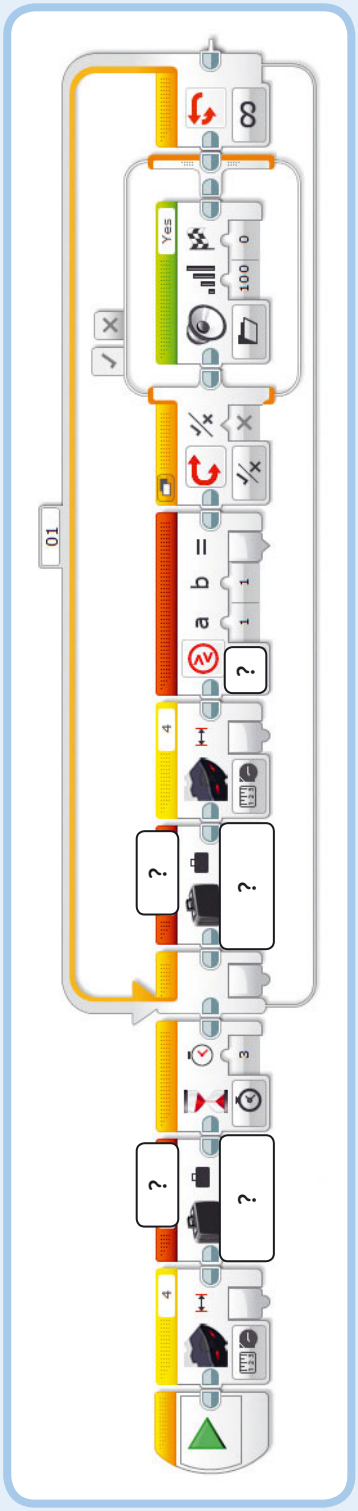


Рис. 16.7. Незаконченная версия программы для практикума № 106

инфракрасного датчика пятисекундной давности. Перед написанием программы определите переменную **Proximity**, если вы еще этого не сделали, как показано на рис. 16.3.

Программа VariableDemo иллюстрирует идею использования переменных, но это очень простая программа. Завершив ее, попрактикуйтесь с переменными в практикумах № 106 и № 107.

ПРИМЕЧАНИЕ **Не путайте блоки Константа (Constant) и Переменная (Variable). У обоих есть значок чемодана, но в блоке Константа (Constant) еще есть значок замка, напоминающий, что константу нельзя изменить во время работы программы.**

Изменение значений переменных

Иногда возникает необходимость изменить значение переменной, к примеру, чтобы с ее помощью отследить наибольшее значение и общее число нажатий кнопки датчика касания. Часто вам потребуется увеличивать значение переменной на 1, что называется *приращением*. Программа TouchCount, которую вы сейчас напишете, покажет, как с помощью переменной определить, сколько раз щелкали по кнопке датчика касания (нажимали кнопку и сразу же отпускали).

Начнем с определения новой числовой переменной с именем **Count**, в которой будет храниться число нажатий кнопки датчика. Программа будет ждать, пока кнопку датчика касания не нажмут, после чего значение переменной **Count** будет увеличиваться на 1. Чтобы программа продолжала считать, нам понадобится блок **Цикл** (Loop).

Но как увеличить значение переменной на 1? Как показано на рис. 16.8, для считывания значения переменной **Count** используется блок **Переменная** (Variable). Затем это значение передается в блок **Математика** (Math), где к нему будет добавлена единица. Полученный результат отправляется в другой блок **Переменная** (Variable), созданный, чтобы записывать (хранить) новое значение переменной **Count**, которое теперь увеличилось на 1. Блок **Цикл** (Loop) повторяет этот процесс в течение 5 секунд, после чего итоговое значение переменной **Count** отображается на экране модуля EV3. Цикл должен быть выполнен по крайней мере один раз, а значит, окончательное значение всегда будет 1 или больше.

С помощью этого метода можно изменять значение переменной как вам заблагорассудится. Данный пример показывает, как добавить к значению 1, но, применив этот же метод, можно и вычитать из значения.

Присвоение начального значения переменным

При программировании с использованием переменных важно присвоить им начальное значение. Вы делали это в программе TouchCount, установив значение переменной **Count** равным нулю в начале программы. Присвоение переменным начального значения повышает надежность программы, гарантируя, что каждый раз при запуске она будет работать одинаково, поскольку начинается с одного и того же момента.

Однако начальное значение необязательно должно равняться нулю. Например, если задать начальное значение

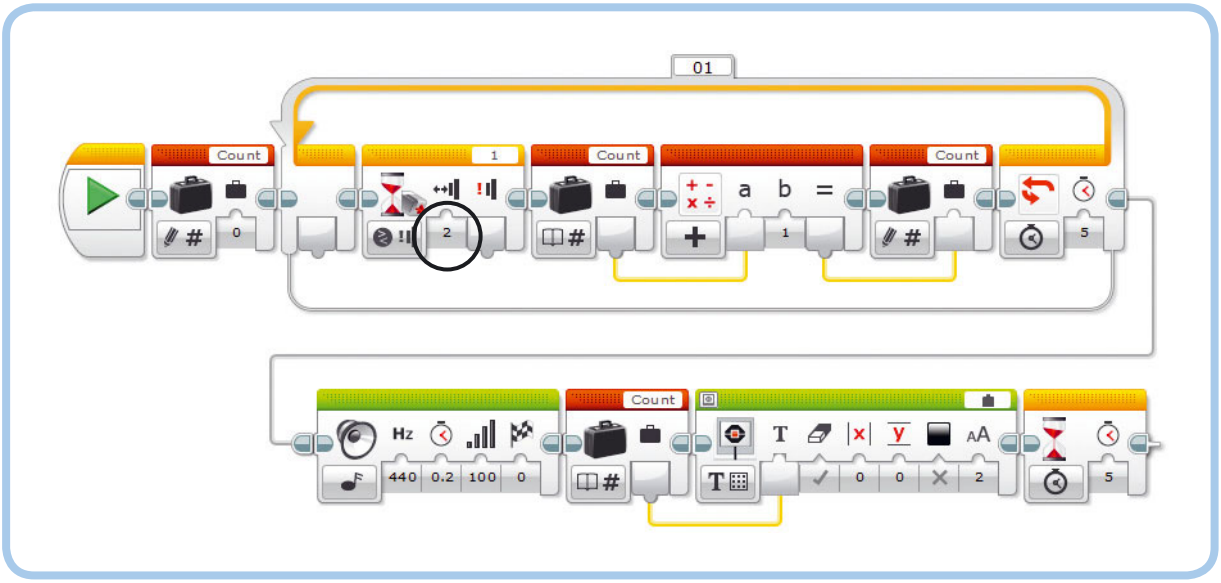


*Рис. 16.8. Программа TouchCount считывает количество нажатий на кнопку датчика касания в течение 5 секунд, а затем отображает итоговое число на экране. Обратите внимание, что для большей наглядности я подключил блоки **Звук** и **Цикл** друг к другу с помощью соединителя, но вы можете просто поместить блок **Звук** сразу после блока **Цикл***

переменной **Count** равным 5, программа будет начинать отсчет с 5.

Если вы не зададите в своей программе начальное значение числовой переменной, программное обеспечение EV3 будет использовать в качестве начального значения 0.

Тем не менее рекомендуется всегда задавать начальные значения в программах, даже если вы хотите, чтобы значения переменных начинались с 0.

Вычисление среднего значения

В следующем примере используем переменную для вычисления *среднего значения* 50 показаний датчика. Чтобы вычислить среднее показание (среднее значение), разделим сумму всех показаний на их количество (в данном случае 50). Программа Average, показанная на рис. 16.9, вычисляет эту сумму путем многократного добавления показаний датчика к переменной с именем **Sum**.

Начальное значение переменной равно нулю, а блоки в цикле каждый раз добавляют в переменную **Sum** последние показания датчика. После 50 повторов сумма делится на 50, чтобы вычислить среднее значение показаний датчика, которое, в свою очередь, отображается на экране.

Вычисление среднего значения может быть полезно для получения более точных показаний датчика. Например, у вас есть робот, огибающий препятствия, который поворачивает, если значение приближения становится меньше 50. В некоторых случаях датчик может показать, что препятствие находится ближе (на значении 40, скажем), хотя на самом деле там ничего нет. Обычно такие данные заставили бы робота повернуть, но если взять среднее значение трех показаний, то получится (100 + 100 + 40) / 3 = 80, и это ложное показание не заставит робота изменить курс.

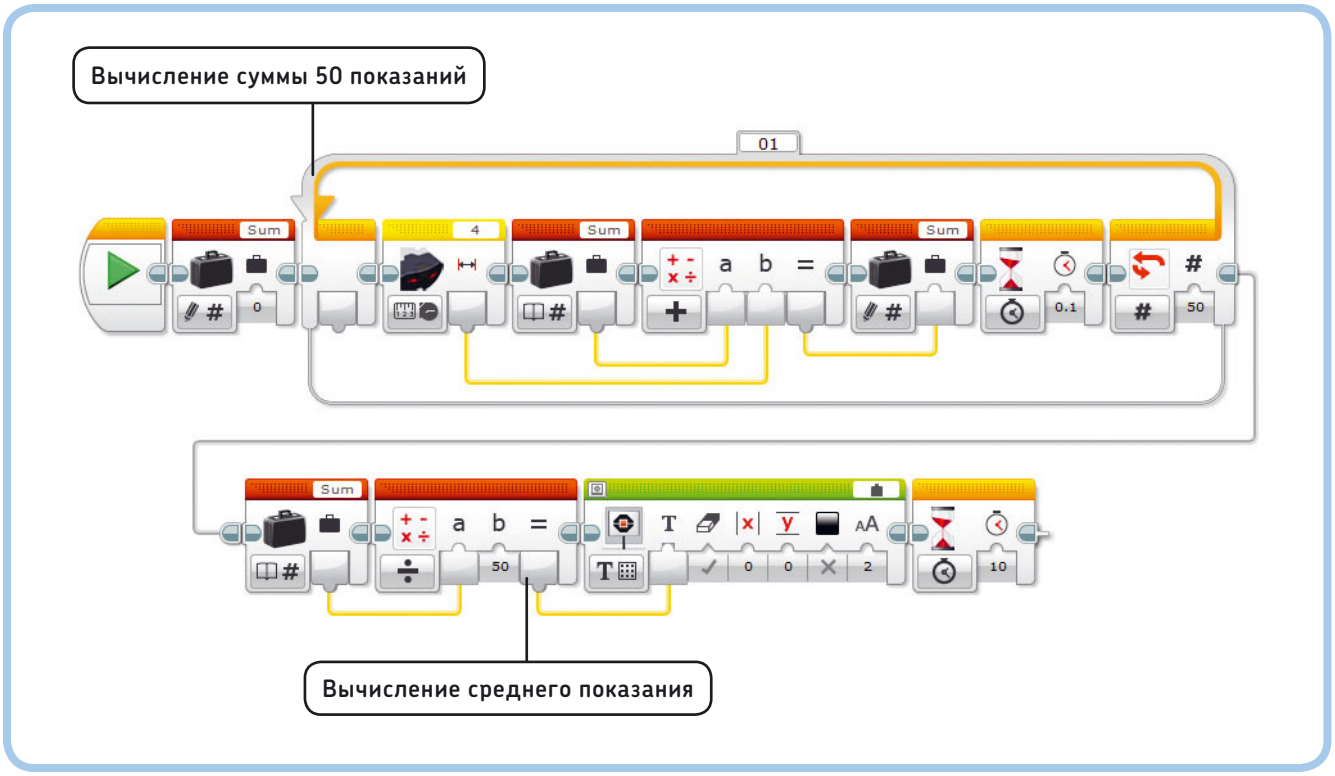


*Рис. 16.9. Программа Average. Блок **Ожидание** в блоке **Цикл** отвечает за паузу в 0,1 секунды между снятием показаний, чтобы программа могла вычислить среднее значение датчика за период 50×0,1 = 5 секунд*

Дальнейшее изучение

Поздравляю! Вы изучили всю теорию по программированию в этой книге и готовы перейти к следующей главе, в которой напишете большую программу, превращающую SK3TCHBOT в устройство наподобие детской игры «Волшебный экран». Но прежде изучите практикумы. Они могут оказаться сложнее, чем те, которые вы выполняли ранее, но помните, что в каждом случае существует несколько решений. Попробуйте их выполнить, а решения многих заданий можно найти по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**.

ПРАКТИКУМ № 108: ПРЯМОЙ И ОБРАТНЫЙ СЧЕТ!

Сложность:  **Время:** 

Можно ли на основе программы TouchCount создать другую, которая будет использовать переменную для отслеживания количества нажатий кнопок «Влево» и «Вправо» на корпусе модуля EV3? При нажатии кнопки «Вправо» значение переменной должно увеличиваться на 1, а при нажатии кнопки «Влево» — уменьшаться на 1. Выведите значение переменной на экран EV3.

СОВЕТ **Используйте блок Ожидание (Wait), отслеживающий нажатия кнопок «Влево» и «Вправо». Затем используйте блок Переключатель (Switch), чтобы определить, какая кнопка была нажата. Внутрь блока Переключатель (Switch) добавьте блоки, необходимые для изменения значения переменной, и блоки, ожидающие, когда кнопку EV3 отпустят.**

ПРАКТИКУМ № 109: ОГРАНИЧЕННОЕ СРЕДНЕЕ!

Сложность:  **Время:** 

Можно ли расширить программу Average, показанную на рис. 16.9, таким образом, чтобы она вычисляла среднее значение всех замеров приближения с момента запуска программы до тех пор, пока вы не нажмете кнопку датчика касания?

СОВЕТ **Настройте цикл так, чтобы он повторялся, пока не будет нажата кнопка датчика касания. Задайте переменную с именем Measurements, чтобы отслеживать количество измерений датчика, добавленных к переменной Sum. Когда цикл завершится, вычислите среднее значение, поделив сумму на количество измерений.**

ПРАКТИКУМ № 110: ПОДТВЕРЖДЕНИЕ СЛУЧАЙНОСТИ!

Сложность:  **Время:** 

В этом практикуме вам нужно проверить, насколько хорошо работает блок **Случайное значение** (Random) в режиме **Логическое значение** (Logic). Для этого робот должен сгенерировать 10 000 случайных логических значений и определить, сколько из них являются истинными, а сколько — ложными. Поэкспериментируйте со значением параметра **Вероятность значения «истина»** (Probability of True) и посмотрите, удастся ли вам добиться ожидаемого результата.

СОВЕТ **Создайте две переменные типа Числовое значение (Numeric) с именами TrueCount и FalseCount. Создайте случайное логическое значение с помощью блока Случайное значение (Random). Если оно окажется истинным, то увеличьте значение переменной TrueCount; если ложным — переменной FalseCount. Выведите на экран значения обеих переменных после 10 000 повторений. Какие значения вы ожидаете увидеть?**

ПРАКТИКУМ № 111: МАКСИМАЛЬНОЕ СБЛИЖЕНИЕ!

Сложность: **Время:**

Можно ли создать программу, которая будет определять наименьшее зафиксированное значение приближения из 50 измерений? Задайте в программе паузу на 0,1 секунды между измерениями и выведите на экран наименьшее значение.

СОВЕТ **Создайте переменную с именем Lowest, в которой будет храниться наименьшее зарегистрированное показание. Регулярно сравнивайте сохраненное измерение с новыми показаниями датчика. Если новое показание окажется меньше, сохраните его значение в переменной Lowest. Какое значение нужно указать в качестве начального для переменной Lowest в начале программы?**

СДЕЛАЙ САМ № 27: СЧЕТЧИК КЛИЕНТОВ!

Сборка: **Программирование:**

Можно ли создать робота, который будет подсчитывать количество находящихся в комнате людей? Спроектируйте устройство, которое будет открывать дверь, когда гости нажимают кнопку датчика касания, чтобы войти в комнату, и попросите их поднести руку к инфракрасному датчику, чтобы открыть дверь, когда они захотят выйти. Затем измените программу, которую вы написали на практикуме № 108, чтобы подсчитать число людей в комнате: при нажатии кнопки датчика касания количество на счетчике должно увеличиваться, а при поднесении руки к инфракрасному датчику — уменьшаться.

СОВЕТ **Вместо того чтобы создавать робота, который будет поворачивать дверную ручку, гораздо проще спроектировать перемещающегося робота, который будет открывать дверь, толкая ее вперед, а закрывать — потянув назад. Убедитесь, что дверь закрывается не до конца. Как вариант, с помощью скотча или шнура закрепите дверную ручку, чтобы язычок замка не выступал и дверь не закрывалась.**

17

Играем в игры на EV3

Теперь напишем программу, для создания которой потребуется применить многие из изученных ранее техник программирования. Выше я познакомил вас со всеми техниками программирования и блоками на коротких примерах, а теперь вы узнаете, как объединить их в более сложную программу.

В этой главе вы создадите программу, которая позволит вам запускать на модуле EV3 игру наподобие «Волшебного экрана». С помощью стрелок и больших моторов вы будете рисовать на экране модуля EV3, а датчики касания и цвета будут использоваться для дополнительного ввода данных (рис. 17.1).

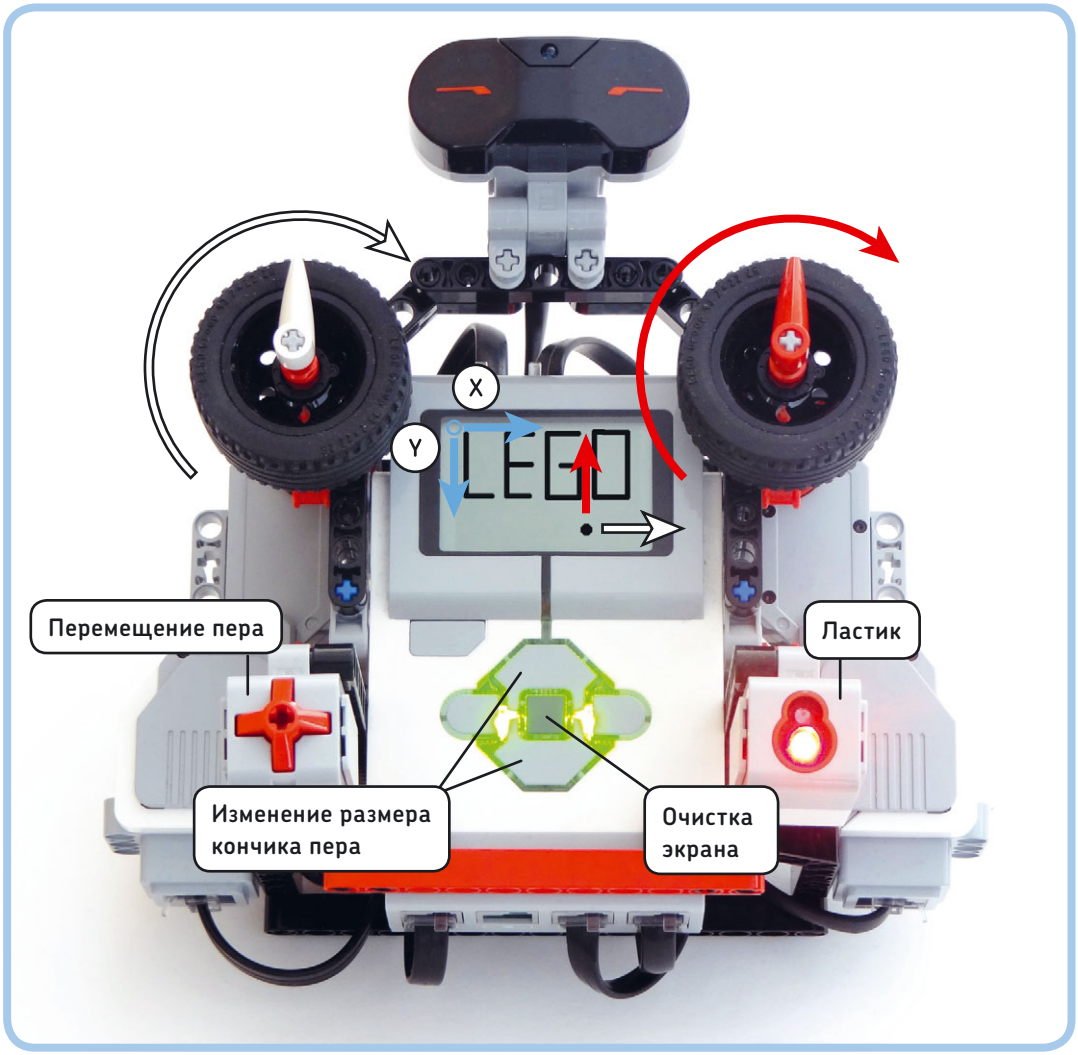


Рис. 17.1. Играем на SK3TCHBOT в «Волшебный экран»

Шаг первый: создание основной программы

В игре «Волшебный экран» можно рисовать на экране устройства линии, поворачивая стрелки, как будто вы рисуете пером на листе бумаги. Левая стрелка управляет положением пера по горизонтали (ось *x*), а правая — по вертикали (ось *y*). Такое поведение на удивление просто воссоздать в программе EV3, как показано на рис. 17.2.

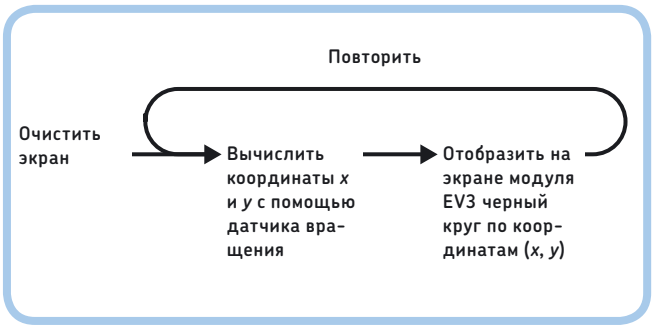


Рис. 17.2. Схема простой программы волшебного экрана

Программа постоянно отображает на экране модуля EV3 новую точку (кружок) в позиции, соответствующей координатам *x* и *y*. Программа определяет эти координаты с помощью датчиков вращения в больших моторах. Белая стрелка (мотор B) контролирует положение по оси *x*, а красная (мотор C) — по оси *y*. По мере того как программа добавляет точки на экране, можно создавать рисунки, «перемещая перо» с помощью стрелок на корпусе робота SK3TCHBOT.

Для начала создайте новый проект под названием SK3TCHBOT-Games, а в нем — программу Etch-A-Sketch. Чтобы отобразить точку, используйте блок **Экран** (Display) в режиме **Фигуры** ▸ **Круг** (Shapes ▸ Circle), но вам нужно будет создать два контейнера «Мой блок», позволяющих очищать экран и рассчитывать координаты, где должна отображаться точка.

Контейнер «Мой блок» № 1: Очистка экрана

У блока **Экран** (Display) нет функции очистки экрана без вывода на него нового изображения, но ее можно сымитировать, отобразив один белый пиксел и очистив остальную часть экрана. Создайте блок **Экран** (Display), как показано на рис. 17.3, и превратите его в контейнер «Мой блок» с именем **Clear**.

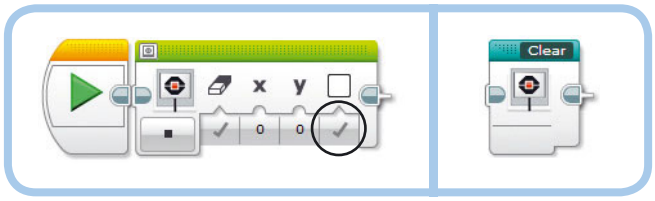


*Рис. 17.3. Контейнер **Clear** очищает экран EV3. Готовый контейнер «Мой блок» показан справа*

Контейнер «Мой блок» № 2: Определение координат

Теперь создайте контейнер «Мой блок» с именем **Coordinates**, как показано на рис. 17.4. У него есть два числовых вывода (**X** и **Y**), которые вы будете использовать для управления положением каждой новой точки. Давайте посмотрим, как этот блок вычисляет координату *x*.

Напомним, что датчик вращения сообщает, на сколько градусов повернулся мотор с момента запуска программы. В принципе, можно использовать значение датчика от мотора B, чтобы управлять непосредственно координатой *x*. Но тогда наше перо будет двигаться слишком быстро: поворот белой стрелки всего на 180 градусов заставит перо переместиться через весь экран. (Ширина экрана по горизонтали составляет 178 пикселов.)

Для уточнения движения разделите количество градусов на 3, чтобы при повороте белой стрелки на 180 градусов линия появлялась только поперек одной трети экрана. Добавьте к получившемуся частному число 89 (половина от 178), чтобы при значении, равном нулю градусов, при запуске программы точка появлялась в центре экрана.

Процедура определения координат *y* аналогична, за исключением того, что вы будете использовать мотор C и добавите к частному число 64 (по вертикали экран содержит 128 пикселов). Так как мотор C перевернут, то, поворачивая его в направлении, указанном красной стрелкой на рис 17.1, вы получите отрицательное значение градусов. Следовательно, перо будет двигаться вверх (вертикальная красная стрелка), а это противоположно положительному направлению координаты *y* (вертикальная синяя стрелка).

Чтобы создать контейнер «Мой блок», поместите в рабочую область два блока **Вращение мотора** (Motor Rotation) и два блока **Математика** (Math) и превратите их в контейнер «Мой блок» с двумя числовыми выводами и именем **Coordinates**, как показано на рис. 17.4.

Завершение основной программы

Теперь, когда вы создали основные компоненты (см. схему на рис. 17.2), завершите программу, как показано на рис. 17.5. Кроме двух контейнеров «Мой блок» программа содержит

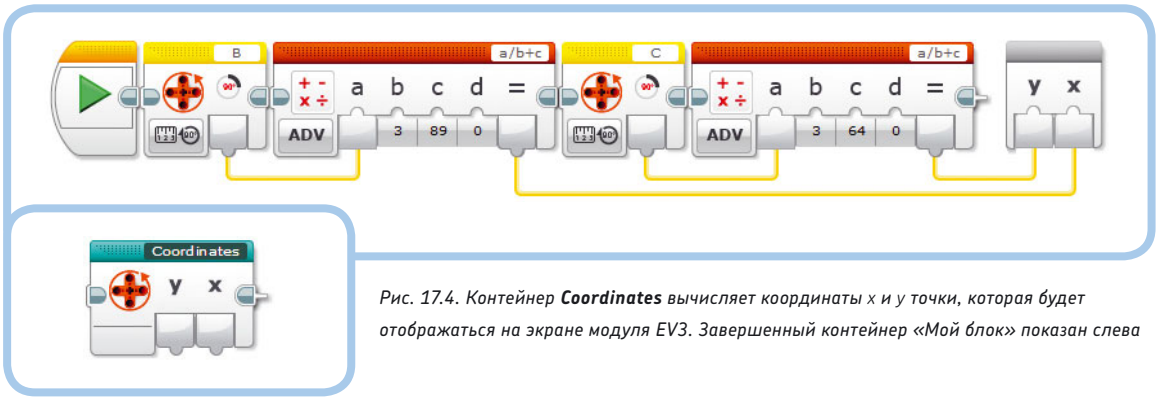


*Рис. 17.4. Контейнер **Coordinates** вычисляет координаты x и y точки, которая будет отображаться на экране модуля EV3. Завершенный контейнер «Мой блок» показан слева*

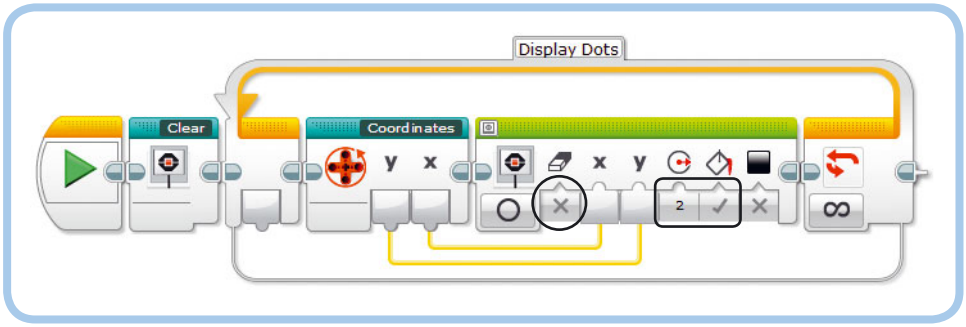


*Рис. 17.5. Основная программа Etch-A-Sketch. Обратите внимание, что блок **Экран** запрограммирован не для очистки экрана; он гарантирует, что ранее напечатанные точки останутся видимыми, когда на экран добавится еще одна точка*

блок **Экран** (Display), запрограммированный на отображение точки (кружочка) по заданным координатам, и блок **Цикл** (Loop) для повтора программы. Загрузите программу в робота и протестируйте ее перед тем, как развивать дальше в следующем разделе.

ПРИМЕЧАНИЕ Если у вас возникли трудности при создании этой программы, загрузите готовую программу по ссылке eksmo.ru/files/Lego_Mindstorms_Primers.zip и сравните ее с вашей.

Шаг второй: добавление элементов управления пером

Основная программа «Волшебного экрана» интересна, но можно придать ей еще больше функциональных возможностей, немного доработав, как показано на рис. 17.6.

Используйте датчик касания, чтобы на время прекращать процесс рисования, датчик цвета, чтобы превратить перо в ластик, центральную кнопку модуля EV3 для очистки экрана, а кнопки «Вверх» и «Вниз» для изменения размера пера.

Чтобы упростить поиск потенциальных проблем, обязательно тестируйте программу каждый раз после добавления новых функций. Когда закончите, в новых практикумах вас ждут задания по дальнейшему дополнению программы.

Перемещение пера без рисования

В процессе рисования иногда требуется «оторвать перо от бумаги», чтобы перейти в другую позицию, не чертя за собой линию. Для этого укажите в программе, что рисовать можно, только если кнопка датчика не нажата (рис. 17.7). В противном случае ничего не произойдет. (На вкладке **Ложь** (False) блока **Переключатель** (Switch) нет блоков.)

Превращение пера в ластик

Теперь добавьте в программу ластик, который будет рисовать белые точки при активации датчика цвета, как показано на рис. 17.8. Рисуя белые точки, можно эффективно стирать фрагменты изображения, что очень удобно, если вы вдруг сделаете ошибку.

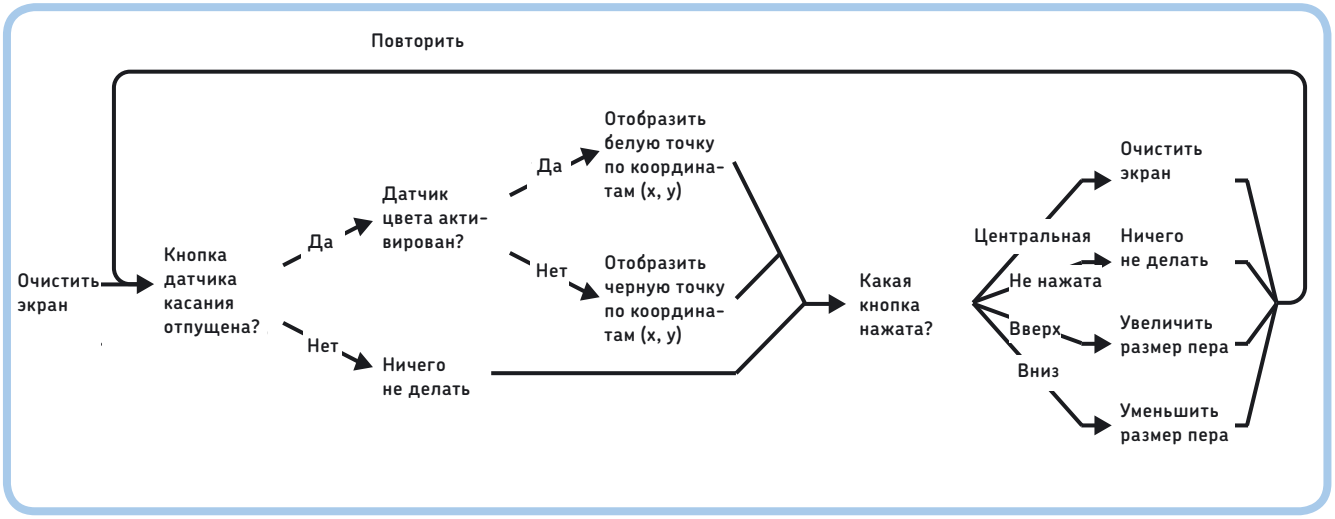


Рис. 17.6. Схема расширенной версии программы Etch-A-Sketch

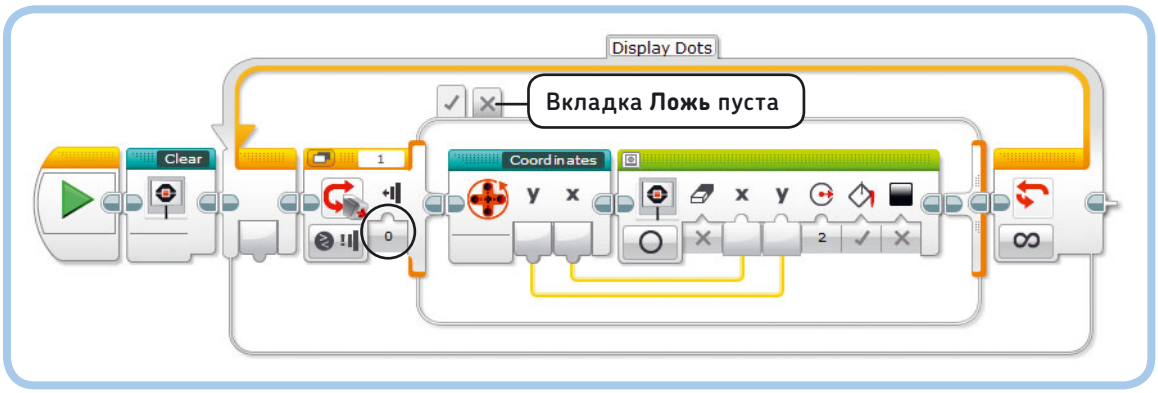


Рис. 17.7. Программа рисует новые точки только в том случае, если кнопка датчика касания не нажата. Чтобы переместить перо в другую позицию, не чертя при этом линию, нажмите и удерживайте кнопку датчика касания, поверните стрелки и отпустите кнопку датчика касания, чтобы продолжить рисовать

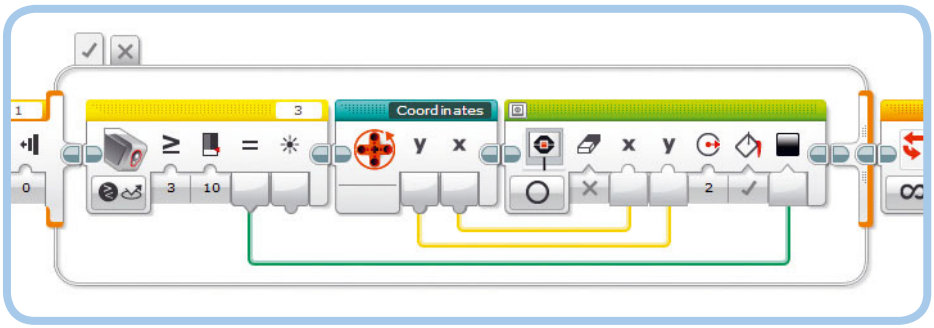


*Рис. 17.8. Добавьте блок **Датчик цвета**, настроенный, как показано на рисунке, и подключите логическую шину данных к блоку **Экран**. Теперь, когда вы поднесете палец к датчику цвета, перо будет работать как ластик*

Если поднести к датчику цвета палец, показатель яркости отраженного света вырастет до значений выше 10%, а параметр **Результат сравнения** (Compare Result) блока **Датчик цвета** (Color Sensor) примет значение **Истина** (True), в результате чего параметр **Цвет** (Color) блока **Экран** (Display) примет значение **Белый** (White).

Когда вы уберете палец, вывод передаст значение **Ложь** (False), и блок **Экран** (Display) продолжит добавлять на экран черные кружки.

Очистка экрана

Для очистки экрана нажатием центральной кнопки добавьте блоки **Переключатель** (Switch) и **Clear** (Очистить) (рис. 17.9). Если ни одна кнопка на корпусе модуля EV3 не нажата, то ничего не произойдет. Вариант **Ничего** (No Buttons) задан по умолчанию, и в нем не используется ни одного блока.

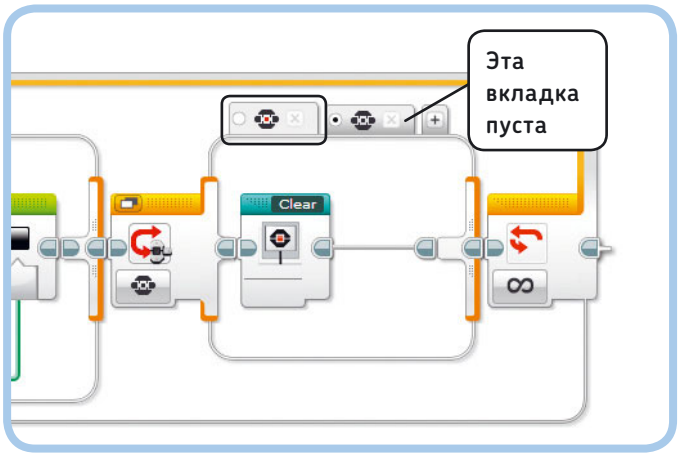


Рис. 17.9. Эти блоки очищают экран, когда вы нажимаете центральную кнопку

Установка размера пера

И наконец, дополним программу, чтобы можно было задавать размер пера с помощью кнопок «Вверх» и «Вниз» на корпусе модуля EV3. Для этого создадим переменную с именем **Size**, которая будет управлять значением параметра **Радиус** (Radius) блока **Экран** (Display). Затем добавим блоки, которые позволят менять значение переменной **Size** с помощью кнопок на корпусе модуля EV3.

Для начала определите переменную типа **Числовое значение** (Numeric) с именем **Size** и добавьте в текущую программу два блока **Переменная** (Variable), как показано на рис. 17.10. В первом блоке **Переменная** (Variable) задается начальное значение 1.

Второй блок передает значение размера на ввод **Радиус** (Radius) блока **Экран** (Display). Следовательно, радиус будет равен 1, пока вы не поменяете значение переменной.

Затем добавьте две ситуации (вкладки) к блоку **Переключатель** (Switch), который вы настроили ранее. Добавьте одну вкладку для кнопки «Вверх», а вторую — для кнопки «Вниз».

При нажатии кнопки «Вверх» переменная **Size** должна увеличиваться на 1, как показано на рис. 17.11. Согласно величине экрана, нужно ограничить значение размера числом 25. Другими словами, вы будете увеличивать переменную, только если ее текущее значение меньше 25, как это определено в блоке **Сравнение** (Compare). Если значение переменной **Size** достигнет 25, а вы попытаетесь его еще увеличить, блок **Сравнение** (Compare) выдаст значение **Ложь** (False), вы услышите звуковой сигнал, и значение не изменится.

Блок **Ожидание** (Wait) прекратит выполнение программы до тех пор, пока вы не отпустите кнопку «Вверх». Если не установить режим ожидания до тех пор, пока кнопка не будет отпущена, программа продолжит повторять действие, увеличивая значение переменной с каждым новым циклом.

Значение переменной должно уменьшаться (на 1) при нажатии кнопки «Вниз», но только если текущее значение переменной **Size** больше 1, чтобы в итоге у вас не получился круг с отрицательным или равным нулю значением радиуса (который не будет работать). Если значение переменной **Size** равно 1, а вы попытаетесь еще его уменьшить, блоки **Сравнение** (Compare) выведут значение **Ложь** (False), вы услышите звуковой сигнал, и значение не изменится (рис. 17.12).

Поздравляем! Вы завершили программу Etch-A-Sketch. После проверки работоспособности программы попробуйте еще дополнить ее, выполняя задания из практикумов № 112–114.

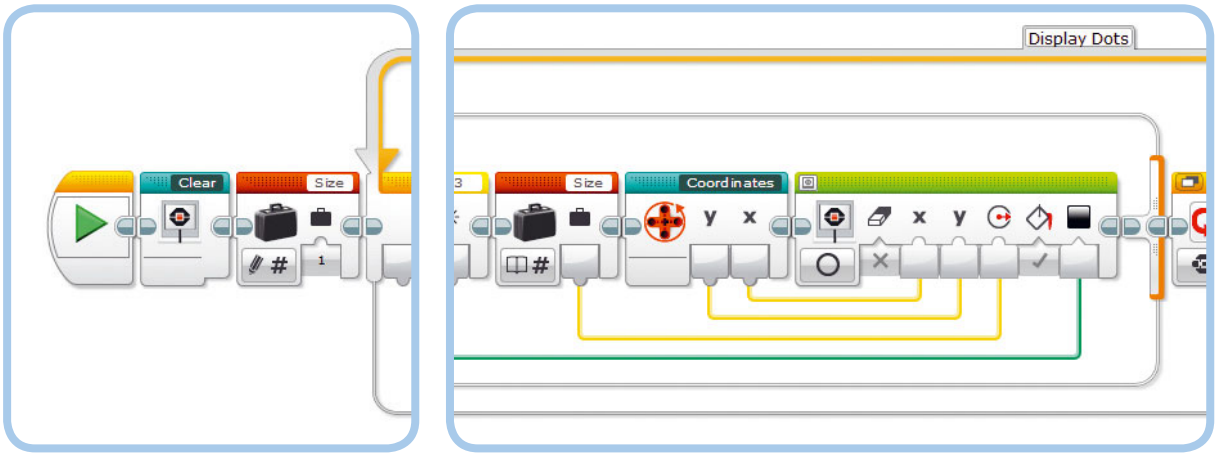


*Рис. 17.10. Добавьте в программу два блока **Переменная**, как показано на рисунке*

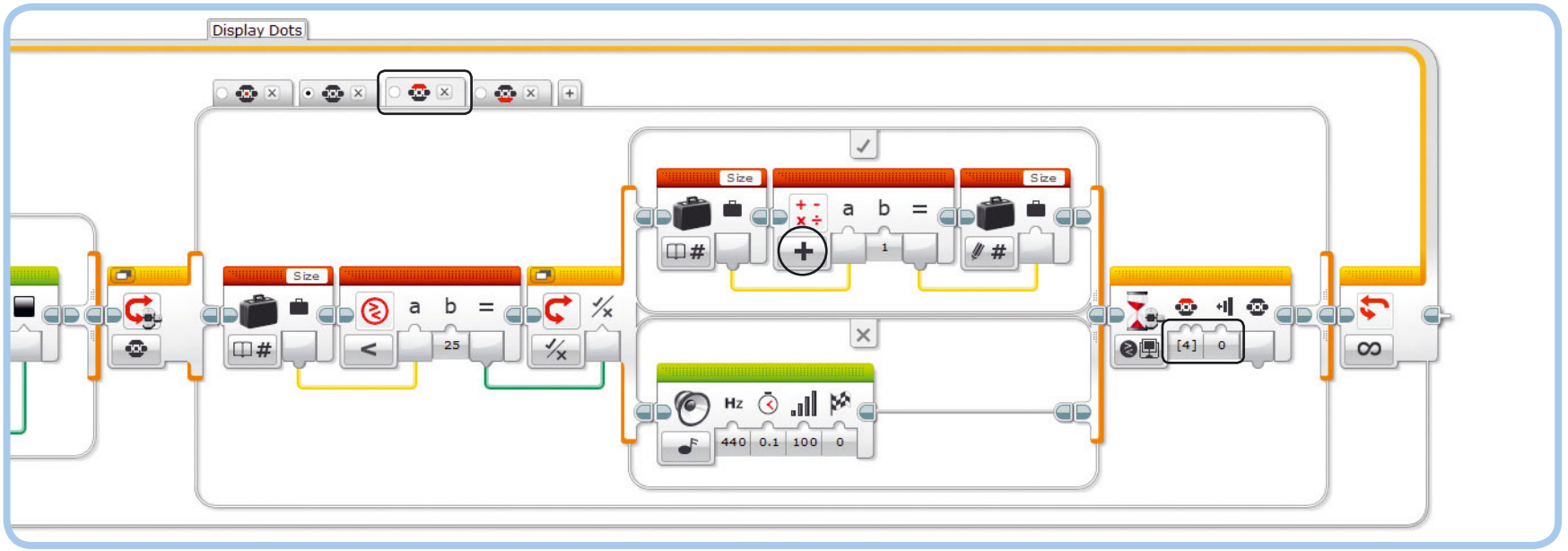


*Рис. 17.11. Если текущее значение переменной **Size** меньше 25, то при нажатии кнопки «Вверх» оно увеличится на 1. Обратите внимание, что вторая вкладка (**Ничего**) остается выбранной по умолчанию*

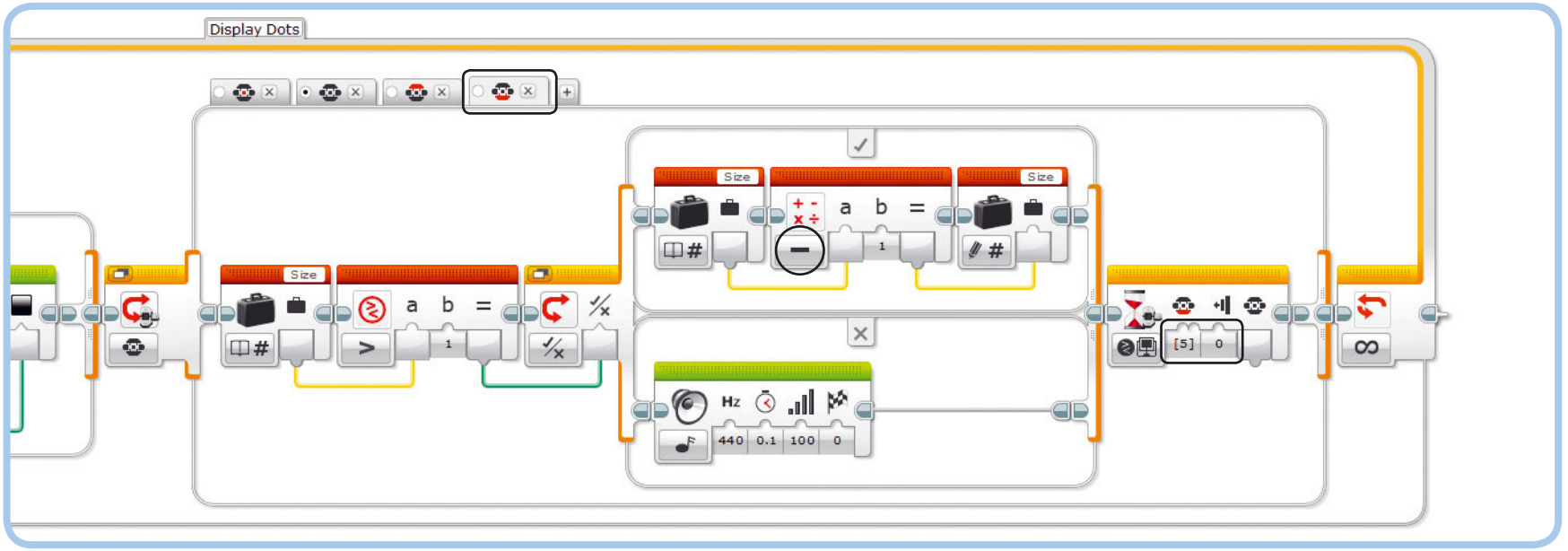


*Рис. 17.12. Если текущее значение переменной **Size** больше 1, то при нажатии кнопки «Вниз» оно уменьшится на 1*

Дальнейшее изучение

В этой главе вы разработали программу, в которой объединены многие изученные ранее техники программирования. Если вам понравилось, рекомендую изучить практикумы № 115 и 116. Далее вы узнаете, как сконструировать и запрограммировать еще двух классных роботов.

ПРАКТИКУМ № 112: РОБОТ-ХУДОЖНИК!

Сложность: **Время:**

Нарисовать красивый рисунок с помощью программы Etch-A-Sketch непросто, но можно применить маленькую хитрость — заставить программу рисовать идеально прямые линии. Для этого нужно с помощью программных блоков управлять движением моторов B и C. Можно ли запрограммировать робота, чтобы он нарисовал на экране домик? Как нарисовать диагональные линии?

СОВЕТ **Управляйте движением моторов в программе, работающей параллельно с Etch-A-Sketch (см. раздел «Многозадачность» в главе 5).**

ПРАКТИКУМ № 113: РЕАКЦИЯ НА СИЛУ!

Сложность: **Время:**

Можно ли расширить контейнер **Coordinates** так, чтобы пользователь не мог вывести перо за пределы экрана? Перо должно автоматически возвращаться на экран, если пользователь повернет стрелки слишком сильно.

СОВЕТ **Укажите, что скорость мотора B в активном режиме должна составлять –5% при значениях координаты *x* больше 178 и 5%, если значение *x* оказывается меньше 0. Если значение координаты *x* находится в диапазоне от 0 до 178, запрограммируйте остановку мотора и присвоение значения Ложь (False) параметру Тормозить в конце (Brake at End). Такое же поведение задайте и в отношении координаты *y*.**

ПРАКТИКУМ № 114: ПЕРО-УКАЗКА!

Сложность: **Время:**

Диаметр самых маленьких точек, которые программа Etch-A-Sketch может отобразить на экране, 2 пиксела (радиус 1 пиксел). Можно ли дополнить программу так, чтобы в качестве самых маленьких точек на экране использовались отдельные пикселы? Это позволит прорисовать мельчайшие детали.

СОВЕТ **На экране можно нарисовать отдельную точку, если отобразить один пиксел по координатам *x* и *y*. Это можно сделать с помощью блока Экран (Display) в режиме Фигуры ▸ Точка (Shapes ▸ Point).**

ПРАКТИКУМ № 115: АРКАДНАЯ ИГРА!

Сложность: **Время:**

На этом практикуме вы узнаете, как написать программу аркадной игры для SK3TCHBOT. Для начала загрузите программу по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**, запустите ее и посмотрите, что произойдет. На экране модуля EV3 должна появиться цель (местоположение ее случайно) и игрок, которым управляет пользователь (рис. 17.13).

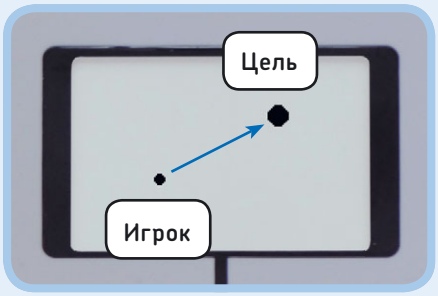


Рис. 17.13. Аркадная игра

Для попадания в цель нужно управлять игроком с помощью стрелок на корпусе робота SK3TCHBOT. Если вам удастся поразить цель в течение 4 секунд, вы заработаете 1 очко. Если вам потребуется более 4 секунд, вы потеряете 1 очко. После каждого попадания в другой позиции экрана появляется новая цель. После 10 ходов программа сообщит вам результат.

Изучите программу, чтобы понять, как она работает, а схема, приведенная на рис. 17.14, вам в этом поможет. Если вы любите трудности, попробуйте написать программу самостоятельно!

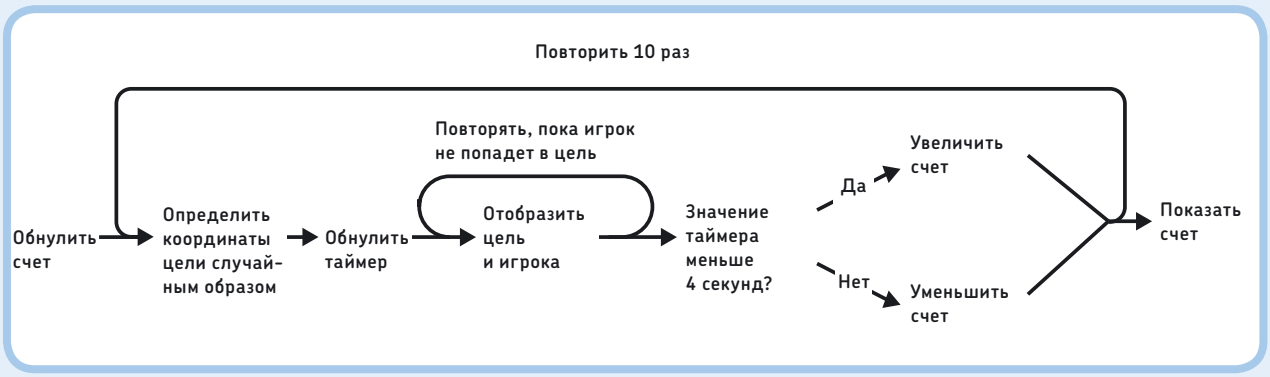


Рис. 17.14. Схема аркадной игры из практикума № 115

ПРАКТИКУМ № 116: ЗАРЯДКА ДЛЯ УМА!

Сложность: **Время:**

В этом практикуме вы освоите программу для тренировки ума, которую можно запустить на экране модуля EV3. Вы можете скачать ее по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**. Программа отображает на экране модуля EV3 в случайном порядке примеры на сложение, вычитание, умножение или деление (например, 7 × 3).

Пользователь решает, верен ли отображаемый ответ, и нажимает кнопку «Влево» (неверно) или кнопку «Вправо» (верно), как показано на рис. 17.15. Правильный ответ высвечивается примерно в половине случаев.

Программа создана с использованием только тех программных блоков, которые обсуждались ранее в этой книге, но ее все равно трудно понять поначалу.

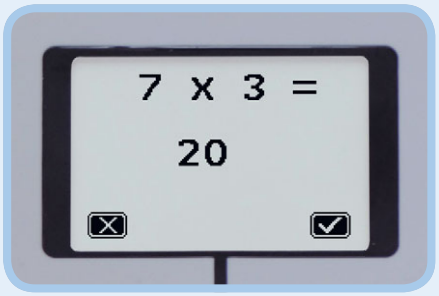


Рис. 17.15. Игра для тренировки ума. Нажав кнопку «Влево» в этом примере, вы заработаете 1 очко, так как данный ответ — неверный

Обратите внимание на комментарии, они помогут изучить программу.

Как только вы разберетесь, как она работает, дополните программу, чтобы отслеживать, сколько верных и неверных ответов вы даете в течение 30 секунд. Сможете улучшить результат, немного потренировавшись?

СДЕЛАЙ САМ № 28: ПЛОТТЕР!

Сборка: **Программирование:**

Можно ли сконструировать робота, способного делать чертежи на листе формата А4? С помощью одного мотора перо будет перемещаться по всей ширине листа для рисования горизонтальных линий (*x*), а с помощью другого мотора — перемещать бумагу, чтобы чертить вертикальные линии (*y*). Наконец, используйте блок **Средний мотор** (Medium Motor), чтобы поднимать и опускать перо. Это позволит перемещать перо и продолжать рисование в другом месте. Чтобы упростить управление устройством, создайте контейнеры «Мой блок» для перемещения пера с вводами, позволяющими указать координаты *x* и *y*.

СОВЕТ Для вдохновения изучите механизм работы обычного струйного принтера. Принтер перемещает картридж влево и вправо по неподвижной направляющей (так печатаются горизонтальные линии) и передвигает бумагу вперед-назад с помощью колесиков (так печатаются вертикальные линии).

ЧАСТЬ VI

Роботы-машины и гуманоиды

18

SNATCH3R: автономный робот-манипулятор

Из предыдущих глав вы узнали много нового о том, как программировать роботов EV3. Теперь можно приступить к созданию и программированию более сложных роботов. В этой главе вы построите и запрограммируете SNATCH3R, робота-манипулятора, способного находить и поднимать предметы. Он показан на рис. 18.1.

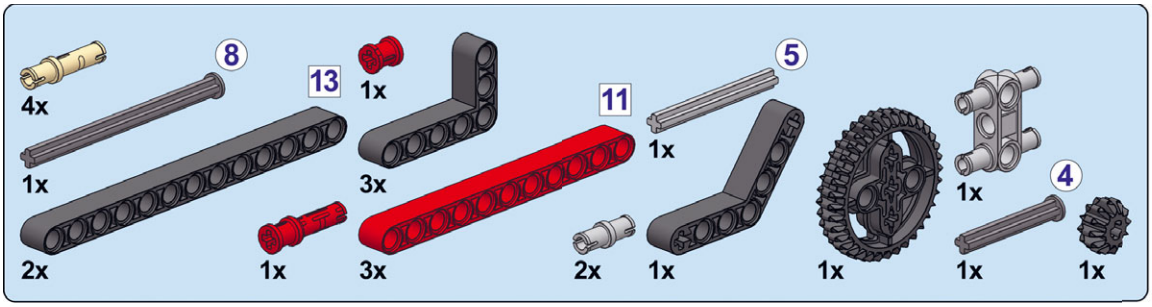
Сначала вы создадите программу, позволяющую управлять роботом удаленно, — так вы сможете протестировать его механические возможности. Затем вы запрограммируете робота таким образом, чтобы он находил и поднимал инфракрасный маяк. С помощью шин данных и переменных можно создать такой алгоритм, чтобы робот исследовал окружающую обстановку и был в состоянии обнаружить маяк на расстоянии вплоть до 2 метров, даже если тот находится позади робота.

Практикумы продемонстрируют вам, как работают эти программы, и вы сможете дополнить их новыми функциями. Например, перед вами будет стоять задача запрограммировать робота так, чтобы он, ориентируясь на датчик цвета, двигался по определенным линиям и собирал предметы вдоль маршрута.

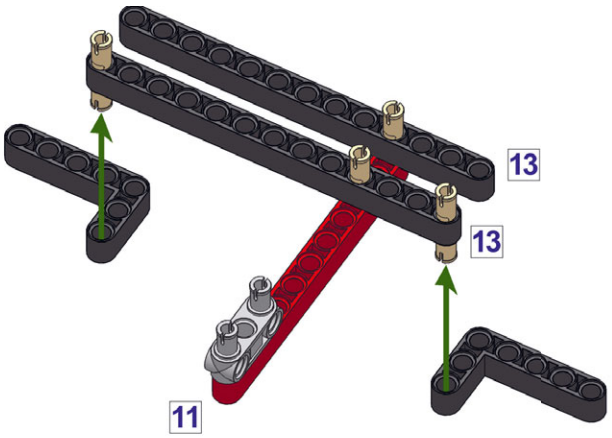
Рис. 18.1. Робот SNATCH3R может захватывать и поднимать легкие предметы, например пустую пластиковую бутылку

Захватное устройство

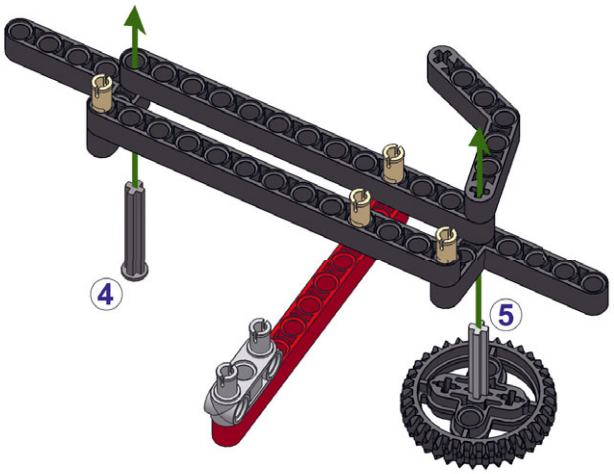
В роботе SNATCH3R вращением колес управляют два больших мотора, но самая потрясающая особенность этого робота — его многофункциональное захватное устройство. Как правило, для захвата и подъема предметов требуется два мотора: один для захвата, а второй, соответственно, для подъема. Роботу SNATCH3R для выполнения обеих задач достаточно одного среднего мотора, что обеспечивается особенностями используемых для его создания балок, осей и зубчатых колес LEGO. Чтобы лучше понять, как этот робот функционирует, вы с помощью инструкций на следующей странице построите упрощенную версию захватного устройства.


8
4x
1x
2x
13
1x
1x
1x
3x
3x
11
2x
5
1x
1x
1x
1x
4
1x
1x

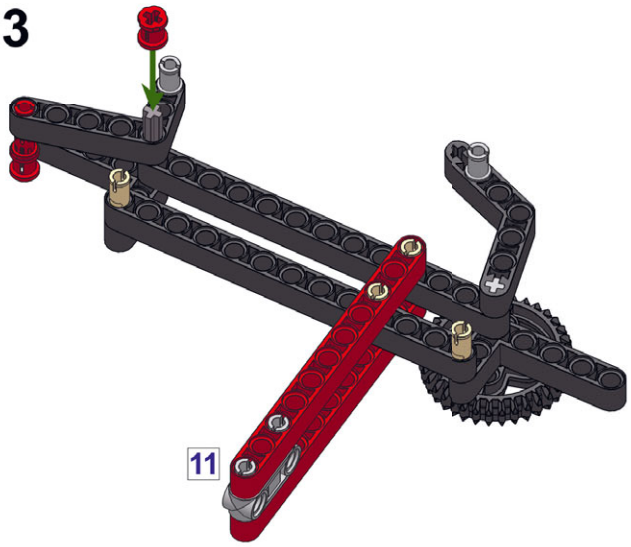
1


13
13
11

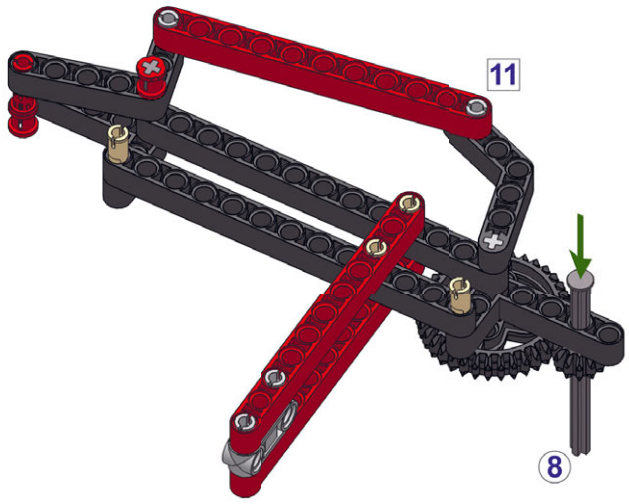
2


4
5

3


11

4


11
8

Механизм захватного устройства

Чтобы посмотреть, как действует захватное устройство, одной рукой держите красные балки механизма-образца, а второй поверните зубчатое колесо с 12 зубьями (12T), прокрутив его ось (рис. 18.2). Старайтесь не толкать ось ни вниз, ни вверх, механизм должен делать это самостоятельно.

Когда вы повернете маленькое зубчатое колесо, колесо с 36 зубьями (36T) станет тянуть красную балку вверх, и «клешня» закроется. Примерно по такому принципу функционирует и захватное устройство робота SNATCH3R. Средний мотор приводит в движение червячное колесо и зубчатое колесо с 24 зубьями (24T), в результате запускается цепная реакция, которая в итоге приводит к тому, что захватное устройство сжимает предмет, расположенный между клешнями. Когда мотор крутится в противоположную сторону, происходит обратное, и захватное устройство раскрывается.

Механизм подъема

Когда захватное устройство в закрытом положении, балки и зубчатые колеса, участвовавшие в процессе его закрытия, больше не могут двигаться. В упрощенном варианте механизма (рис. 18.3, сверху) это приводит к тому, что колесо 36T (*a*) практически не перемещается относительно голубой балки (*b*). В результате можно крутить голубую балку относительно зеленых балок (*c* и *d*) путем проворачивания колеса 36T, при этом захватное устройство будет подниматься.

На рис. 18.3 (снизу) видно, что в роботе SNATCH3R колесо 24T (*a*) точно так же блокируется балками захватного механизма (*b*), и, если прокручивать вперед средний мотор, проворачивая зубчатое колесо относительно моторного отсека (*c*), захватное устройство (*d*) поднимается. Когда мотор прокручивается в обратную сторону, захватное устройство опускается.

Обратите внимание, что две зеленые балки в механизме-примере (*c* и *d*) всегда располагаются горизонтально, независимо от положения манипулятора. Моторный отсек (*c*) и захватное устройство (*d*) робота SNATCH3R тоже всегда находятся в горизонтальном положении.

Датчик касания в основании робота SNATCH3R определяет, поднято ли захватное устройство до верхнего предела. Поэтому для захвата и подъема предметов необходимо прокручивать средний мотор вперед, пока не будет прижат датчик касания. Если манипулятор находится в самом низу, а его клешни раскрыты, мотор необходимо будет повернуть на 14,2 оборота, прежде чем он достигнет датчика касания. Соответственно, чтобы опустить и поставить предмет обратно, нужно прокрутить мотор назад на 14,2 оборота.

Но откуда механизм знает, что нужно закрыть захватное устройство, прежде чем пониматься, и наоборот, что нужно опустить захватное устройство, прежде чем открывать его? Разумеется, механизм ничего не «знает». Просто он спроектирован так, чтобы эти действия происходили естественным образом, под влиянием гравитации. На закрытие клешней затрачивается меньше энергии, чем на подъем захватного устройства, поэтому, когда мотор прокручивается вперед, сначала всегда закрываются клешни. Чтобы опустить захватный механизм, нужно меньше энергии, чем для открытия клешни,

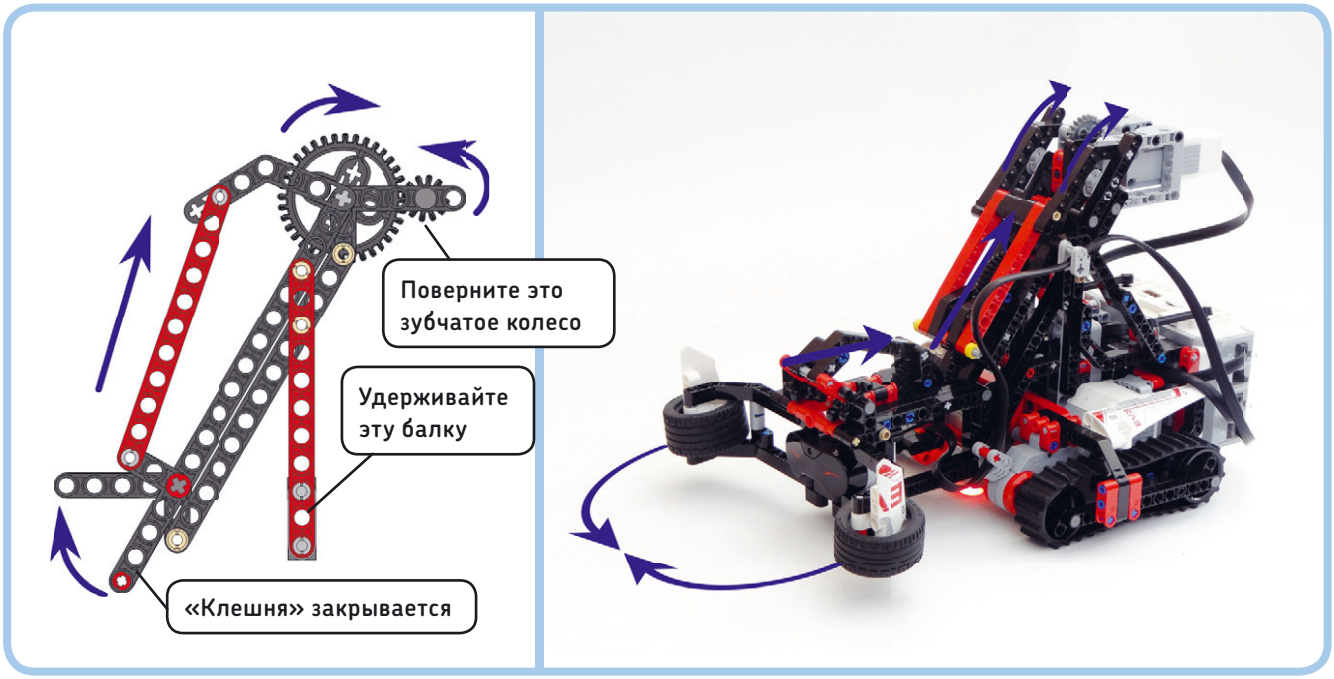


Рис. 18.2. Поверните ось с маленьким зубчатым колесом, чтобы «клешня» механизма-образца закрылась (слева). По аналогии, вращение среднего мотора заставляет робота SNATCH3R захватывать клешнями предметы (справа)

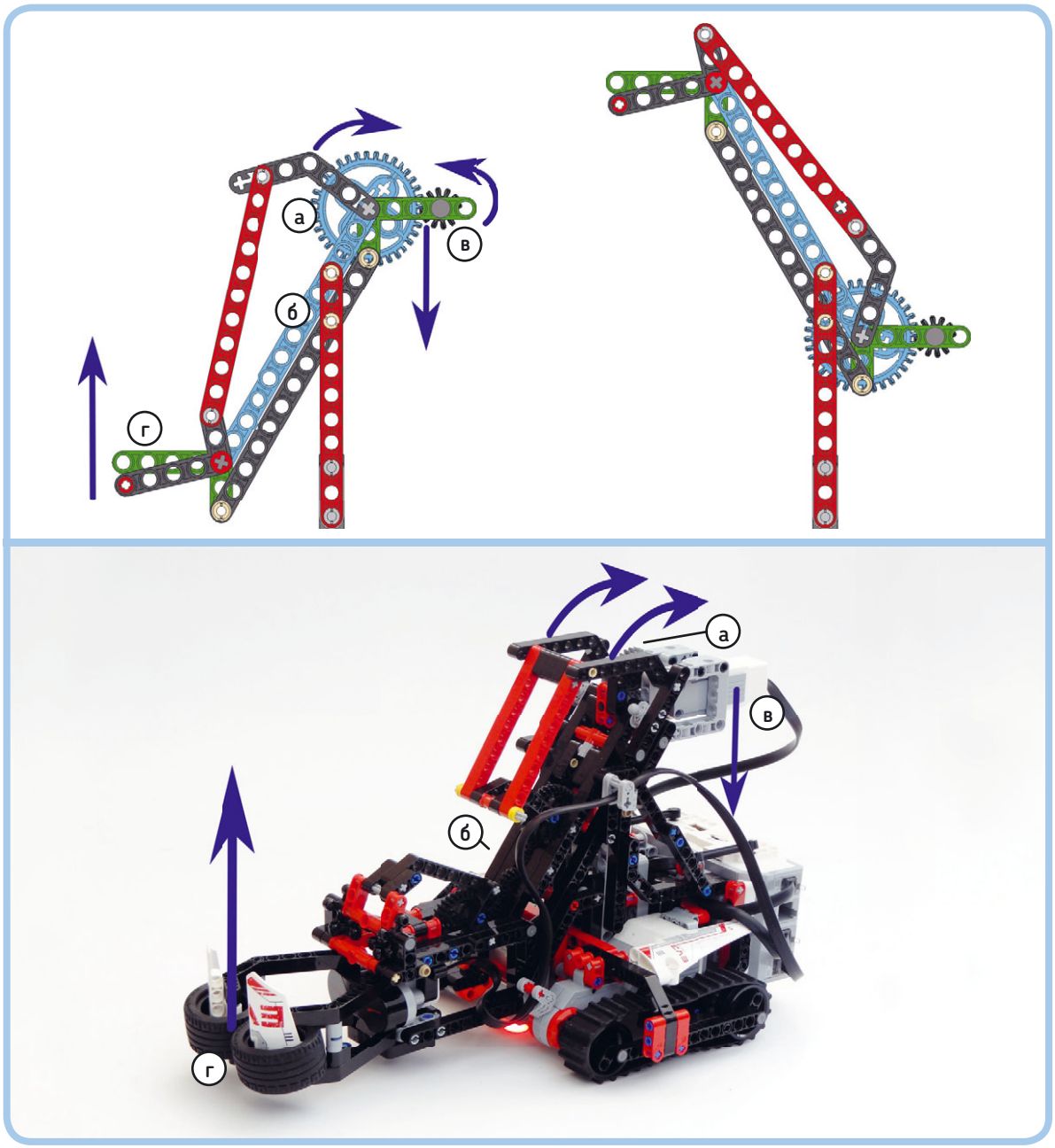


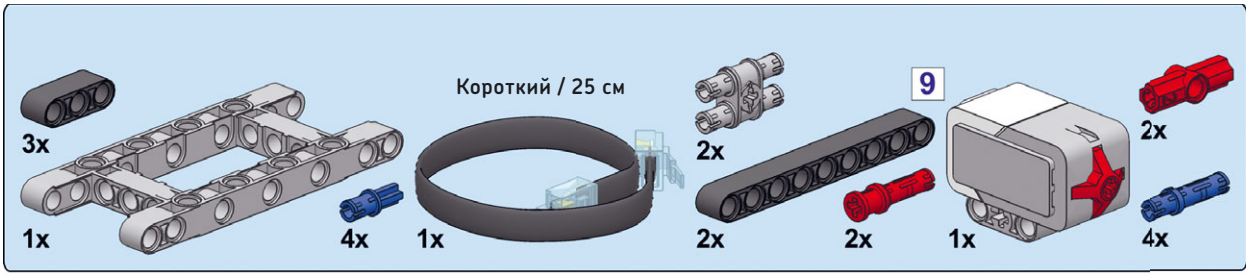
Рис. 18.3. Если вы будете дальше поворачивать колесо 12T, колесо 36T и голубая балка будут прокручиваться относительно зеленой балки, и захватное устройство будет подниматься (сверху). То же самое произойдет, если поворачивать средний мотор дальше вперед: захватное устройство робота SNATCH3R будет подниматься (снизу)

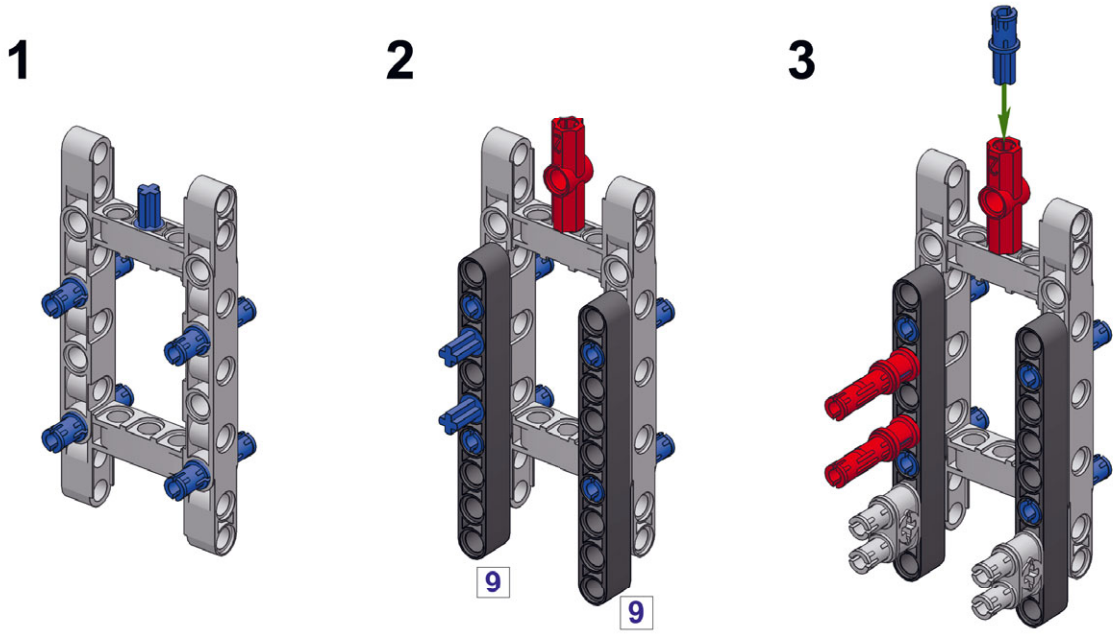
поэтому, когда мотор крутится в обратную сторону, сначала всегда опускается захватное устройство. Более подробное описание этого процесса не имеет непосредственного отношения к теме книги, но вы можете наблюдать гравитацию в действии, если положите механизм набок. Вы убедитесь, что, когда клешню не тянет вниз сила тяжести, механизм не работает.

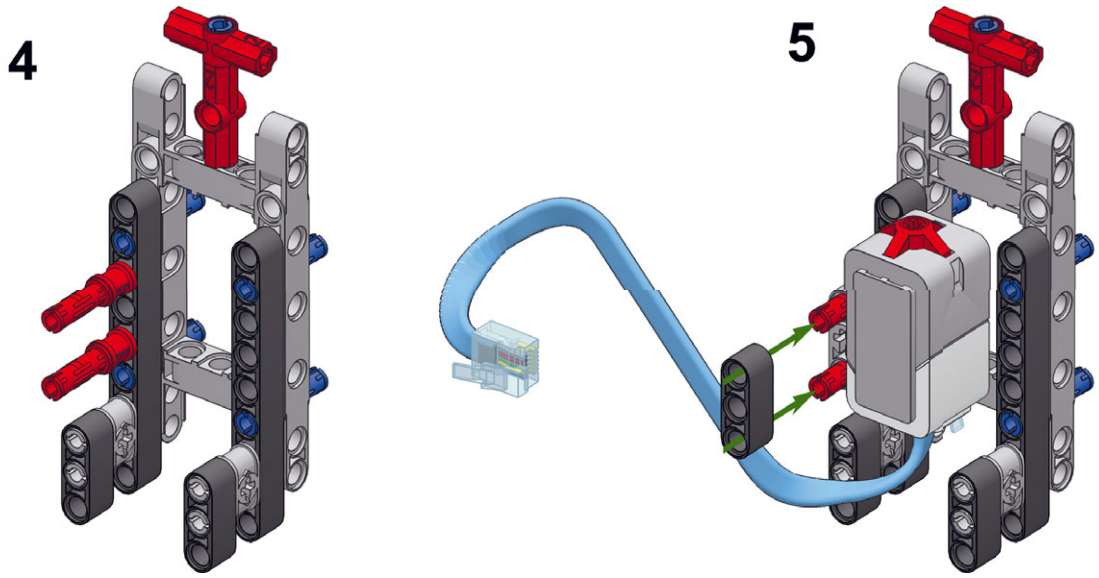
Сборка SNATCH3R

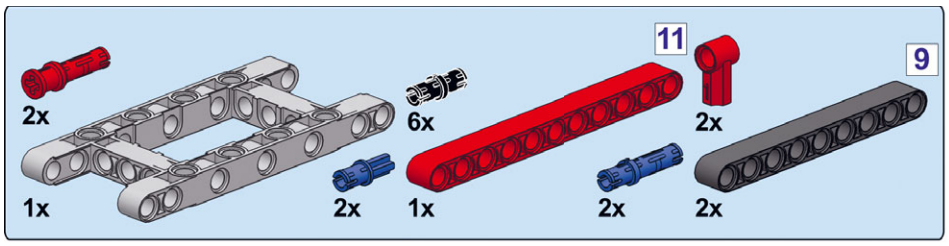
Теперь, когда у вас появилось понимание того, как функционирует захватное устройство робота SNATCH3R, пора построить самого робота и посмотреть, как работает вся конструкция в целом. Выполняйте указания на следующей странице. Но сначала разберите механизм-образец и подготовьте необходимые для этой модели детали, которые показаны на рис. 18.4.

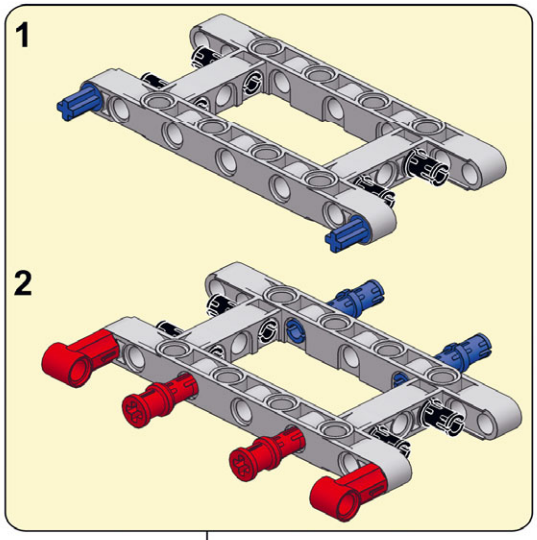
Рис. 18.4. Детали, необходимые для постройки робота SNATCH3R. Когда робот будет собран, останется несколько лишних деталей — они вам пригодятся позднее


3x
1x
4x
Короткий / 25 см
1x
2x
9
2x
2x
1x
2x
4x

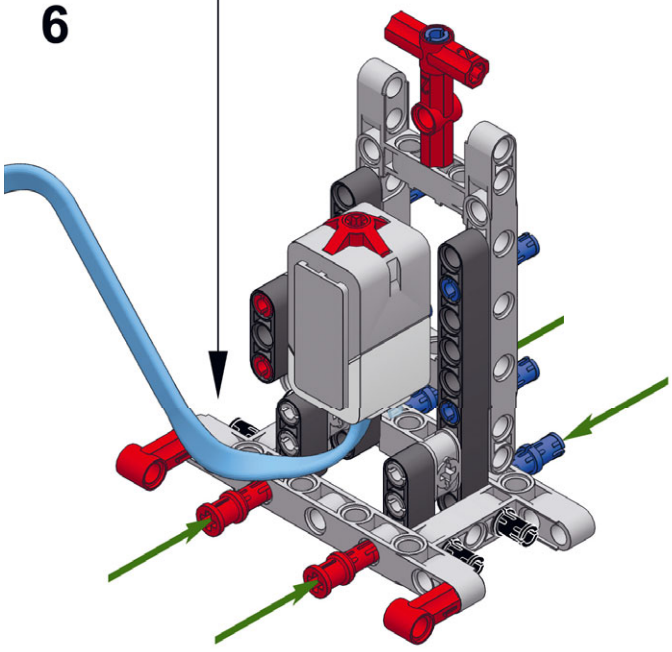

1
2
9
9
3


4
5

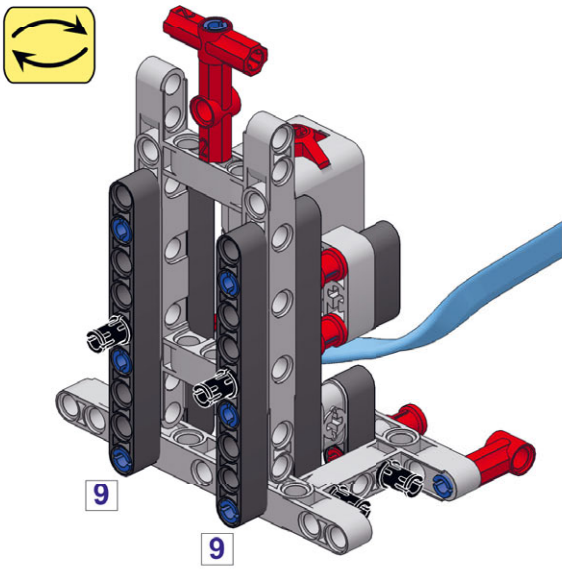




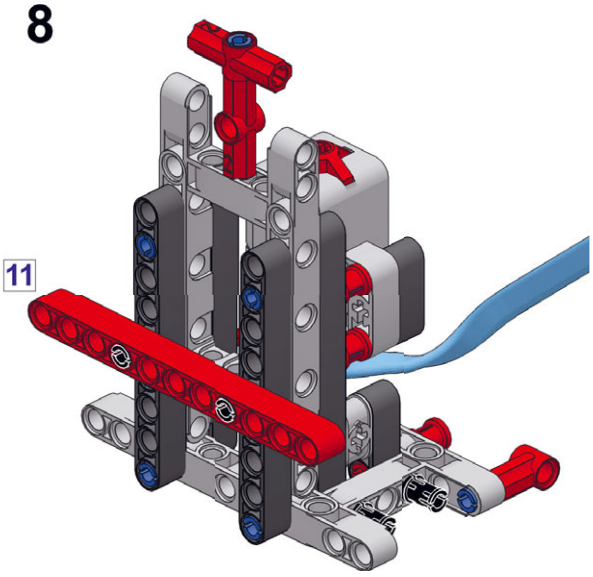
6

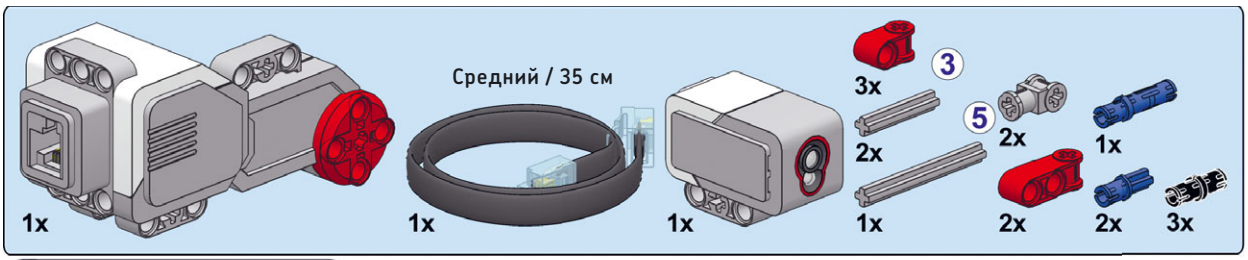


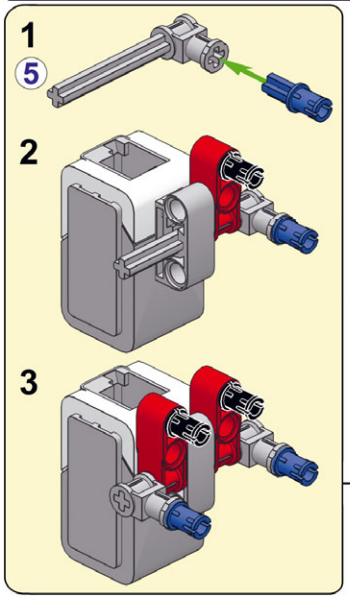
7



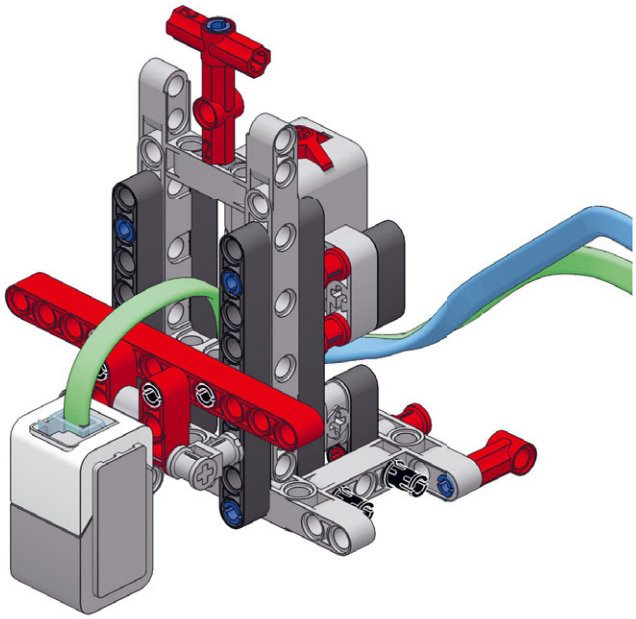
8

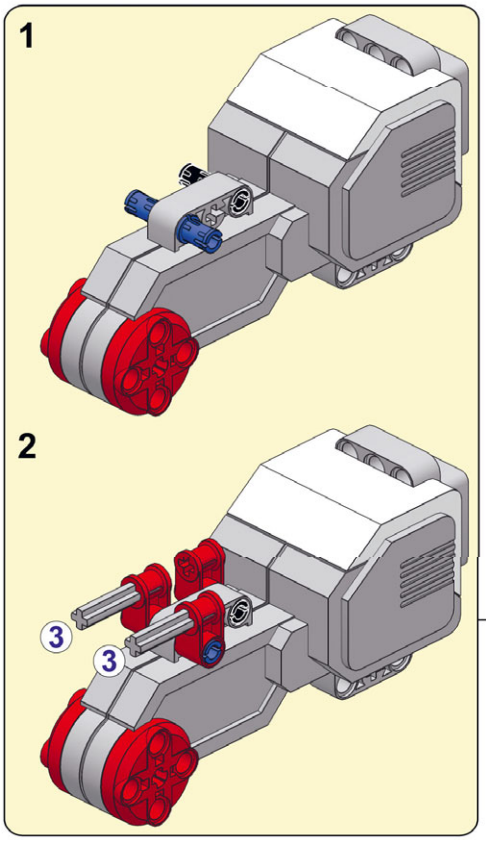




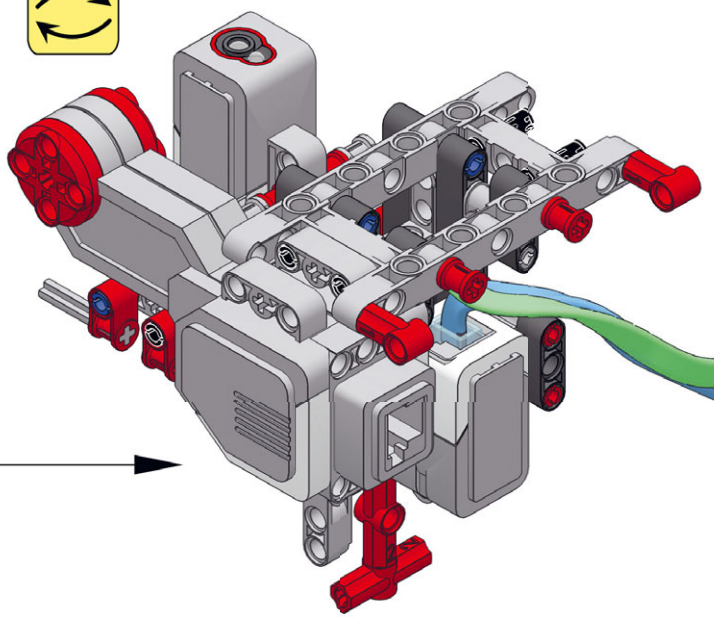


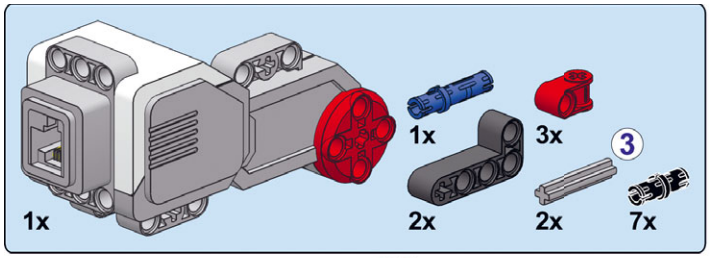
9

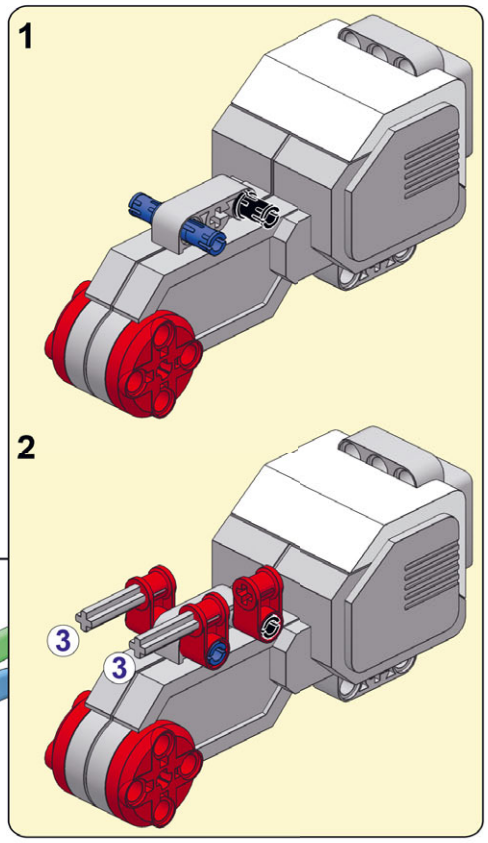




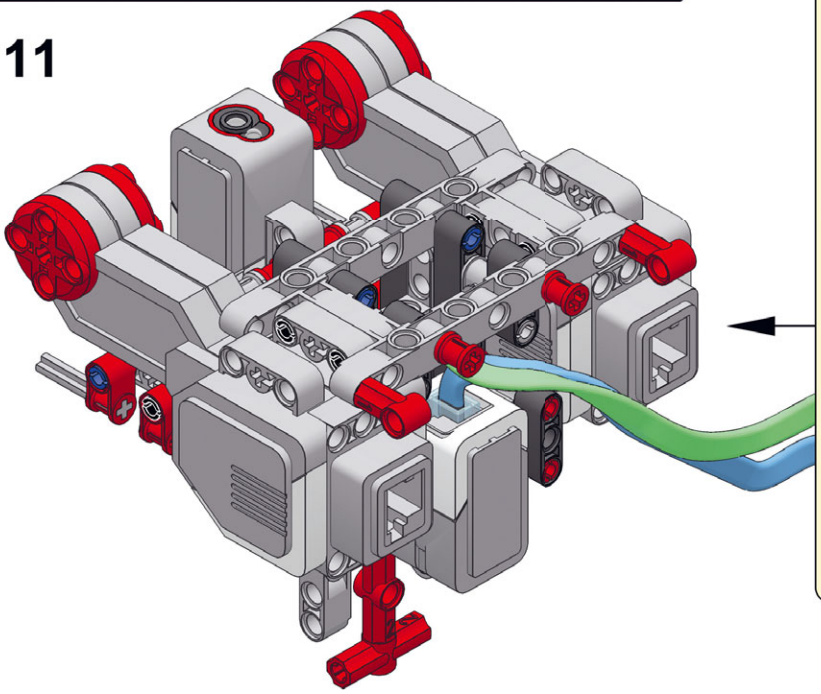
10



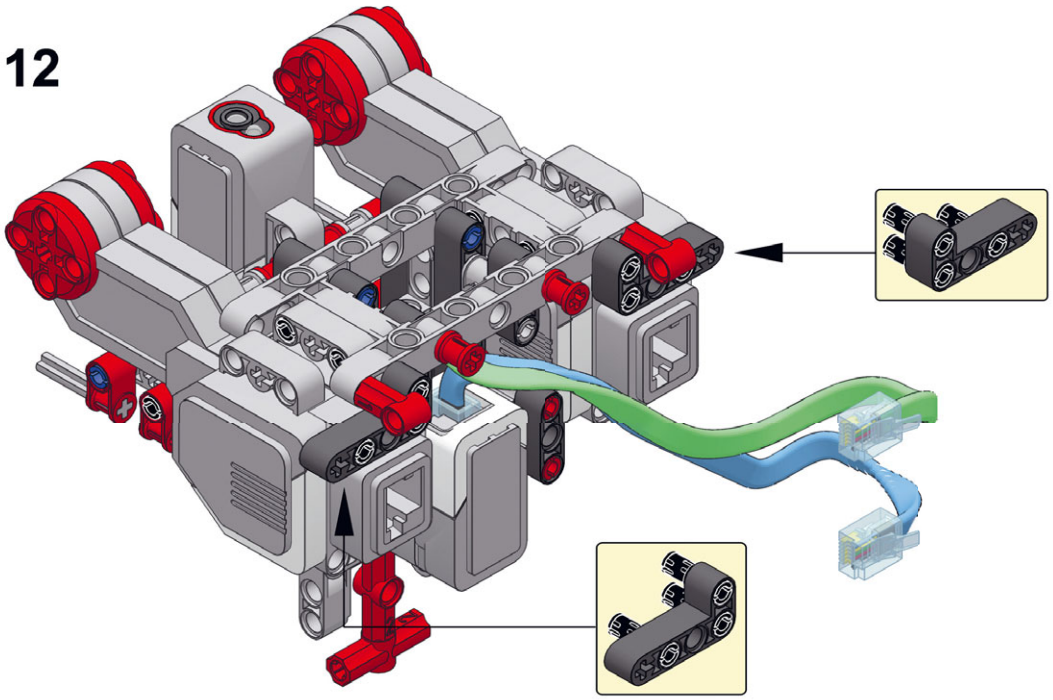


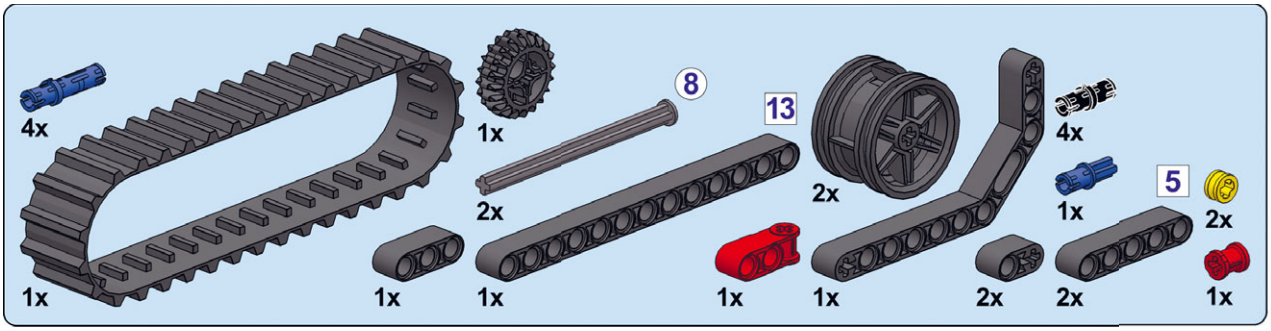


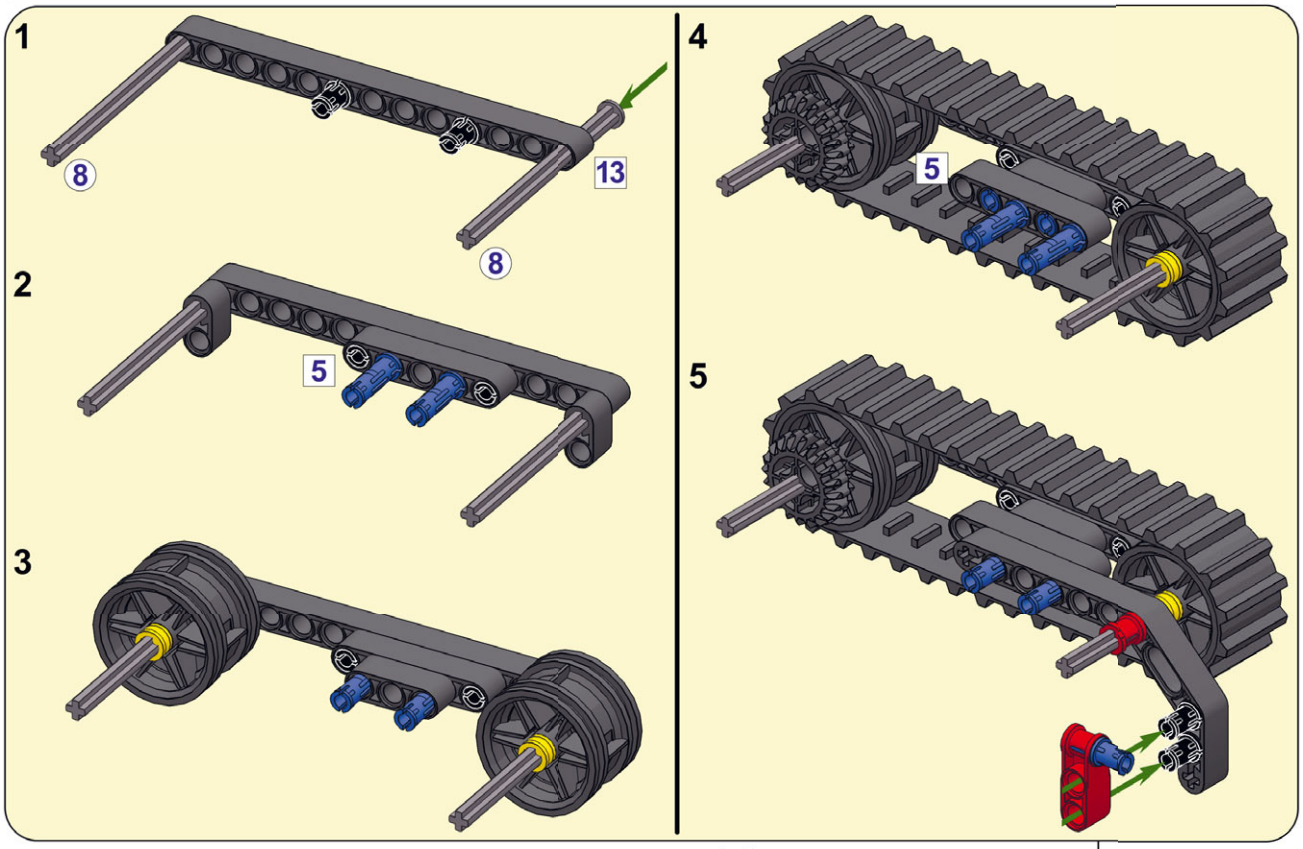
11



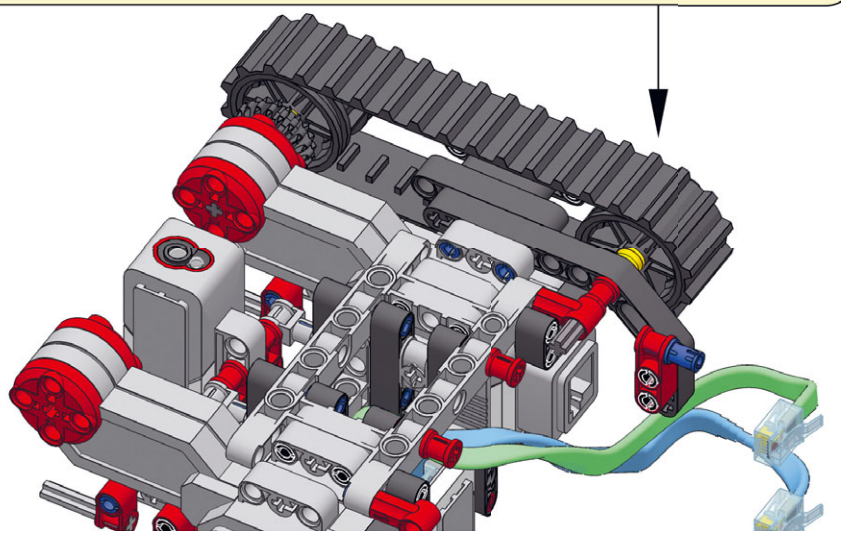
12

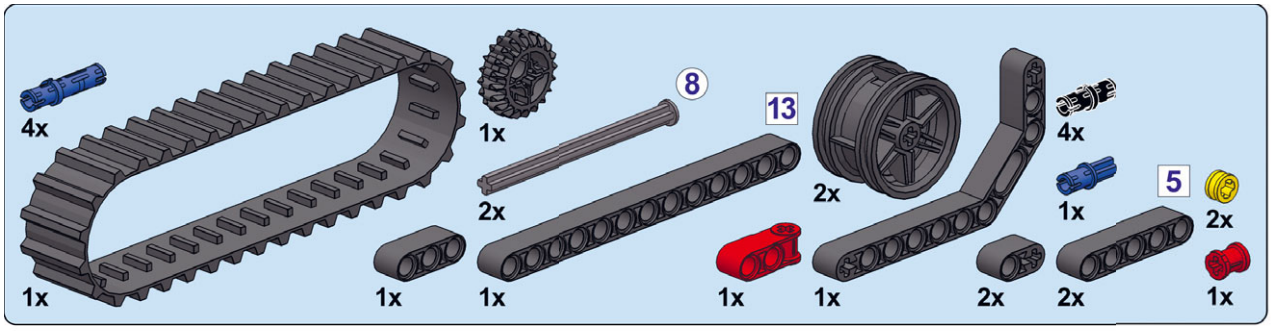


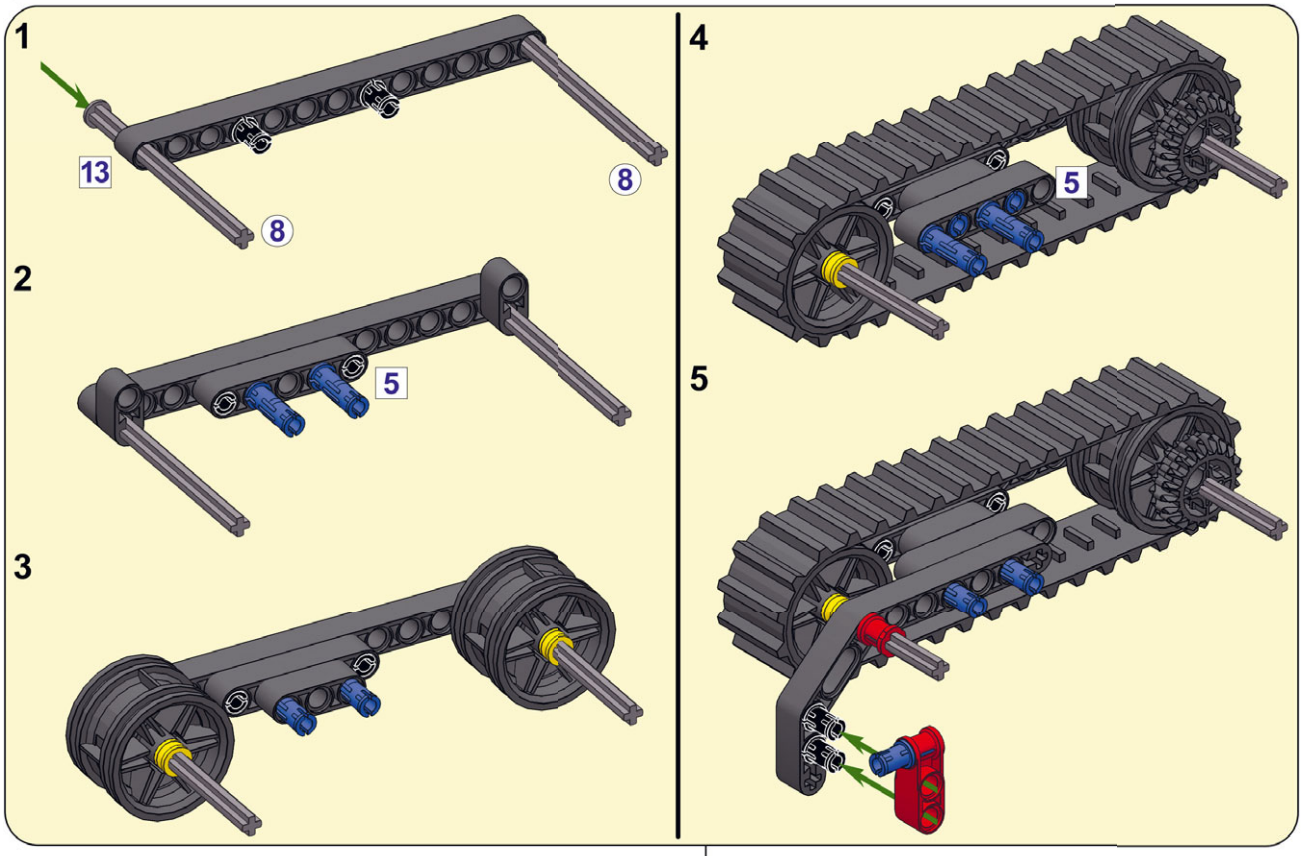




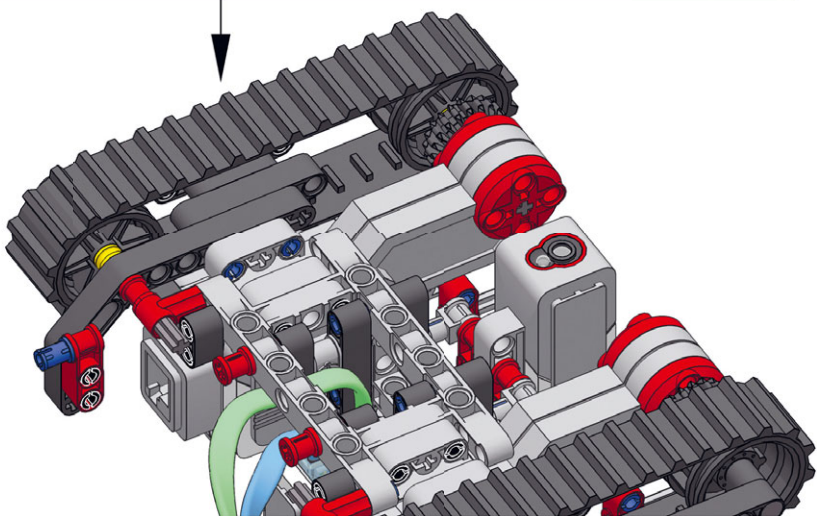
13

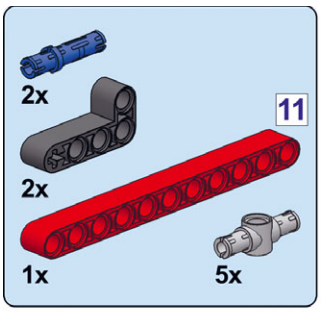


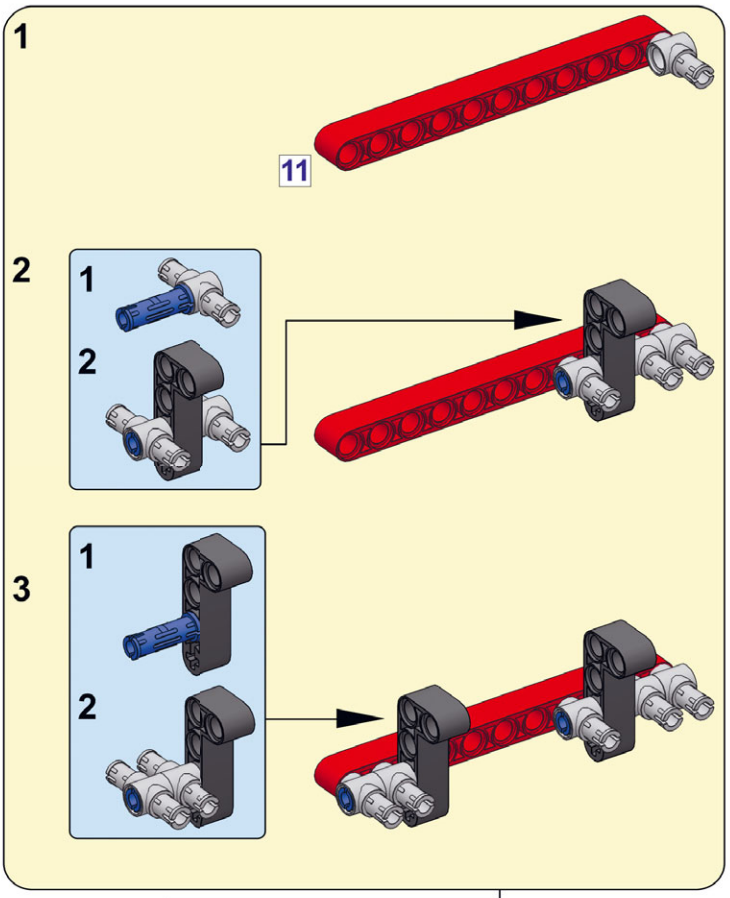




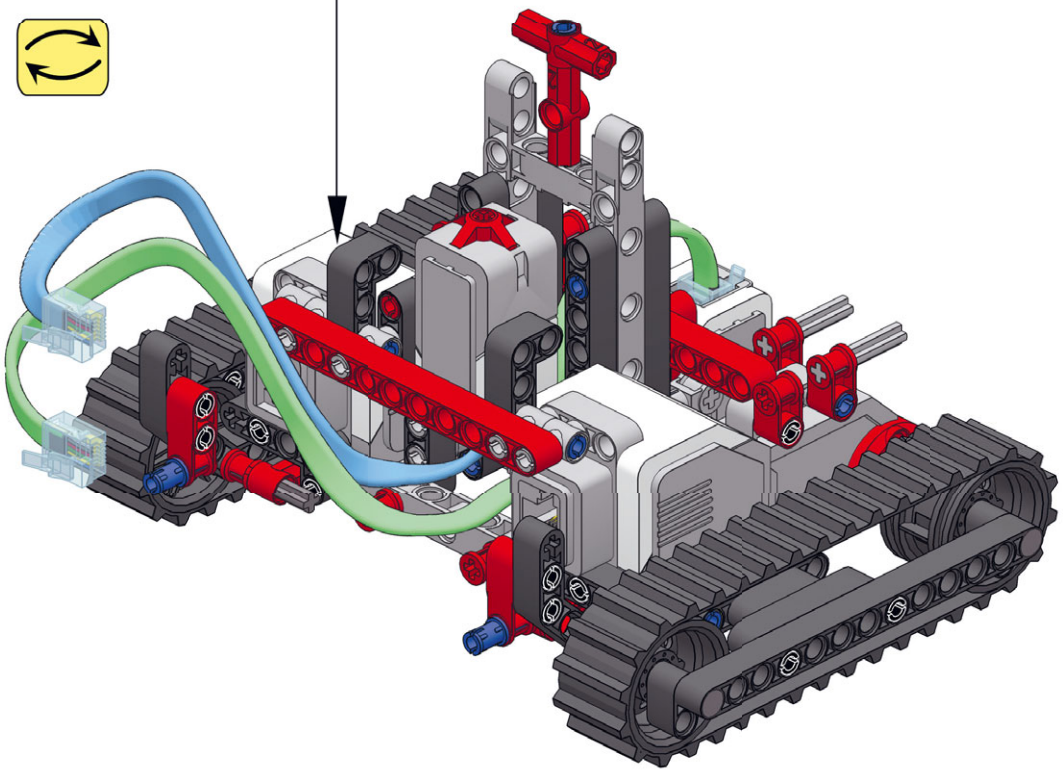
14

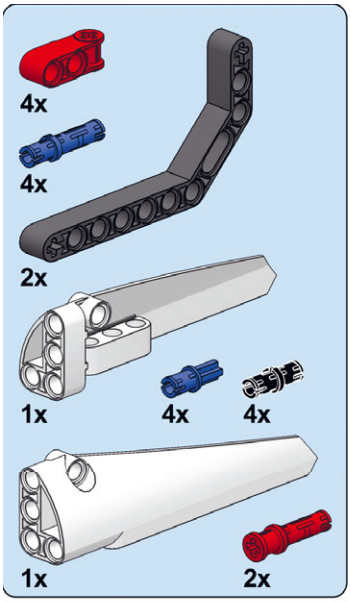


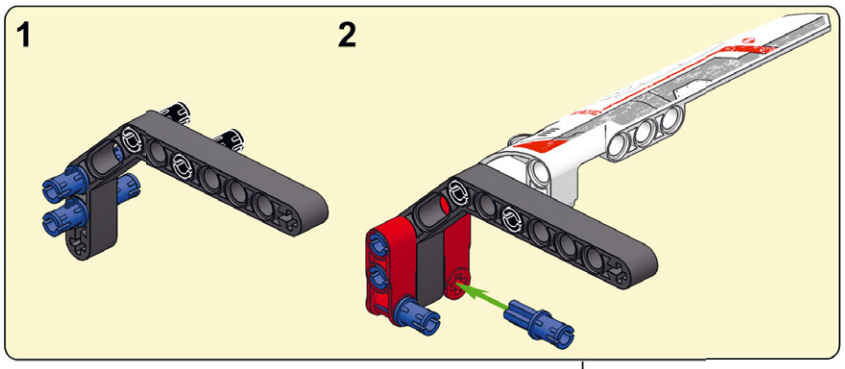




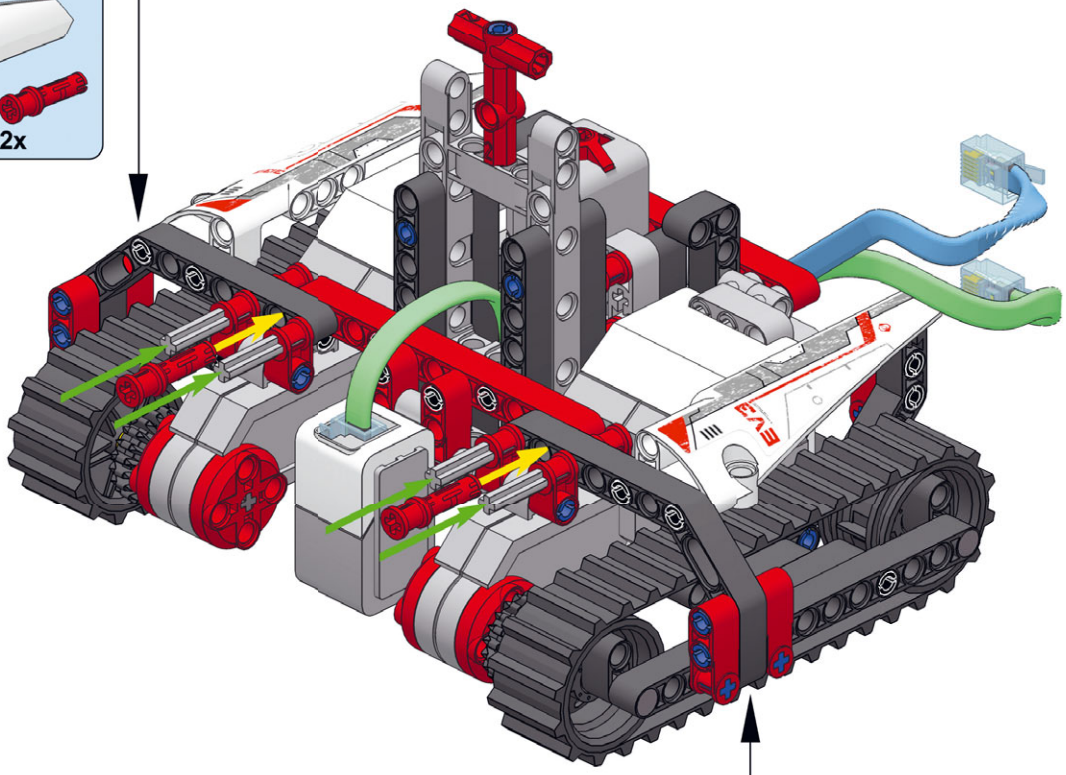
15

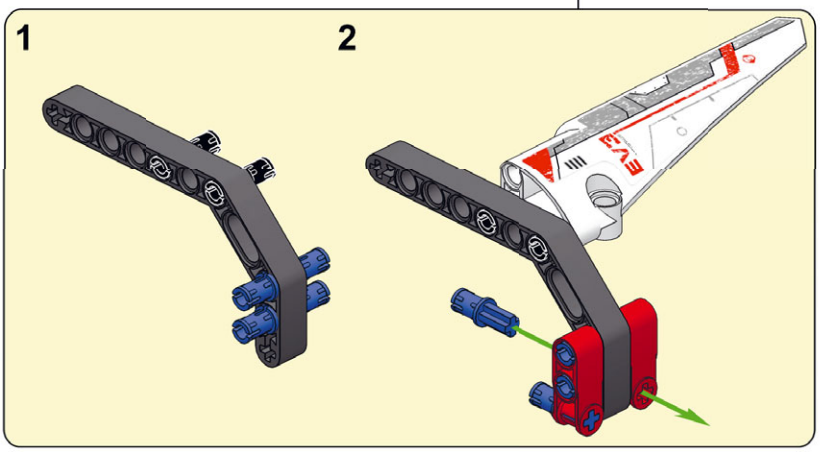


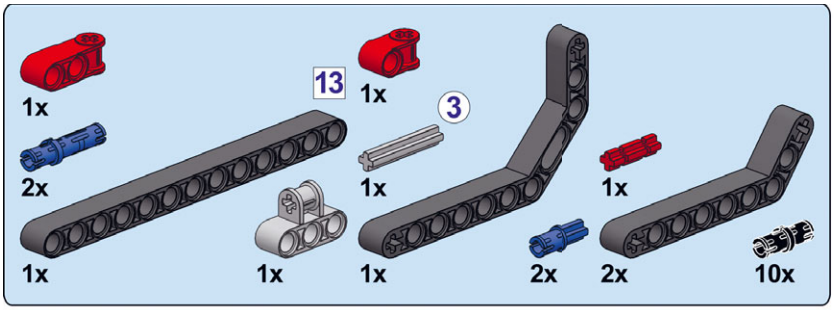


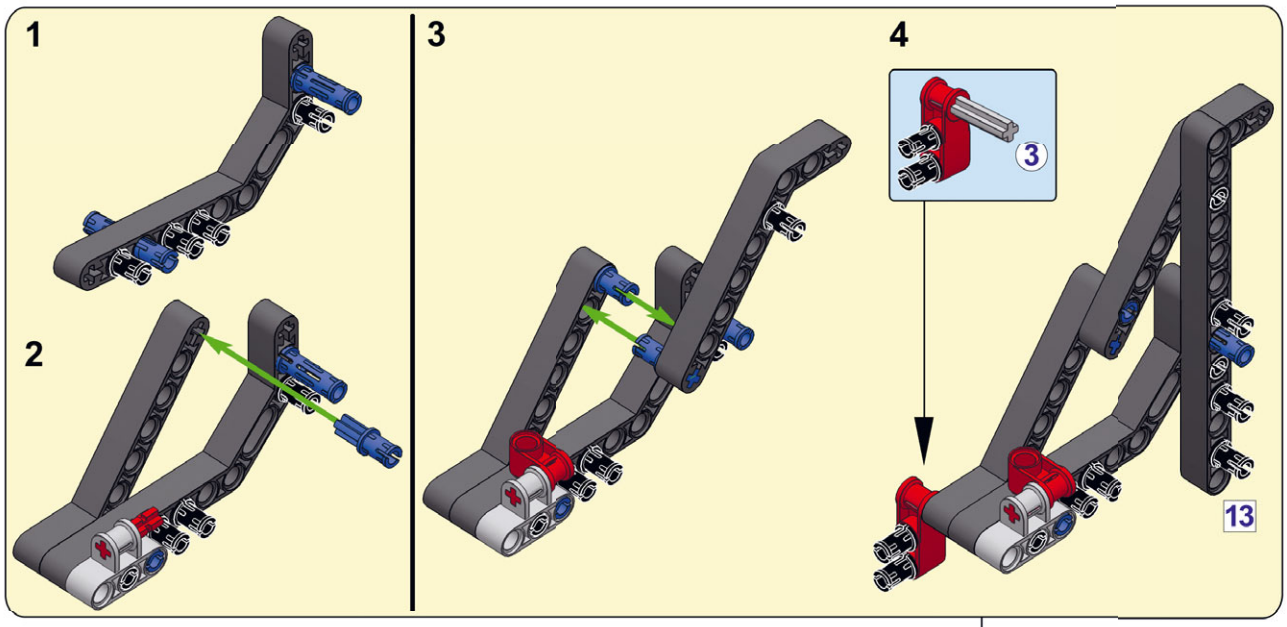


16

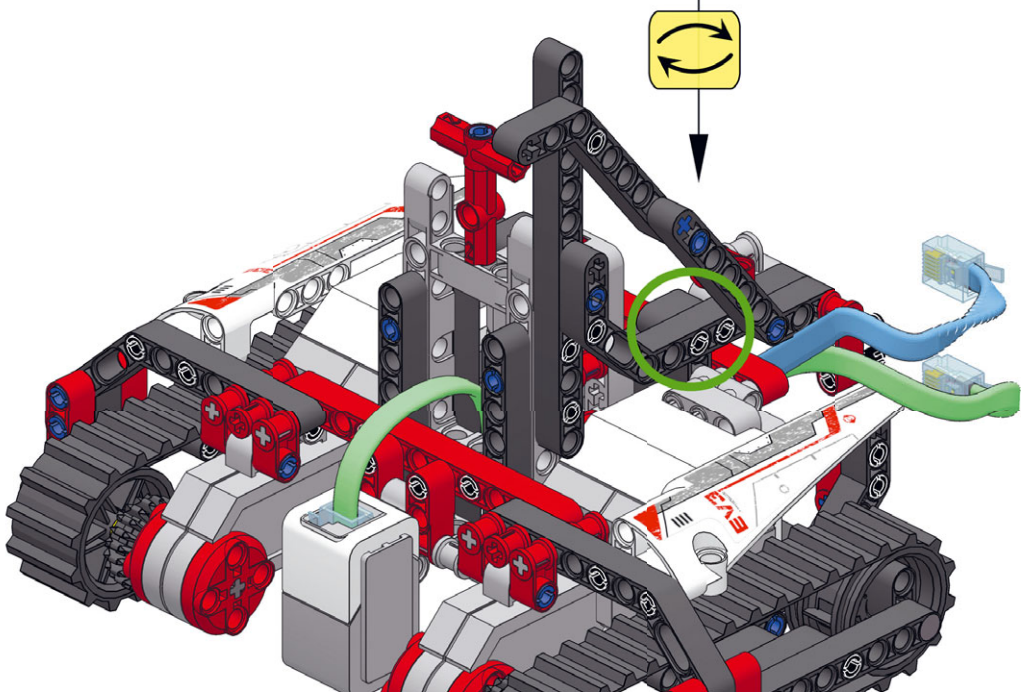


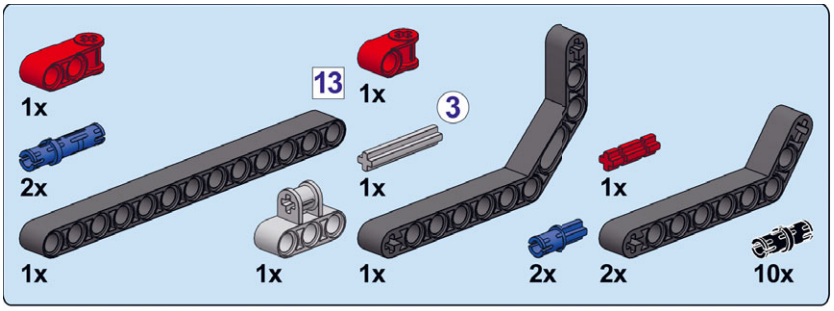


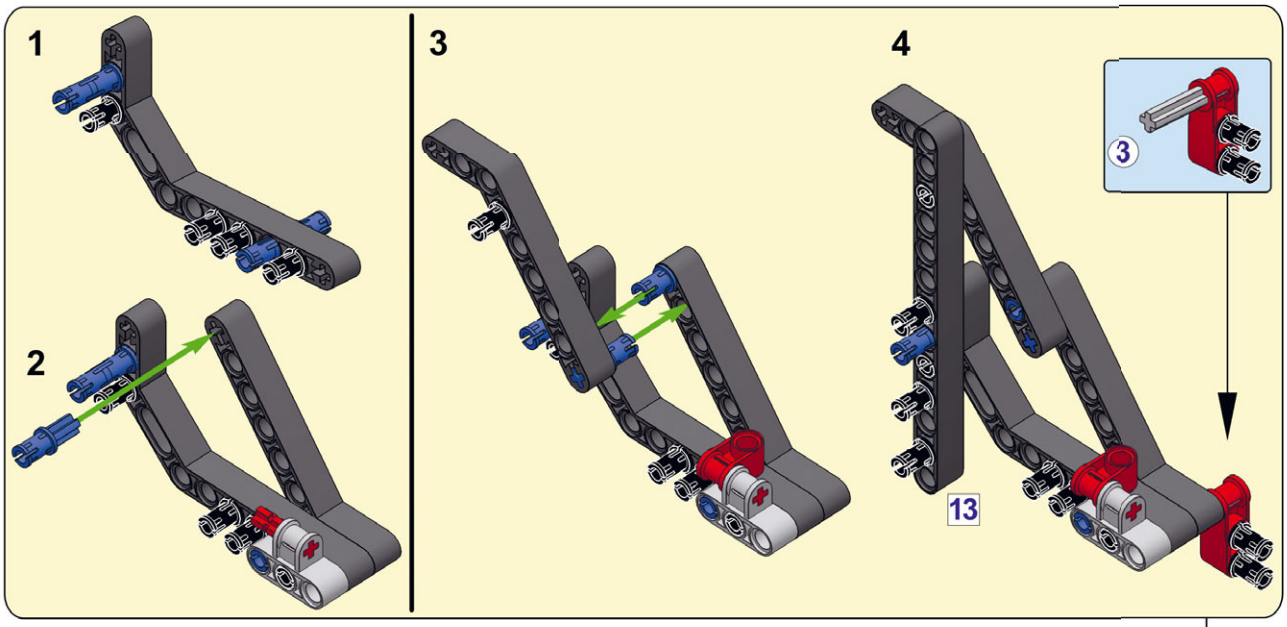




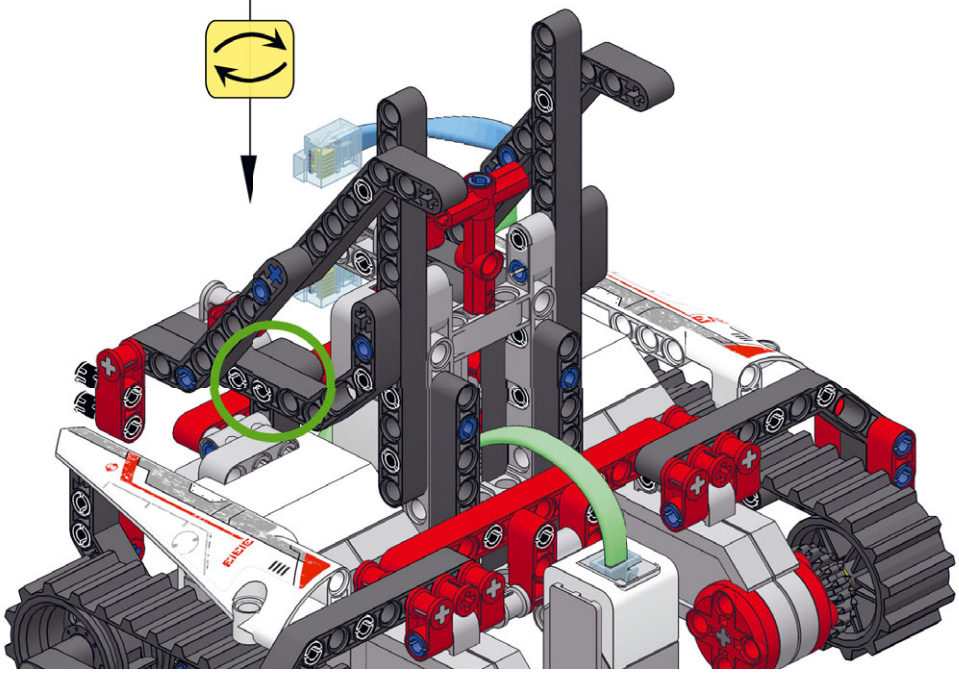
17



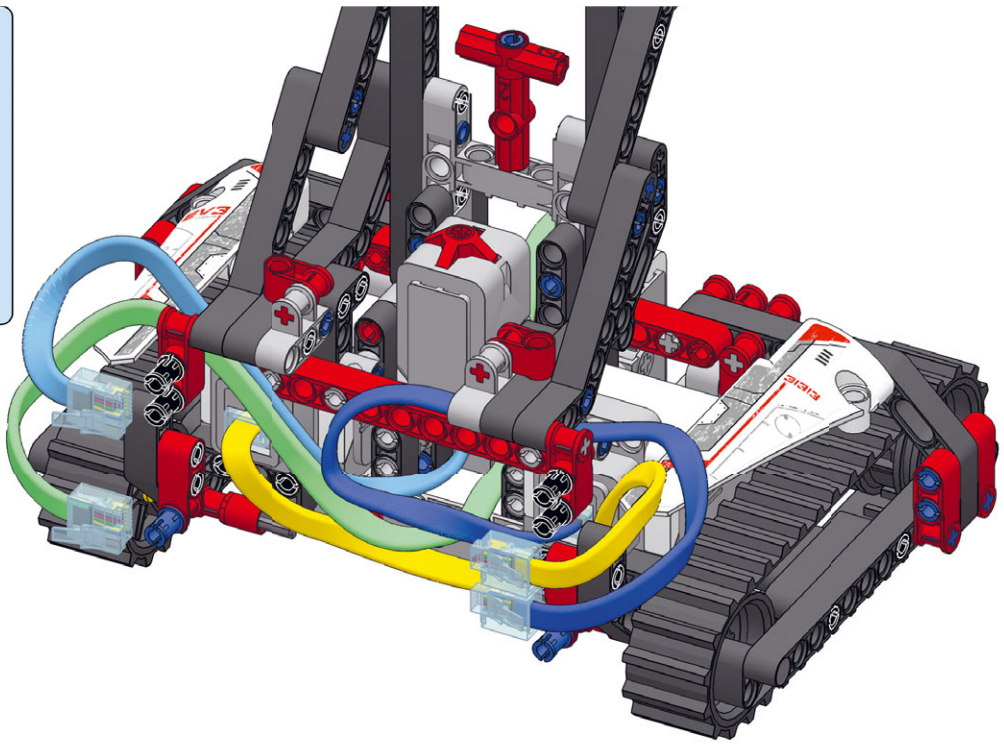




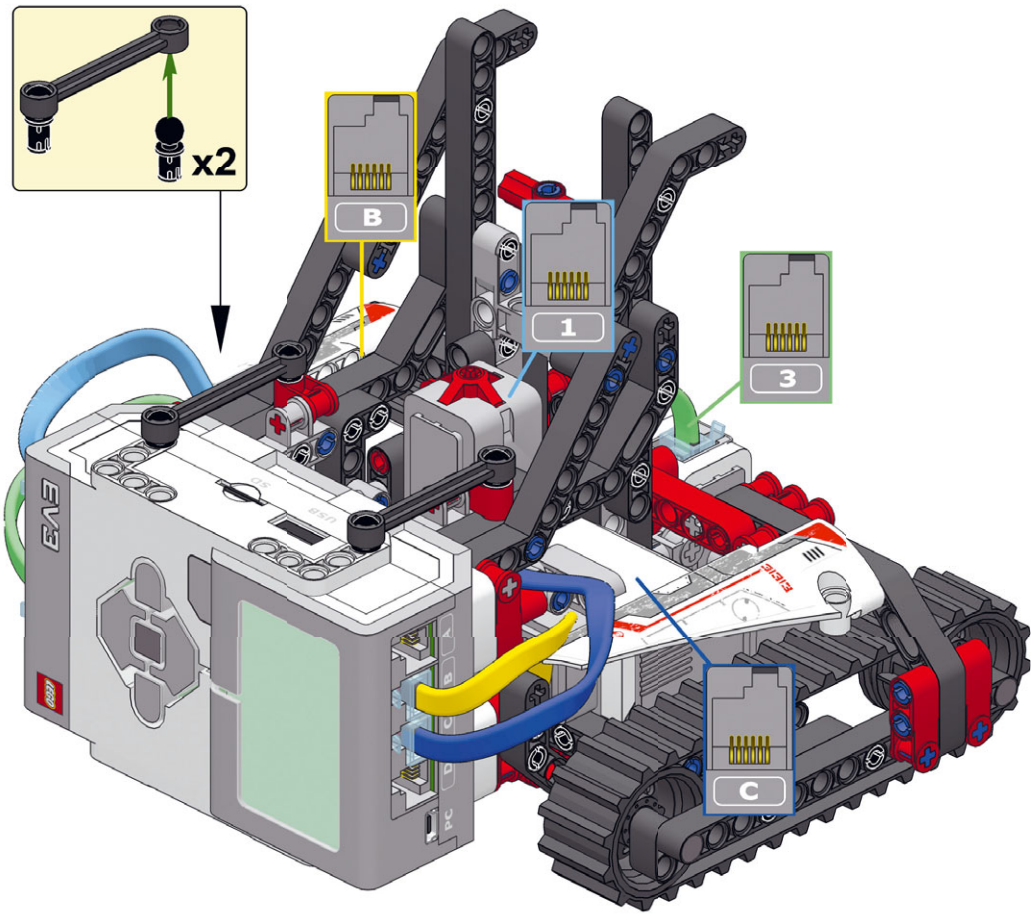
18

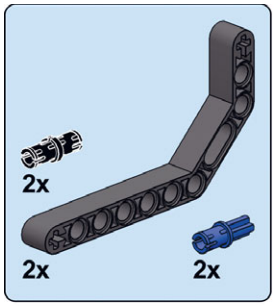


19

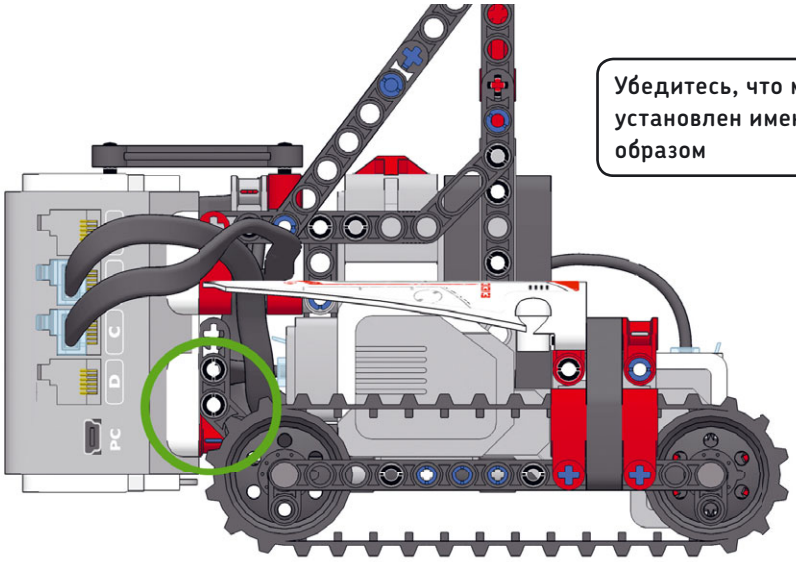


20

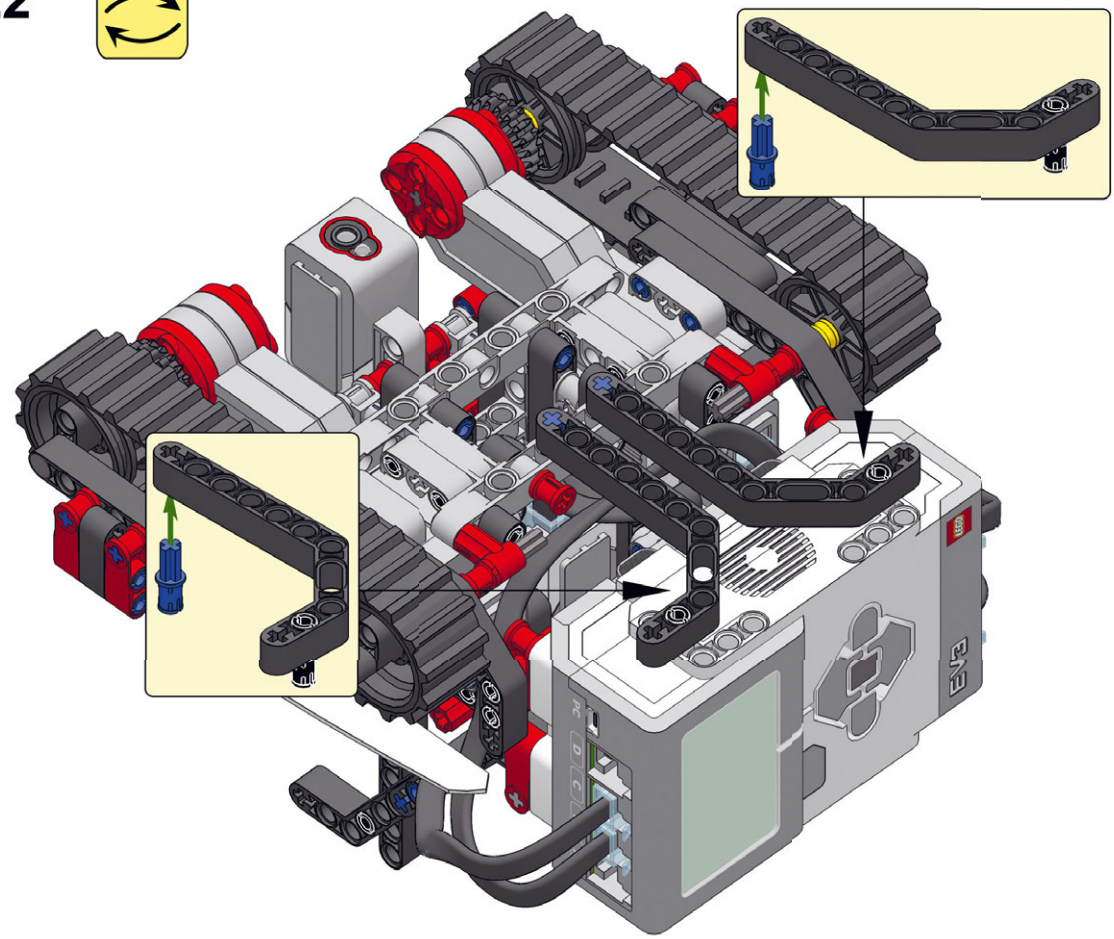


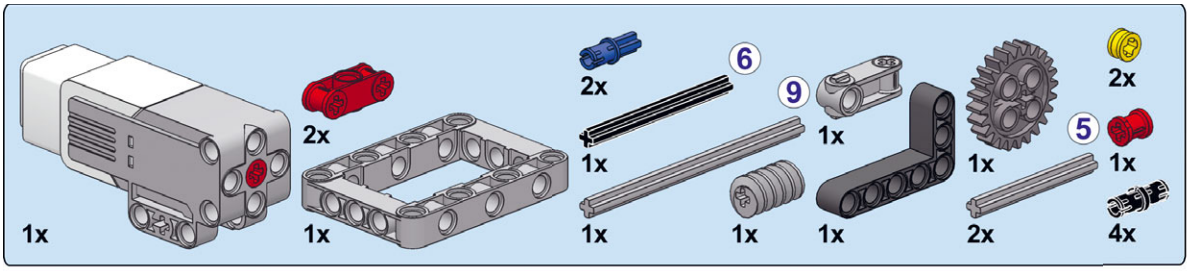

2x
2x
2x

21

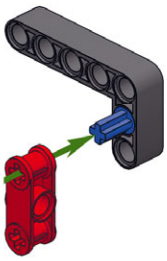

Убедитесь, что модуль EV3 установлен именно таким образом

22

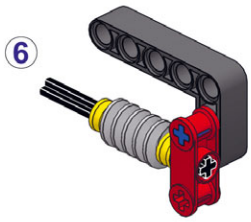




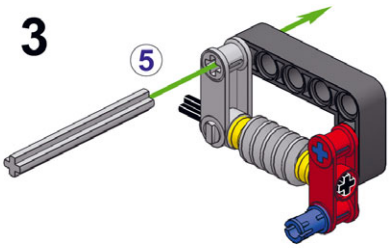
1



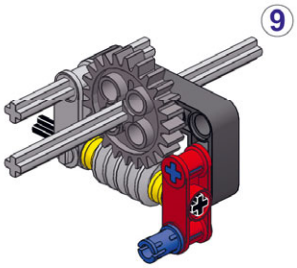
2



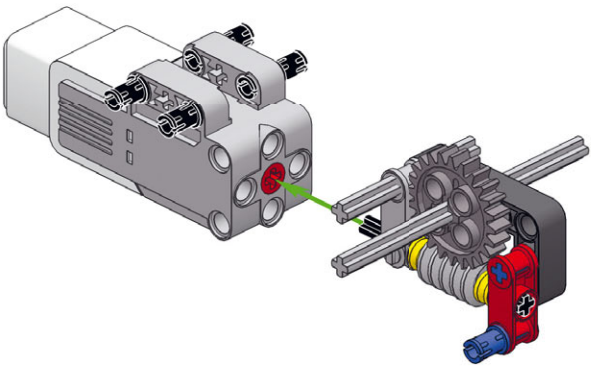
3



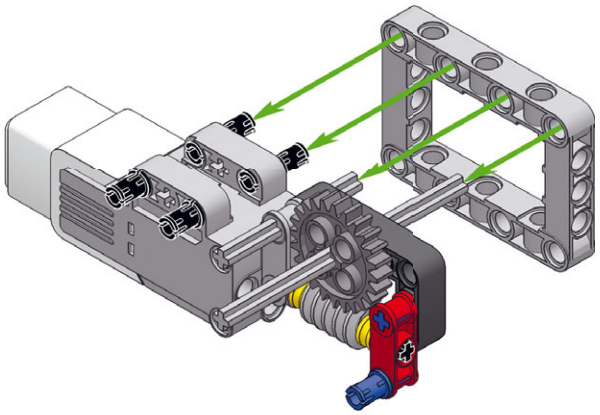
4



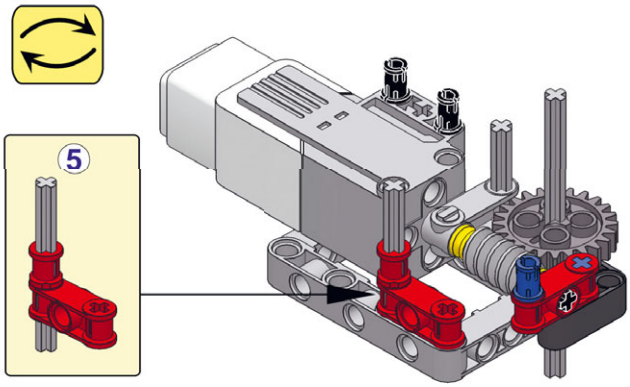
5

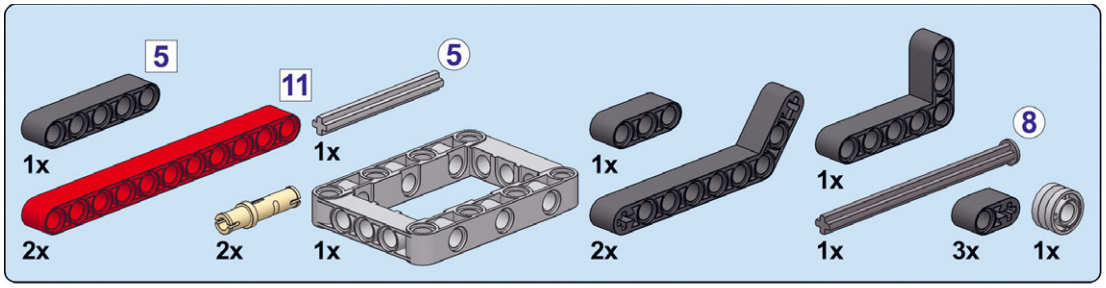


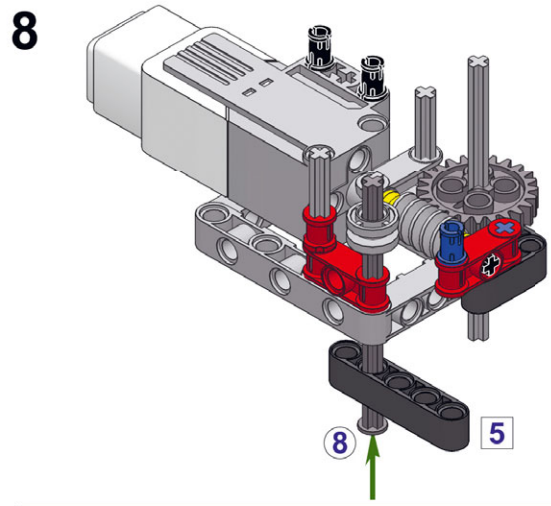
6

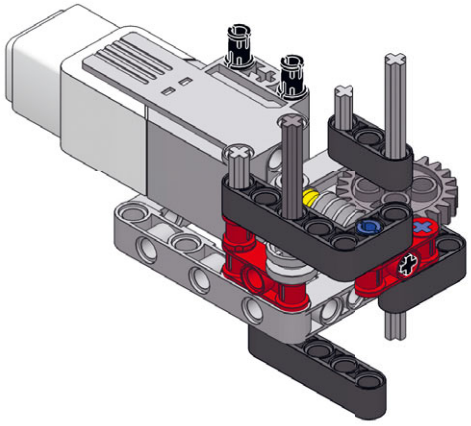


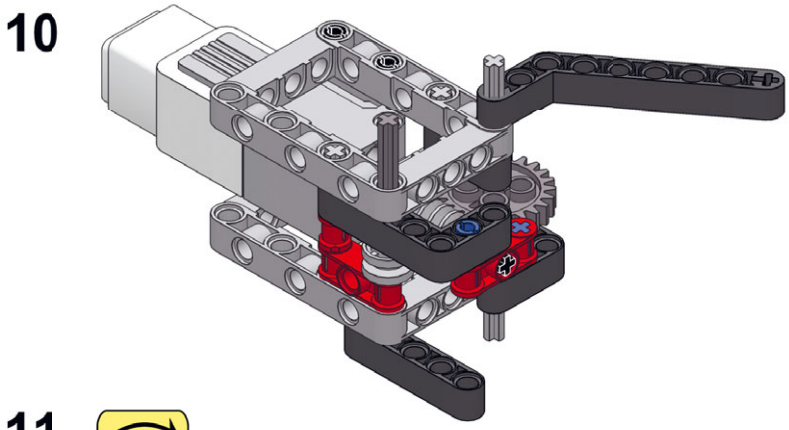
7

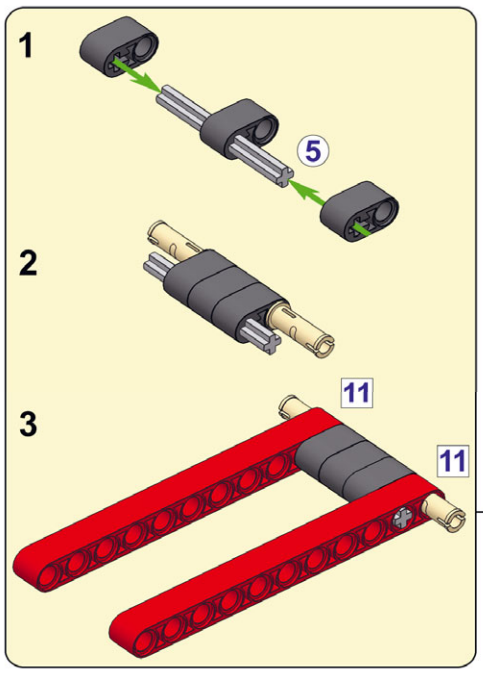


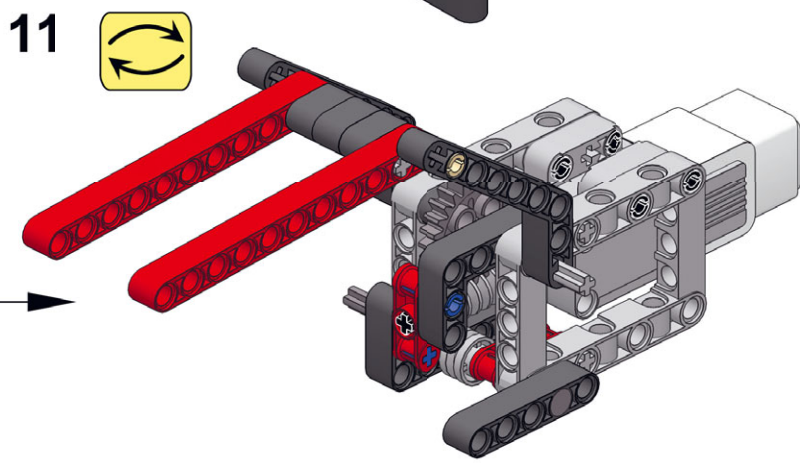

5
11
5
8
1x
1x
1x
1x
2x
2x
1x
2x
1x
3x
1x

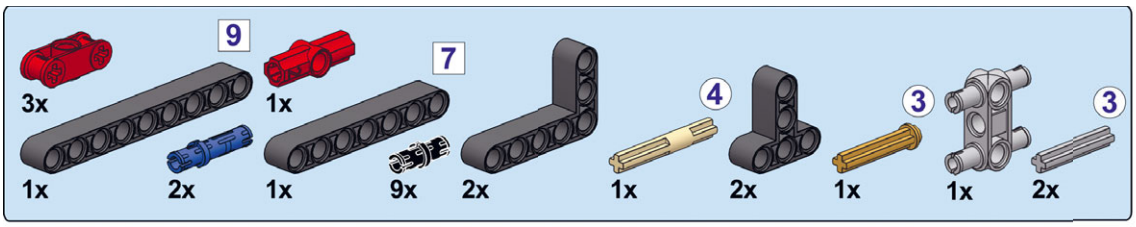

8
8
5

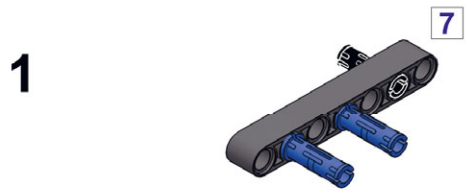

9

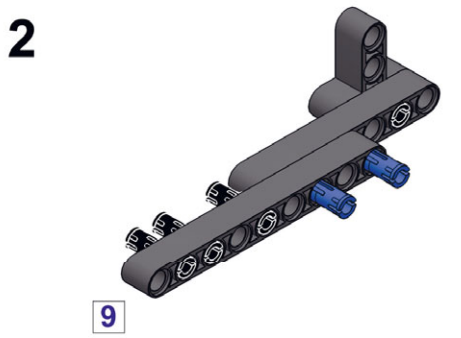

10

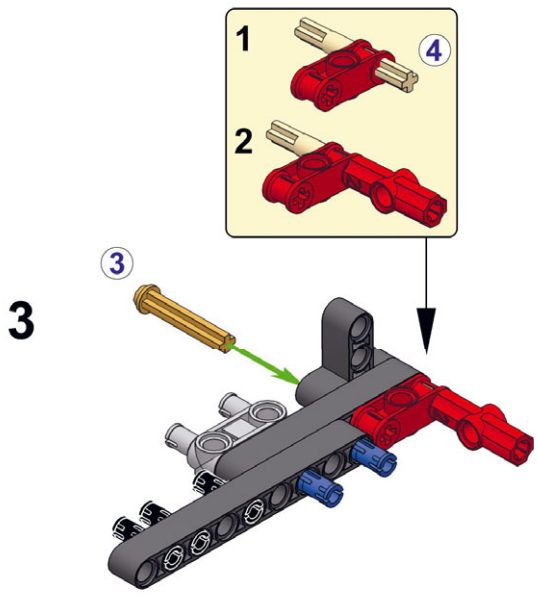

1
5
2
3
11
11

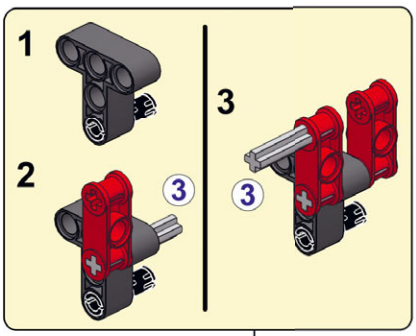

11

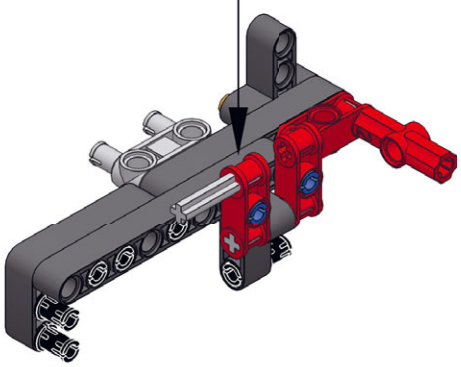

9
3x
1x
2x
7
1x
1x
9x
2x
4
1x
2x
3
1x
3
2x

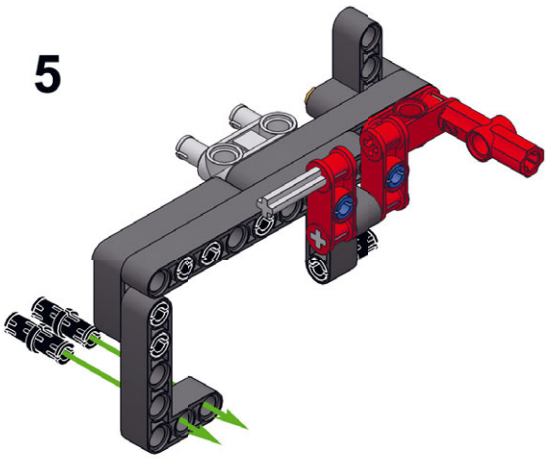

1
7

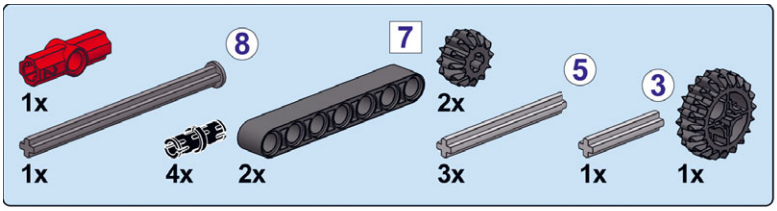

2
9


1
4
2
3
3

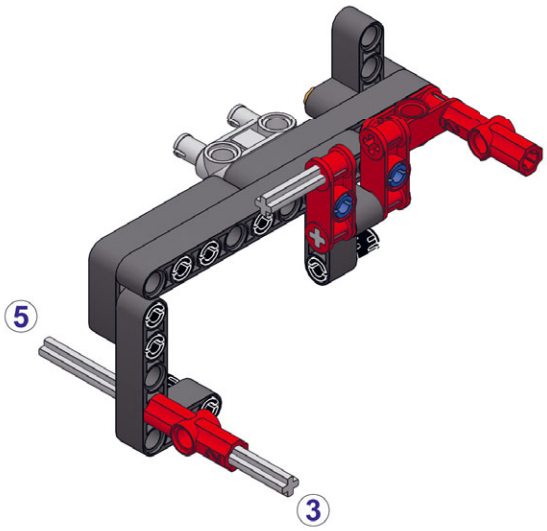

1
3
2
3
3


4

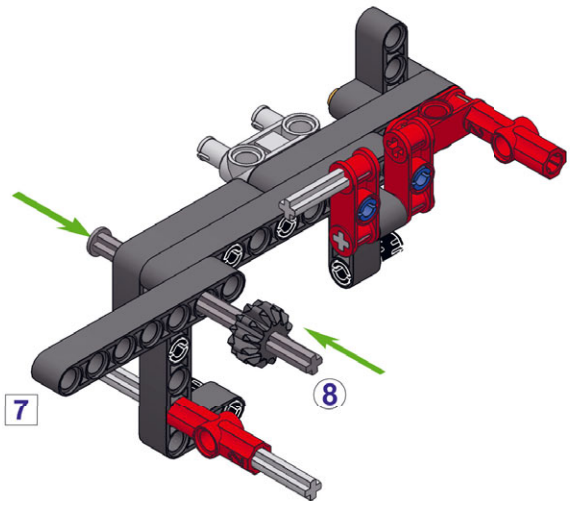

5



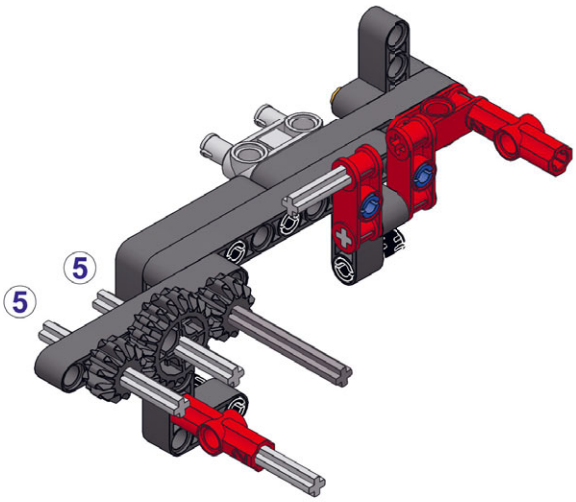
6



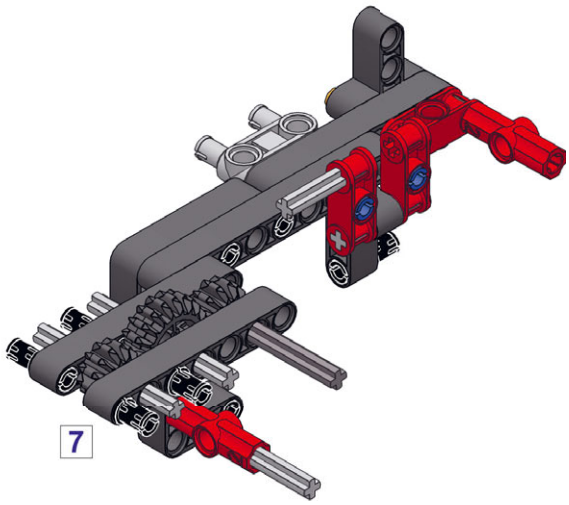
7

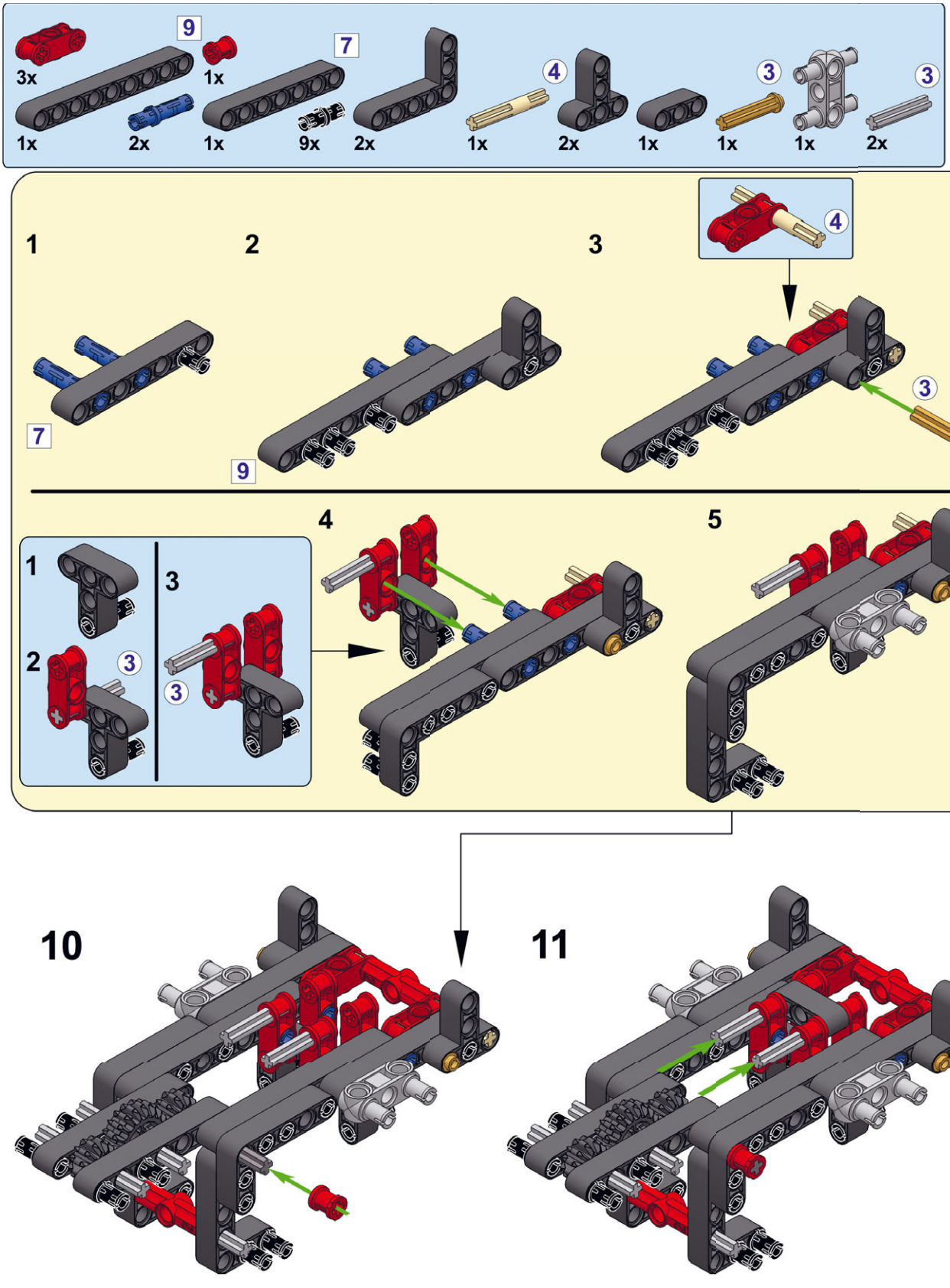


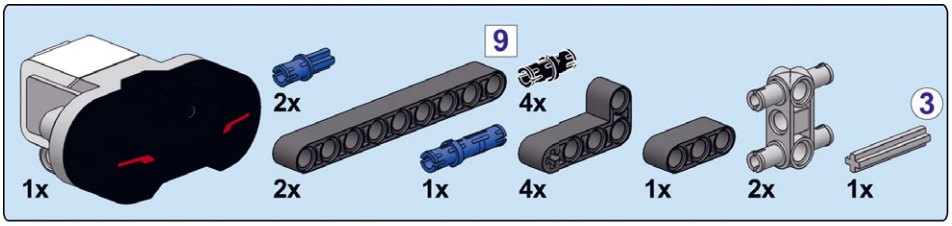
8

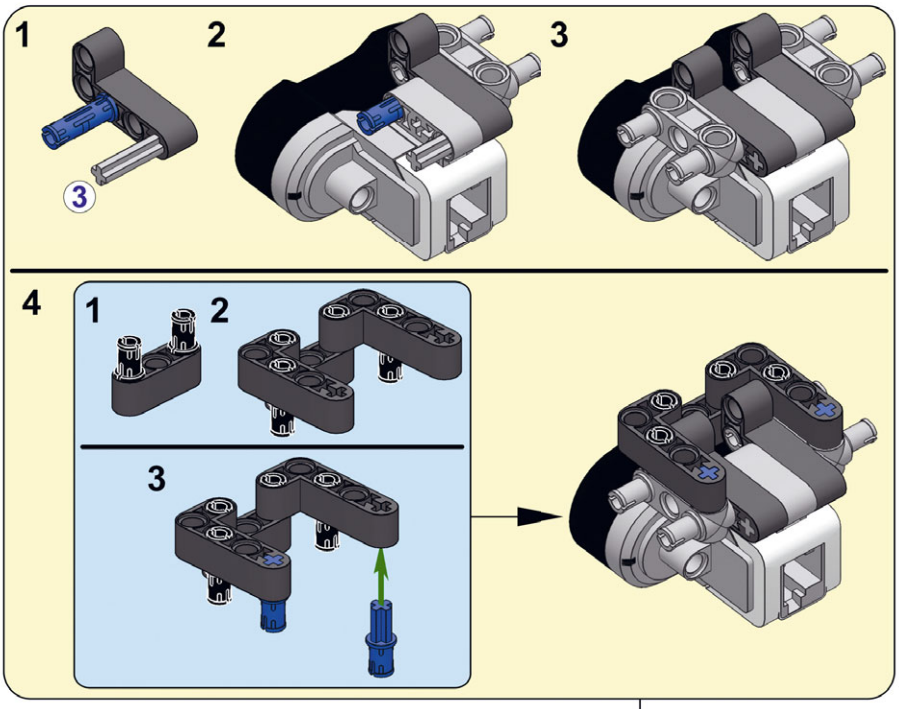


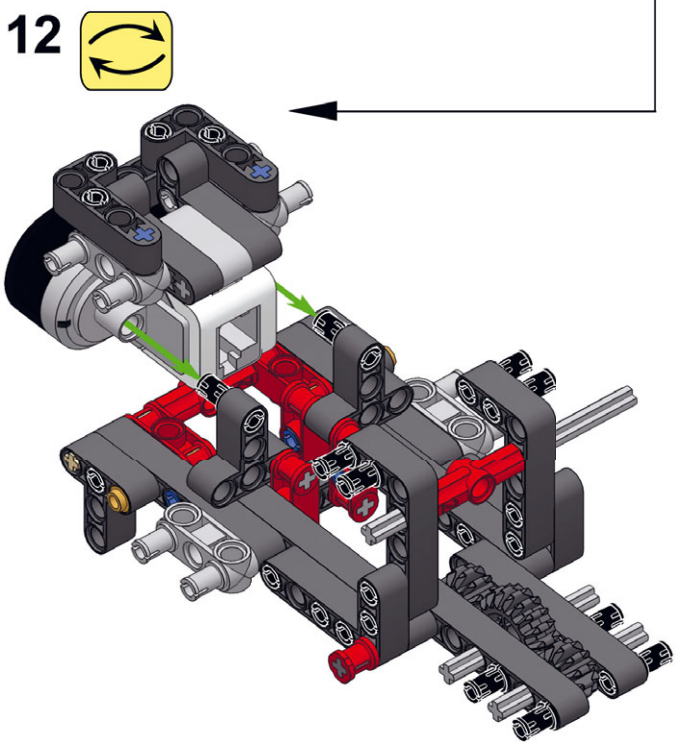
9

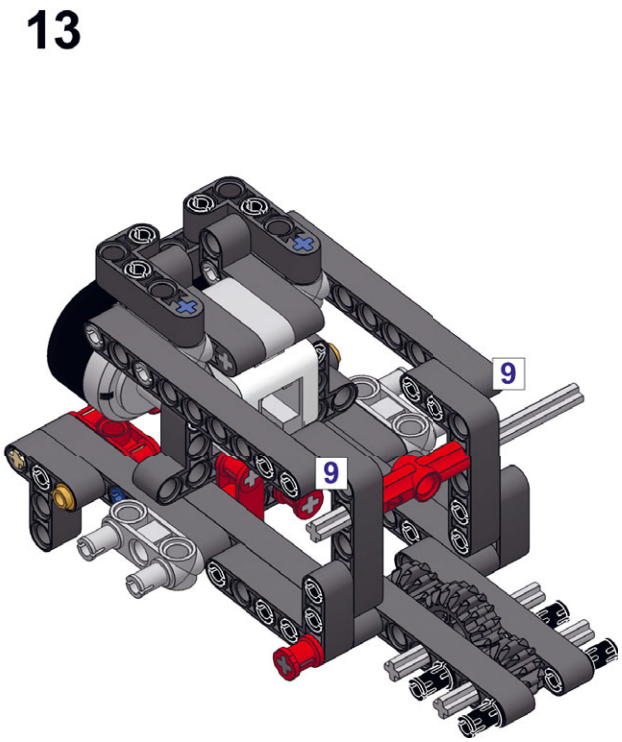


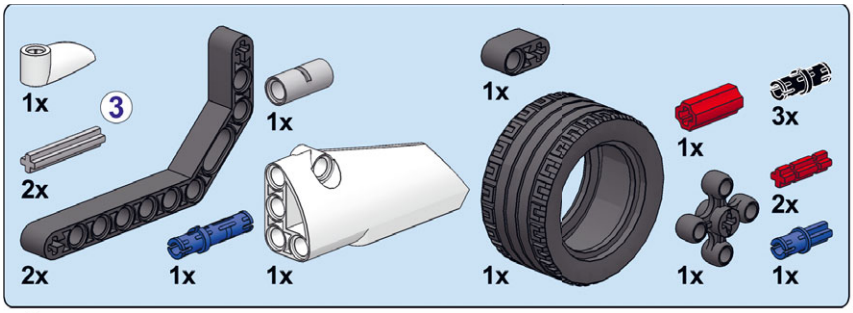

3x
1x
9
1x
2x
7
1x
9x
2x
4
1x
2x
1x
3
1x
1x
3
2x
1
2
3
4
7
9
3
1
2
3
3
3
4
5
10
11

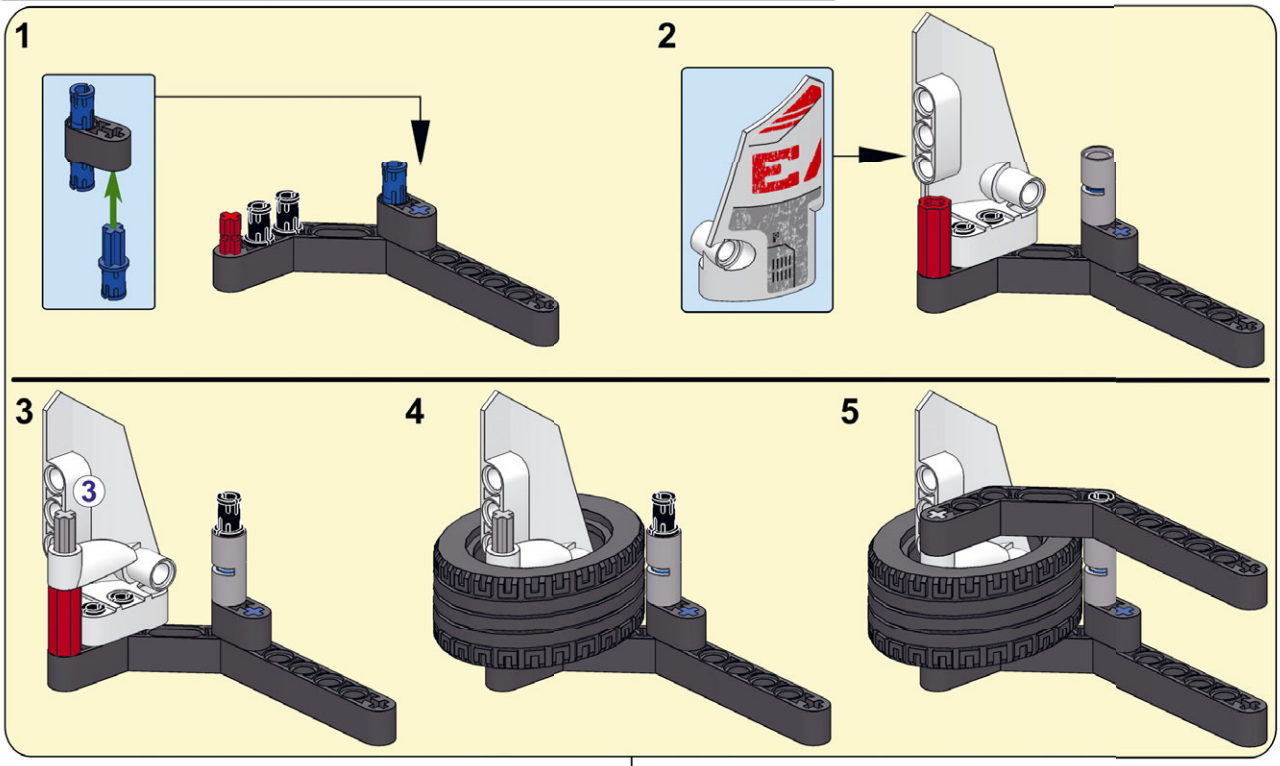

9
2x
4x
1x
2x
1x
4x
1x
2x
1x
3

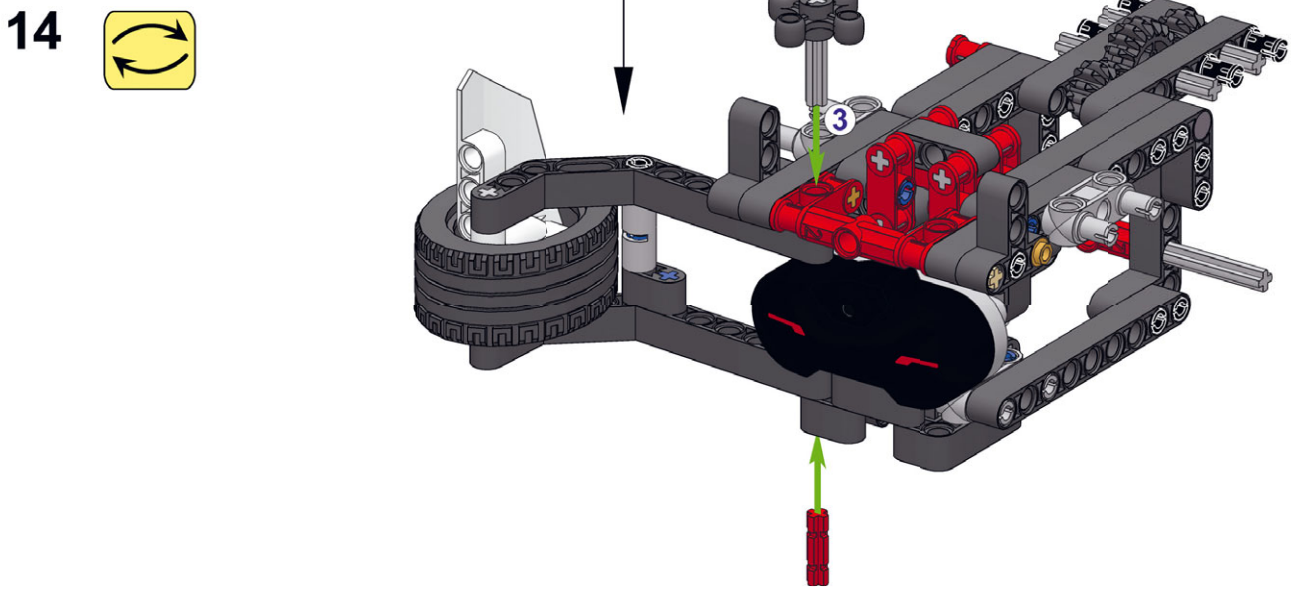

1
2
3
3
4
1
2
3

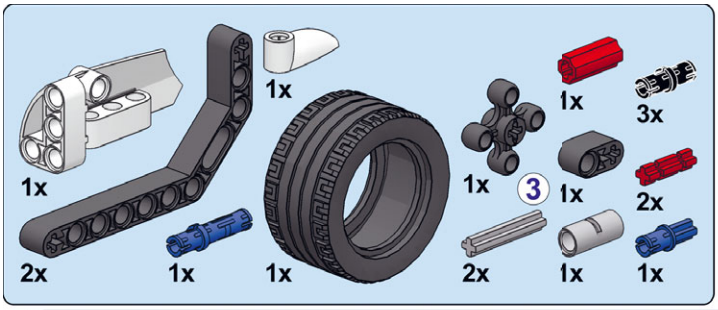

12

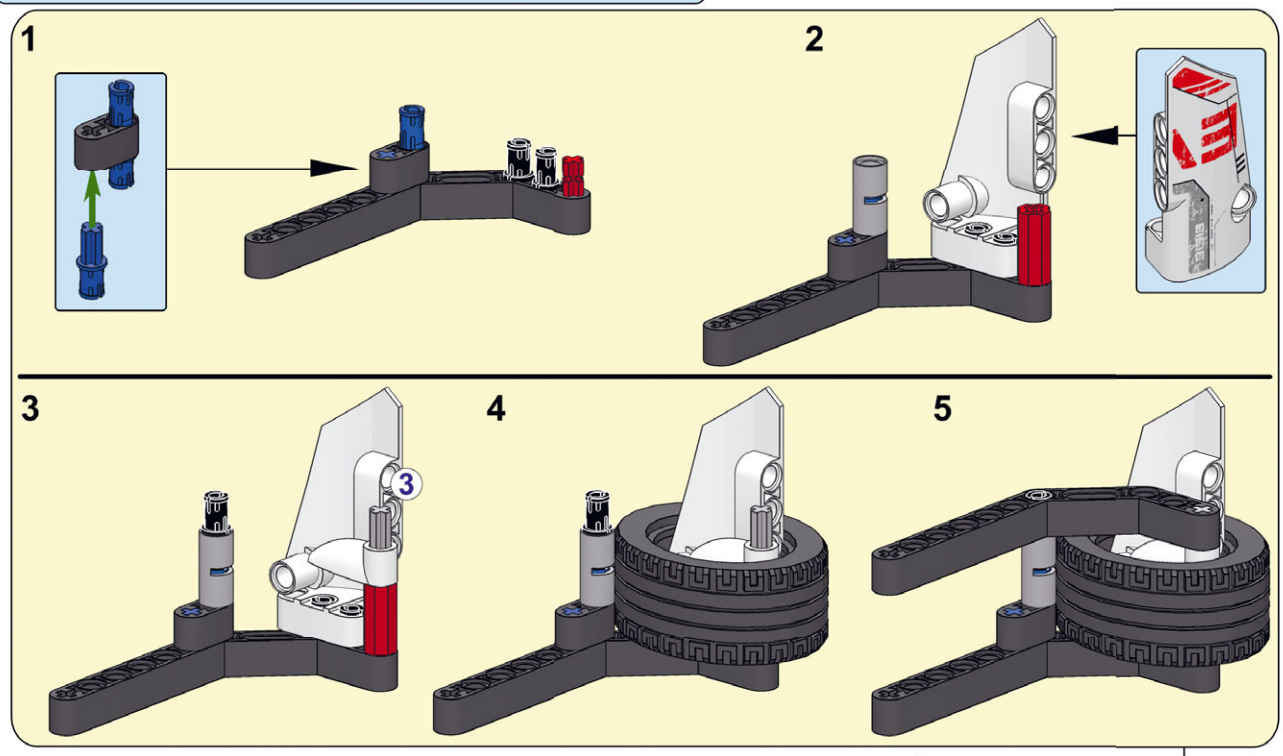

13
9
9


1x
3
2x
2x
1x
1x
1x
1x
1x
1x
3x
2x
1x

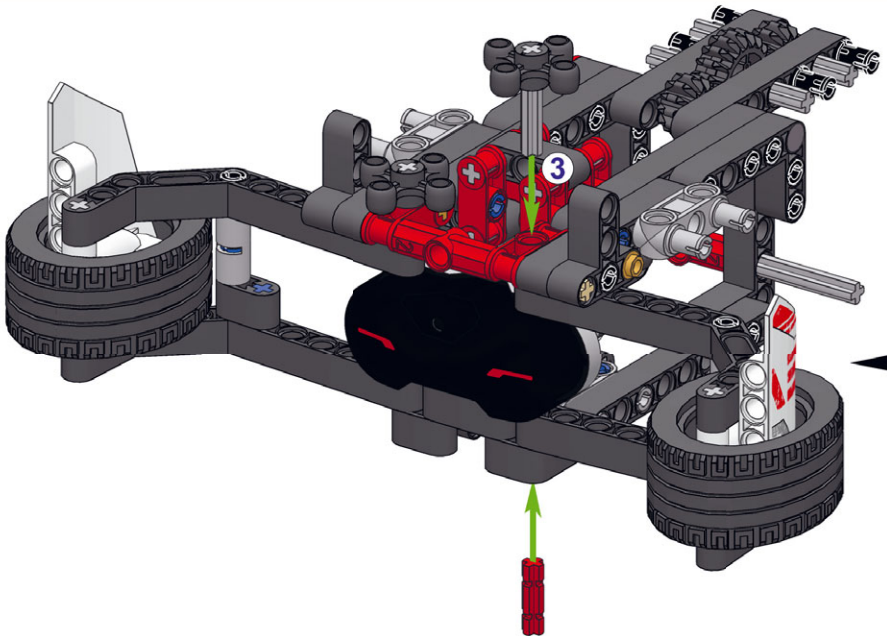

1
2
3
3
4
5

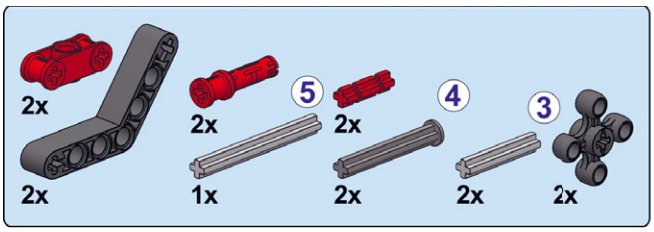

14
3





15

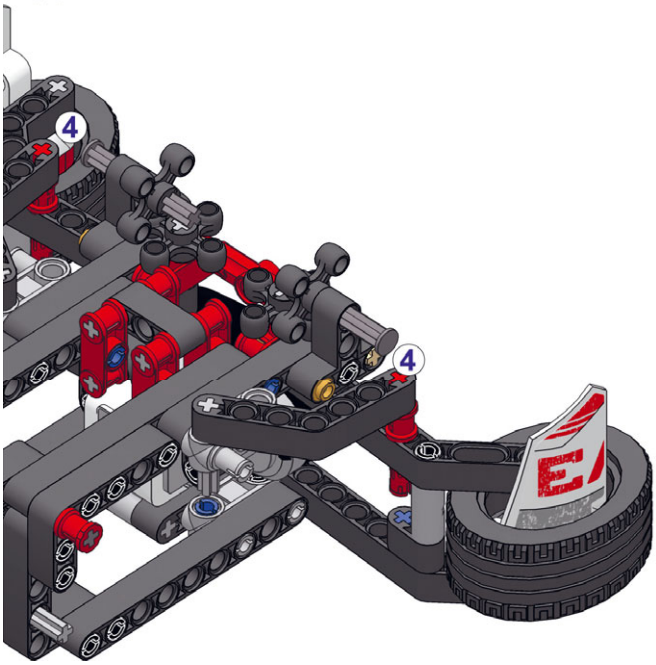




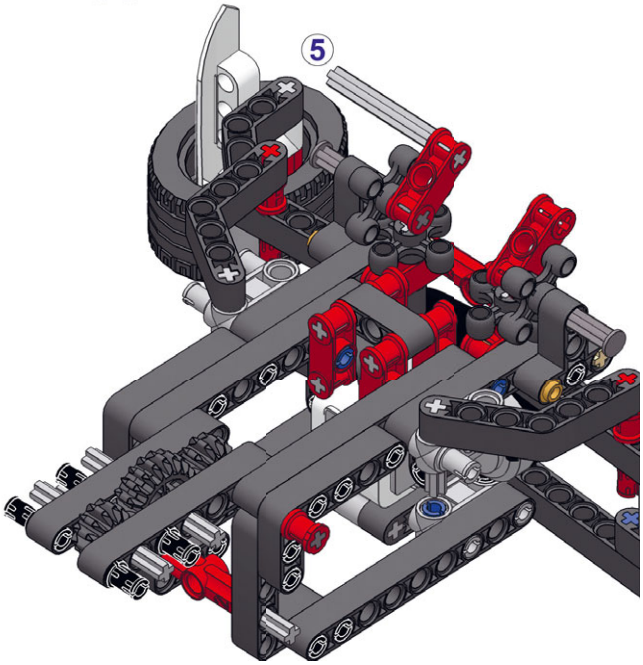
16

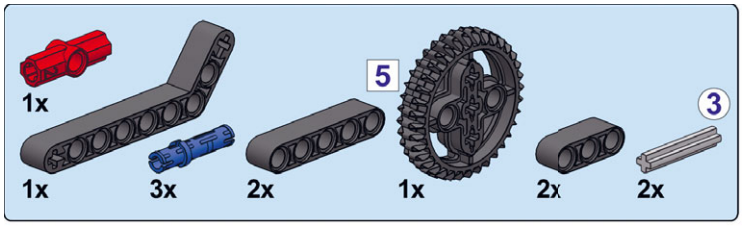
Эти вспомогательные детали удерживают клешню на месте, пока вы собираете манипулятор. Позже вы их снимете

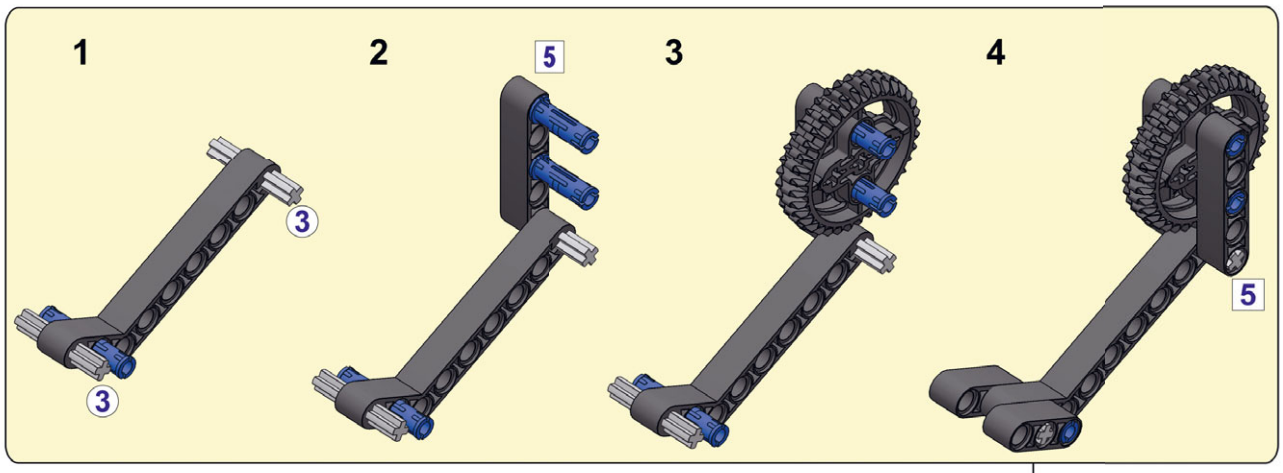
17



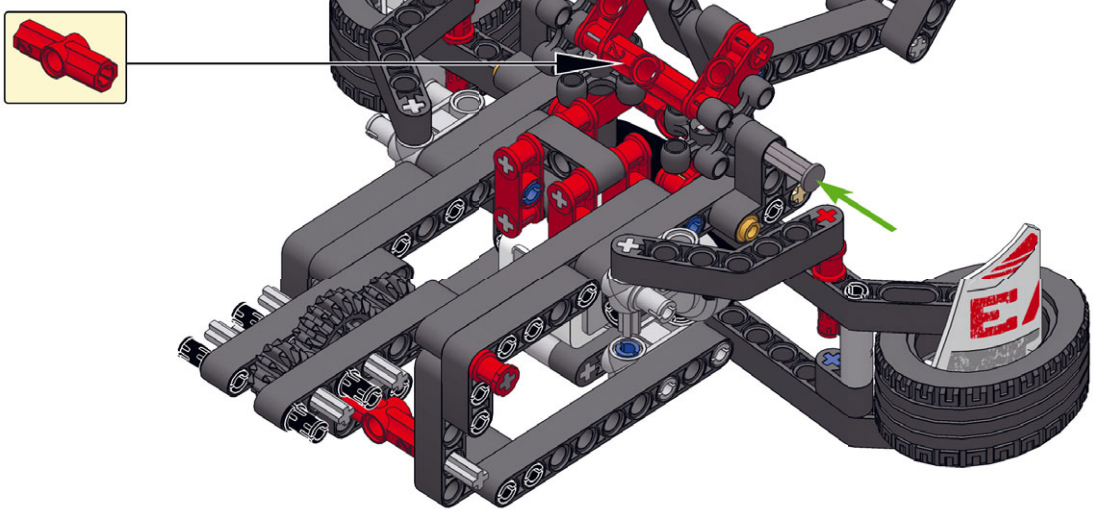
18

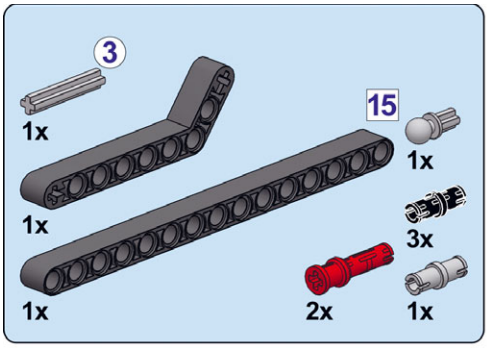


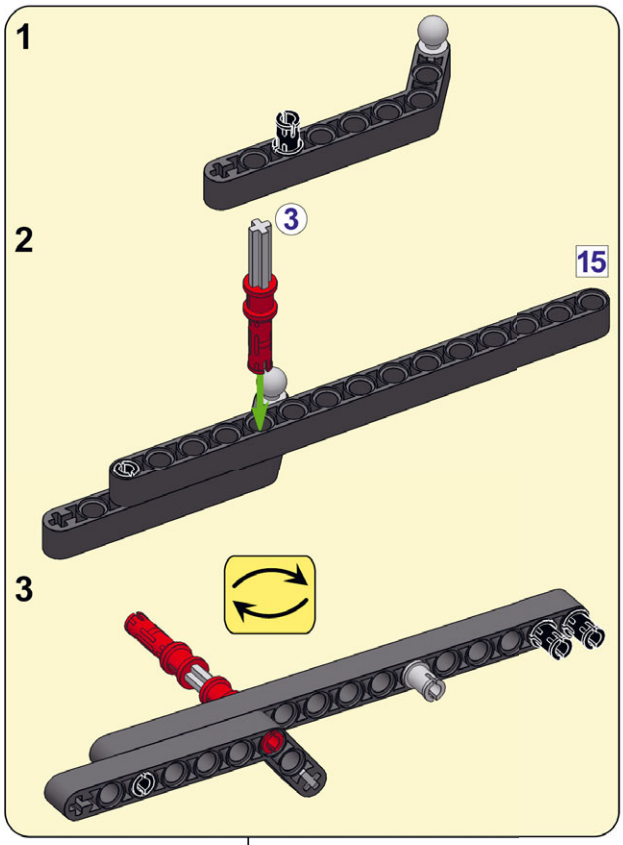




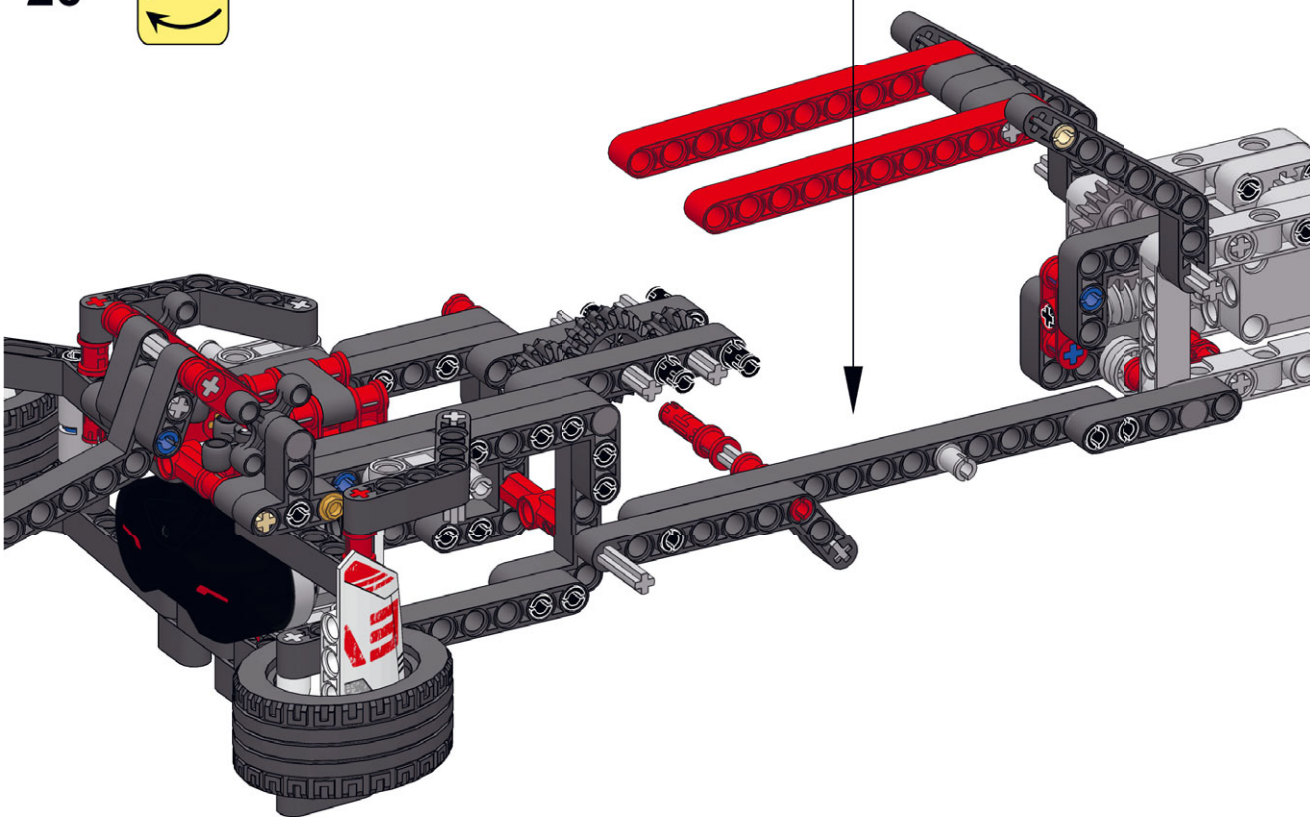
19

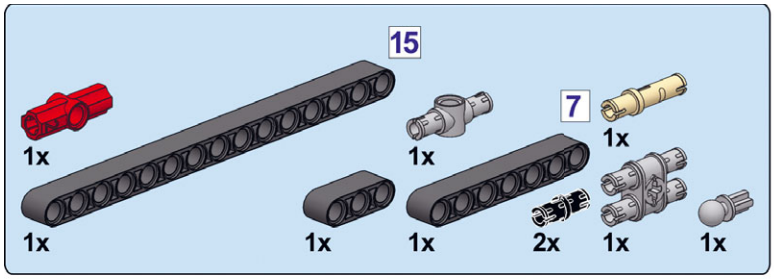


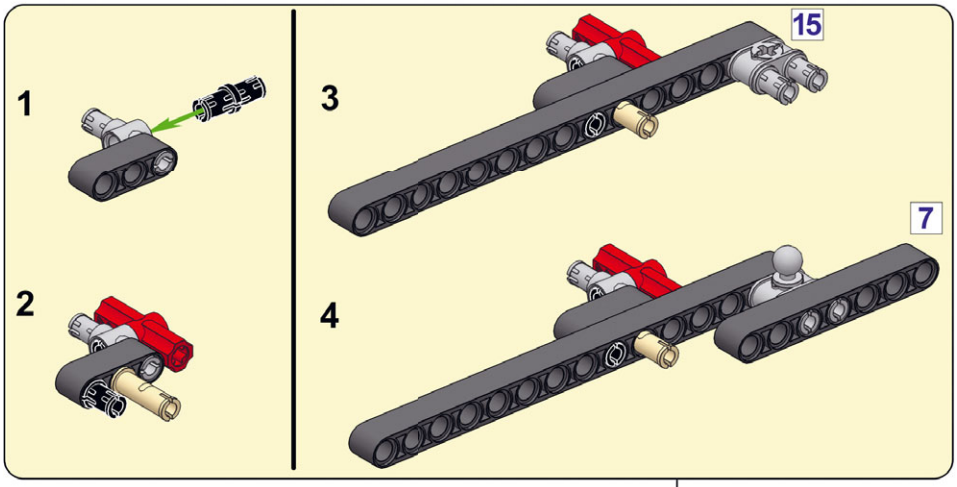




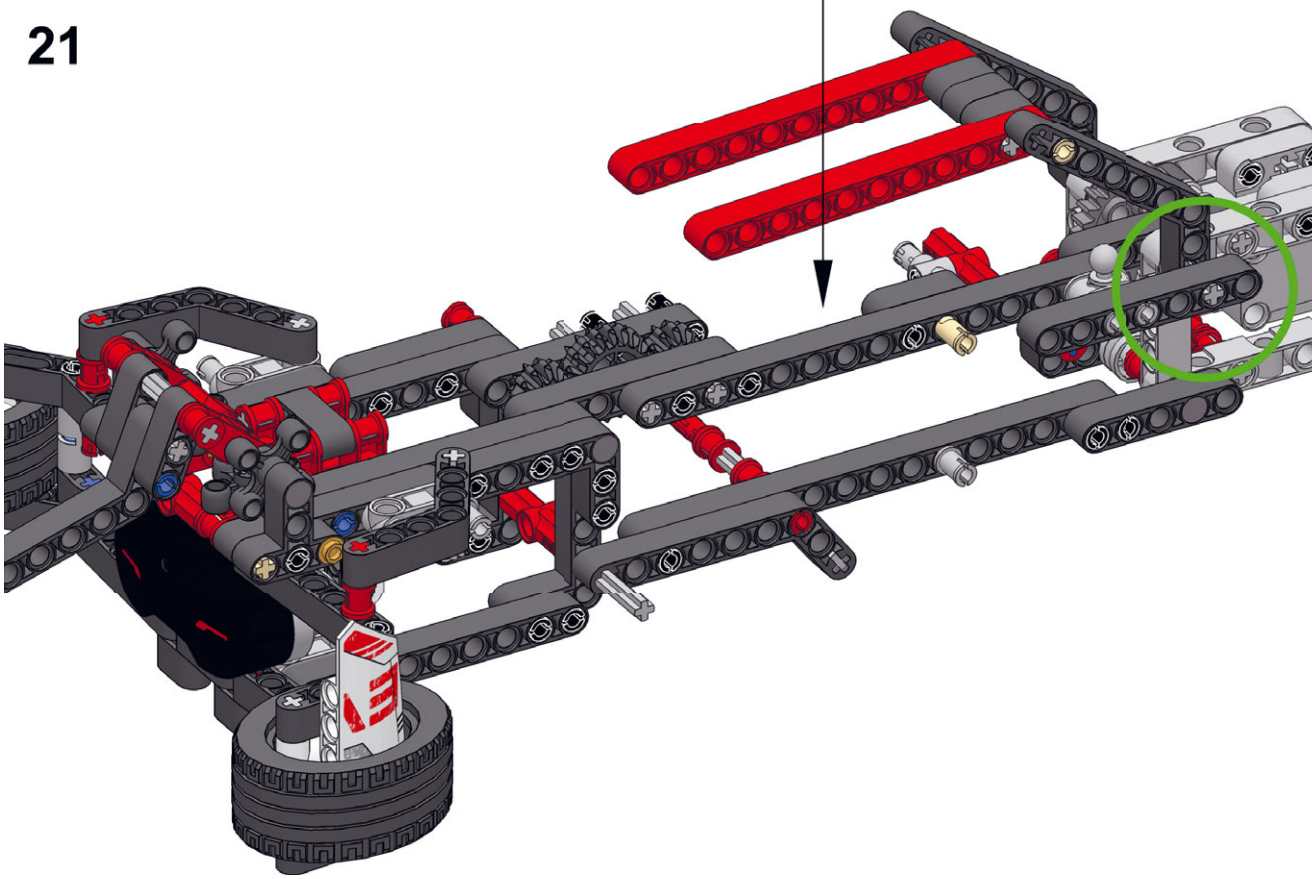
20

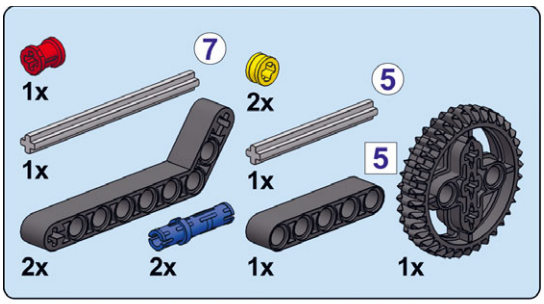


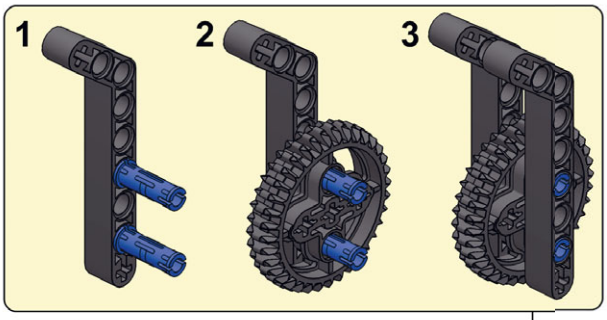




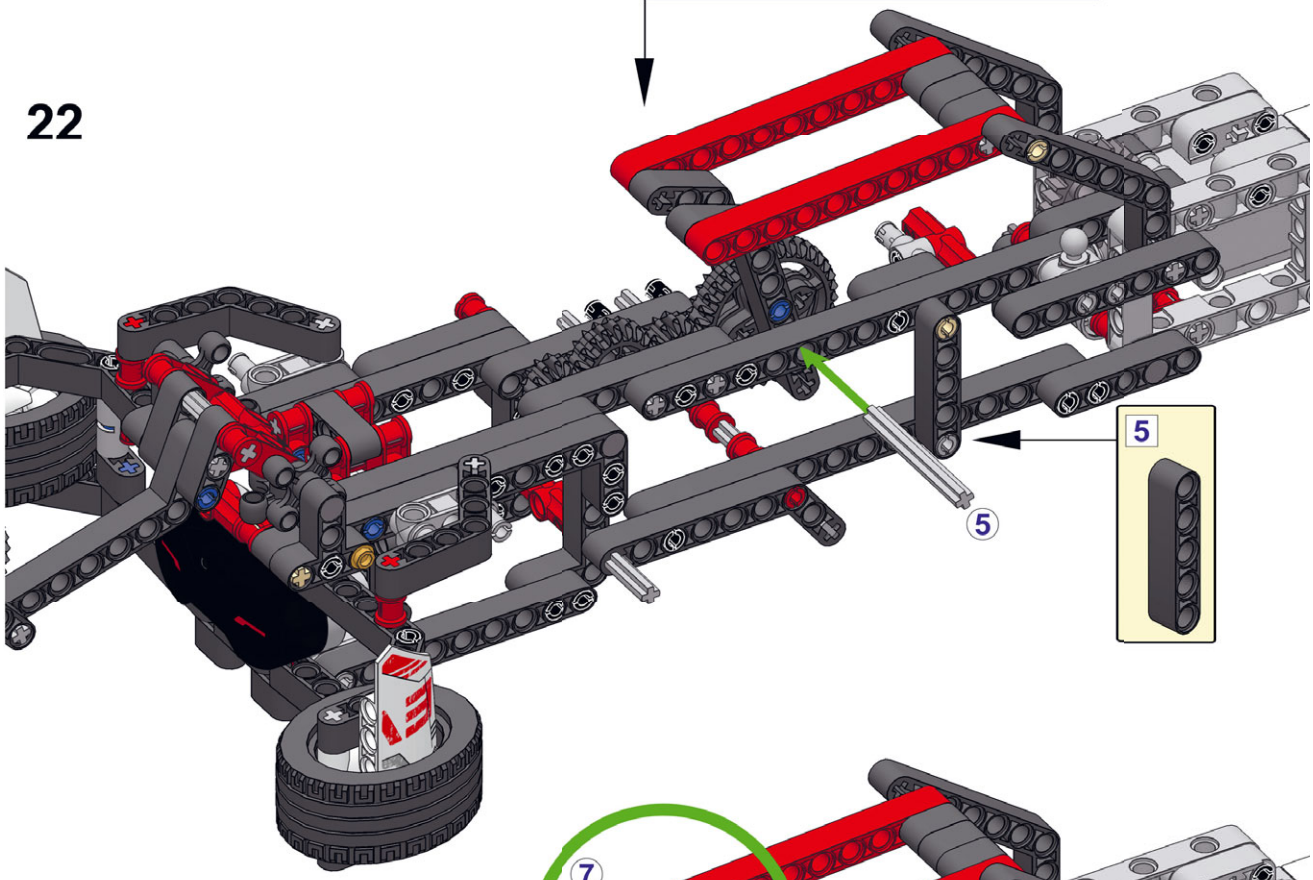
21



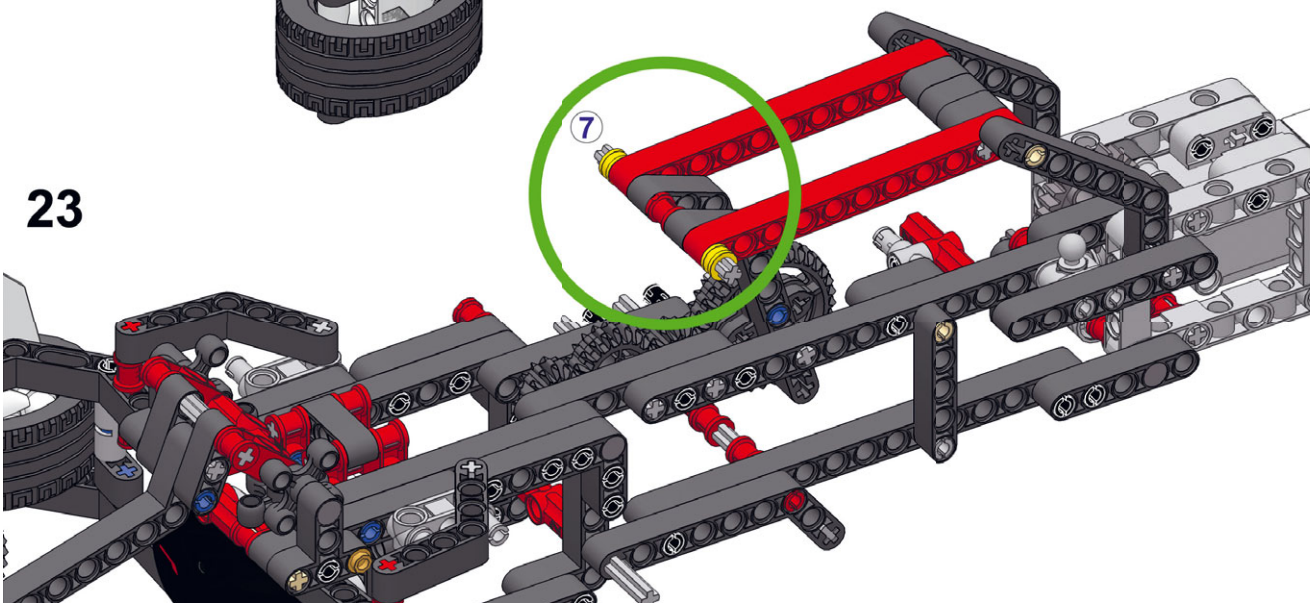


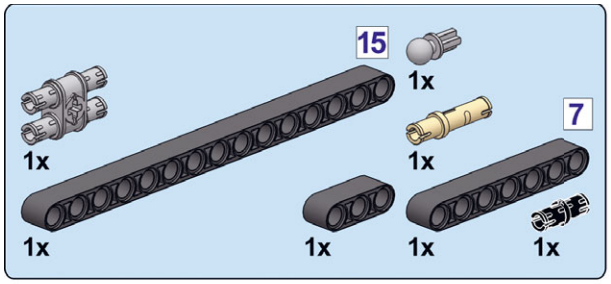


22

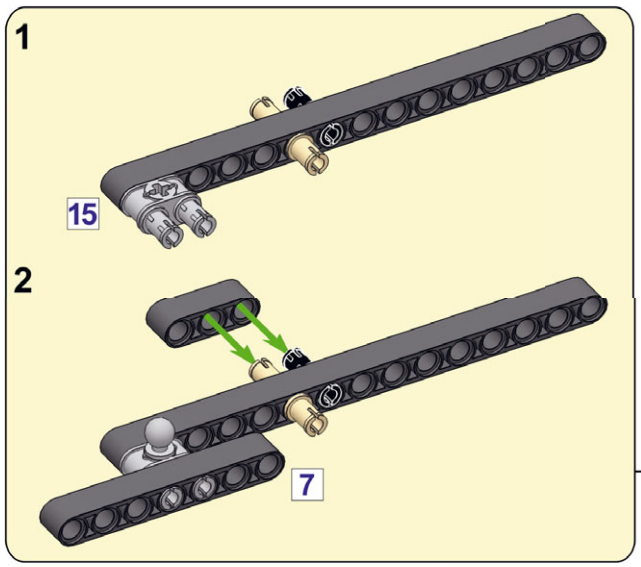


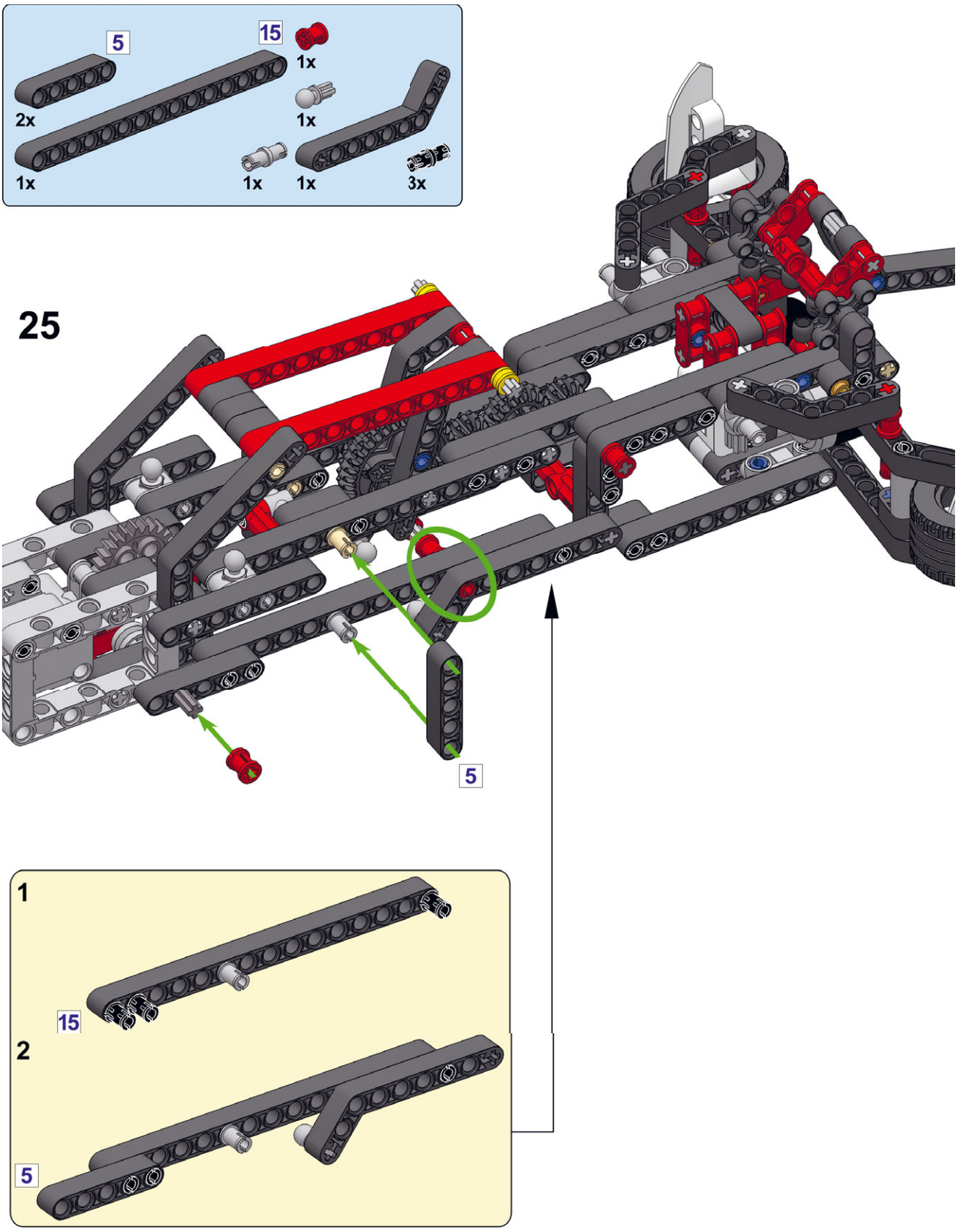
23

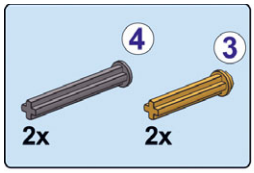


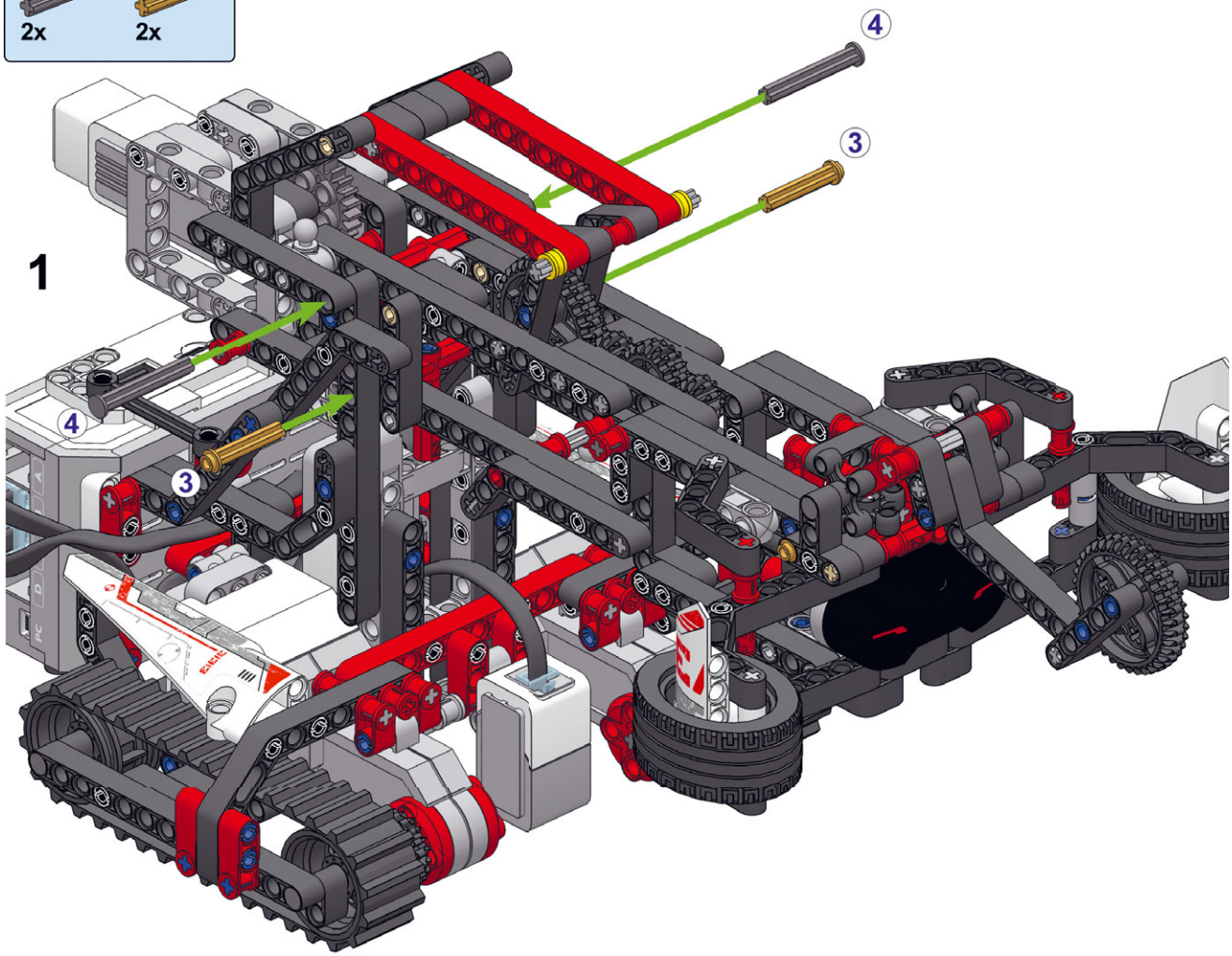

15
1x
1x
7
1x
1x
1x
1x
1x

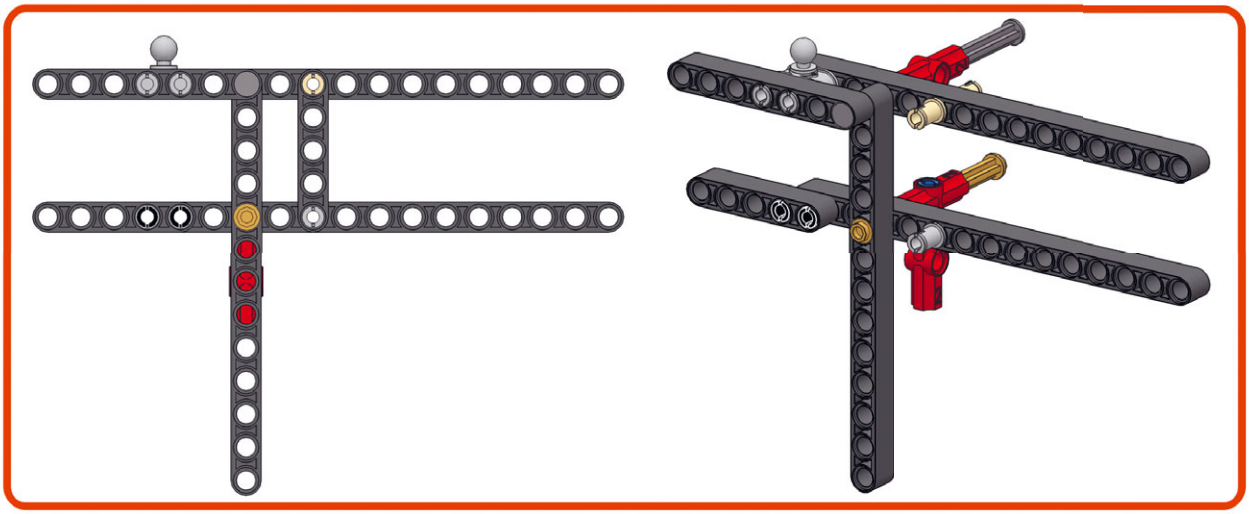
24


1
15
2
7

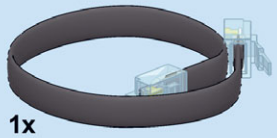

5
2x
1x
15
1x
1x
1x
1x
3x
25
5
1
15
2
5


4
3
2x
2x


1
4
3
4
3



Короткий / 25 см

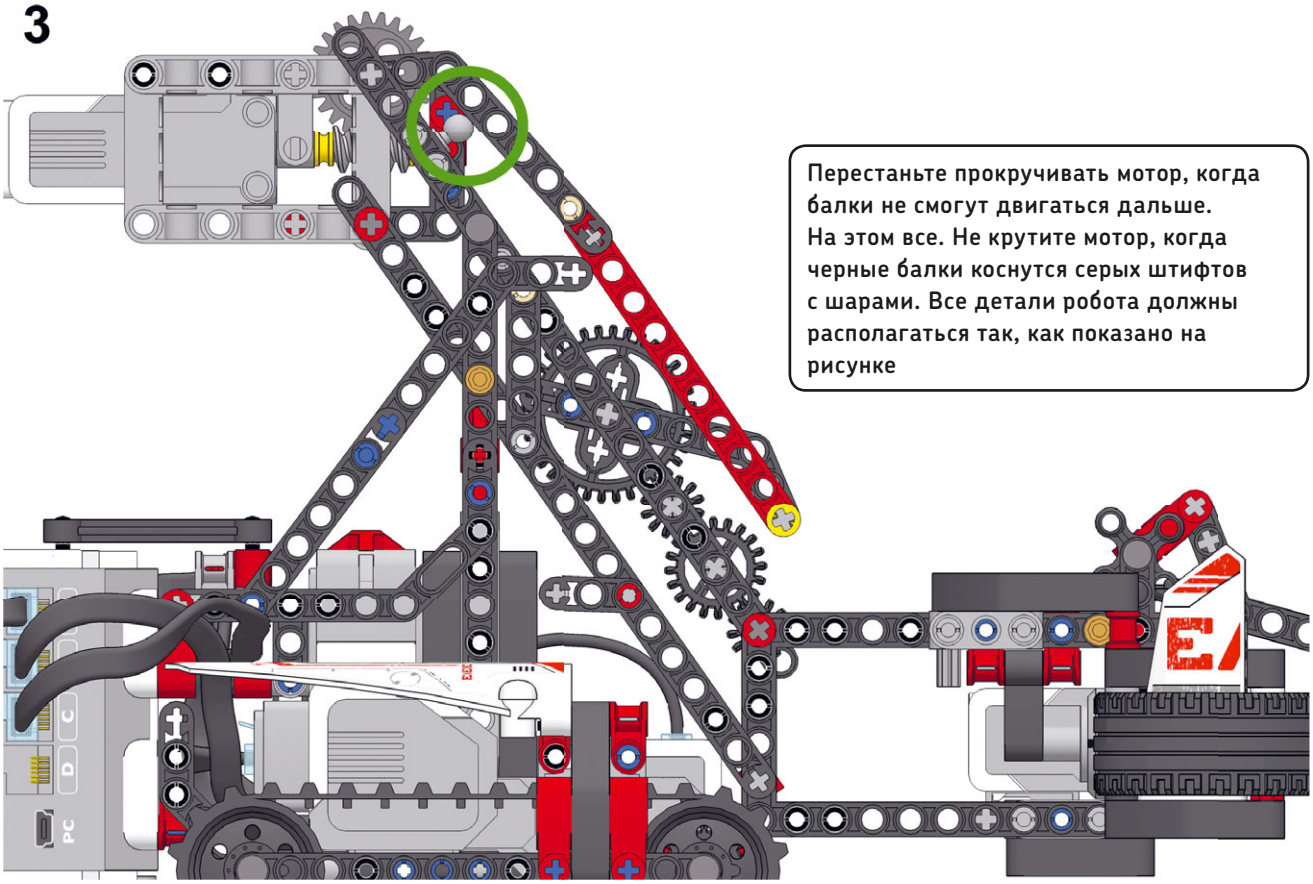


1x

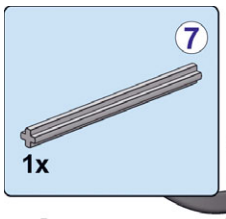
Включите модуль EV3, на вкладке **Приложения модуля** (Brick Apps) выберите приложение Motor Control и с помощью кнопки «Вниз» управляйте средним мотором таким образом, чтобы балки перемещались в направлении стрелки

2

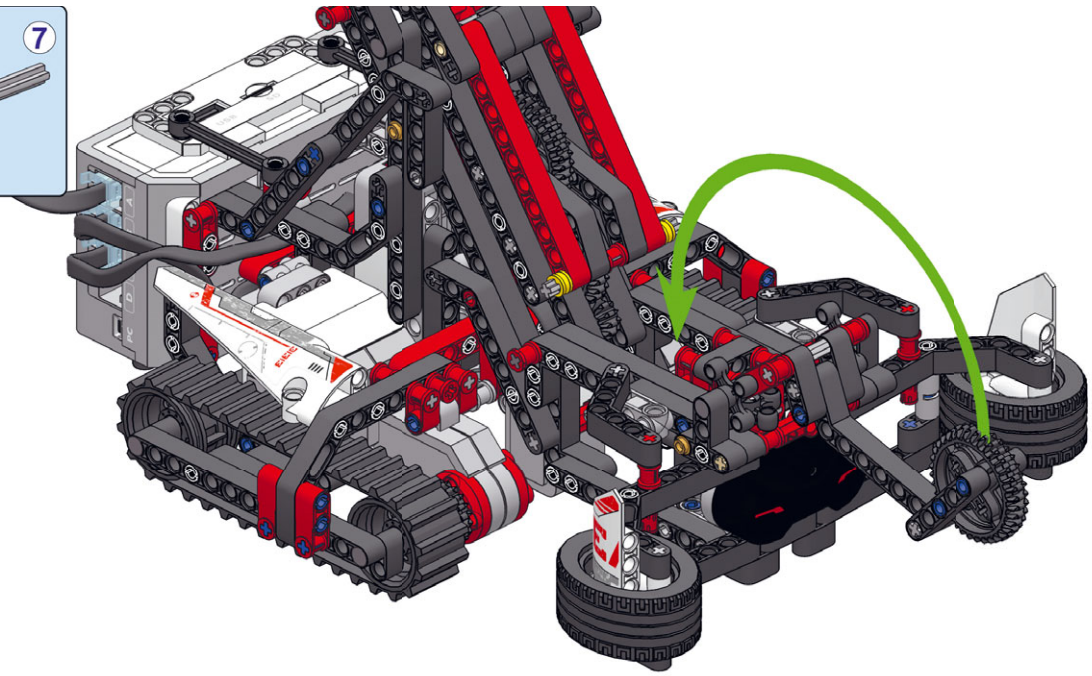
3



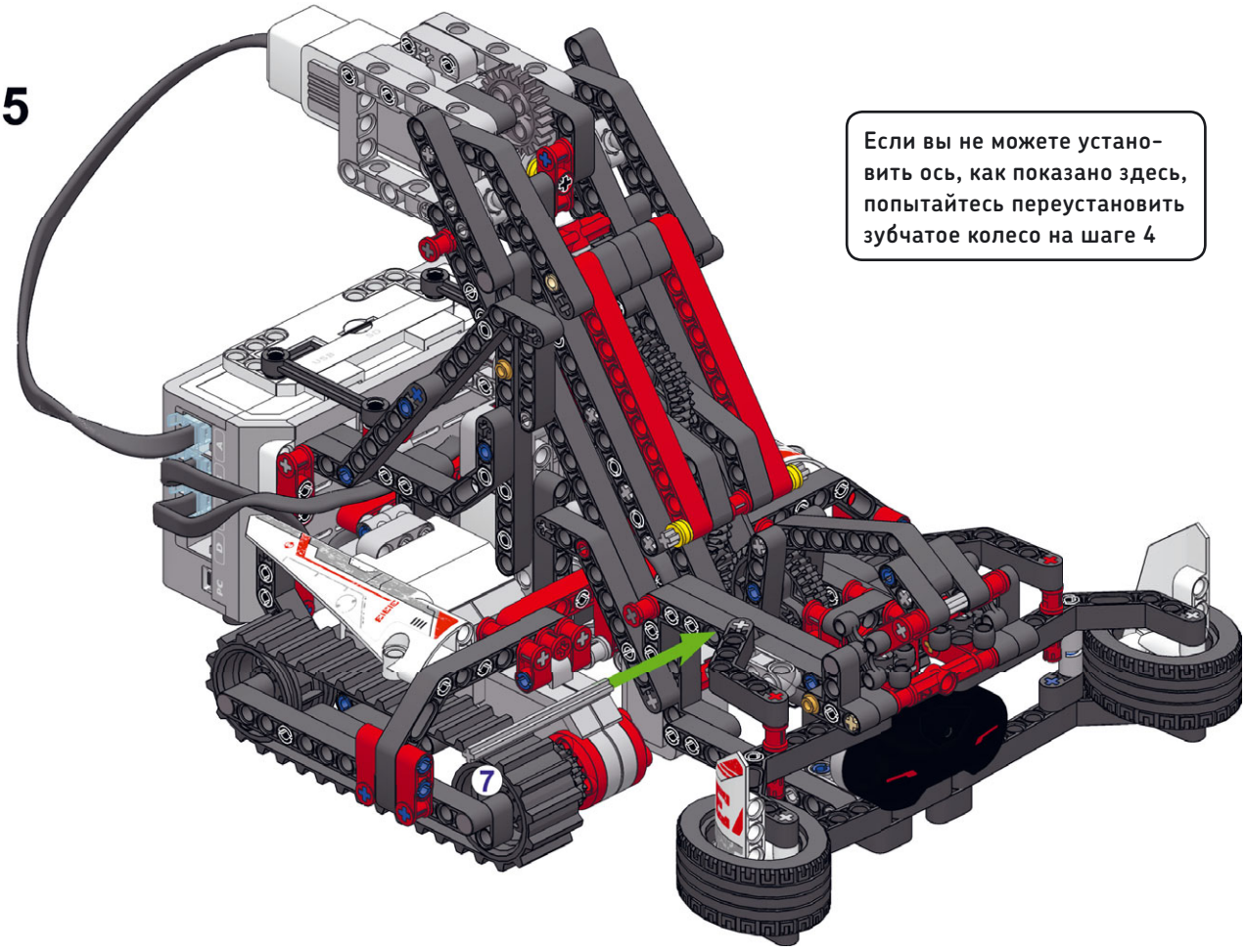
Перестаньте прокручивать мотор, когда балки не смогут двигаться дальше. На этом все. Не крутите мотор, когда черные балки коснутся серых штифтов с шарами. Все детали робота должны располагаться так, как показано на рисунке



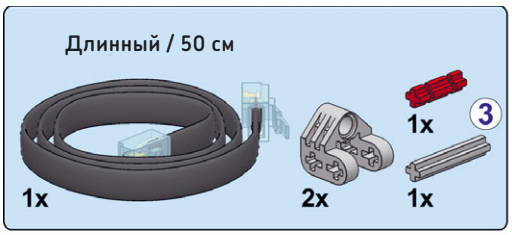
4



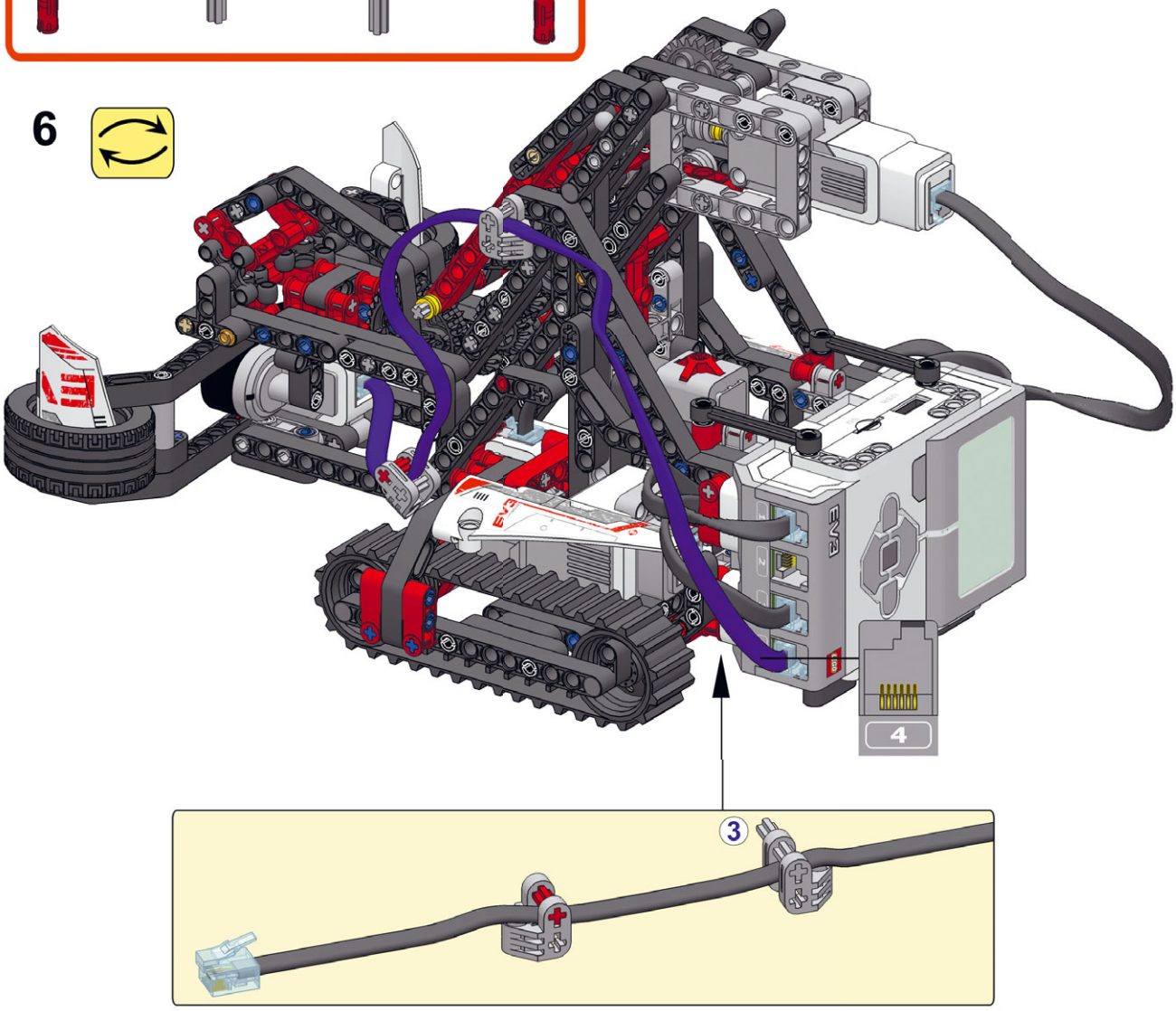
5



Если вы не можете установить ось, как показано здесь, попытайтесь переустановить зубчатое колесо на шаге 4


Длинный / 50 см
1x
2x
1x
1x
3


Уберите все вспомогательные
детали из клешни


6
4
3

Управление захватным устройством

Теперь, когда вы построили робота, можно протестировать его механические возможности, создав программу удаленного управления. Вы будете управлять роботом с помощью блоков **Рулевое управление** (Move Steering), а также создадите три контейнера «Мой блок» для управления захватным устройством.

Контейнер «Мой блок» № 1: Захват

Чтобы захватить предмет и поднять захватное устройство, средний мотор должен крутиться вперед до тех пор, пока не будет нажата кнопка датчика касания в основании робота. Вы ограничите скорость мотора до уровня 40%, чтобы сократить объем энергии, необходимой для приведения мотора в движение.

Создайте новый проект EV3 под названием SNATCH3R, поместите в область программирования три блока, как показано на рис. 18.5, а затем объедините их в контейнер «Мой блок» с именем **Grab**. Поскольку червячное колесо в механизме не позволяет мотору крутиться, когда мотор не подключен к питанию, если параметру **Тормозить в конце** (Brake at End) присвоено значение **Ложь** (False), мотор не будет двигаться, что позволит сэкономить заряд батареи.

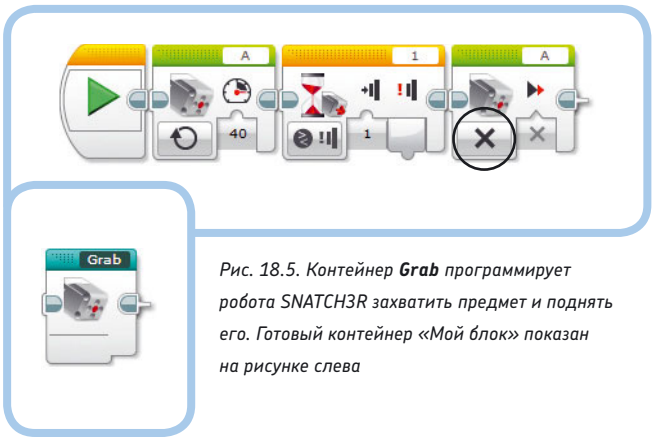


*Рис. 18.5. Контейнер **Grab** программирует робота SNATCH3R захватить предмет и поднять его. Готовый контейнер «Мой блок» показан на рисунке слева*

Контейнер «Мой блок» № 2: Исходное положение

Когда программа запускается, захватное устройство можно либо опустить (клешни будут открыты), либо поднять до верхнего предела (клешни будут закрыты, см. рис. 18.1), либо установить в любое промежуточное положение. Чтобы предотвратить повреждение механизма или мотора, захватное устройство должно оставаться в этих пределах, а для этого необходимо ограничить степень поворота мотора.

Верхнюю границу помогает контролировать датчик касания: если кнопка датчика нажата, мотор не будет двигаться дальше вперед. Нижнюю границу оберегает датчик вращения среднего мотора: если датчик фиксирует значение менее 0 градусов, мотор не будет двигаться дальше назад.

Чтобы все это осуществить на практике, до запуска программы необходимо убедиться, что датчик вращения фиксирует значение 0 градусов в тот момент, когда захватное устройство в опущенном положении. Для этого с помощью контейнера **Grab** поднимите захватное устройство до нажатия кнопки датчика касания, а затем опустите его с помощью блока **Средний мотор** (Medium Motor), настроенного таким образом, чтобы мотор совершил 14,2 оборота в обратную сторону. После этого сбросьте значение датчика касания до 0. Расстояние между верхней и нижней границами составляет 14,2 оборота только тогда, когда при подъеме захватного устройства клешни полностью закрыты, поэтому во время сброса значения в клешнях робота не должно быть никаких предметов.

Создайте контейнер «Мой блок» с именем **Reset**, ориентируясь на инструкции, приведенные на рис. 18.6. Ставьте этот блок в начало каждой программы для робота SNATCH3R.

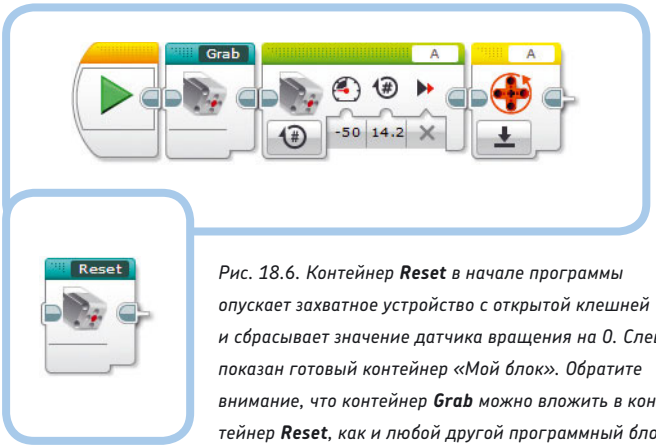


*Рис. 18.6. Контейнер **Reset** в начале программы опускает захватное устройство с открытой клешней и сбрасывает значение датчика вращения на 0. Слева показан готовый контейнер «Мой блок». Обратите внимание, что контейнер **Grab** можно вложить в контейнер **Reset**, как и любой другой программный блок*

Контейнер «Мой блок» № 3: Сброс

Чтобы опустить захватное устройство и высвободить предмет, средний мотор должен вращаться в обратную сторону, пока

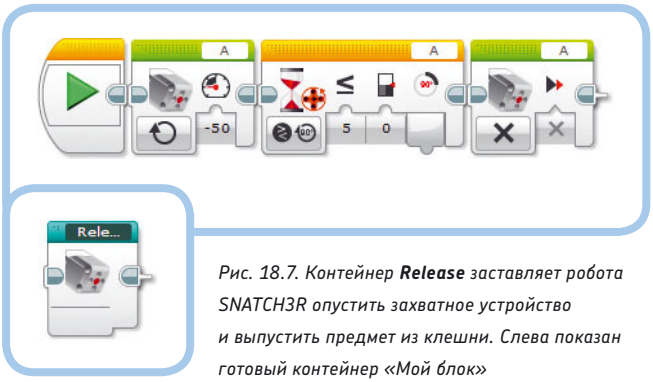


*Рис. 18.7. Контейнер **Release** заставляет робота SNATCH3R опустить захватное устройство и выпустить предмет из клешни. Слева показан готовый контейнер «Мой блок»*

захватное устройство не окажется внизу, или, иными словами, пока значение датчика вращения не станет равно 0 градусов. Создайте контейнер «Мой блок» с именем **Release**, как показано на рис. 18.7.

Разработка программы дистанционного управления

Теперь создадим и запустим программу RemoteControl, как показано на рис. 18.8. Пусть робот ездит, поднимает и перемещает предметы с помощью инфракрасного маяка

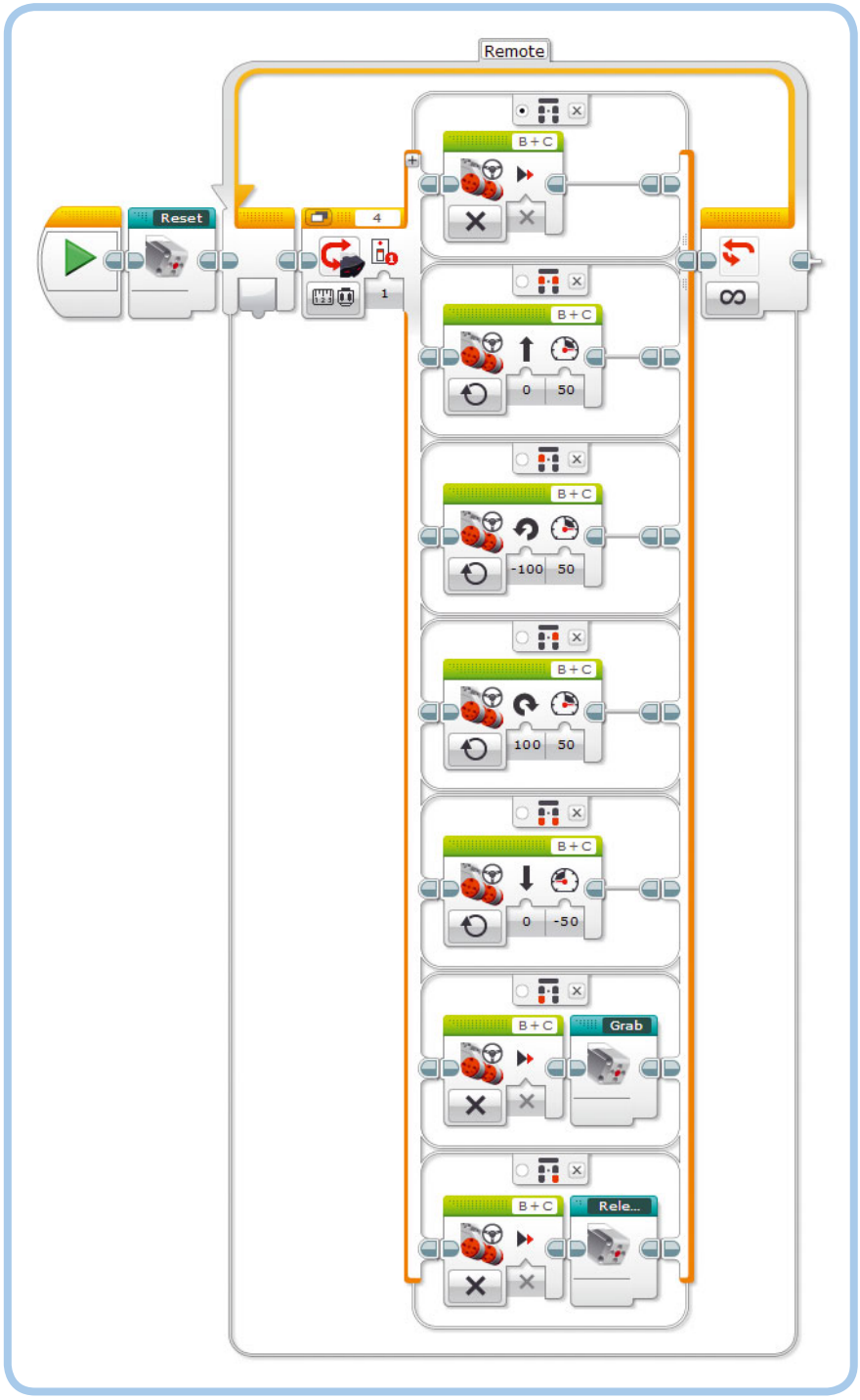


Рис. 18.8. Программа RemoteControl. По умолчанию, когда вы отпускаете кнопки, мотор останавливается, прекращая движение робота

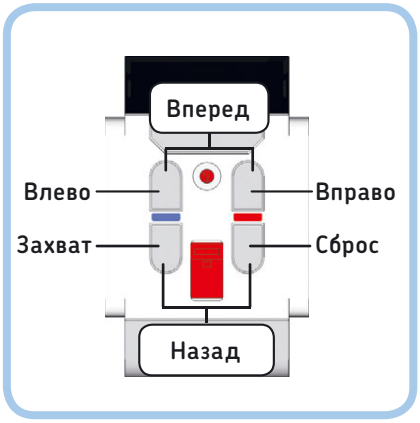


Рис. 18.9. Команды в программе RemoteControl

ПРАКТИКУМ № 117: УЛУЧШЕННОЕ ДИСТАНЦИОННОЕ УПРАВЛЕНИЕ!

Программирование: **Время:**

Программа RemoteControl отлично подходит для проверки функциональности робота SNATCH3R, но сама по себе она очень примитивная. Можете ли вы доработать эту программу, чтобы робот мог ездить в любом направлении, даже когда он захватывает или сбрасывает предмет?

СОВЕТ **Создайте две параллельные последовательности блоков, одна будет управлять движением робота через канал 1, а вторая — захватным устройством через канал 2. Так вы сможете управлять одновременно и движением, и захватом предметов, просто переключаясь между двумя каналами на маяке.**

ПРАКТИКУМ № 118: ДИСТАНЦИОННОЕ УПРАВЛЕНИЕ СКОРОСТЬЮ!

Программирование: **Время:**

Блоки **Рулевое управление** (Move Steering) в программе RemoteControl заставляют моторы крутиться со скоростью 50%. Однако иногда нужна более высокая скорость (75%), чтобы перемещаться на большие расстояния, а иногда нужно замедлиться (25%), чтобы расположить захватное устройство точно напротив предмета. Можете ли вы дополнить список команд для дистанционного управления, добавив возможность менять скорость моторов?

СОВЕТ **Определите числовую переменную с именем Speed и с ее помощью регулируйте скорость каждого блока Рулевое управление (Move Steering). Затем добавьте два дополнительных варианта в блок Переключатель (Switch) и вложите блоки, которые при каждом нажатии кнопки будут увеличивать или уменьшать значение переменной Speed на 10 пунктов.**

(см. рис. 18.9). Робот SNATCH3R должен уметь брать, поднимать и перемещать легкие предметы, например пустые банки из-под лимонада или бутылки для воды.

Проблемы с захватным устройством

Если при запуске программы RemoteControl вы столкнулись с какими-либо проблемами, связанными с захватным устройством робота SNATCH3R, необходимо все исправить, прежде чем вы сможете перейти к следующему этапу. Ниже приведен список наиболее часто возникающих проблем и их решений.

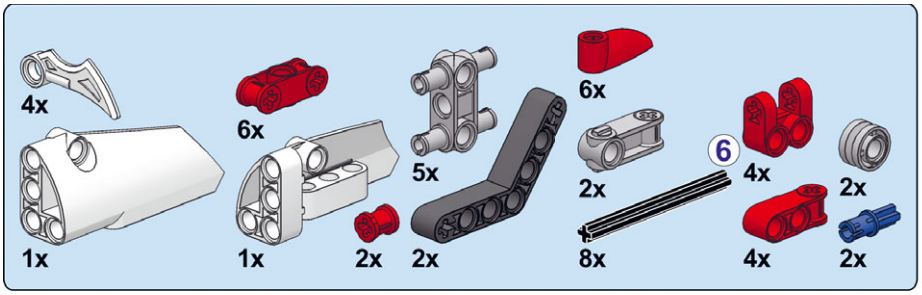
- *Захватное устройство не закрывает клешни перед подъемом.* Это может случиться, если в манипуляторе робота SNATCH3R неправильно установлены зубчатые колеса. Чтобы исправить эту ситуацию, попробуйте заново выполнить последние шаги сборки робота. Сначала снимите ось 7M (которую вы установили на с. 319) — для этого вытолкните ее с помощью другой оси. Затем заново установите вспомогательные детали на клешни (шаг 16 на с. 310). Затем продолжите сборку, начиная со с. 318, и снова протестируйте робота. Обязательно очень внимательно изучите вид механизма сбоку (с. 318) — каждая деталь должна быть установлена именно так, как показано на рисунке.
- *Кабель, подключенный к инфракрасному датчику, мешает клешням закрываться.* Это может случиться, если кабель препятствует вращению зубчатого колеса 36T в захватном устройстве. Чтобы посмотреть, так ли это, попробуйте убрать этот кабель и запустить контейнер **Reset**. Если теперь захватное устройство работает нормально, значит, проблема была в кабеле. Подсоедините кабель так, чтобы он не мешал зубчатым колесам.
- *Средний мотор установлен не параллельно полу.* Это происходит, когда захватное устройство неправильно прикреплено к движущему основанию робота. Чтобы это исправить, снимите оси, установленные на с. 317, и внимательно переустановите их заново в соответствии с инструкциями. Представленные на той странице боковые проекции очень точно отображают, в какие отверстия нужно установить эти оси (для большей наглядности некоторые детали отсутствуют на рисунке).

Поиск инфракрасного маяка

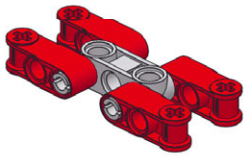
Теперь вы создадите программу, которая научит робота SNATCH3R искать, брать, поднимать и перемещать инфракрасный маяк. Каждое задание должно выполняться автономно, без вмешательства человека.

Сборка инфракрасного маячка

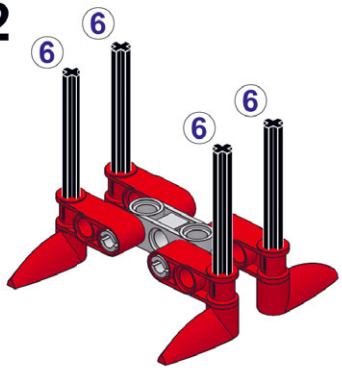
Прежде чем создавать программу, необходимо дополнить инфракрасный маяк несколькими деталями, чтобы роботу было проще захватывать и поднимать его. Выполните представленные на следующей странице инструкции, чтобы



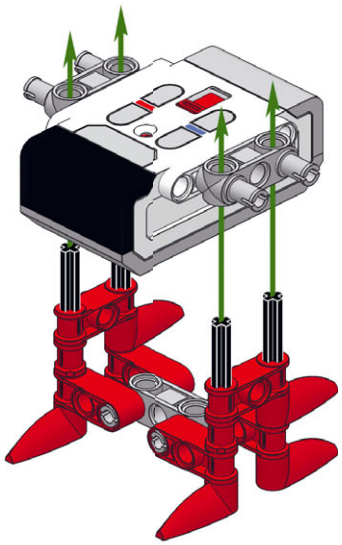
1



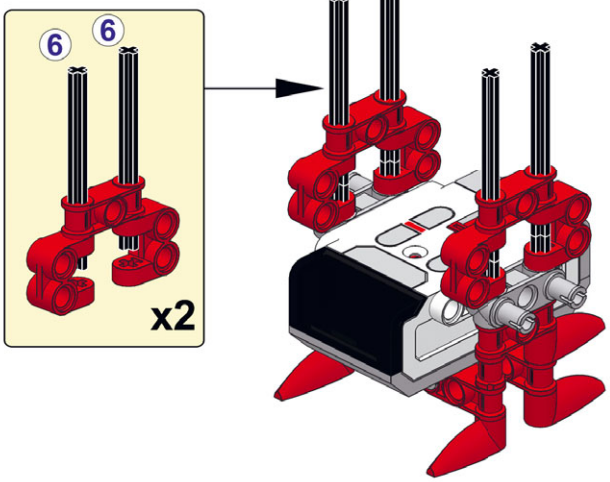
2



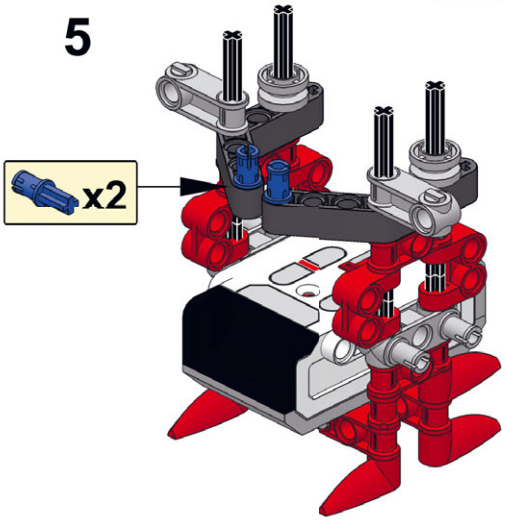
3



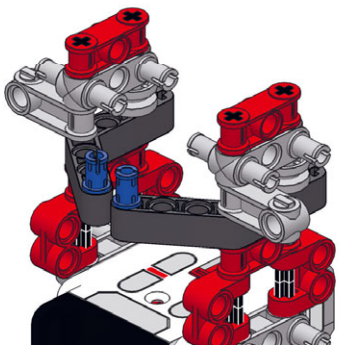
4



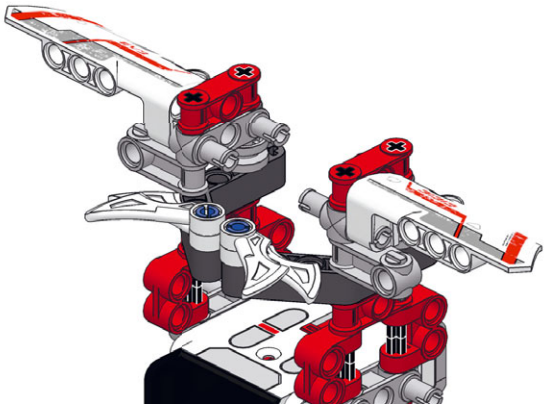
5



6



7



из оставшихся деталей набора MINDSTORMS EV3 построить инфракрасный маячок. Если вы при выборе деталей для своего робота пользовались списком на рис. 18.4, у вас должны были остаться детали для маячка.

Контейнер «Мой блок» № 4: Поиск

Вам уже известно, что робот может найти инфракрасный маяк, если поедет налево, когда фиксируется отрицательное значение параметра **Направление** (Heading) блока **Инфракрасный датчик** (Infrared Sensor), или направо, если это значение положительное. Теперь, когда вы умеете пользоваться шинами данных и переменными, вы можете разработать более сложную программу, с помощью которой робот будет искать маяк в радиусе 2 метров, даже если тот находится позади робота.

Вы создадите контейнер «Мой блок» с именем **Search**, с помощью которого робот будет изучать окружающую обстановку, совершая один полный поворот влево. Затем он поворачивается вправо до той точки, в которой увидел маяк, и направляет свои клешни в сторону маяка. В принципе, теперь робот может найти маяк, просто двигаясь прямо, но, поскольку показания датчика не настолько точны, вы создадите контейнер «Мой блок», который при необходимости упростит новый поиск маяка.

Показания датчика

Чтобы понять действие поискового алгоритма, нужно разобраться в тех данных, которые робот получает, совершая один поворот, как показано на рис. 18.10. Когда датчик повернут в направлении маяка, значение параметра **Направление** (Heading) (**H** на рисунке) равно или близко к нулю. Когда датчик расположен к маяку под углом около 90 градусов, значение параметра **Направление** (Heading) равно 25 или –25. Кроме того, датчик фиксирует значение 0, когда робот повернут к маяку своей тыльной стороной, поскольку детали, из которых собран робот SNATCH3R, закрывают маяк и датчик не может определить направление сигнала.

Чтобы найти маяк, мы должны ориентироваться на значение, близкое к 0, а не равное 0. Нужно игнорировать значение 0, поскольку оно может свидетельствовать о том, что маяк находится позади робота. В результате точность немного понизится, но, по крайней мере, вы будете знать, что робот не едет в противоположную от маяка сторону. (На завершающем этапе поиска вы больше не будете игнорировать значение 0.)

И наконец, не важно, положительное или отрицательное значение мы получили, поэтому мы можем ориентироваться только на абсолютную величину измерения. Например, –3 и 3 находятся на одинаковом расстоянии от 0. Подытожим, что нам нужно найти *наименьшую абсолютную величину*, не равную 0, и сохранить ее в памяти робота.

Робот должен знать не только наименьшее значение параметра **Направление** (Heading), но и в какой точке оно было зафиксировано. С помощью датчика вращения

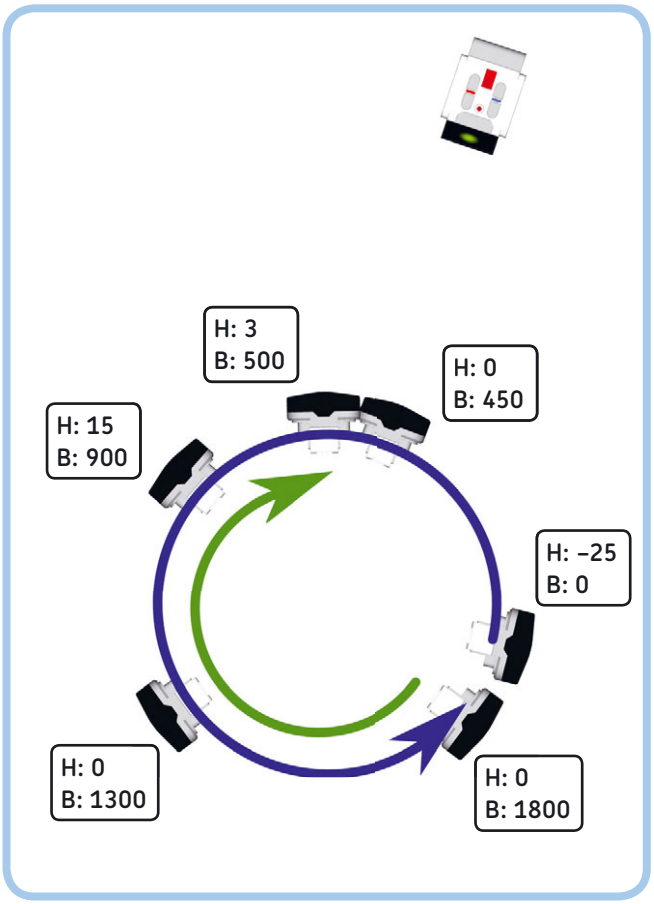


*Рис. 18.10. Когда робот поворачивается влево (синяя стрелка), инфракрасный датчик в режиме **Маяк** непрерывно фиксирует значение параметра **Направление**, но на диаграмме показано только шесть значений. Если не обращать внимания на нулевые значения, в данном случае наименьшей абсолютной величиной (**H**) будет 3. В этом положении значение датчика вращения, встроенного в мотор C (**B**), составляет 500 градусов. Когда робот совершает полный оборот вокруг своей оси (когда значение датчика вращения доходит до 1800 градусов), робот начинает поворачиваться вправо до тех пор, пока снова не достигнет отметки в 500 градусов (зеленая стрелка) и не окажется повернут (примерно) в сторону маяка*

в моторе C робот отслеживает свое положение, как показано на рис. 18.10. Значение датчика вращения (**B** на рисунке) в начальной точке равно 0, оно увеличивается по мере того, как робот поворачивается влево. Чтобы робот совершил полный поворот вокруг себя, каждый мотор должен повернуться примерно на 1800 градусов. Если робот будет помнить значение датчика вращения, полученное в той точке, в которой инфракрасный датчик зафиксировал наименьшее значение параметра **Направление** (Heading), он сможет вернуться в эту точку позже. Для этого ему нужно будет просто поворачиваться вправо до тех пор, пока мотор C не окажется в сохраненном в памяти положении (например, 500 градусов).

Алгоритм поиска

На схеме на рис. 18.11 показано, как робот может определить наименьшее достоверное значение параметра **Направление** (Heading) и соответствующее ему значение датчика вращения. На схеме переменная с именем **Reading** обозначает зафиксированное инфракрасным датчиком абсолютное значение параметра **Направление** (Heading), которое обновляется после каждой итерации цикла.

Чтобы не сохранять все значения и не выбирать из них наименьшее, программа сохраняет только одно из них — соответствующая переменная называется **Lowest**. Каждый раз, когда новая допустимая отметка оказывается меньше переменной **Lowest**, она занимает место последней, а значение датчика вращения сохраняется в виде переменной с именем **Position**. Таким образом, переменная **Lowest** содержит наименьшее зафиксированное значение инфракрасного датчика, а переменная **Position** — данные датчика вращения в той точке, в которой было зафиксировано это значение **Lowest**.

Создание и тестирование контейнера Search

Теперь указанную схему можно реализовать на практике. Для этого воспользуйтесь инструкциями с рис. 18.12–18.16. Чтобы разобраться с обозначенными на схеме двумя вопросами, необходимо создать два блока **Переключатель** (Switch) (на схеме отмечены как **a** и **b** соответственно).

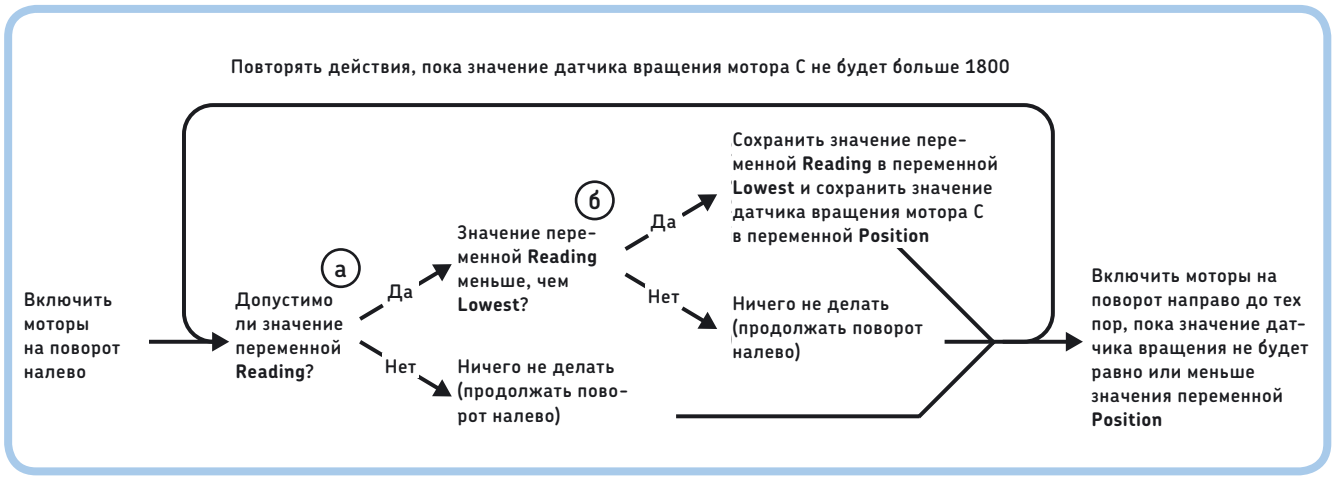


*Рис. 18.11. Алгоритм поиска. Переменная **Reading** — это абсолютное значение параметра **Направление**, которое считается допустимым, если оно не равно 0*

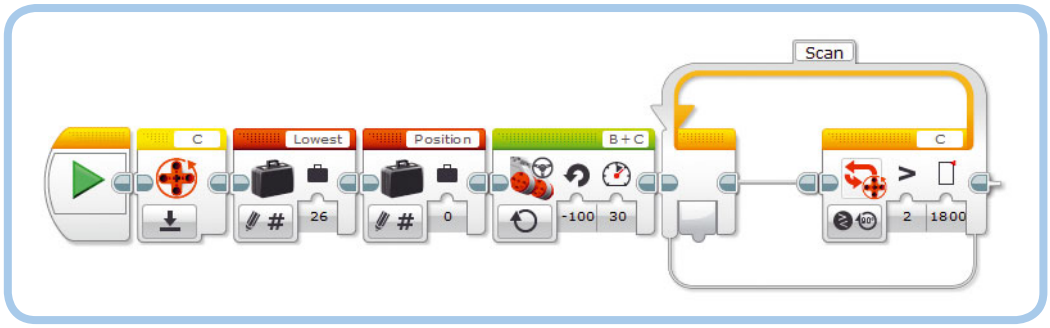


*Рис. 18.12. Шаг 1: определите две числовые переменные, назвав их **Lowest** и **Position**, и инициализируйте их, как показано на рисунке. Переменной **Lowest** присвойте значение **26**, тогда любое зафиксированное значение от 25 и ниже станет следующим наименьшим значением. Обнулите значение датчика вращения мотора C, включите моторы на поворот налево и добавьте блок **Цикл** для поиска маяка. **Цикл** будет длиться до тех пор, пока мотор C не повернется на 1800 градусов. При этом робот должен совершить поворот примерно на 360 градусов*

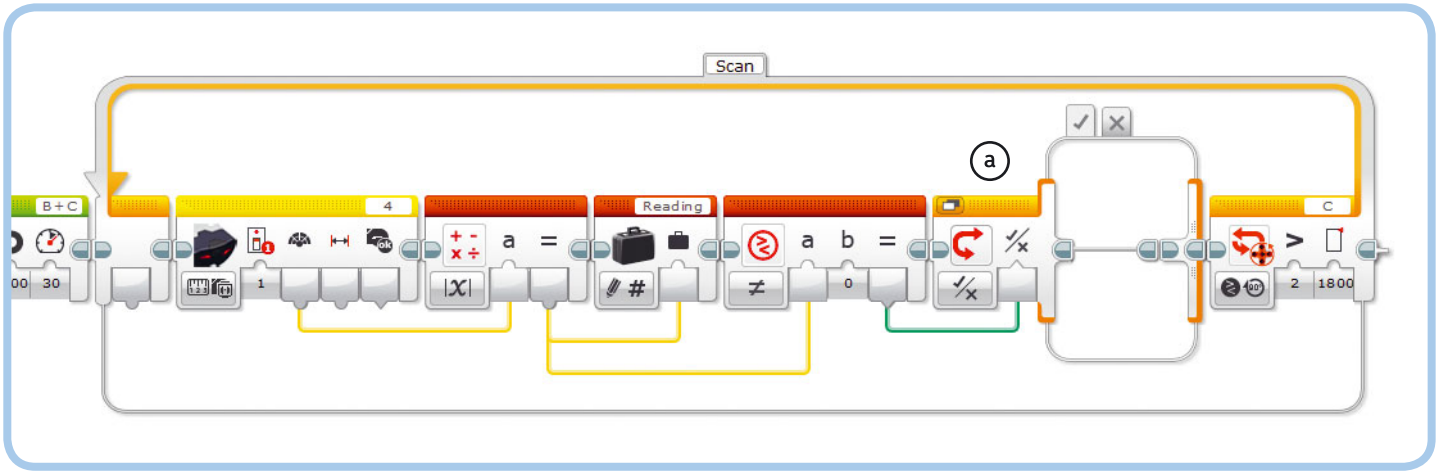


*Рис. 18.13. Шаг 2: теперь добавьте в этот цикл блоки, с помощью которых робот будет сохранять полученное значение, брать его абсолютную величину и определять, не равно ли оно нулю. Определите числовую переменную, назвав ее **Reading**, и настройте блоки, как показано на рисунке. С помощью блока **Сравнение** вы будете проверять, не равно ли значение нулю, и если это так, то будут выполняться блоки на вкладке **Истина** блока **Переключатель** (**a**). В случае если значение равно нулю, ничего происходить не должно, поэтому вкладка **Ложь** остается пустой*

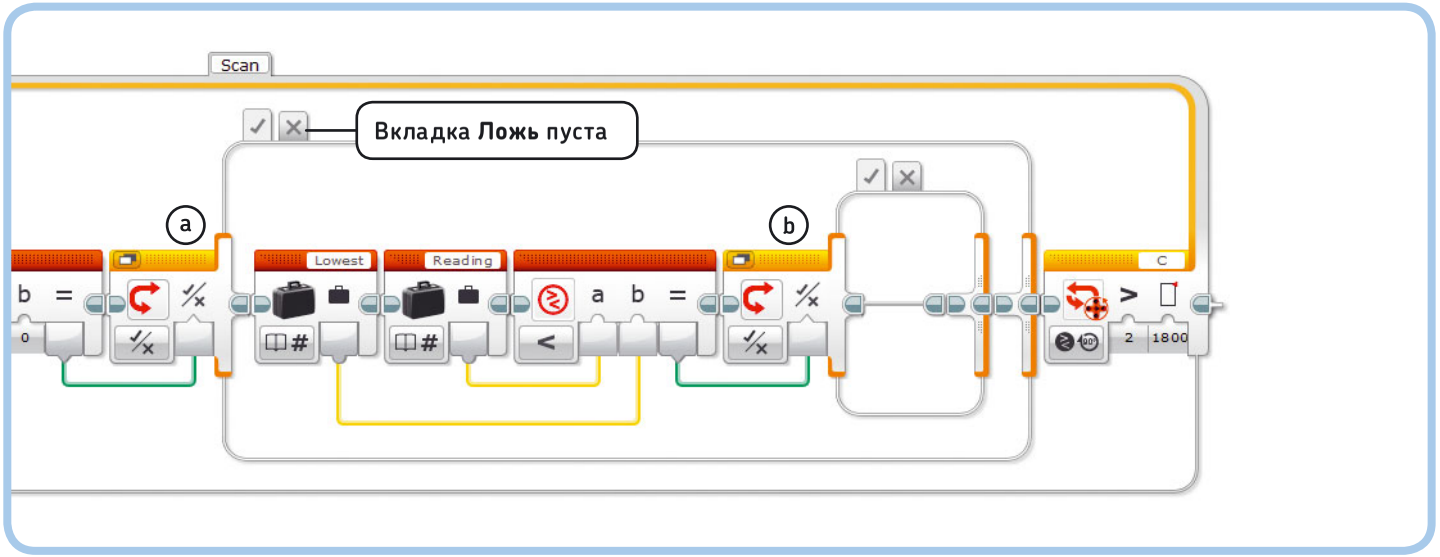


*Рис. 18.14. Шаг 3: теперь, когда у вас есть допустимое значение, можно сравнить его с наименьшим из ранее зафиксированных (**a**). Если значение переменной **Reading** меньше, чем **Lowest**, блок **Сравнение** выдаст результат **Истина**, и будут выполняться блоки на вкладке **Истина** блока **Переключатель** (**b**)*

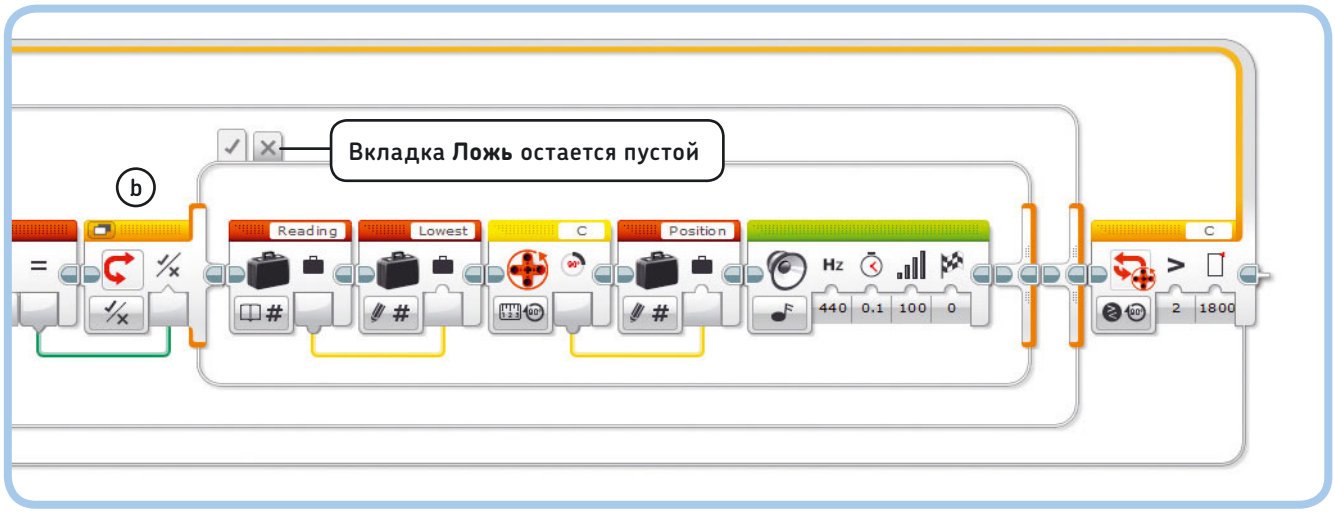


*Рис. 18.15. Шаг 4: теперь, когда вы знаете, что новое значение переменной **Reading** меньше, чем значение переменной **Lowest**, вы должны сохранить значение переменной **Reading** в переменной **Lowest**, а также сохранить текущее значение датчика вращения в переменной **Position**. Далее нужно добавить блок **Звук**, чтобы каждый раз, когда робот обновляет значения переменных **Lowest** и **Position**, раздавался звуковой сигнал. Если новое значение переменной **Reading** не меньше, чем значение переменной **Lowest**, ничего не должно происходить, поэтому вкладка **Ложь** остается пустой*

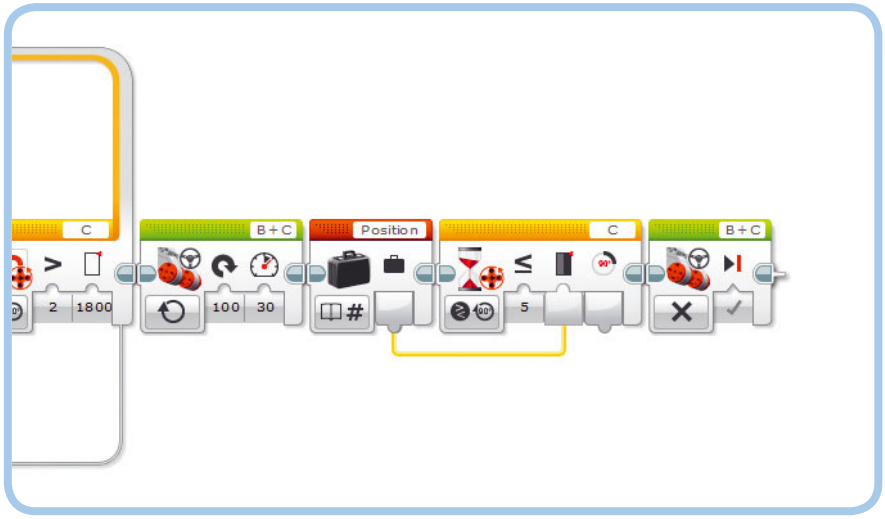


*Рис. 18.16. Шаг 5: совершив полный поворот вокруг своей оси в левую сторону, робот должен начать поворачиваться вправо до той точки, в которой было зафиксировано наименьшее значение датчика вращения (значение переменной **Position**). Моторы отключаются, и робот повернут в сторону маяка*

*Рис. 18.17. Шаг 6: объедините все блоки в контейнер «Мой блок» с именем **Search***

Выполнив указанные инструкции, объедините все блоки в контейнер «Мой блок» с именем **Search**, как показано на рис. 18.17.

Расположите инфракрасный маячок примерно на расстоянии 1 метр от робота. Нажмите кнопку в верхней части маяка (идентификатор 9), чтобы он непрерывно отсылал сигнал. В момент запуска программы робот может быть повернут в любую сторону, но инфракрасный маяк должен быть направлен непосредственно на робота, как показано на рис. 18.18. Если захватное устройство не приведено в исходную позицию, в которой его клешня находится в открытом положении, запустите созданный ранее контейнер **Reset**.

Далее запустите контейнер **Search**, чтобы протестировать его работу. Робот должен повернуться вокруг своей оси и издавать звуковой сигнал каждый раз, когда фиксирует более низкое значение, чем то, которое хранится в его памяти. Выполнив один полный поворот в левую сторону, он должен начать поворачиваться направо до тех пор, пока не окажется напротив маяка. Именно в этой точке вы должны услышать последний звуковой сигнал. Если роботу не удалось

Рис. 18.18. Расположите инфракрасный маячок примерно на расстоянии 1 метр от робота. Робот может быть повернут в любую сторону, но маяк должен быть направлен непосредственно на робота

обнаружить маяк с первой попытки, подвиньте инфракрасный маячок ближе к нему и снова запустите контейнер **Search**.

Создание итоговой программы

Благодаря итоговой программе робот будет искать инфракрасный маячок, подъезжать к нему, захватывать и поднимать его, переносить в другое место, а затем опускать и сбрасывать. Сначала создайте новую программу под названием Autonomous, а затем выполните дальнейшие инструкции.

Поиск маяка

Первая часть программы направлена на то, чтобы робот SNATCH3R нашел маяк (с помощью контейнера **Search**), а затем подъехал к нему. Робот будет искать маяк и двигаться в его сторону до тех пор, пока значение параметра **Приближение** (Proximity) не опустится ниже 50%, т.е. пока маяк не окажется близко (см. рис. 18.19).

Движение в сторону маяка

Когда маяк оказывается в зоне видимости, робот может двигаться в его сторону, регулируя направление своего движения с учетом значения параметра **Направление** (Heading). В данном случае, в отличие от главы 8, параметру **Рулевое управление** (Move Steering) не задано какое-либо определенное значение. Оно будет пропорционально параметру **Направление** (Heading). Чем левее расположен маяк, тем сильнее робот отклоняется влево; чем правее расположен маяк, тем сильнее робот отклоняется вправо.

Поскольку предыдущий цикл поиска заканчивался только тогда, когда значение параметра **Приближение** (Proximity) становилось меньше 50%, мы знаем, что робот хотя бы приблизительно направлен в сторону маяка. Следовательно, нулевые значения должны свидетельствовать о том, что маяк находится прямо напротив робота, и вам больше не

ПРАКТИКУМ № 119: ПРОВЕРКА СИГНАЛА!

Программирование: **Время:**

Если вы забудете включить инфракрасный маяк, робот не будет получать сигнал вообще. Попробуйте в конце контейнера **Search** добавить блоки, благодаря которым робот голосом будет сообщать об ошибке, если не сможет определить местонахождение маяка за целый поворот.

СОВЕТ **Если робот не обнаружит маяк во время выполнения программы, какое значение параметра Lowest будет храниться в его памяти в конце цикла? Ответ вы найдете на рис. 18.11 и 18.12.**

Рис. 18.19. Шаг 1: робот будет искать маяк, пока не приблизится к нему

нужно игнорировать их. По сути, поскольку рулевое управление пропорционально направлению, рулевое управление будет равно 0, если направление будет равно 0, и робот будет двигаться прямо вперед.

Робот продолжит подстраивать значение параметра **Рулевое управление** (Move Steering), пока значение параметра **Направление** (Heading) не станет равным 1%. При этом инфракрасный маячок будет находиться примерно между клешнями робота, как показано на рис. 18.20. Протестируйте эту часть программы, выбрав только блок **Цикл** (Loop) и нажав кнопку **Запустить выбранное** (Run Selected).

Подъем и перемещение инфракрасного маячка

Робот почти готов взять маяк, но сначала он продвигается еще на один оборот вперед, чтобы инфракрасный маячок оказался точно между клешнями. Когда захватное устройство поднято с помощью контейнера **Grab**, робот поворачивается вокруг себя, проезжает немного вперед, а затем опускает и сбрасывает предмет в другом месте с помощью контейнера **Release**, как показано на рис. 18.21. Когда все будет готово, запустите программу, чтобы посмотреть, как робот SNATCH3R самостоятельно может найти инфракрасный маячок.

Дальнейшее изучение

Вы собрали одного из самых сложных роботов в книге. Поздравляю! В этой главе вы увидели, как можно объединить множество методов сложного конструирования и программирования, чтобы получить по-настоящему автономно функционирующего робота. Теперь, когда вы собрали робота SNATCH3R и разработали для него программу, посмотрите, что еще можно с ним сделать. Для начала ознакомьтесь со следующими практикумами и закрепите свои навыки в конструировании роботов.

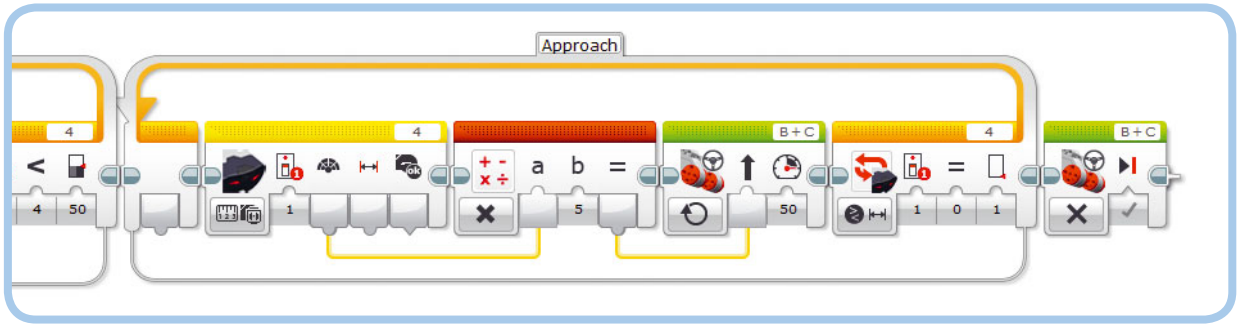


*Рис. 18.20. Шаг 2: робот движется вперед, подстраивая значение параметра **Рулевое управление** в зависимости от параметра **Направление**, пока маяк не окажется между его клешней*

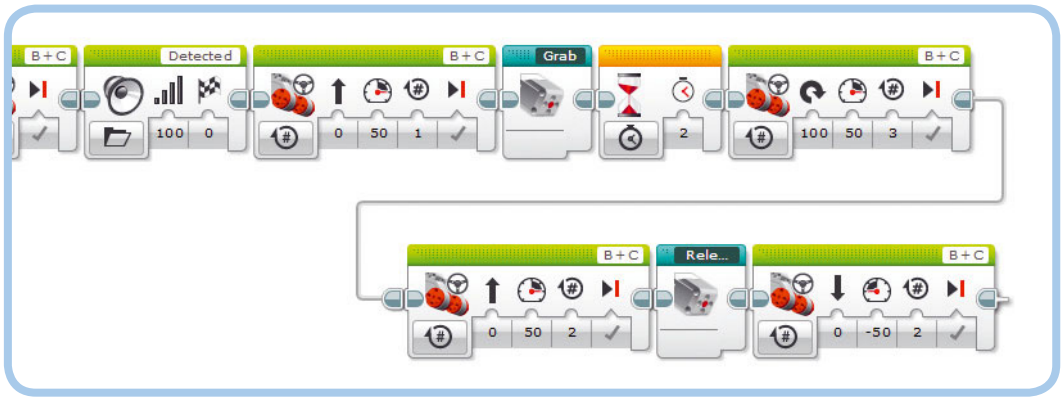


Рис. 18.21. Шаг 3: робот захватывает, поднимает и перемещает инфракрасный маячок

ПРАКТИКУМ № 120: РАБОТА БЕЗ ОСТАНОВКИ!

Сложность: **Время:**

Можете ли вы дополнить программу Autonomous так, чтобы робот SNATCH3R постоянно находил и перемещал инфракрасный маячок? Пусть робот отъезжает от объекта после того, как сбросил его, ищет его снова и так далее. Если робот SNATCH3R не может нормально поставить инфракрасный маячок, прежде чем отъехать от него, запрограммируйте робота так, чтобы он ехал назад зигзагами, вытряхивая объект из клешней.

СОВЕТ **Чтобы управлять количеством поворотов, совершаемых роботом после того, как он взял предмет, применяйте блоки Случайное значение (Random). Так робот каждый раз будет оставлять предмет в новом месте.**

ПРАКТИКУМ № 121: ПОИСК ПУТИ!

Сложность: **Время:**

Датчик цвета в основании робота SNATCH3R позволяет ему различать цвета поверхности, по которой он перемещается, и двигаться по линиям. Попробуйте запрограммировать робота ехать по определенному маршруту, захватить объект в конце линии, а затем вернуться к ее началу.

СОВЕТ **Тестовая трасса, которую вы сделали в главе 7 (см. рис. 7.4), вряд ли подойдет в данном случае, поскольку гусеницы робота SNATCH3R могут порвать мягкую бумагу. Чтобы справиться с этой задачей, наклейте трассу на плотный картон или сделайте новую на куске фанеры (черным скотчем или маркером). Или же можно прибегнуть к помощи коробки из-под набора, которая разворачивается в полноценную трассу, и уже на ее основе создать свою.**

ПРАКТИКУМ № 122: ОПРЕДЕЛИТЕЛЬ ПРИБЛИЖЕНИЯ!

Сложность: **Время:**

Попробуйте запрограммировать робот SNATCH3R так, чтобы он самостоятельно искал не только инфракрасный маяк, но и другие предметы, например бутылку из-под воды. Чтобы определить ближайший к роботу объект, ориентируйтесь на значение параметра **Приближение** (Proximity). Затем робот должен подъехать к предмету и взять его.

СОВЕТ **Сначала создайте и протестируйте модифицированную версию контейнера Search. В каком режиме должен работать инфракрасный датчик? Каким должно быть начальное значение переменной Lowest?**

СДЕЛАЙ САМ № 29: ЭКСКАВАТОР!

Сборка: **Программирование:**

Получится ли у вас построить робот-экскаватор? Снимите с робота SNATCH3R манипулятор, оставив только основание (см. с. 301). Для управления манипулятором и ковшом используйте средний мотор.

19

LAVA R3X: шагающий и болтающий гуманоид

Ранее в книге вы собирали роботов, представляющих собой транспортные средства, животных и механизмы, но, наверное, самый классный робот, которого можно собрать из набора MINDSTORMS EV3, — это гуманоид с двумя ногами. В этой главе вы сконструируете и запрограммируете робота LAVA R3X, показанного на рис. 19.1. Робот LAVA R3X ходит на двух ногах, управляемых большими моторами, и может поворачивать голову и двигать руками с помощью среднего мотора.

Когда робот научится ходить, вы должны будете применить все методы, с которыми познакомились в этой книге, дополнив программу таким образом, чтобы сделать робота интерактивным и реалистичным. Робот LAVA R3X может ходить, беспрерывно перемещая свой вес с одной ноги на другую и одновременно передвигая свободную ногу вперед. Механизм каждой ноги преобразует безостановочное движение мотора вперед в попеременное движение стопы вперед-назад, а также попеременное движение голеностопа вправо-влево (рис. 19.2).

Чтобы робот мог уверенно ходить, механизмы обеих ног должны находиться точно друг напротив друга, а моторы — вращаться с одинаковой скоростью. Когда эти требования выполняются, одна стопа оказывается под

Рис. 19.1. Робот LAVA R3X ходит на двух ногах, двигая при этом головой и руками. Он приветствует вас, когда вы жмете ему руку

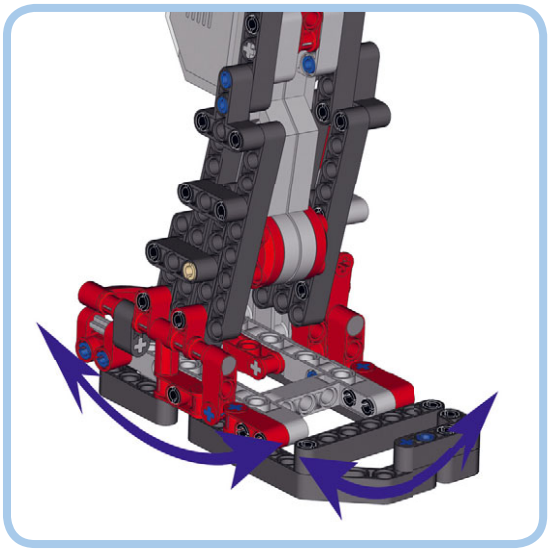


Рис. 19.2. Когда мотор каждой ноги выполняет один полный оборот, стопа движется назад-вперед, а голеностоп — влево-вправо

наклоном и не касается пола, при этом двигаясь вперед, а вторая стопа наклонена так, что несет весь вес робота, при этом отталкиваясь назад и в результате толкая робота вперед.

Пока мотор каждой ноги выполняет один оборот, механизм движется влево, вправо, вперед и назад, после чего возвращается в исходную позицию, сделав один шаг.

С помощью датчика касания, расположенного между ногами робота, вы установите механизмы точно друг напротив друга, как было с моторами робота ANTY, которого вы собирали в главе 13. Еще вы создадите регулятор скорости, чтобы оба мотора вращались с одинаковой скоростью.

Сборка ног

Прежде всего нужно собрать ноги робота и создать контейнеры «Мой блок», с помощью которых робот будет ходить и поворачиваться. Далее вы соберете верхнюю часть туловища и напишете программу, с помощью которой робот будет взаимодействовать с тем, что его окружает, с помощью датчиков. Возьмите необходимые для сборки робота детали (они показаны на рис. 19.3), а затем выполните инструкции на следующих страницах.

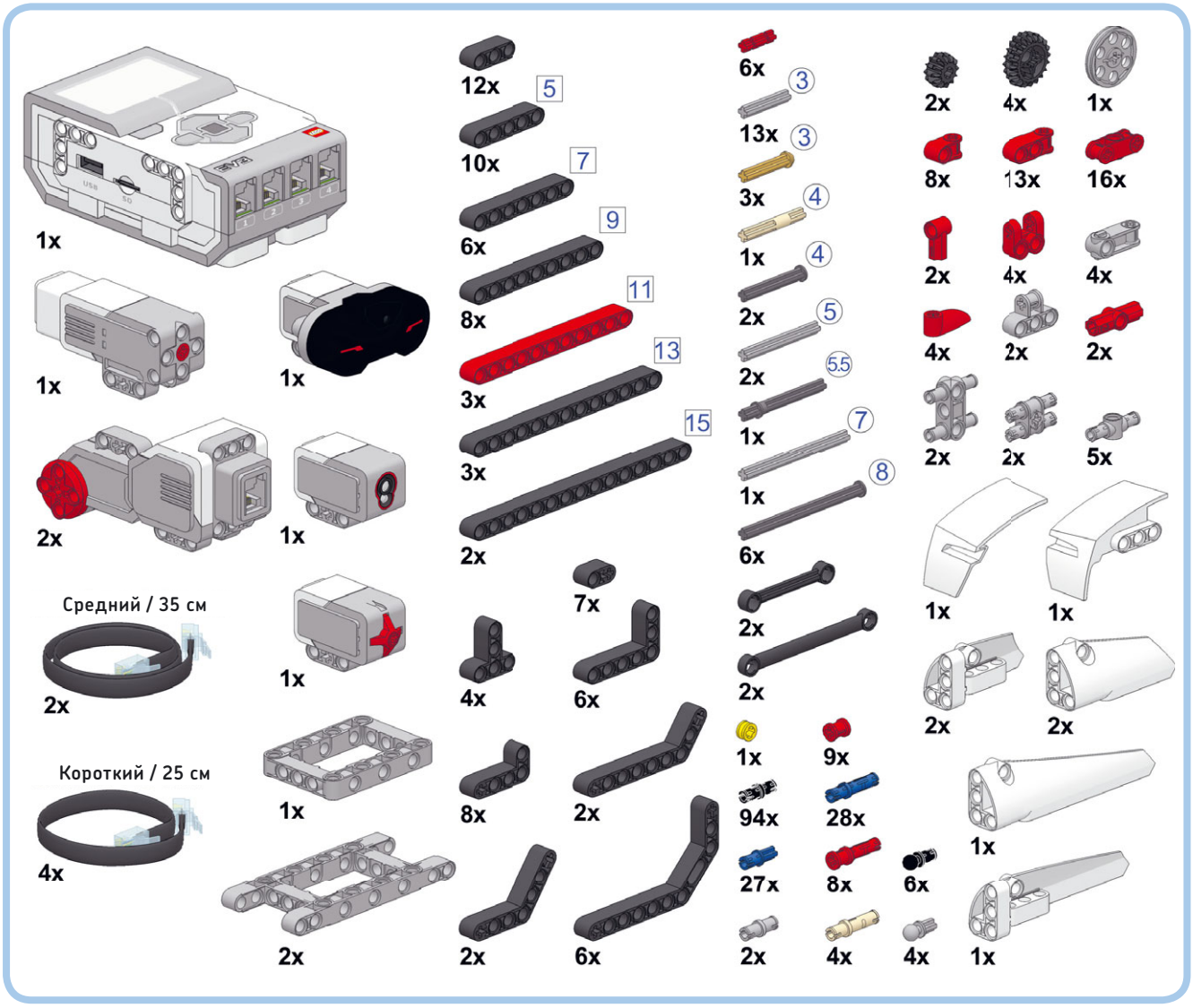
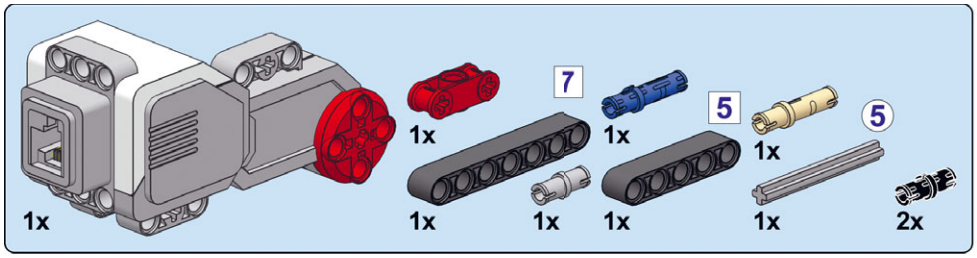
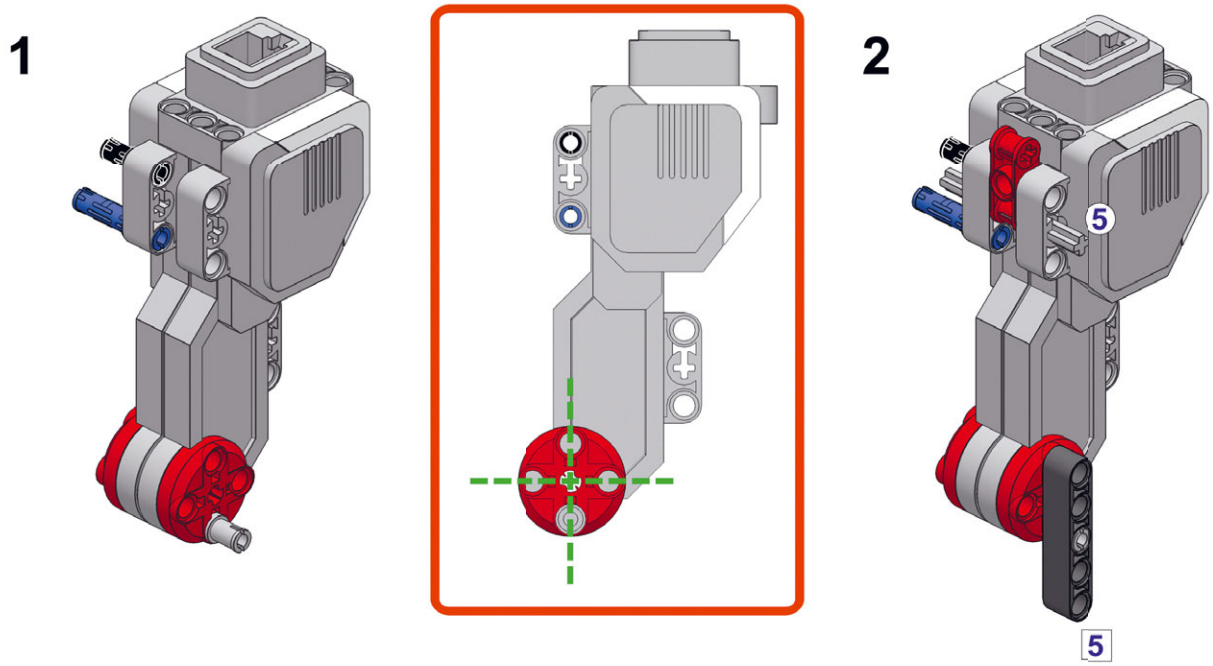
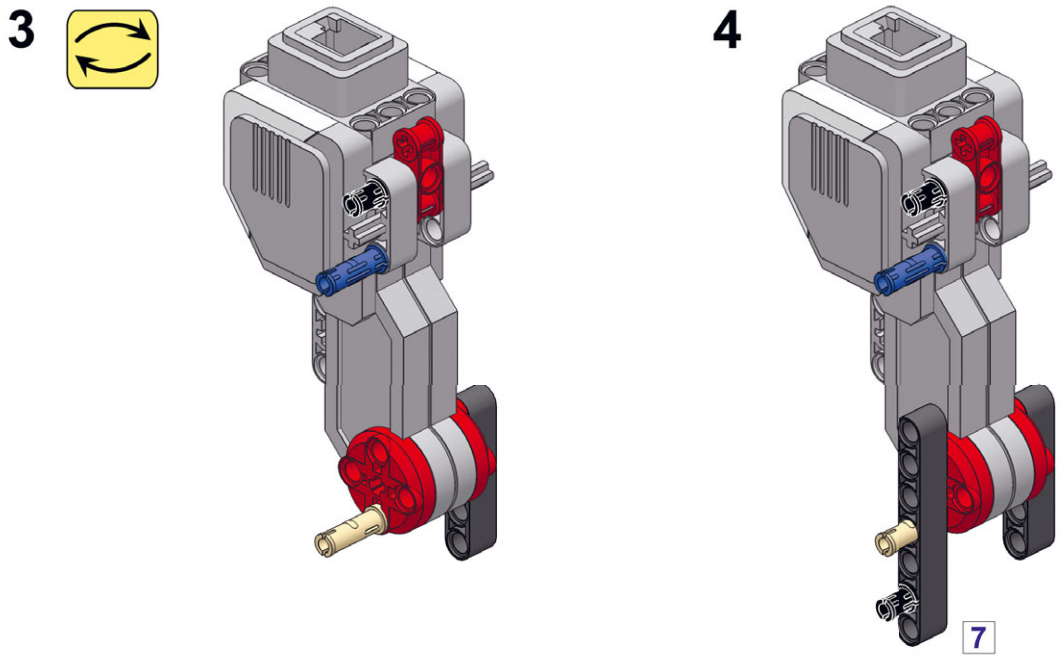
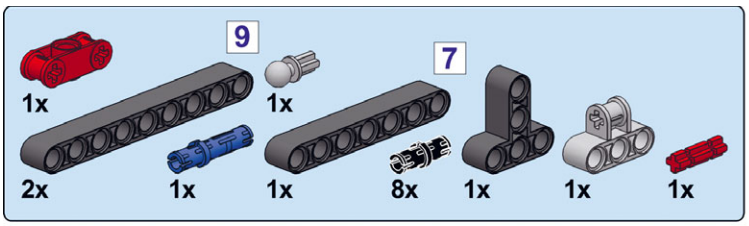


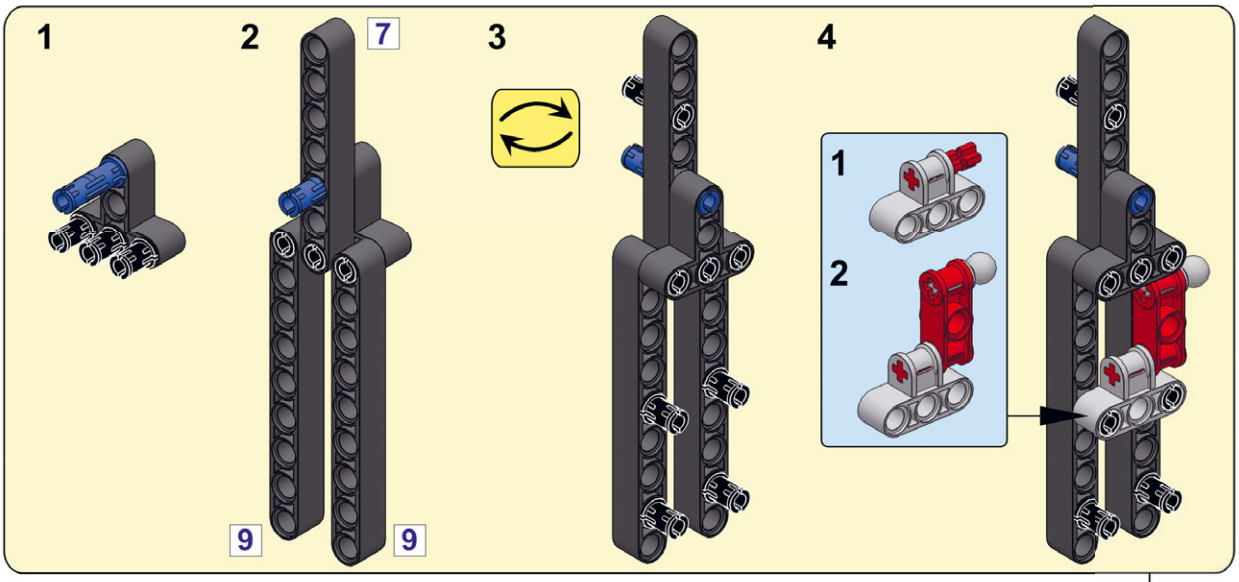
Рис. 19.3. Детали, необходимые для сборки робота LAVA R3X. После сборки ног у вас должны остаться незадействованные детали, они пригодятся вам позже, когда вы будете собирать голову и руки робота

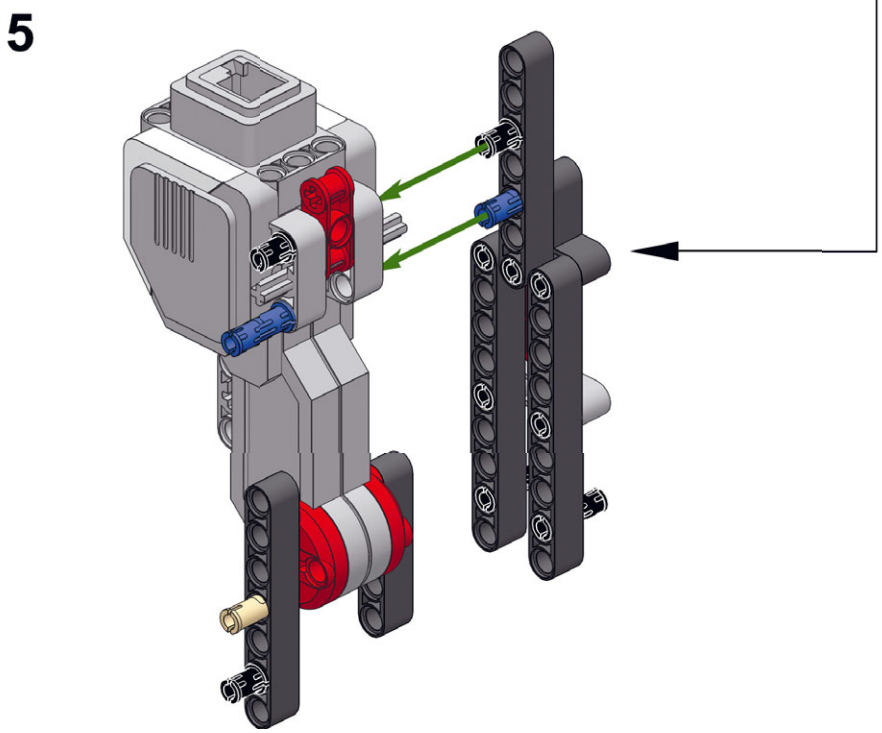

1x
7
1x
1x
1x
5
1x
1x
5
1x
2x

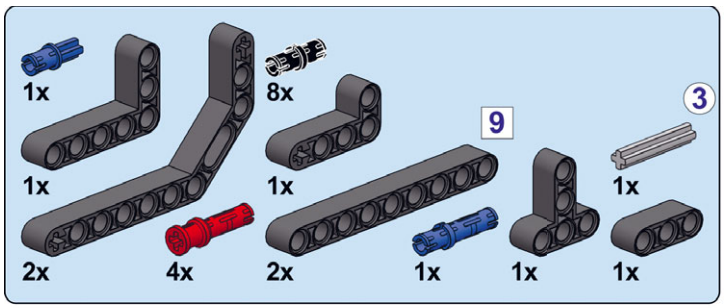

1
2
5
5

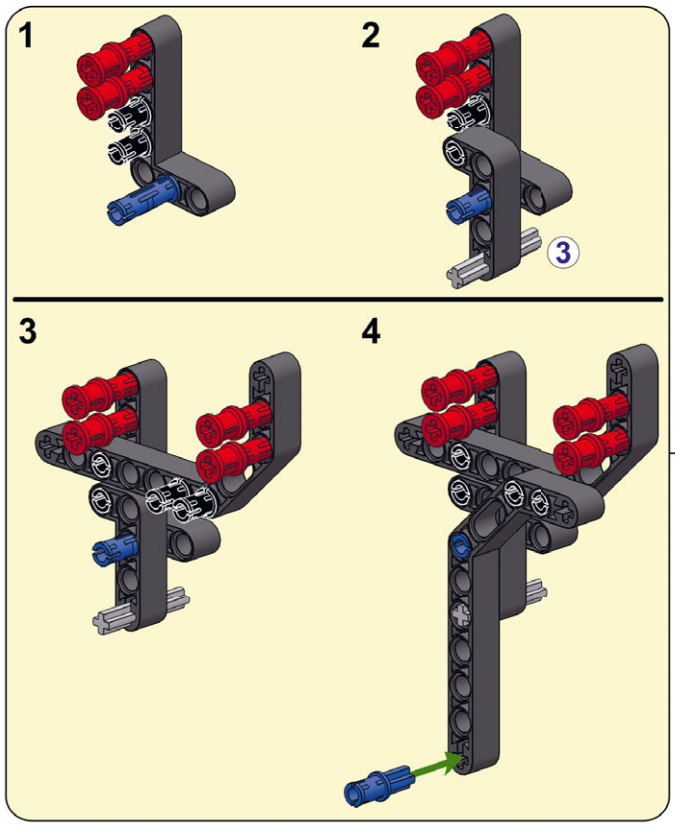

3
4
7


9
7
1x
1x
2x
1x
1x
8x
1x
1x
1x

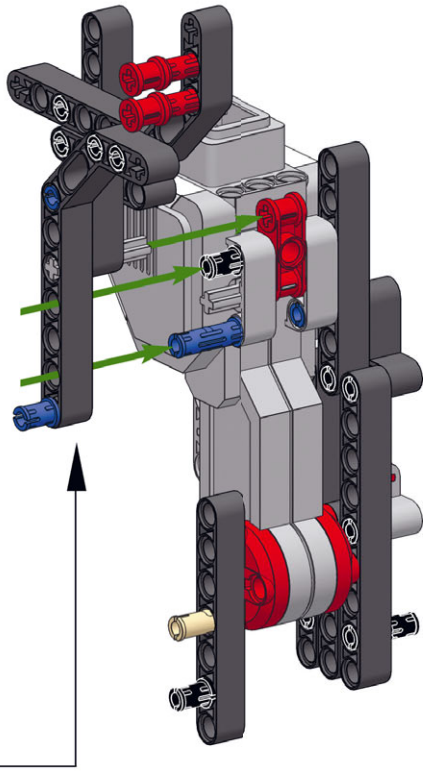

1
2
7
3
4
1
2
9
9


5

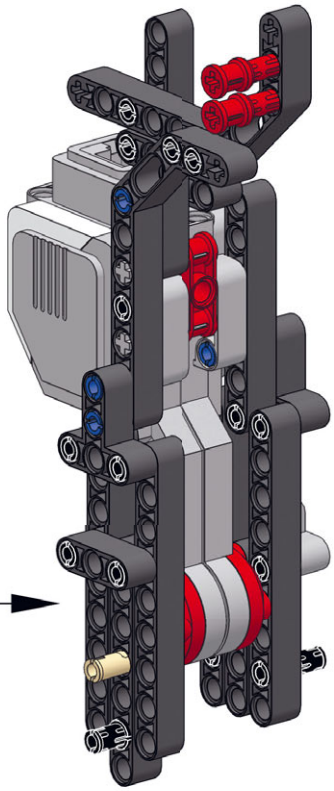


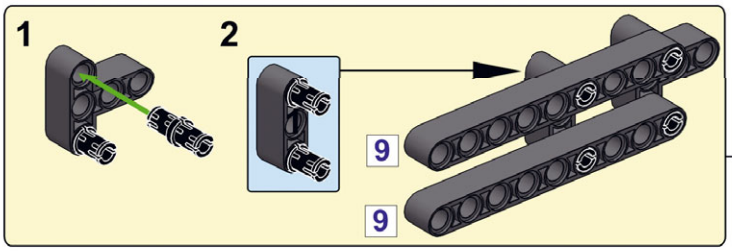


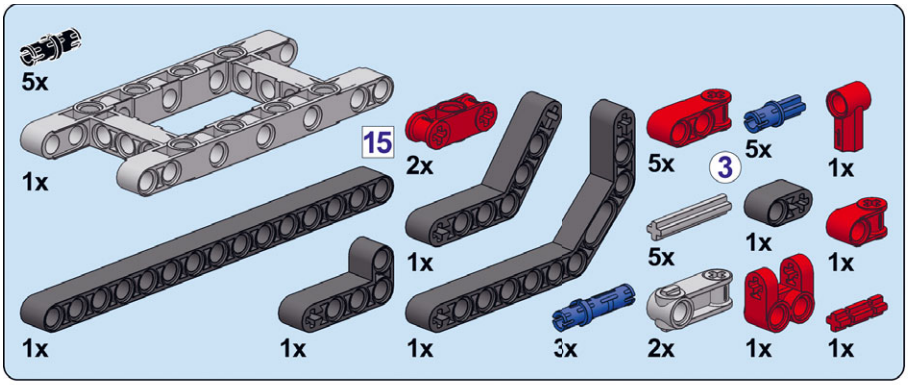
6

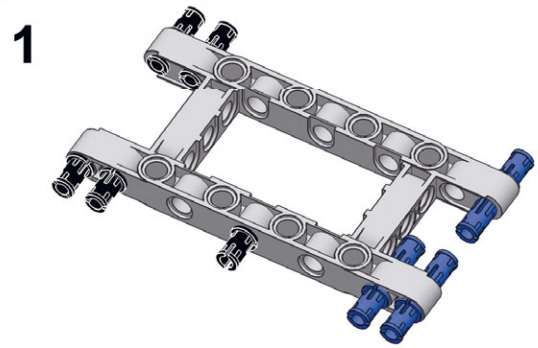


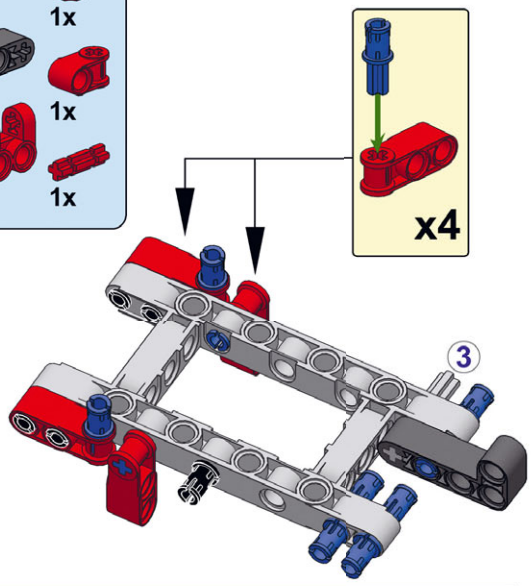
7

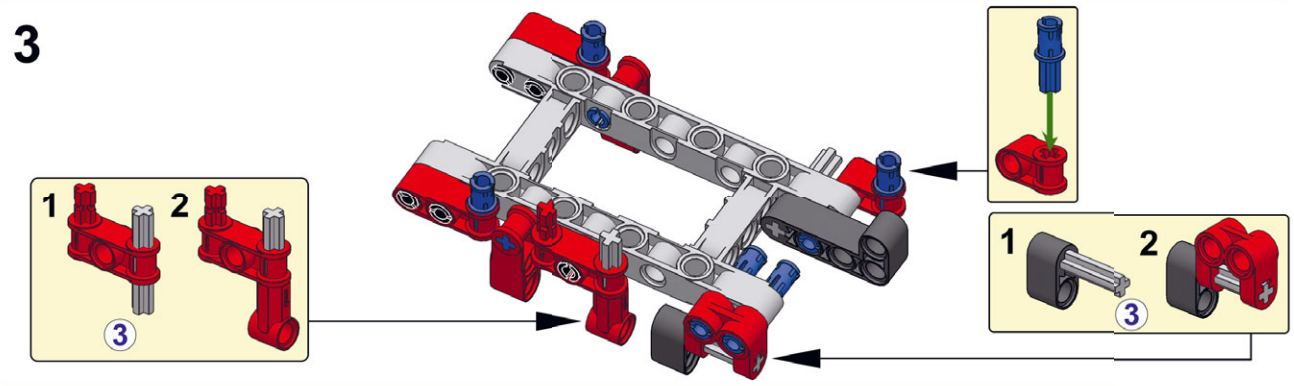


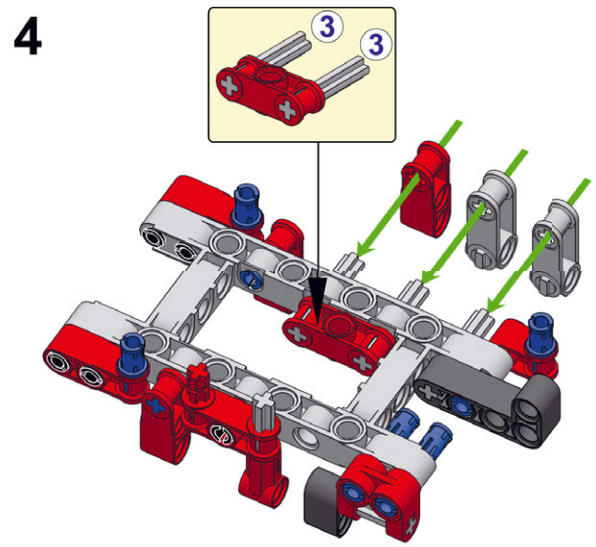


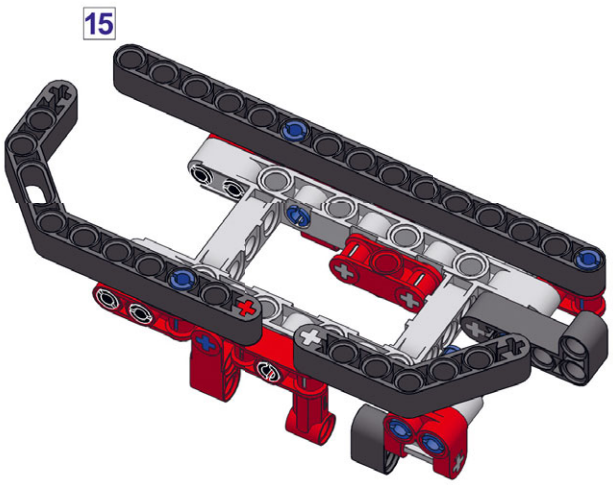

5x
1x
15
2x
1x
1x
1x
1x
3x
5x
3
5x
5x
2x
1x
1x
1x
1x
1x

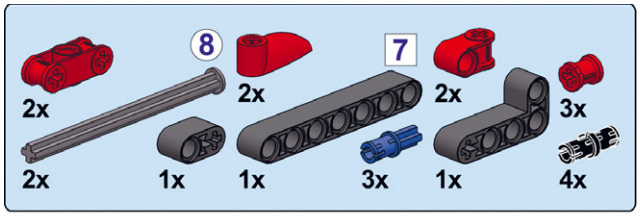

1


x4
2
3

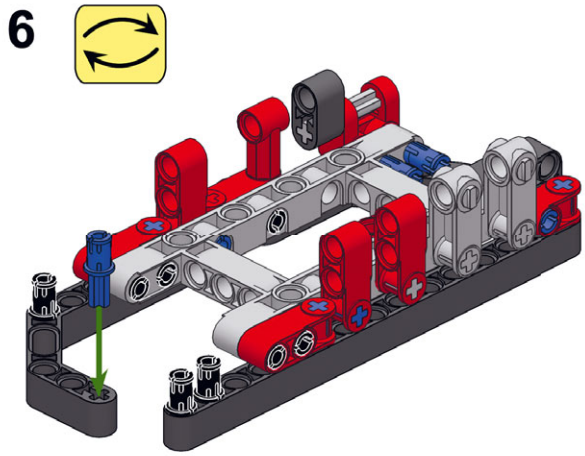

3
1
2
3
1
2
3


4
3
3

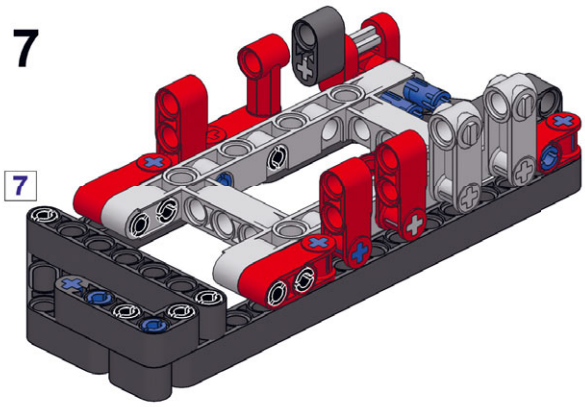

5
15

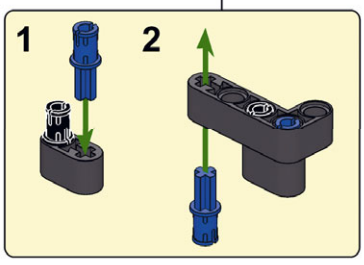


6

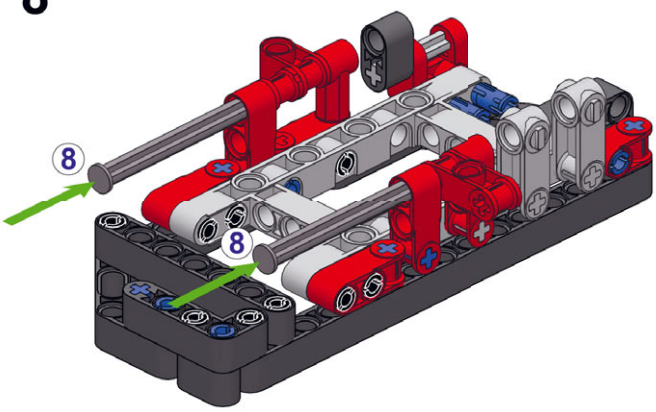


7

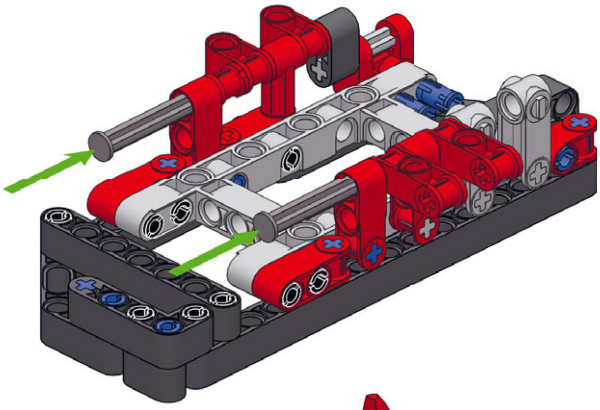




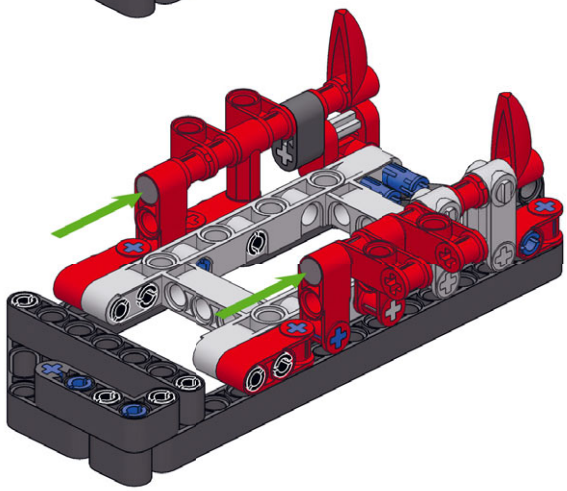
8

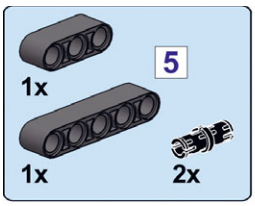


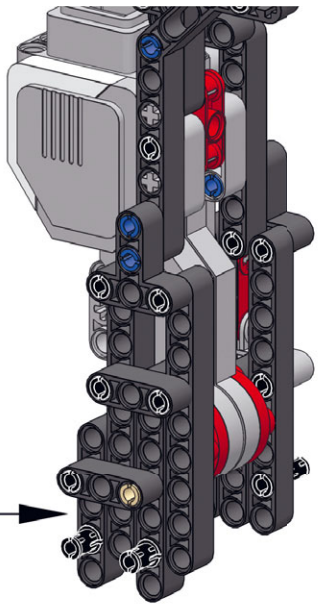
9



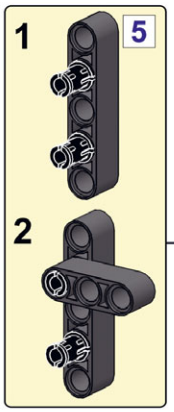
10



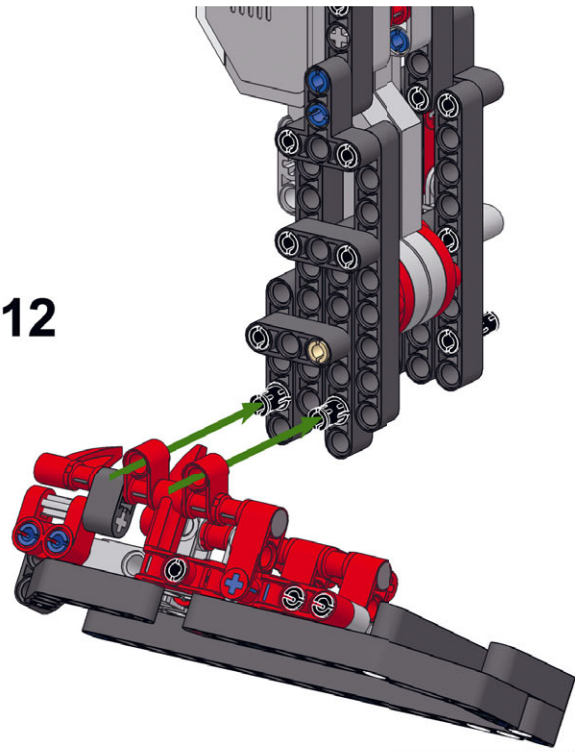




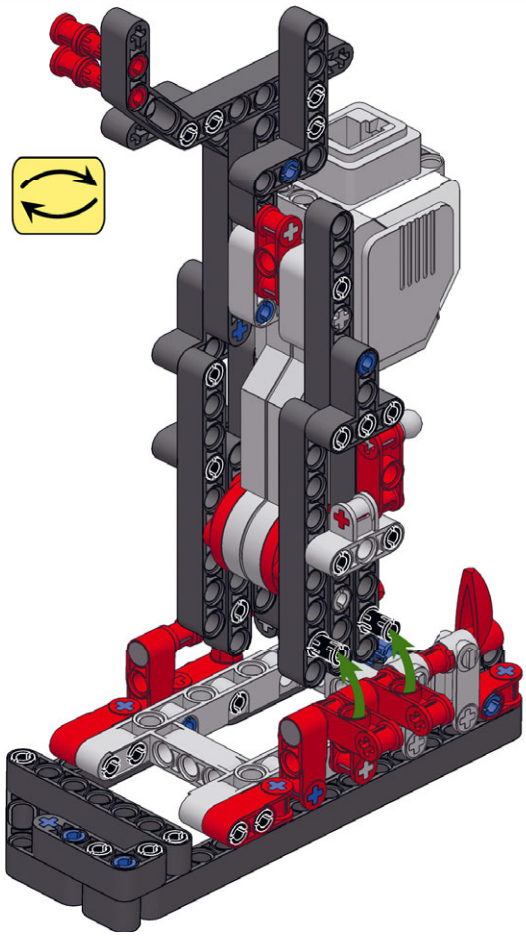
11

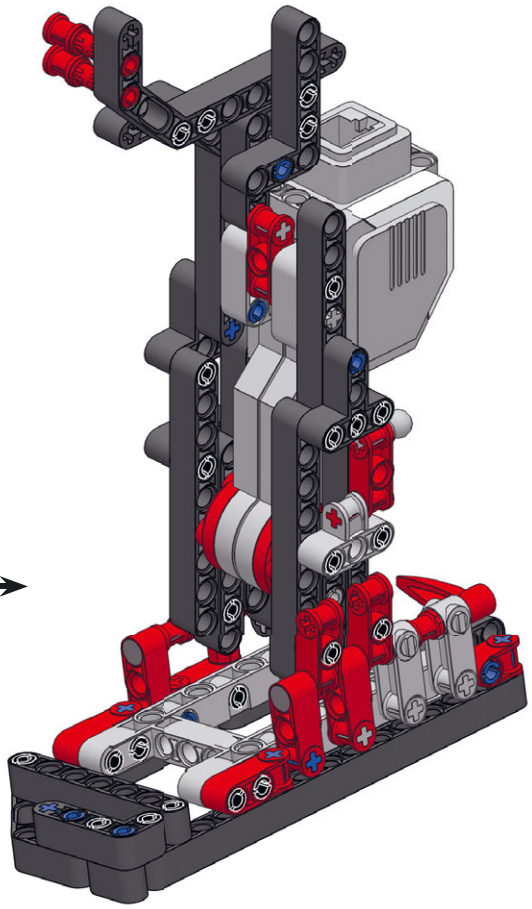


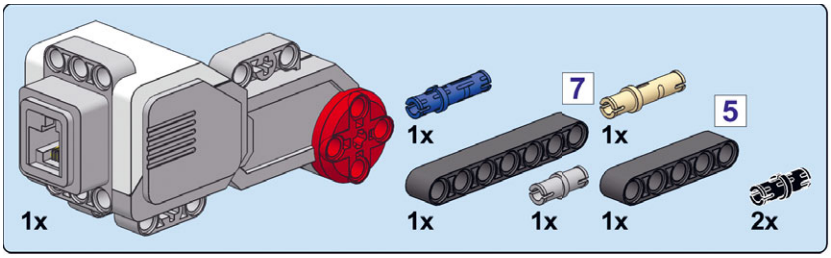
12



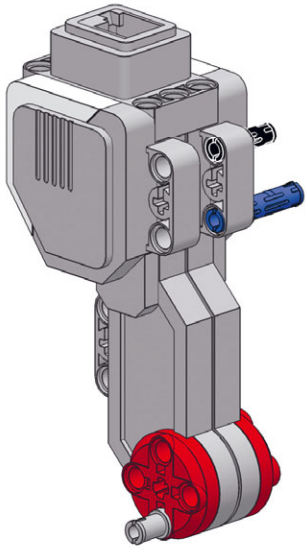
13

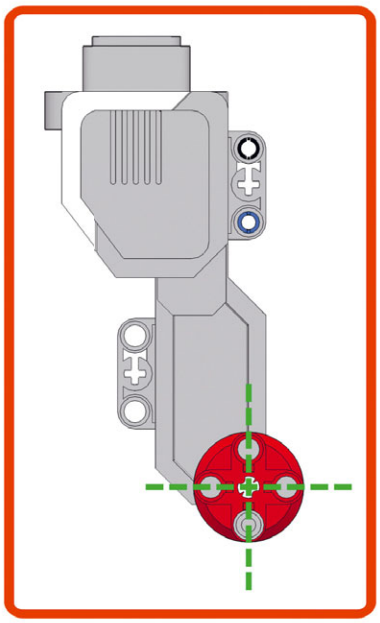




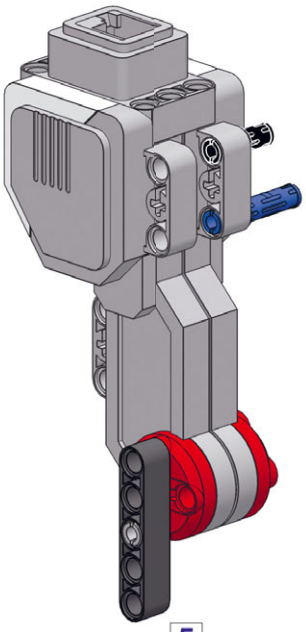


1

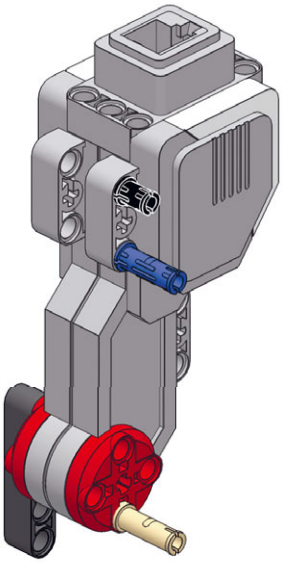




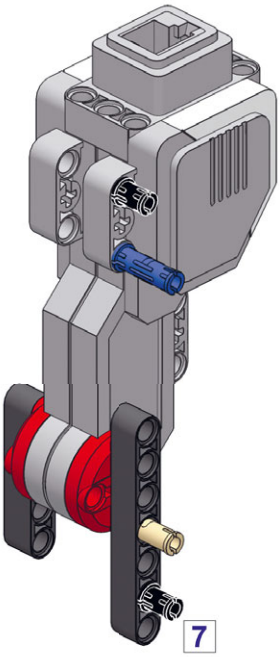
2

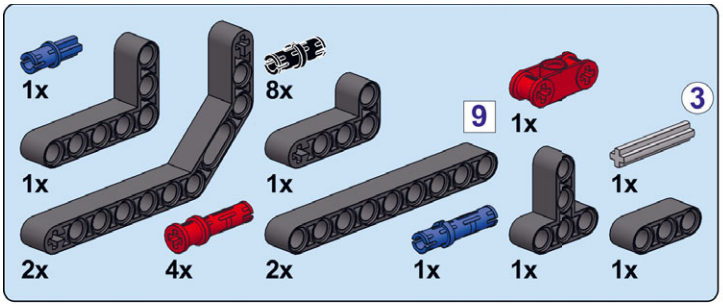


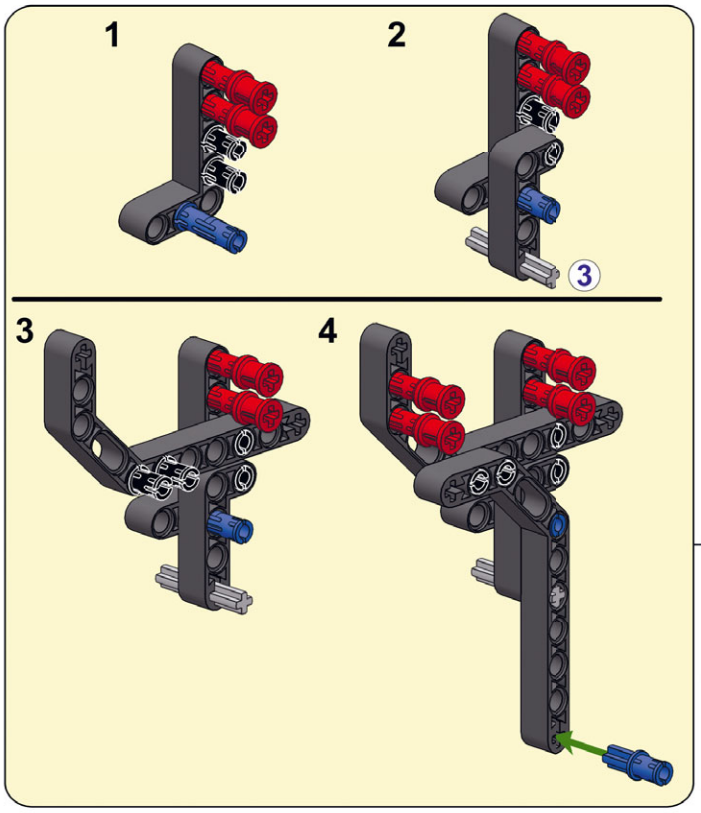
3

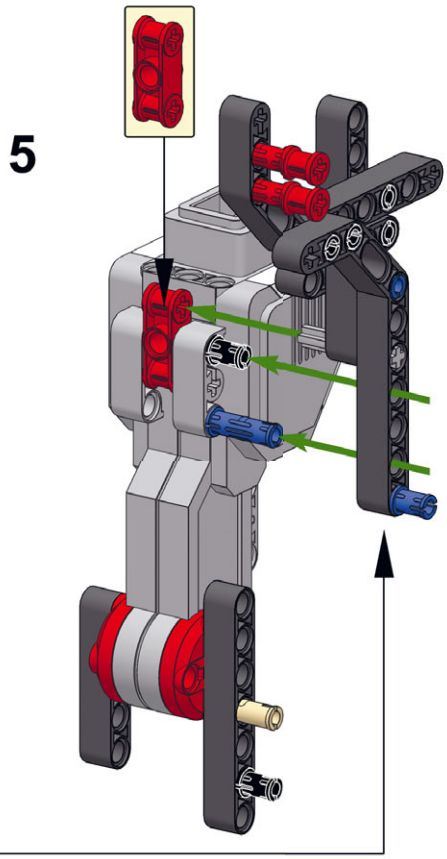


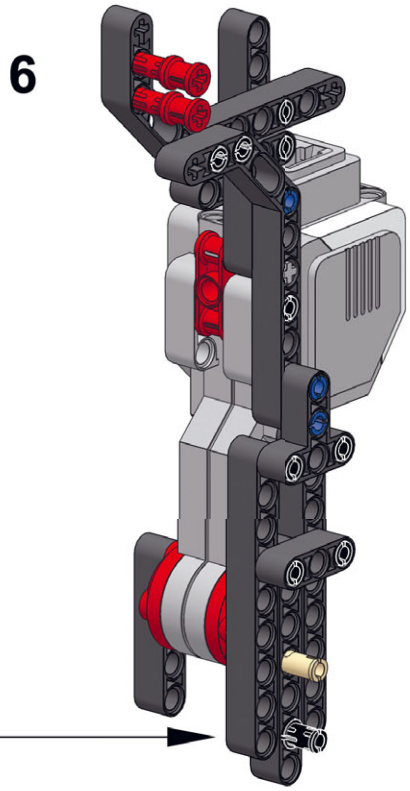
4

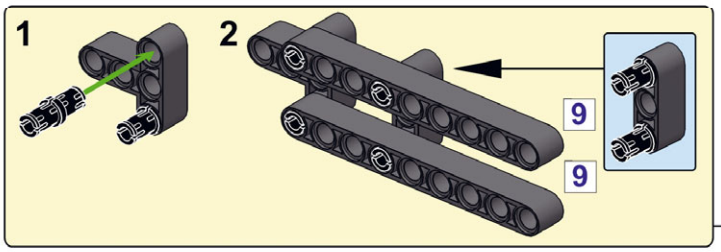


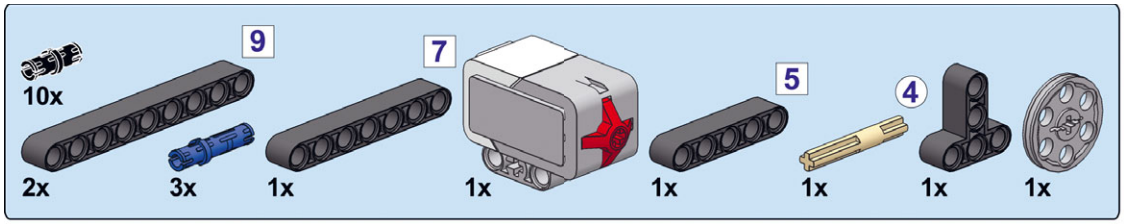

1x
1x
2x
4x
8x
1x
2x
9
1x
1x
1x
3
1x
1x


1
2
3
3
4

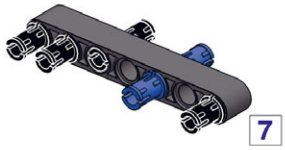

5


6

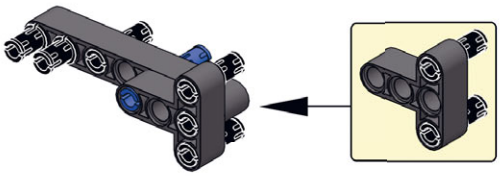

1
2
9
9



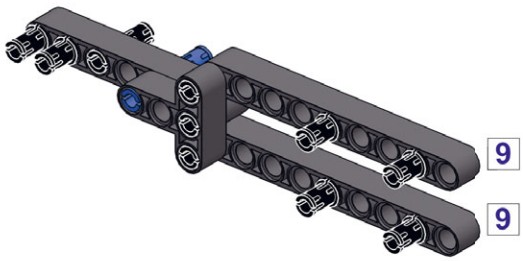
1



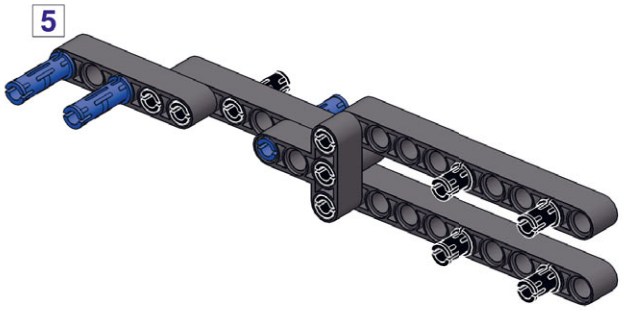
2



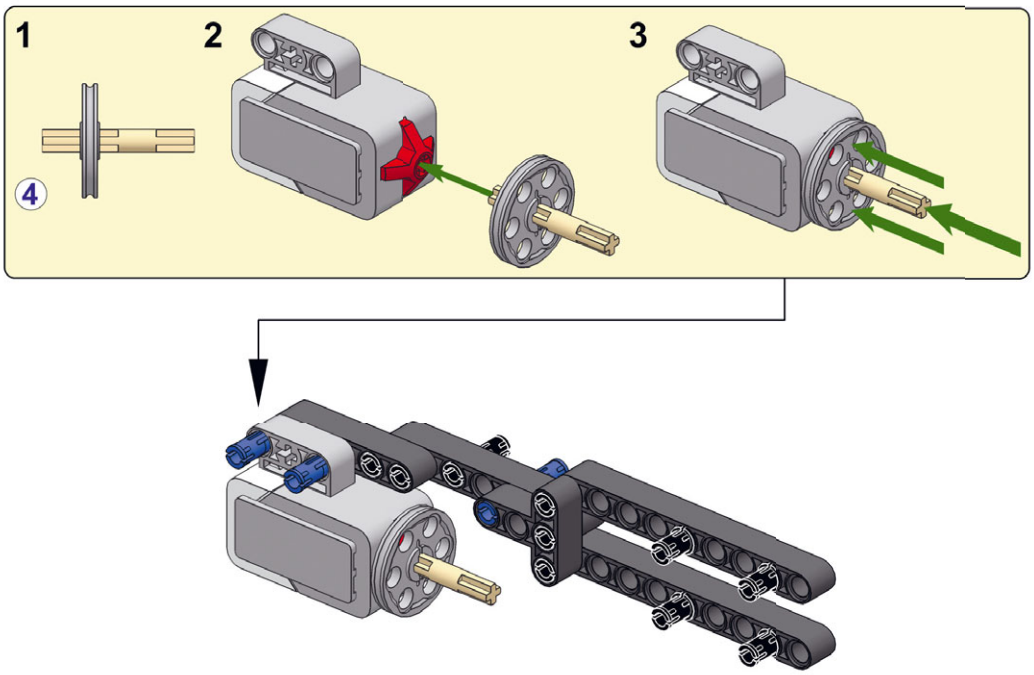
3

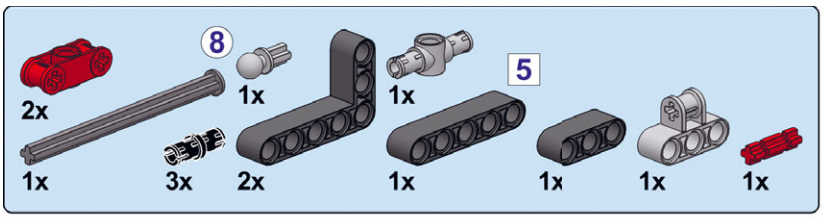


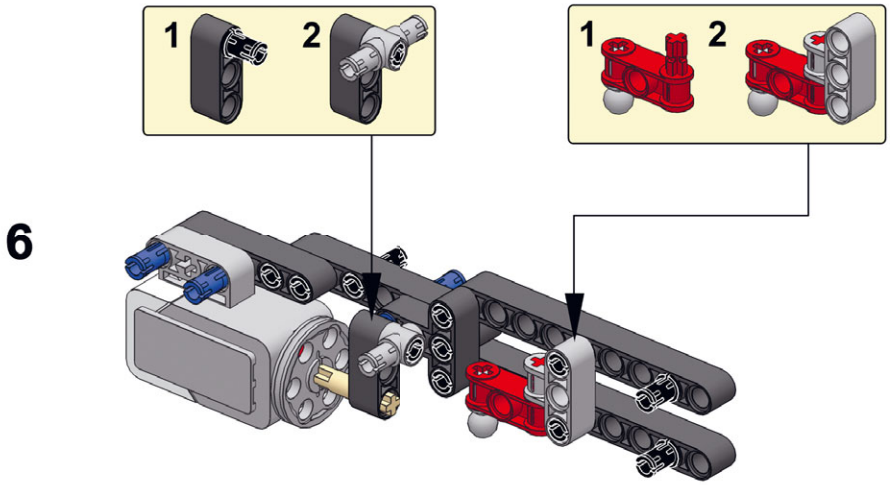
4

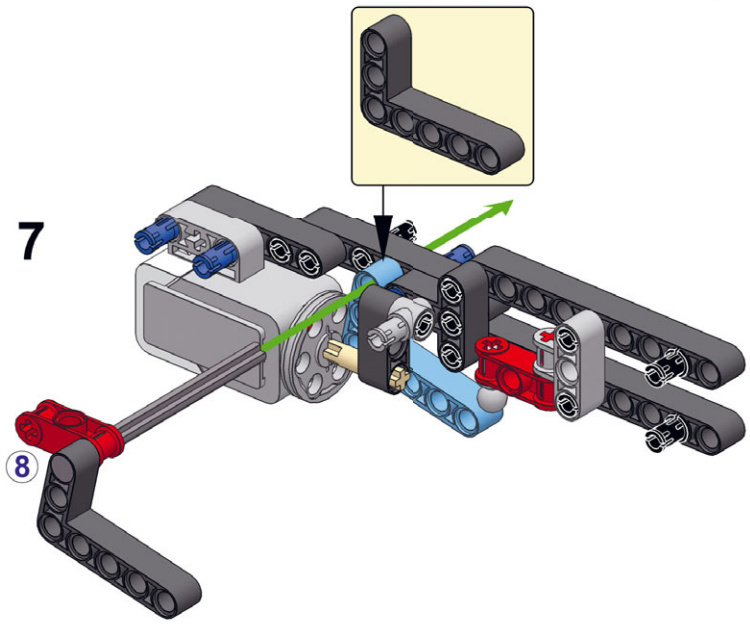


5

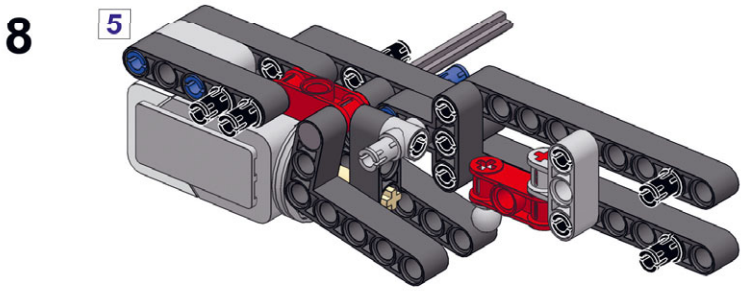


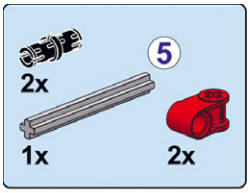




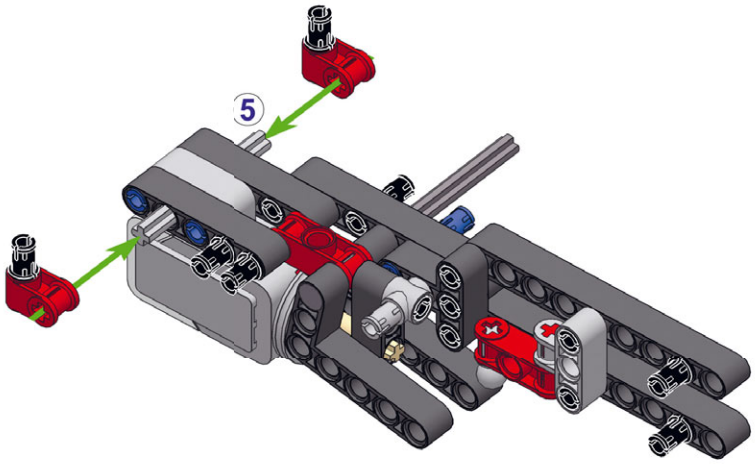


Одна из угловых балок для наглядности нарисована голубым цветом, но тут нужно использовать обычные черные балки

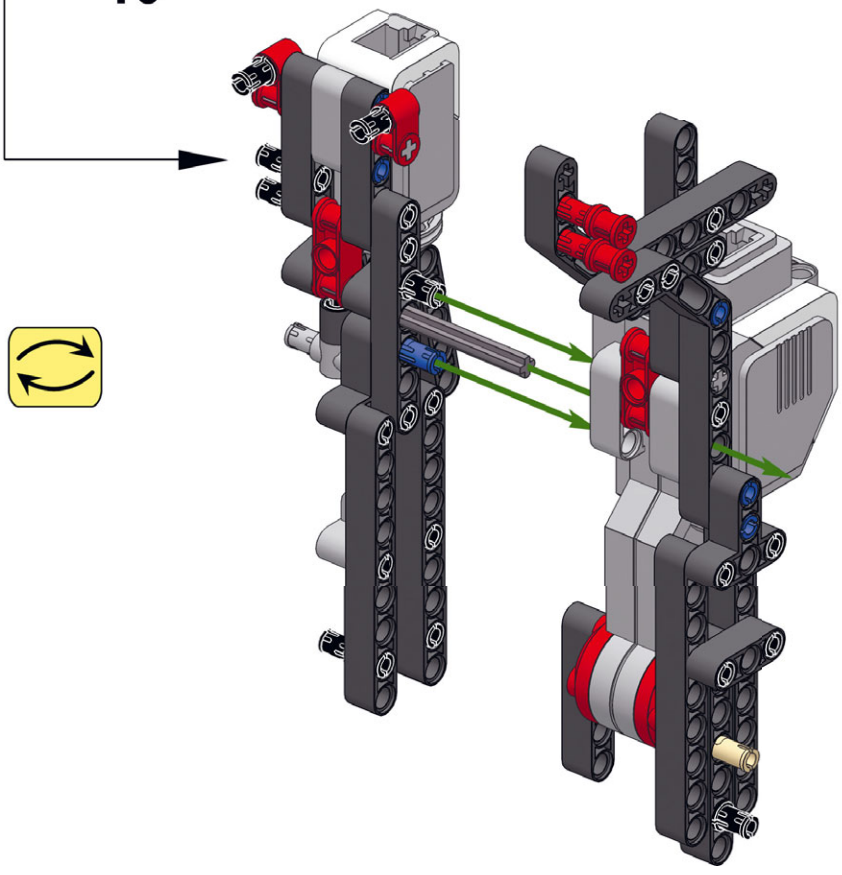


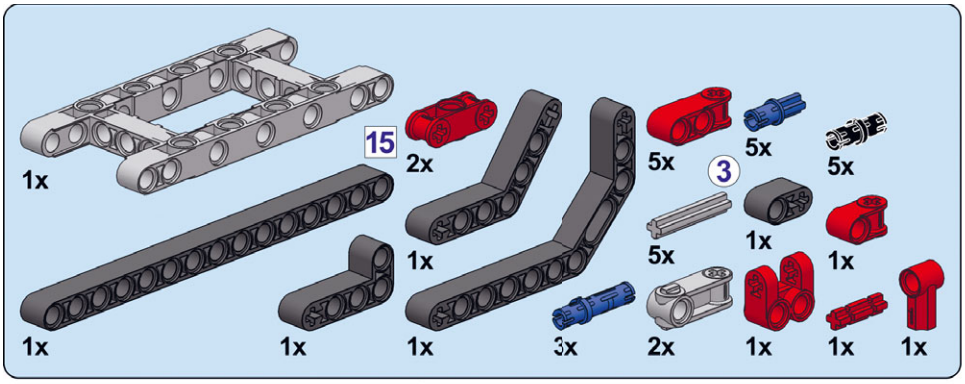


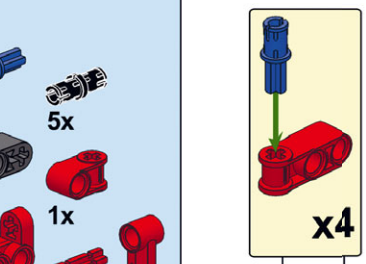
9



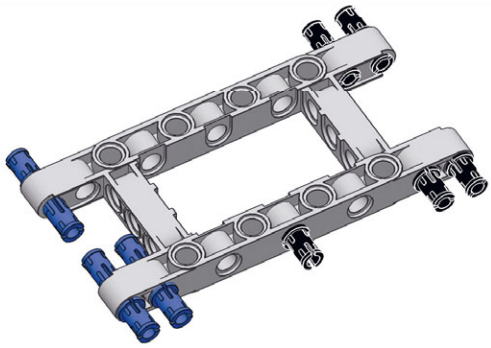
10



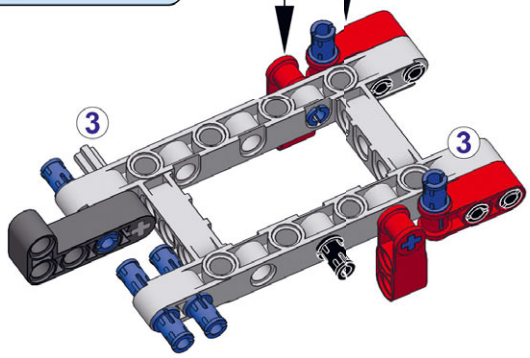




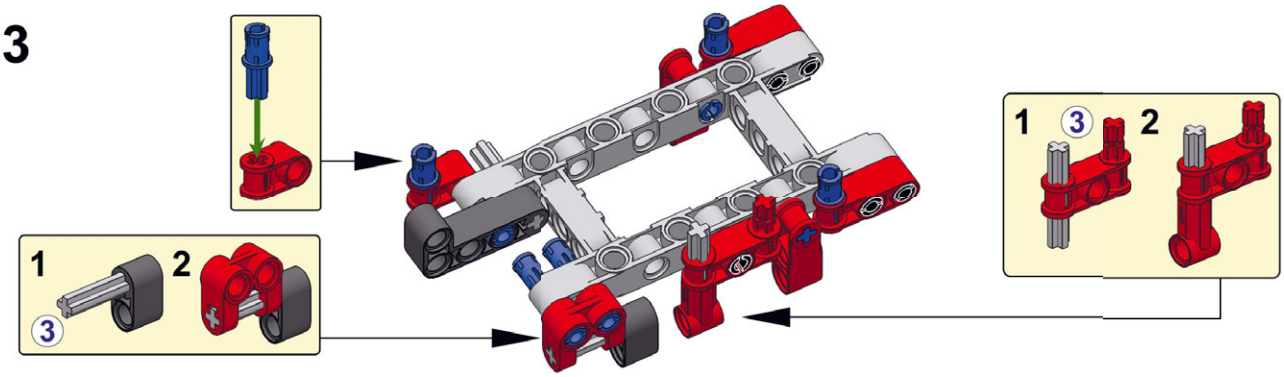
1



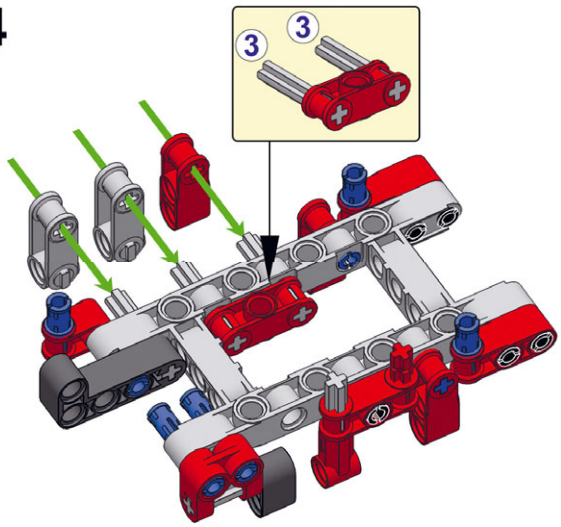
2



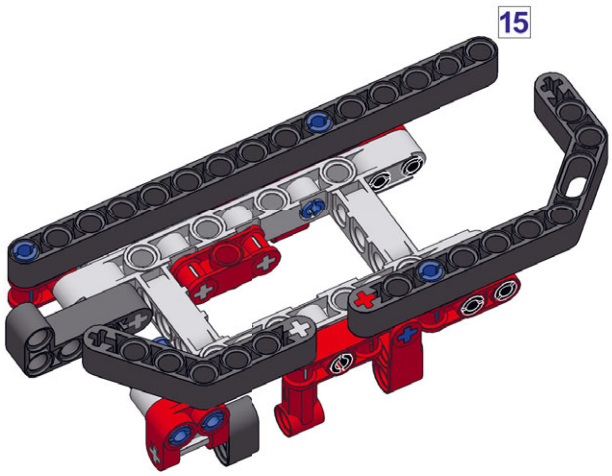
3

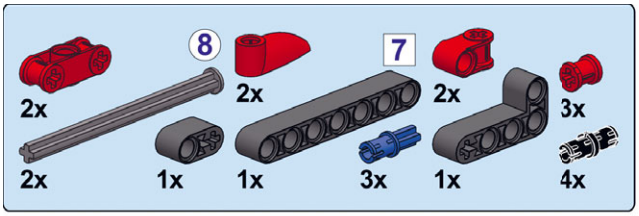


4

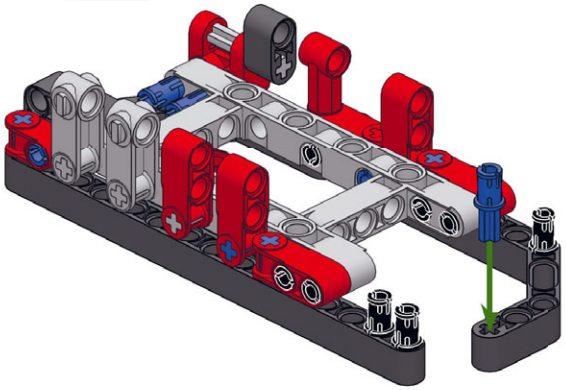


5

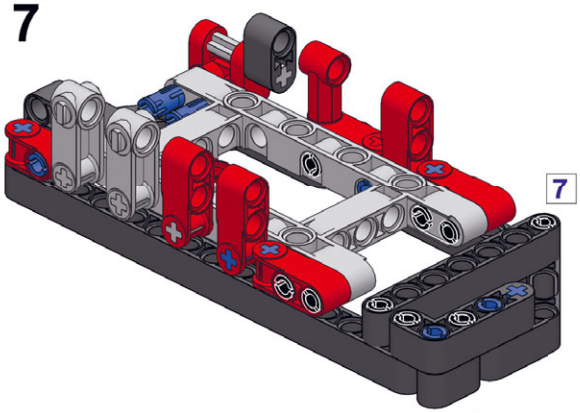


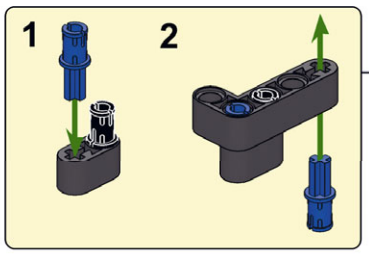


6

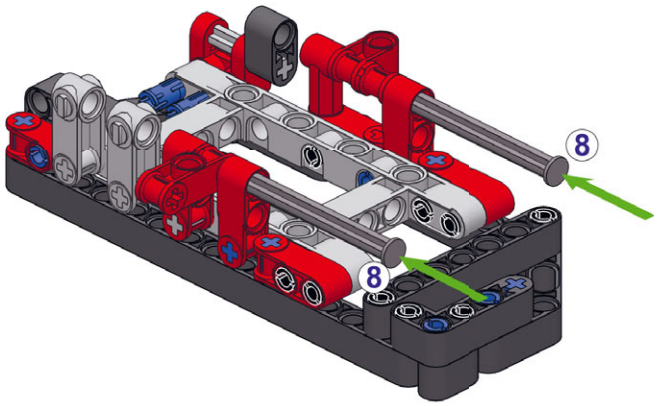


7

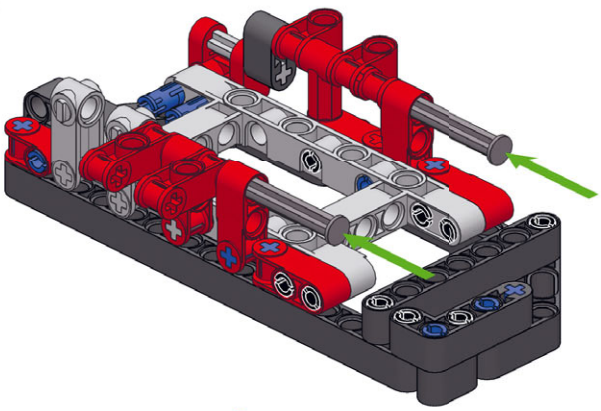




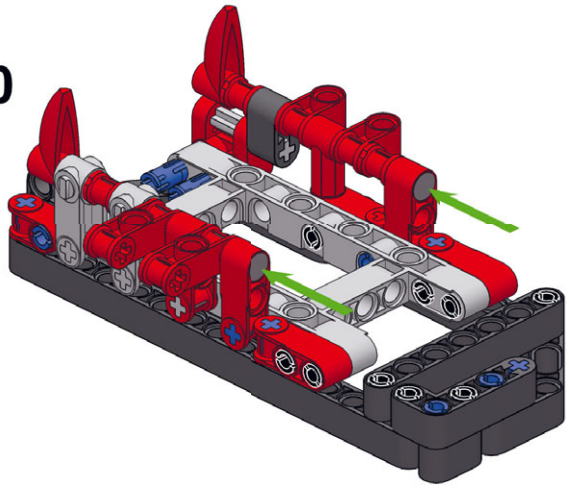
8

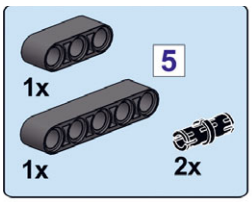


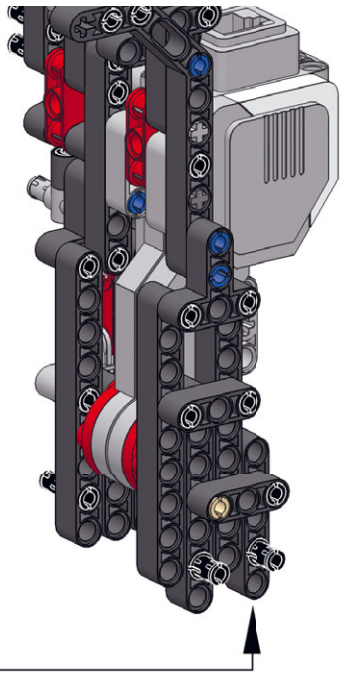
9



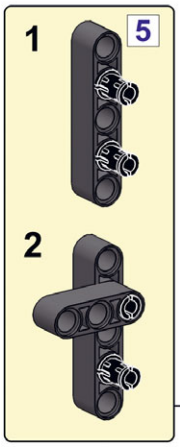
10



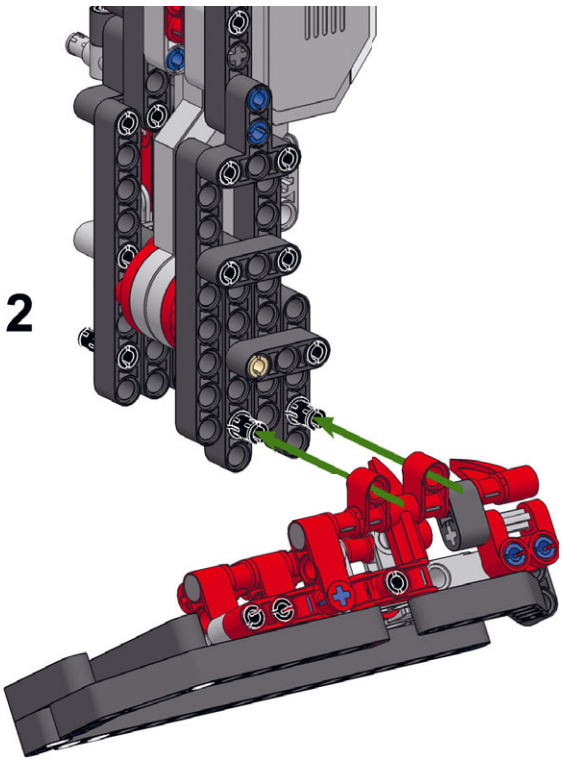




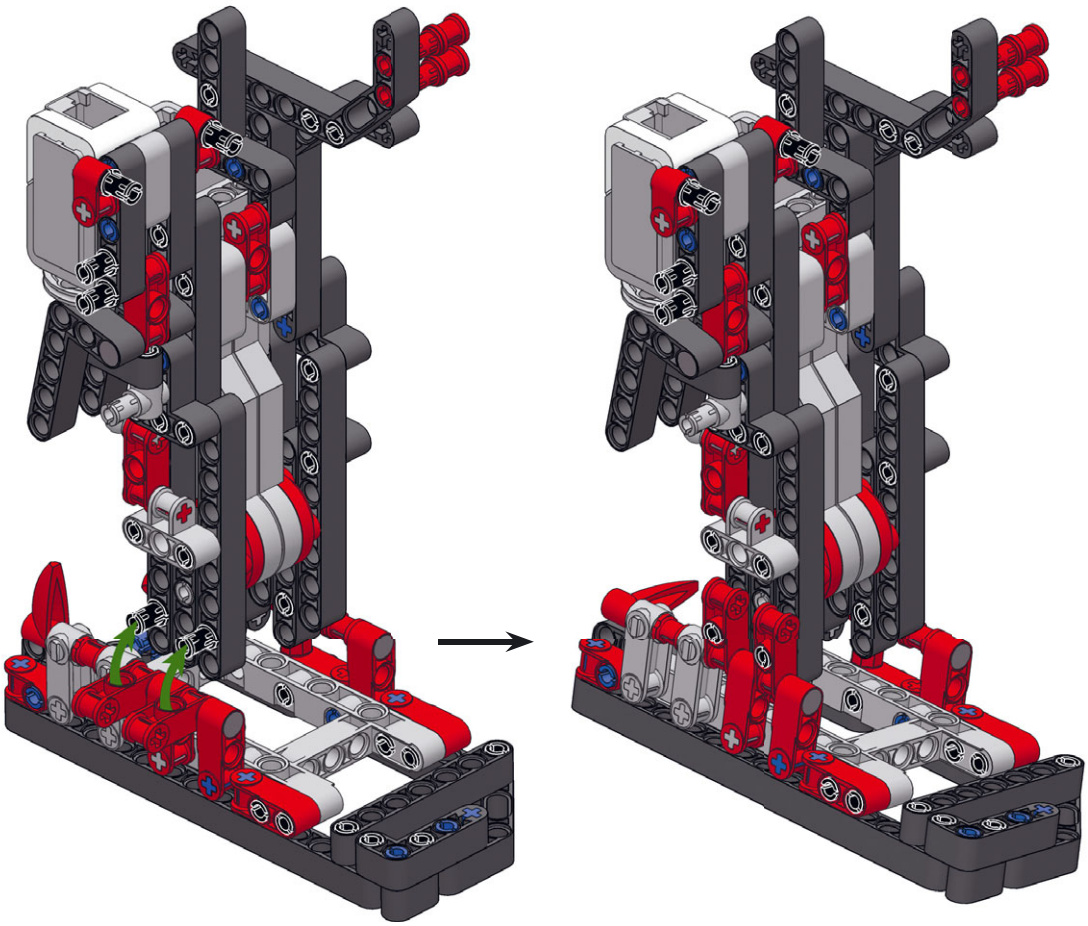
11



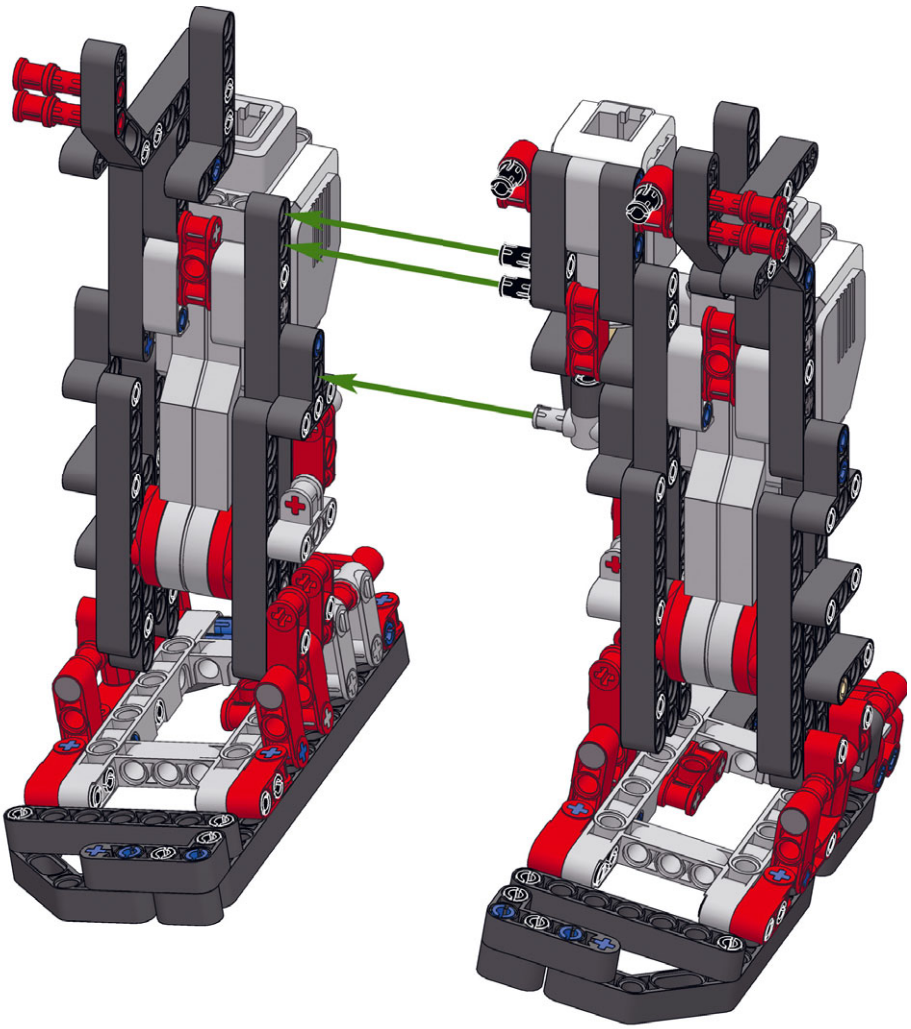
12

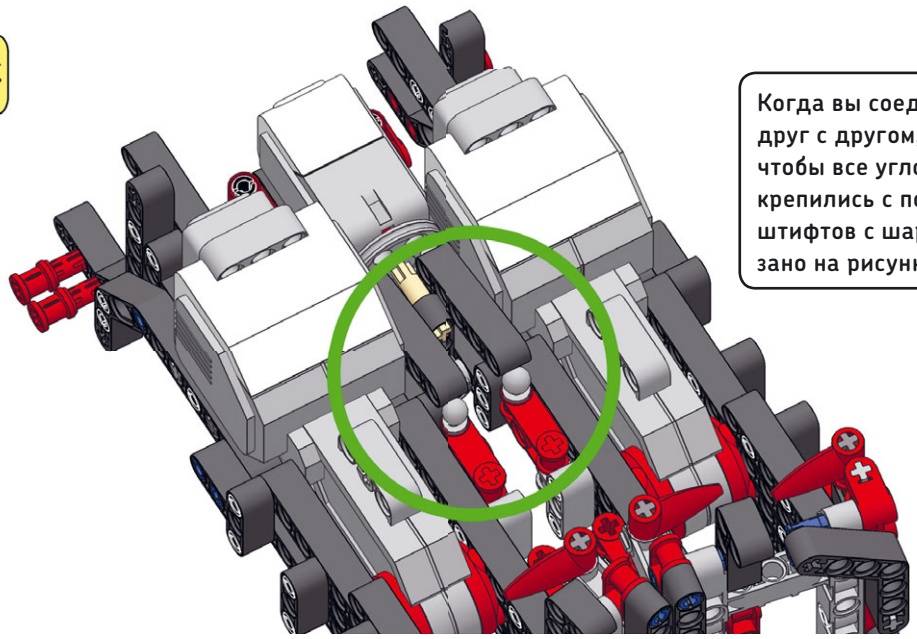


13

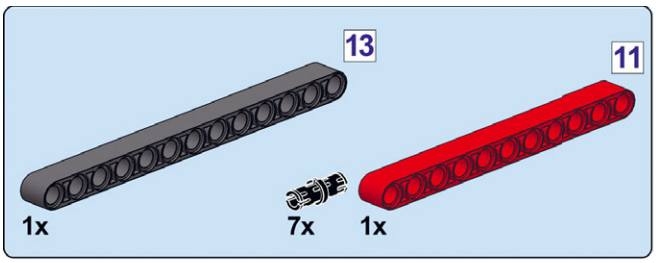


1

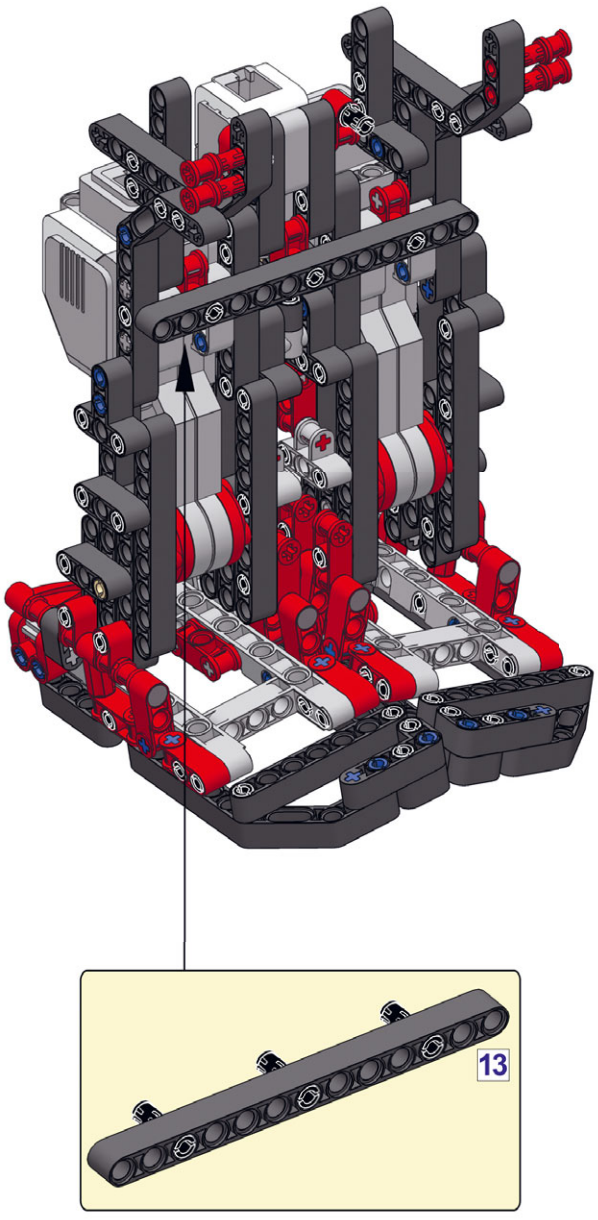




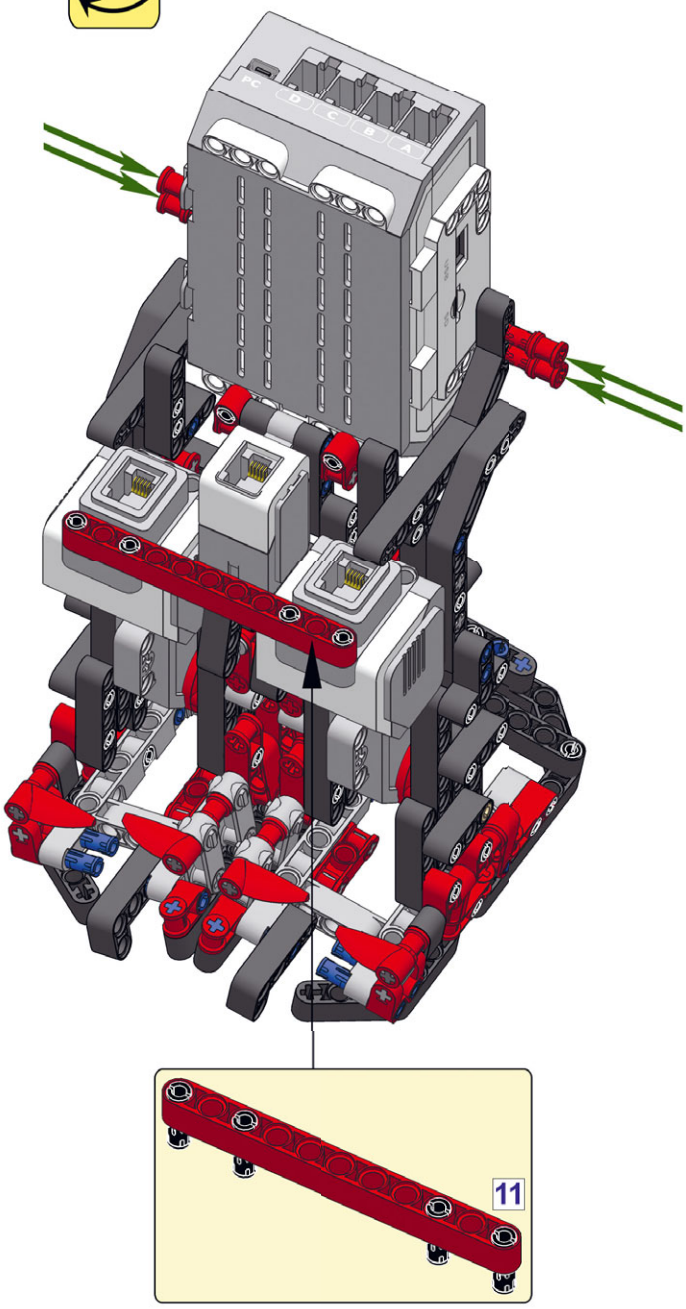
Когда вы соедините ноги друг с другом, проверьте, чтобы все угловые балки крепились с помощью серых штифтов с шарами, как показано на рисунке



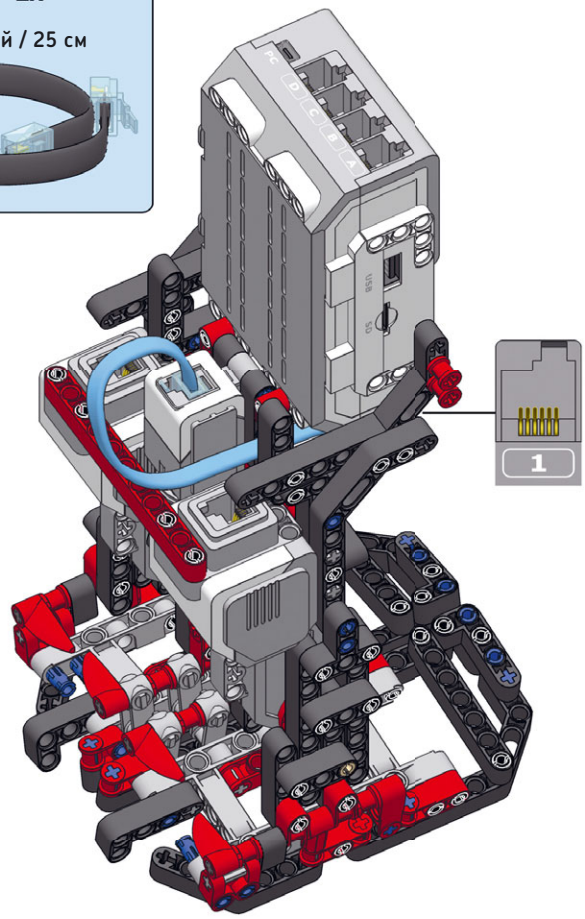
2



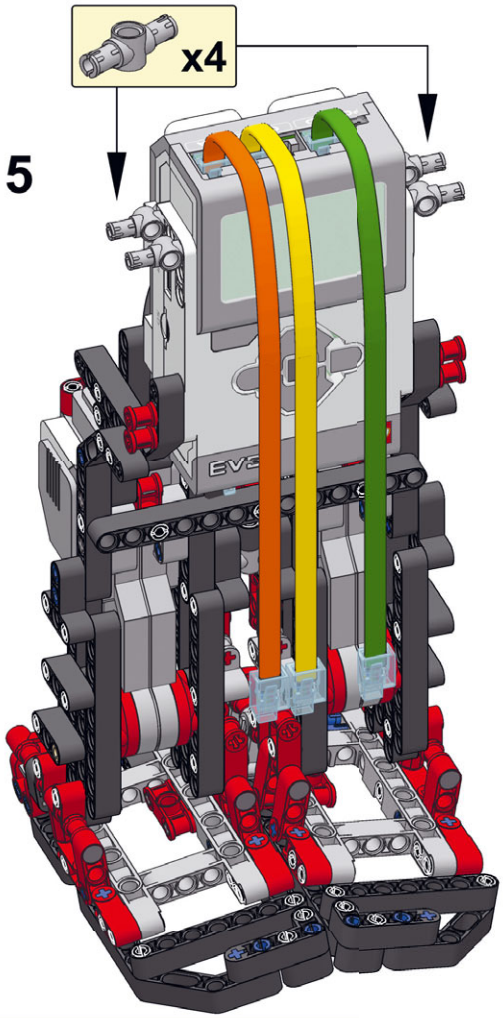
3



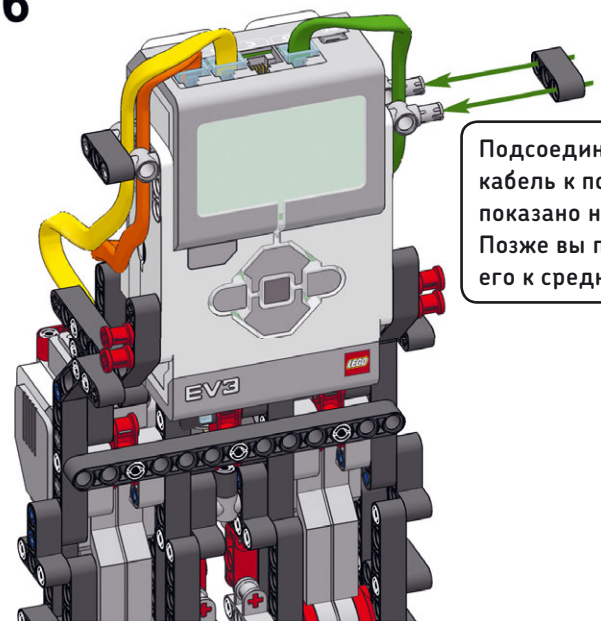
4



5

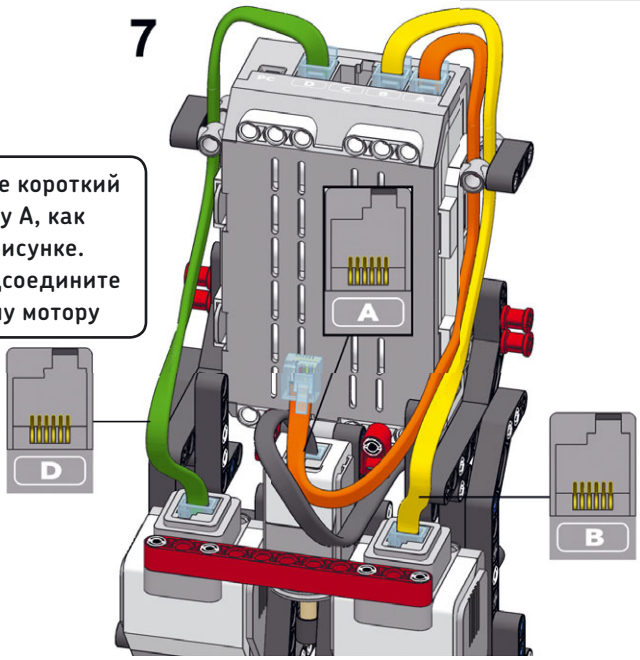


6



Подсоедините короткий кабель к порту A, как показано на рисунке. Позже вы подсоедините его к среднему мотору

7



Обучение робота ходьбе

Теперь вы создадите несколько контейнеров «Мой блок», задача которых — размещать ноги робота в противоположных направлениях, чтобы он мог идти вперед и поворачивать налево. Еще вы напишете небольшую программу для тестирования каждого блока.

Чтобы протестировать эти блоки без тяжелой верхней части робота, нужно меньше усилий, поскольку в этом случае меньше вероятность того, что робот упадет. Когда робот благодаря контейнерам «Мой блок» будет уверенно ходить, вы сможете завершить его сборку.

Контейнер «Мой блок» № 1: Сброс

Каждый раз, когда мотор выполняет один полный оборот, связанный с этим мотором механизм однократно нажимает кнопку датчика касания (рис. 19.4). Робот LAVA R3X, используя показания этого датчика, может поставить левую и правую ноги противоположно друг другу.

Поскольку робот не может определить, механизм которой ноги нажимает кнопку датчика касания в данный момент, он должен сначала расположить ноги таким образом, чтобы ни один из механизмов не касался датчика. Чтобы это сделать, робот прокручивает моторы до тех пор, пока датчик касания не перестает испытывать давление, а затем останавливает движение моторов. Иногда после остановки моторов в результате инерции какой-то из механизмов снова нажимает кнопку датчика касания. Чтобы избежать этого, робот ждет 0,1 секунды. Затем с помощью блока **Цикл** (Loop) он перепроверяет состояние датчика касания. Если кнопка датчика так и осталась отпущенной, цикл завершается. Если же один из механизмов в течение этого времени все-таки нажал кнопку датчика, цикл запускается снова.

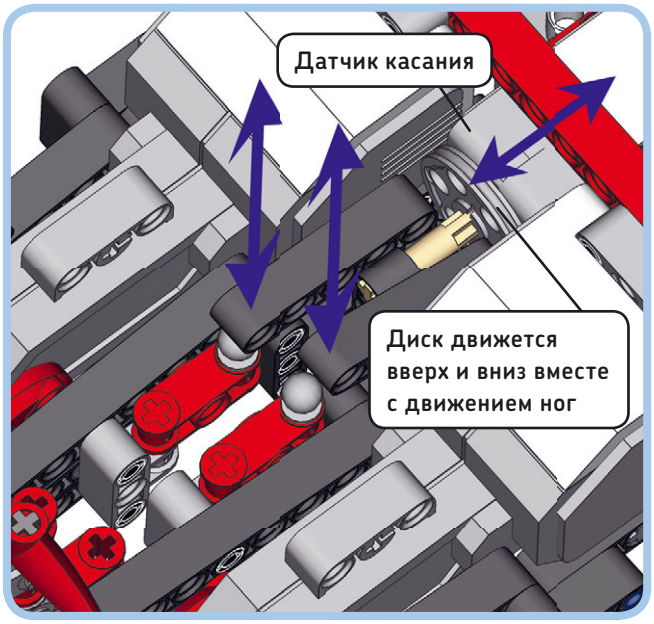


Рис. 19.4. Когда мотор делает оборот, он толкает черную угловую балку на серый диск, который в свою очередь нажимает кнопку датчика касания. Бежевая ось нужна, чтобы серый диск оставался на своем месте

Когда проверка выявила, что ни один из механизмов не нажимает кнопку датчика касания, программа продолжает процесс сброса настроек. Сначала робот прокручивает левый мотор (мотор D) вперед до тех пор, пока тот не нажмет кнопку датчика касания, а затем проворачивает этот мотор еще на 90 градусов вперед. Далее робот проворачивает вперед правый мотор (мотор B) до нажатия кнопки датчика касания, а затем прокручивает его на 90 градусов назад, так чтобы механизмы ног робота отставали друг от друга на 180 градусов, оказавшись в начальном положении для ходьбы. Когда ноги готовы, значения обоих датчиков вращения сбрасываются до 0. То есть во время ходьбы механизмы будут находиться в противоположных позициях, и при этом значения датчиков вращения обоих моторов будут одинаковыми. Обнуление значений датчиков позже позволит без труда вернуться в начальную позицию.

Создайте контейнер «Мой блок» с именем **Reset**, с помощью которого будут выполняться все эти действия (рис. 19.5). Он должен располагаться в начале каждой программы, разработанной для робота LAVA R3X.

Контейнер «Мой блок» № 2: Возврат

Программа предусматривает, что робот LAVA R3X будет не только ходить по прямой, но и сможет поворачивать налево. После поворота механизмы ног могут изменить свое положение по отношению друг к другу, поэтому, прежде чем робот продолжит идти, нужно вернуть ноги на исходные, противоположные друг другу, позиции. Но запускать блок **Reset** после каждого поворота неудобно, на это будет уходить слишком много времени. К счастью, того же результата можно достичь, просто прокрутив каждый мотор назад, к нулевой позиции.

Чтобы мотор вернулся в исходное положение, робот должен вычислить свою текущую позицию и прокрутить мотор назад на соответствующее значение. Например, если мотор повернулся на 25 градусов вперед, он должен вернуться на 25 градусов назад. Если мотор совершил больше одного оборота, он должен вернуться назад лишь на то количество градусов, которое превышает число полных оборотов (рис. 19.6). Например, если датчик зафиксировал 450 градусов, мотор должен прокрутиться назад всего на 90 градусов, к той точке, в которой значение датчика будет равно 360 градусам. При таком положении мотора стопа робота окажется в той же позиции, в которой она находилась бы при нулевом значении датчика, поэтому результат одинаков.

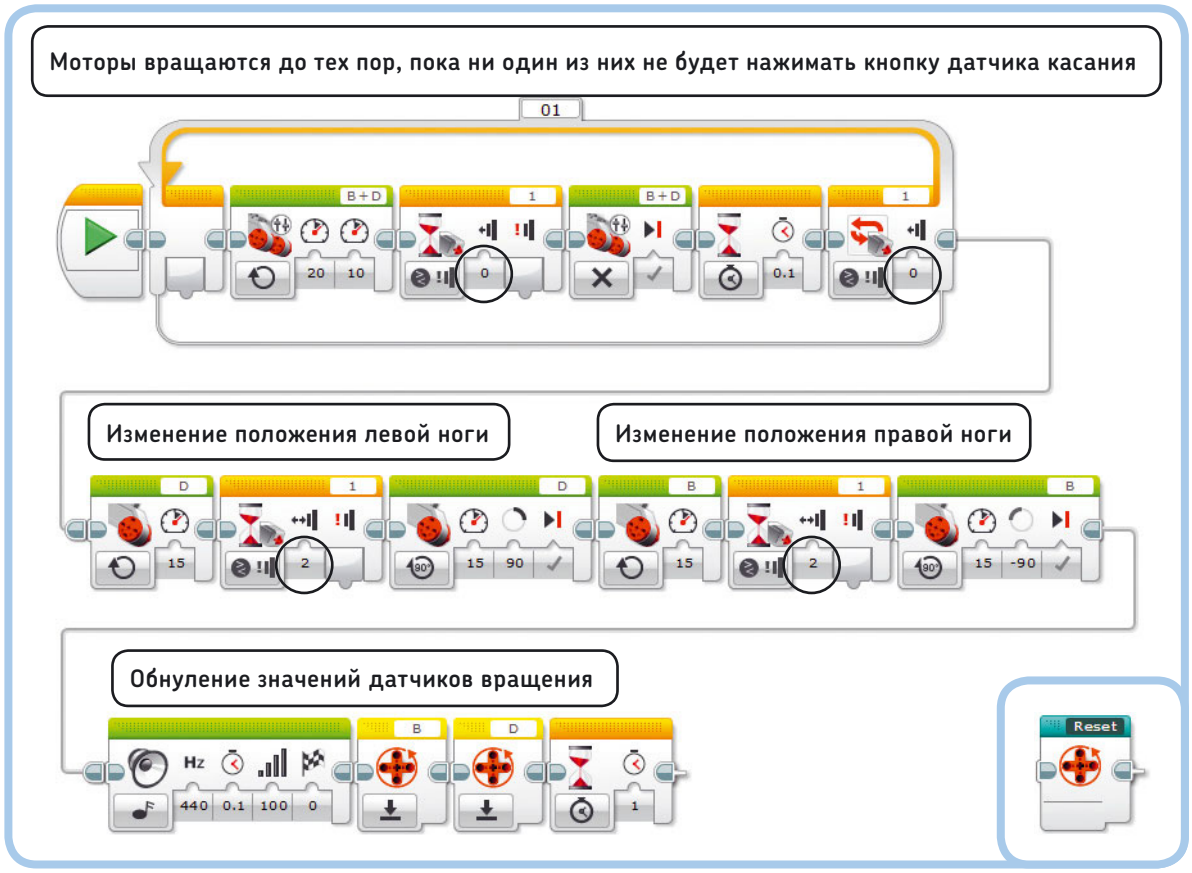


*Рис. 19.5. Конфигурация блоков в контейнере **Reset** (слева) и готовый контейнер «Мой блок» (внизу справа)*

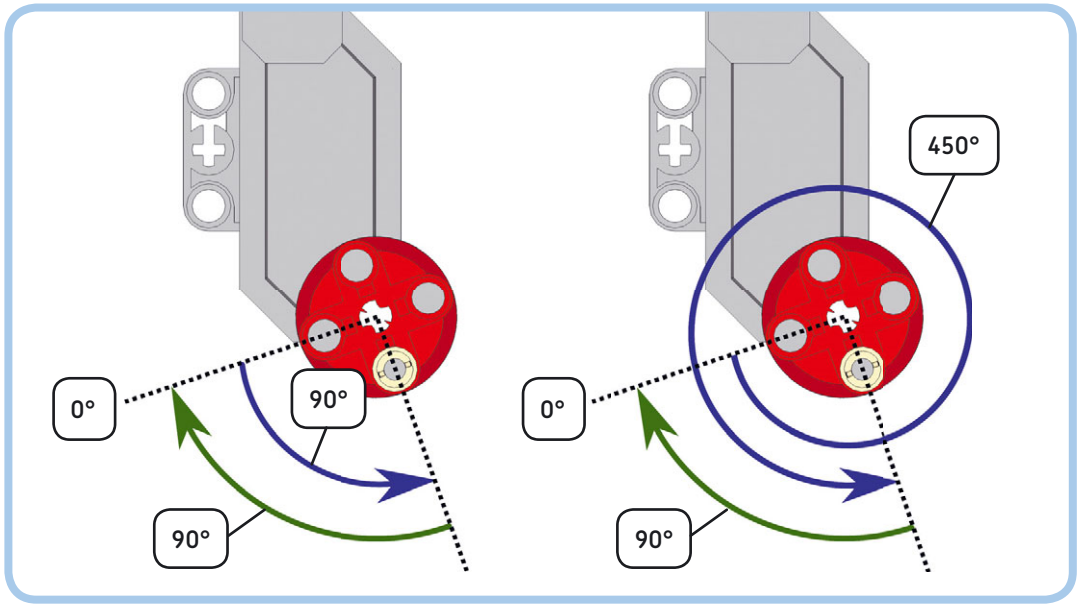


Рис. 19.6. Когда мотор провернулся вперед (синяя стрелка), он может вернуться в нулевую позицию (0°), вычислив свое текущее положение (90°) и прокрутившись назад (зеленая стрелка) на это же число градусов. Если мотор сделал более одного оборота, не нужно прокручивать его обратно на это же количество оборотов, достаточно вернуть его назад на то количество градусов, которое превышает целое число оборотов, как показано справа (450° – 360° = 90°)

Вы можете подсчитать, на сколько градусов текущее положение мотора превышает число полных оборотов, с помощью оператора деления по модулю, %. Оператор деления по модулю позволяет вычислить остаток от деления нацело. Например, 7 разделить на 3 равно 2, 1 в остатке, т. е. 7 % 3 = 1. Оператор деления по модулю доступен в режиме **Дополнения** (Advanced) блока **Математика** (Math).

Чтобы определить требуемое число, вычислите остаток от деления на 360. Например, 450 % 360 = 90.

Другими словами, полные обороты исключаются из подсчета, чтобы сократить обратный путь мотора (который в этом случае будет меньше 360 градусов). В результате механизм возвращается в начальную позицию.

Создайте контейнер «Мой блок» с именем **Return**, с помощью которого оба механизма будут возвращаться в исходную позицию (рис. 19.7).

ПРИМЕЧАНИЕ **В блоке Математика (Math) деление (÷) обозначается наклонной чертой (/). Умножение (×) обозначается звездочкой (*). Оператор деления по модулю обозначается символом процента (%). Чтобы не вводить каждый символ вручную, можно выбрать его из списка, который раскрывается при вводе вычислительной задачи.**

Контейнер «Мой блок» № 3: Синхронизация

Чтобы робот пошел после того, как механизмы ног оказались в противоположных друг другу позициях, оба его мотора должны начать крутиться вперед со скоростью 20% (34 оборота в минуту). Может показаться, что эту задачу отлично решит блок **Рулевое управление** (Move Steering) в режиме **Включить** (On), но небольшие расхождения в скорости в конечном итоге приведут к тому, что один мотор начнет отставать от другого, и механизмы ног утратят свое противопоставленное друг другу положение. Поэтому, чтобы моторы работали синхронно, вам придется придумать что-то другое. Вам нужно создать блок, который будет прокручивать оба мотора со средней скоростью 20%, при этом отслеживая, чтобы значение датчика вращения мотора D было (приблизительно) равно значению датчика вращения мотора B.

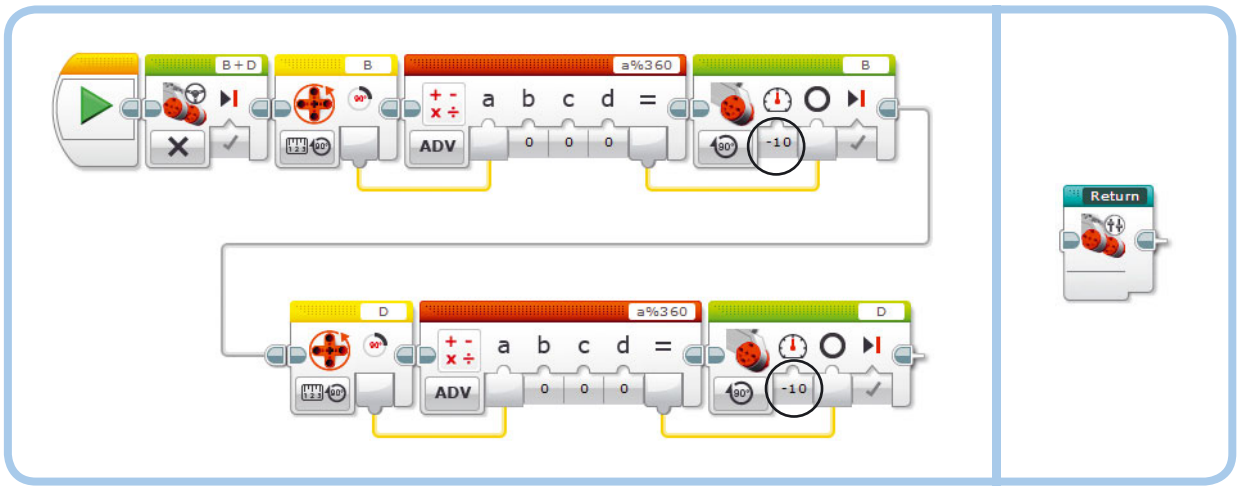


*Рис. 19.7. Конфигурация блоков в контейнере **Return** (слева) и готовый контейнер «Мой блок» (справа)*

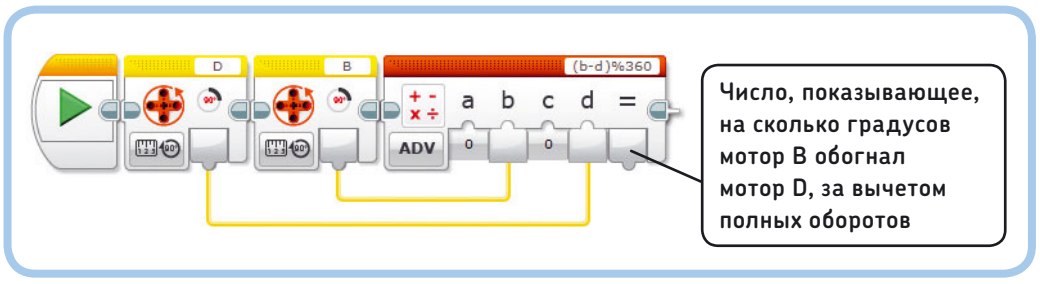


*Рис. 19.8. Шаг 1: первые блоки, входящие в контейнер **OnSync**, предназначены для определения расхождения в положении моторов*

Для этого необходимо, чтобы скорость вращения мотора становилась немного выше, чем 20%, если он отстает от другого, а скорость второго мотора становилась немного ниже, чем 20%. Чем сильнее расхождение в движении моторов, тем сильнее нужно корректировать скорость. Например, если датчик вращения мотора B фиксирует 790 градусов, а датчик вращения мотора D — 750 градусов, нужно увеличить скорость мотора D до 22%, а скорость мотора B снизить до 18%, чтобы мотор D мог нагнать мотор B.

Чтобы вычислить, на какую величину необходимо изменить скорость (в данном случае на 2%), сначала нужно определить различие в положении моторов. Для этого вычитаем положение мотора D из положения мотора B (в нашем примере это 790 – 750 = 40 градусов). Как и в случае с контейнером **Return**, к этому числу мы применяем операцию деления по модулю. Дело в том, что, если мотор отстает более чем на один оборот, ему будет гораздо проще нагнать второй мотор, прокрутившись лишь на то значение, которое превышает число полных оборотов (вы получите тот же результат при меньших усилиях). Расположите и настройте необходимые для решения этой задачи блоки, как показано на рис. 19.8.

Теперь вы знаете, на сколько градусов мотор B обогнал мотор D. Иногда эта цифра больше 180 градусов, иногда меньше –180. Например, мотор B может на 220 градусов опережать мотор D. В этом случае из указанного числа нужно вычесть 360 градусов (220 – 360 = –140), чтобы определить оптимальное значение и направление для вращения мотора. Другими словами, теперь можно сказать, что мотор B на 140 градусов отстает от мотора D. Поскольку моторы вращаются по кругу, суть будет та же, а корректировка скорости — меньше, и движение робота будет менее резким.

Точно так же, если различие меньше –180 градусов, нужно прибавить к получившемуся числу 360 градусов. Чтобы воплотить эти шаги на практике, добавьте в программу блоки **Сравнение** (Compare) (рис. 19.9). Блок **Сравнение** (Compare) выдаст результат **Истина** (True) (1), если различие между моторами будет больше 180 градусов, и **Ложь** (False) (0), если нет. В первом случае блок **Математика** (Math) вычтет 1 × 360 = 360

Рис. 19.9. Шаг 2: эти блоки затем определяют различие в положении моторов и находят самый быстрый способ преодолеть это различие. Для этого они вычитают 360 градусов из разницы значений, если она больше 180 градусов, или прибавляют 360 градусов, если она меньше –180 градусов

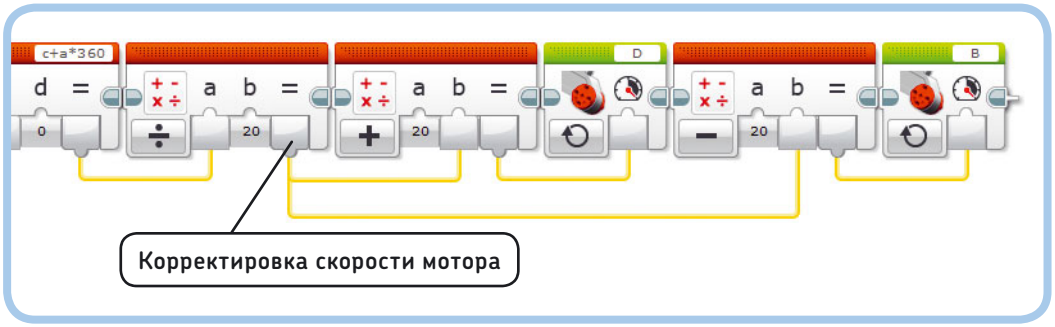


Рис. 19.10. Шаг 3: чтобы понять, как нужно скорректировать скорость, разделите разницу между моторами на 20. Затем прибавьте получившееся число к скорости мотора D, чтобы ускорить его вращение, и вычтите из скорости мотора B, чтобы его замедлить. Этот блок эффективен и в том случае, если мотор D опережает мотор B: разница между ними будет отрицательной, и мотор B разгонится, чтобы нагнать мотор D

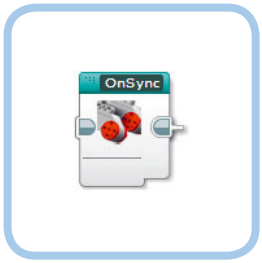


Рис. 19.11. Шаг 4: преобразуйте блоки в контейнер «Мой блок» ***OnSync***

из разницы значений, а в последнем — вычтет из разницы значений 0 × 360 = 0, т.е. оставит все без изменений. Вторая группа блоков **Сравнение** (Compare) и **Математика** (Math) действует точно так же, но тут, если разница значений меньше –180 градусов, к ней приплюсовывается 360.

Теперь мы знаем разницу значений (в градусах) между мотором B и мотором D, и это число корректируется с оглядкой на то, какое различие в положении моторов быстрее всего преодолеть. Если разница составляет 40 градусов, нужно изменить скорость каждого мотора на 2%, т. е. мы делим разницу на 20 и получаем число, которое нужно прибавить к скорости мотора D и вычесть из скорости мотора B (рис. 19.10). Преобразуйте созданные блоки в контейнер «Мой блок» с именем **OnSync**, как показано на рис. 19.11.

Может показаться, что контейнер **OnSync** слишком сложно устроен, но он действует очень понятно. Просто помните, что, если поместить его в блок **Цикл** (Loop), он будет функционировать как блок **Рулевое управление** (Move Steering) в режиме **Включить** (On), но при этом оба мотора будут работать синхронно, их скорость будет корректироваться, если один из них окажется впереди другого. Это обязательно, если вы хотите, чтобы робот LAVA R3X нормально ходил.

Контейнер «Мой блок» № 4: Поворот налево

Робот LAVA R3X может поворачивать налево. Для этого правый мотор должен крутиться в обратном направлении, а левая стопа должна быть зафиксирована. Положение левой стопы (на 120 градусов назад относительно исходной позиции) должно быть таким, чтобы стопа касалась пола каждый раз, когда правый мотор совершает один оборот. Каждый раз, когда левая стопа касается пола, робот немного уходит влево. То, насколько сильно робот поворачивается, зависит от типа покрытия, по которому он идет, но в среднем при вращении правого мотора на 10 оборотов назад робот поворачивается налево примерно на 90 градусов.

Чтобы соответствующий контейнер «Мой блок» работал независимо от текущей позиции моторов, поместите в его начало и конец контейнер **Return**. Создайте контейнер «Мой блок» с именем **Left**, как показано на рис. 19.12.

Первые шаги

В программе WalkTest описанные контейнеры «Мой блок» нужны, чтобы робот LAVA R3X в течение 15 секунд шел вперед, а затем поворачивал налево, и так постоянно (рис. 19.13).

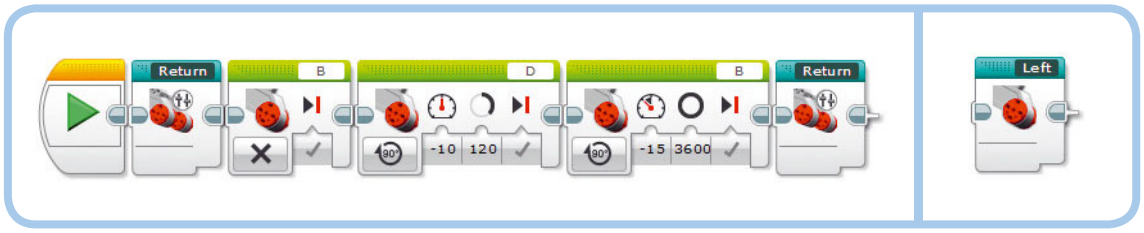


*Рис. 19.12. Конфигурация блоков в контейнере **Left** (слева) и готовый контейнер «Мой блок» (справа)*

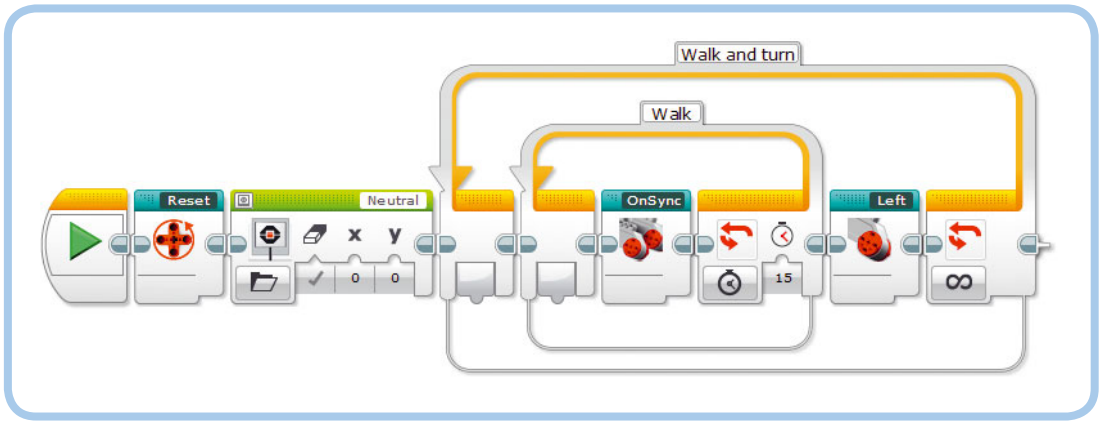


Рис. 19.13. Программа WalkTest

Поставьте робота на гладкую ровную поверхность, например на деревянный пол, и запустите программу. Контейнер **Reset** в начале программы ставит ноги робота противоположно друг другу. Звуковой сигнал, за который отвечает блок **Звук** (Sound) в контейнере **Reset**, указывает на то, что процесс обнуления настроек завершен и робот готов к ходьбе. Внутренний блок **Цикл** (Loop) и контейнер **OnSync** заставляют робота идти вперед в течение 15 секунд, а внешний блок **Цикл** (Loop) приводит к тому, что ходьба и поворот повторяются.

ПРИМЕЧАНИЕ **Если робот ходит неправильно, посмотрите готовую программу в архиве, доступном по ссылке eksmo.ru/files/Lego_Mindstorms_Primers.zip, чтобы сравнить ее с версией, созданной вами.**

ПРАКТИКУМ № 123: КОНТЕЙНЕР «ПРОГУЛКА»!

Сложность: **Время:**

Попробуйте создать контейнер «Мой блок», благодаря которому робот будет идти вперед в течение заданного количества секунд. Создайте контейнер с именем **Walk** с одной числовой переменной с именем **Seconds**, как показано на рис. 19.14.

СОВЕТ **Преобразуйте внутренний блок Цикл (Loop), показанный на рис. 19.13, в контейнер «Мой блок» и с помощью числовой переменной настройте количество итераций цикла.**

*Рис. 19.14. Контейнер **Walk***

ПРАКТИКУМ № 124: ЗАДОМ НАПЕРЕД!

Сложность: **Время:**

Попробуйте создать модифицированную версию контейнера **OnSync**, чтобы робот LAVA R3X шел задом наперед. Пусть моторы крутятся со скоростью –20% вместо 20%, при этом они должны работать синхронно.

СОВЕТ **Для этого нужно всего лишь сделать копию контейнера OnSync (назовем ее OnRev) и изменить значения двух параметров. От каких параметров зависит средняя скорость моторов?**

ПРАКТИКУМ № 125: ПОВОРОТ НАПРАВО!

Сложность: **Время:**

Создайте контейнер «Мой блок» с именем **Turn** с одной логической переменной с именем **Direction**, как показано на рис. 19.15. Пусть робот поворачивает налево, если вы выберете значение **Истина** (True), и направо, если выбрано значение **Ложь** (False). Чтобы посмотреть, как этот блок работает, замените им контейнер **Left** в программе WalkTest.

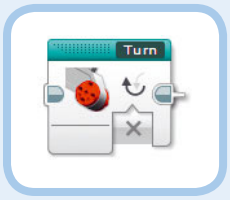
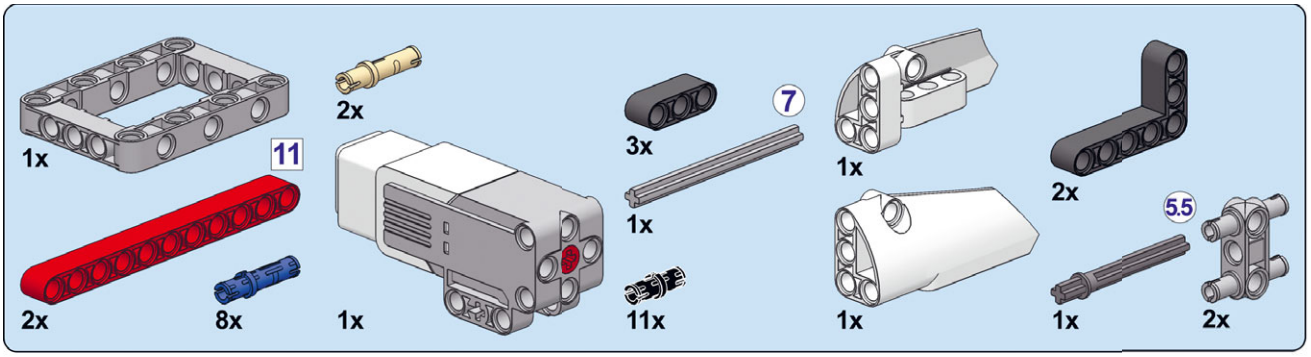
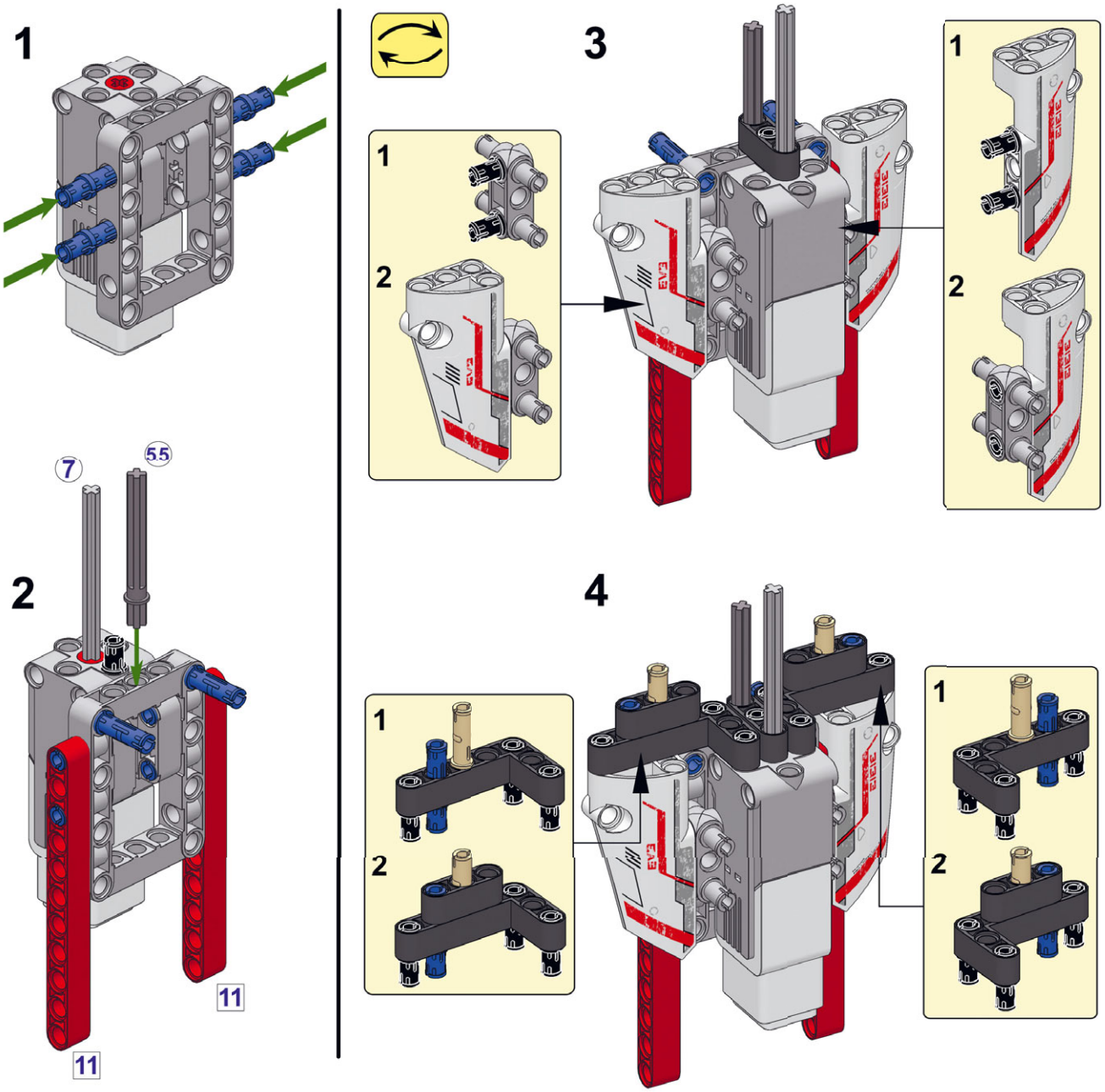


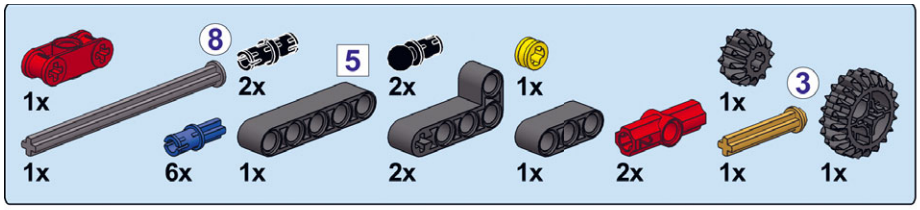
*Рис. 19.15. Контейнер **Turn***

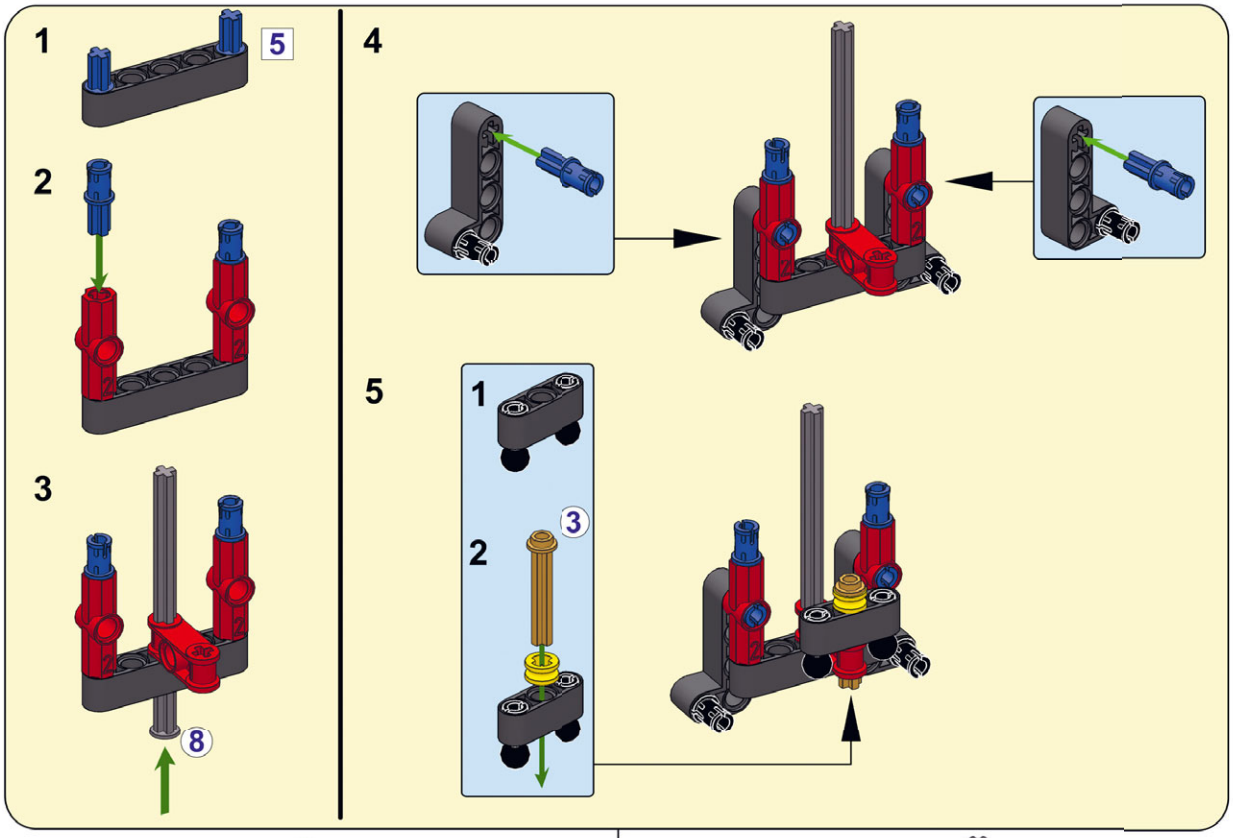
Сборка головы и рук

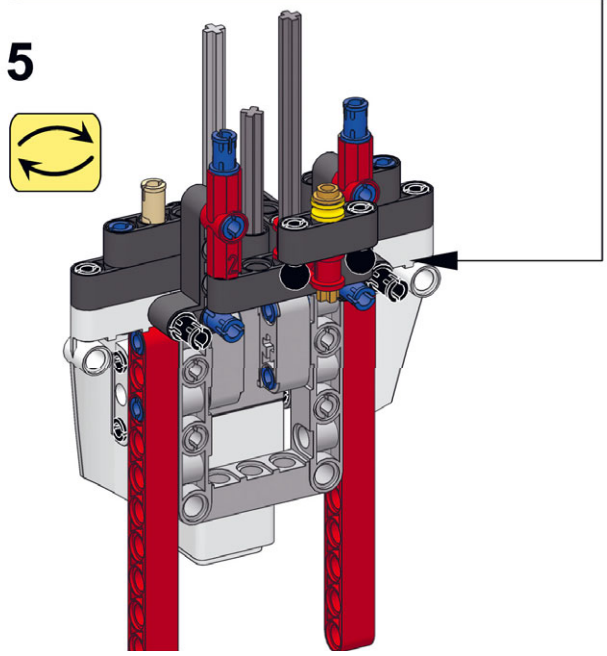
Теперь соберем голову и руки и построим целого робота с помощью инструкций на следующих страницах. Когда все будет готово, убедитесь, что подвижные элементы руки не мешают кабелям в верхней части модуля EV3. Чтобы это проверить, вручную поверните ось, подсоединенную к среднему мотору, и передвиньте кабели, если есть такая необходимость.

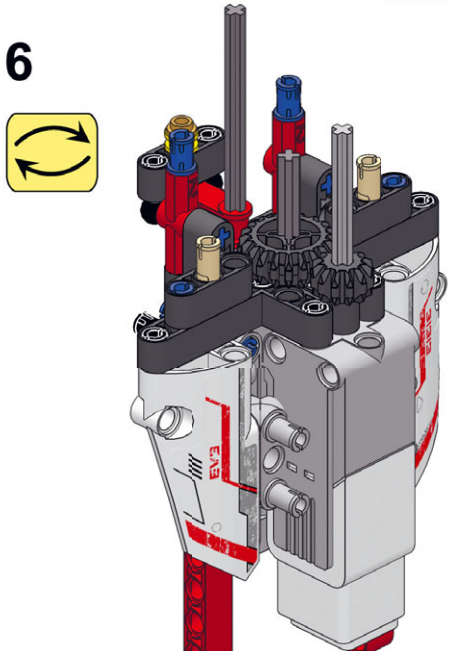

1x
11
2x
8x
2x
1x
3x
7
1x
11x
1x
2x
1x
5.5
2x

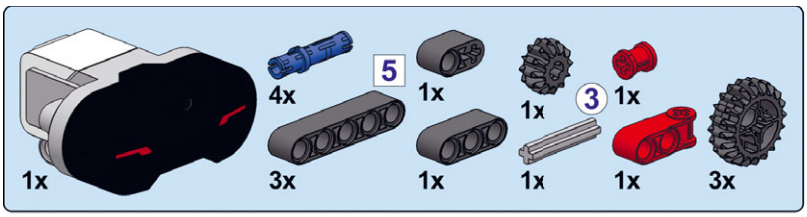

1
2
7
5.5
11
11
3
1
2
1
2
4
1
2
1
2


1x
8
1x
6x
2x
5
1x
2x
2x
1x
1x
2x
1x
3
1x
1x

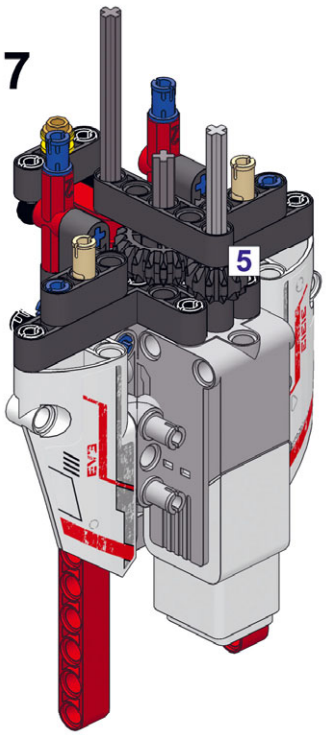

1
5
2
3
8
4
5
1
2
3


5

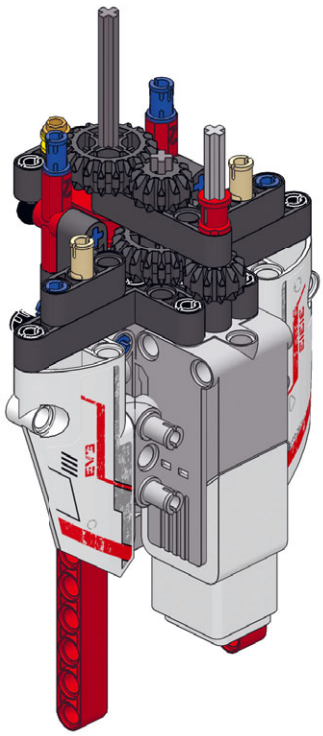

6
EV3



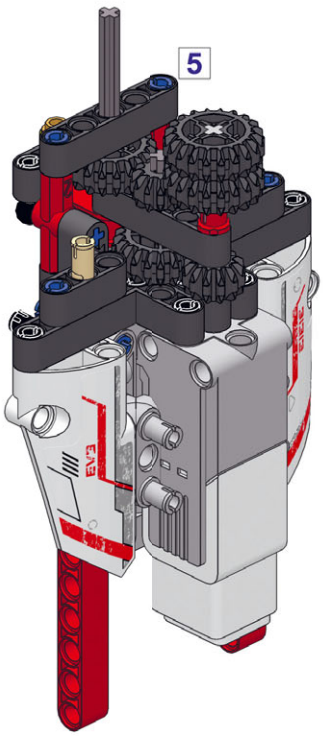
7



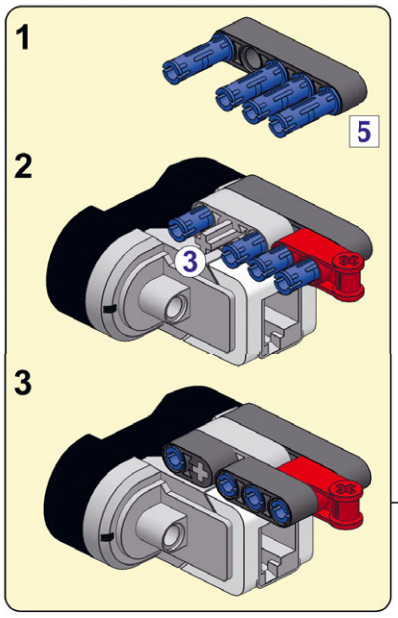
8

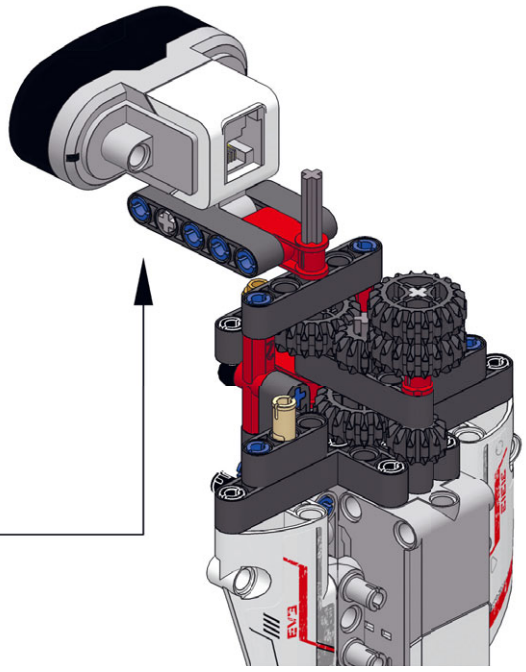


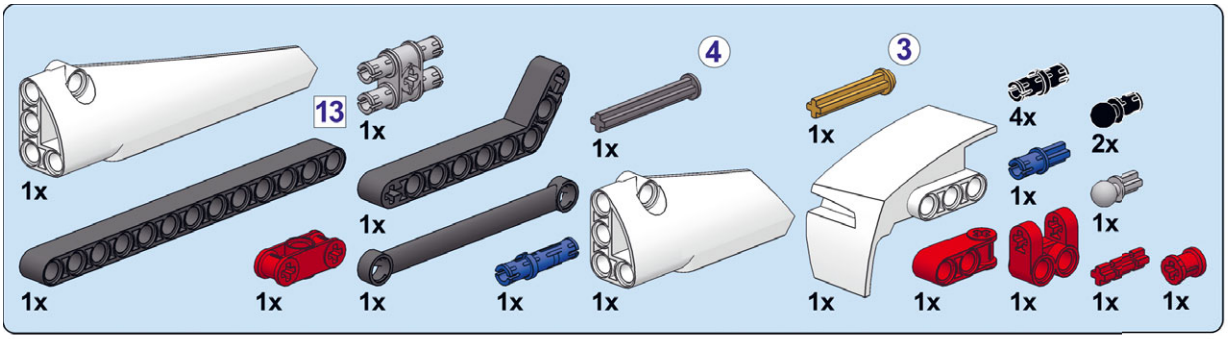
9

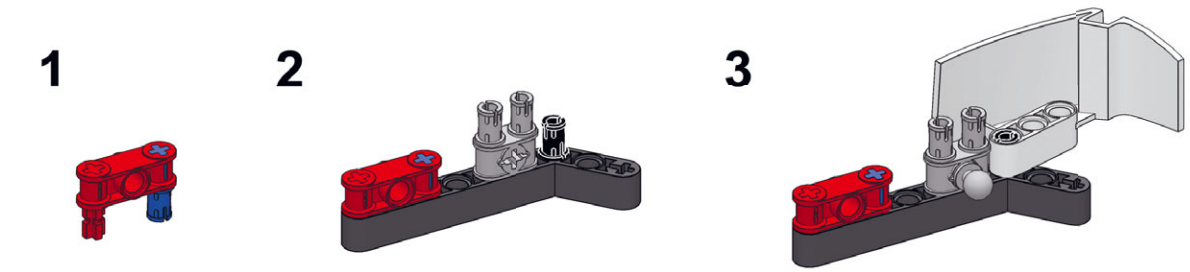


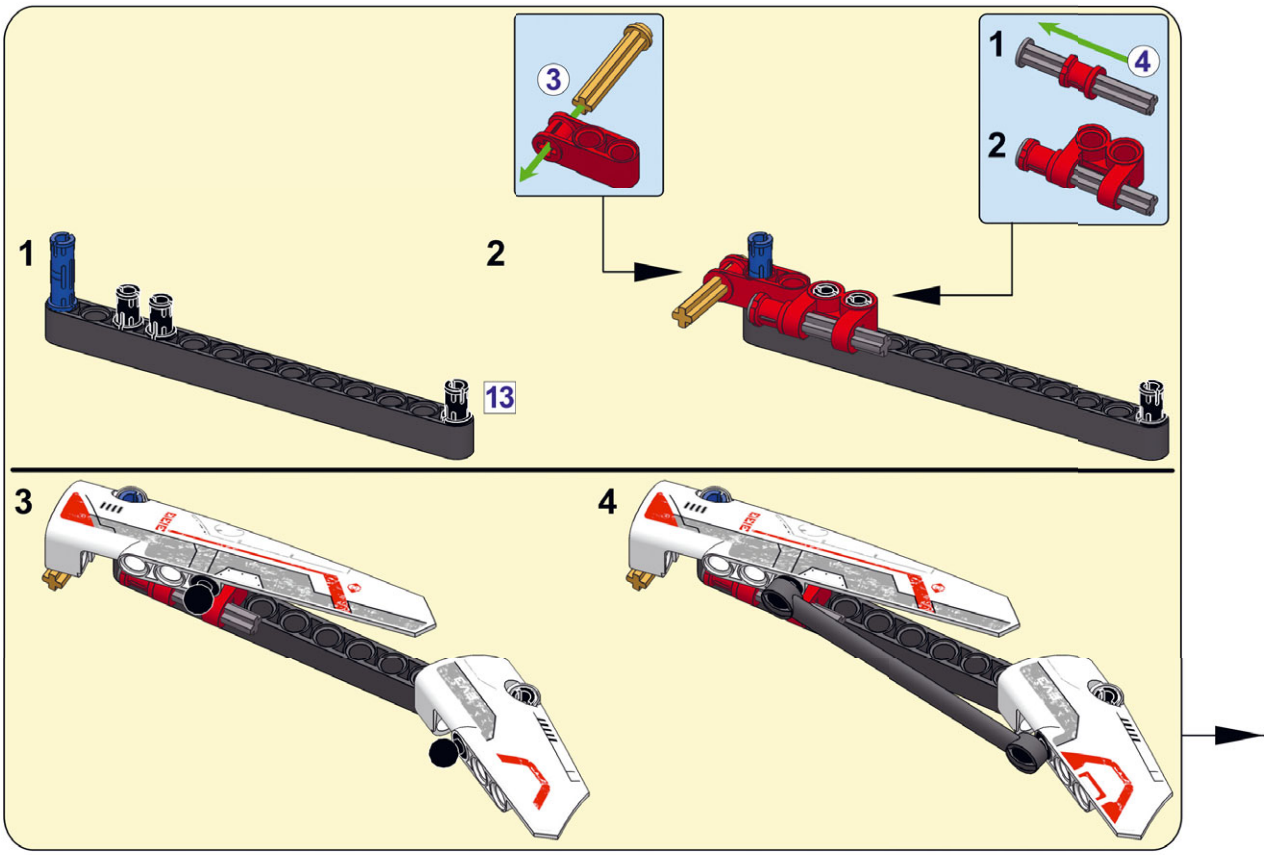
10



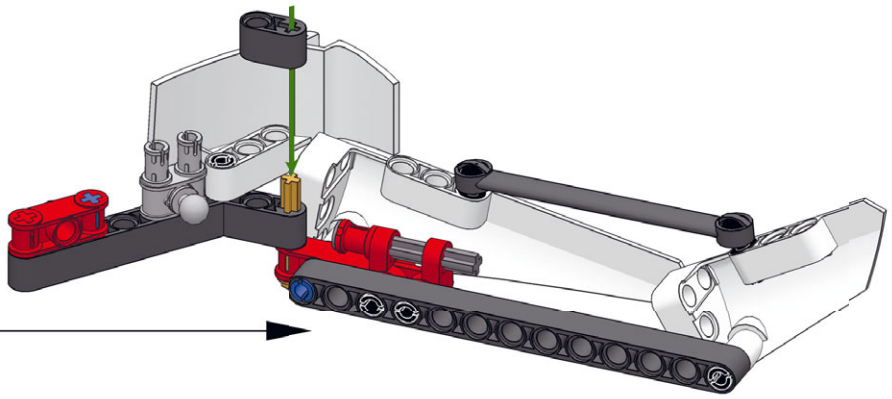


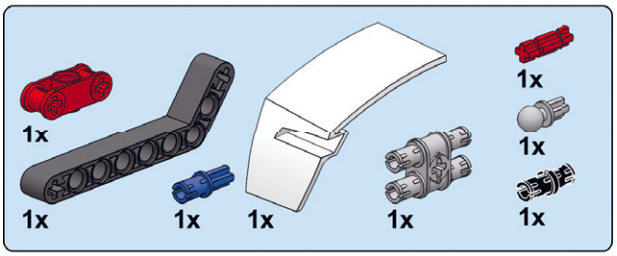

13
1x
1x
1x
1x
1x
1x
4
1x
1x
3
1x
1x
4x
1x
2x
1x
1x
1x
1x


1
2
3

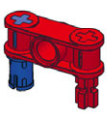

3
1
4
2
1
13
2
3
4

4

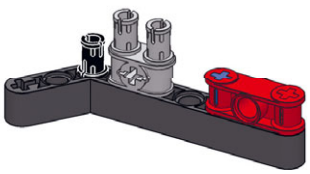




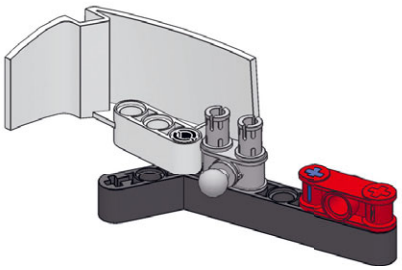
1

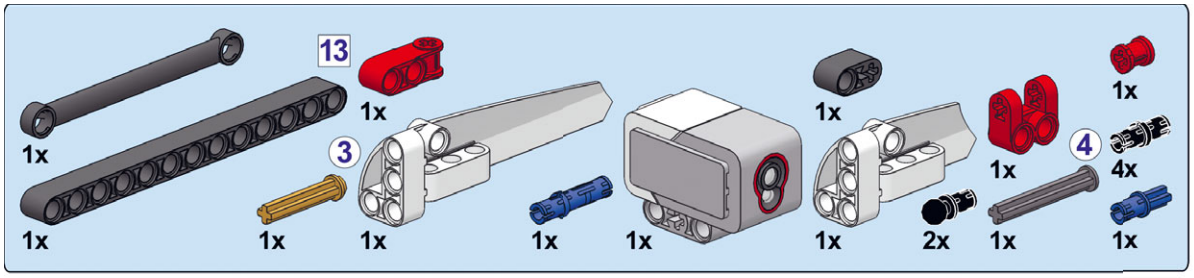


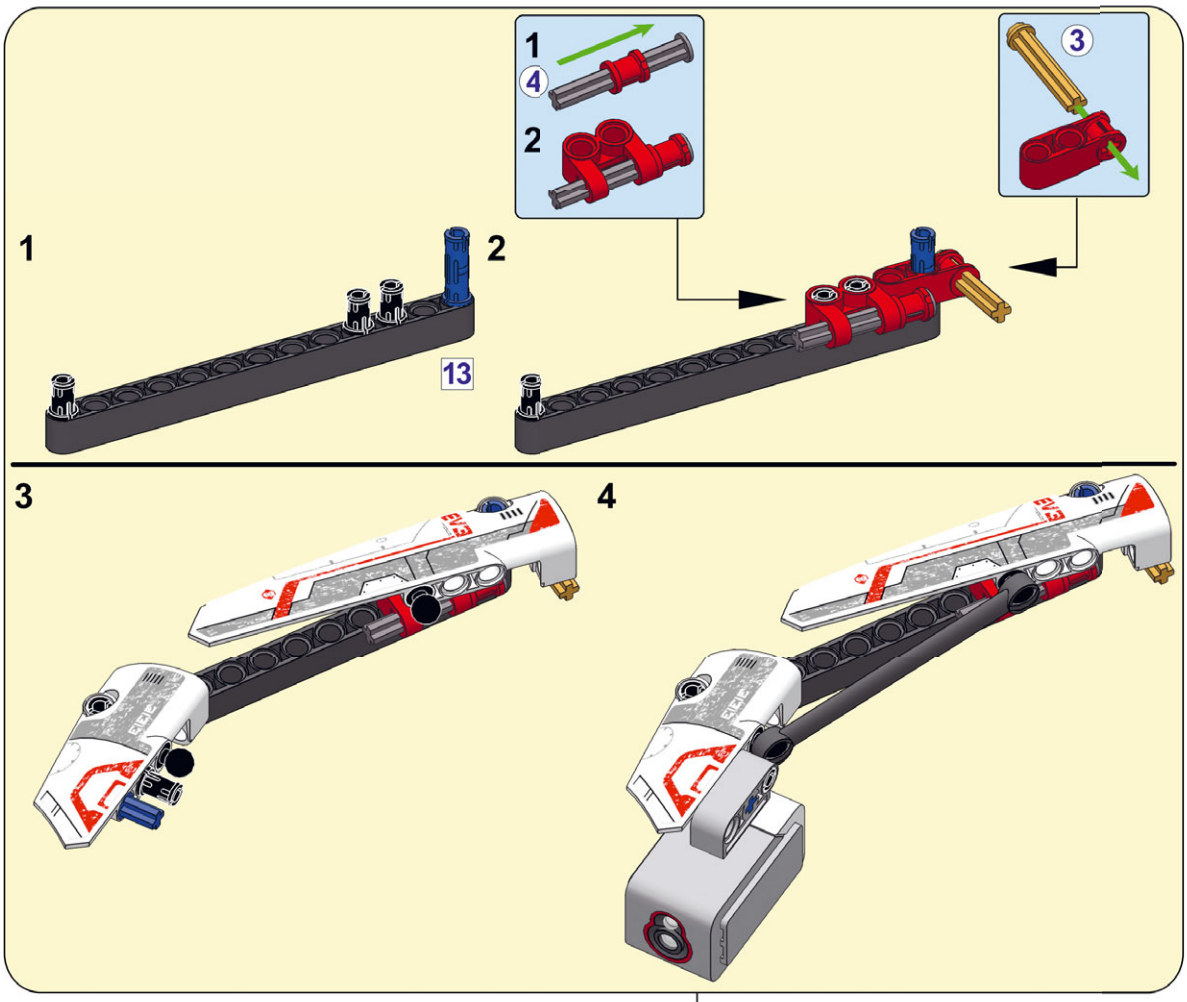
2

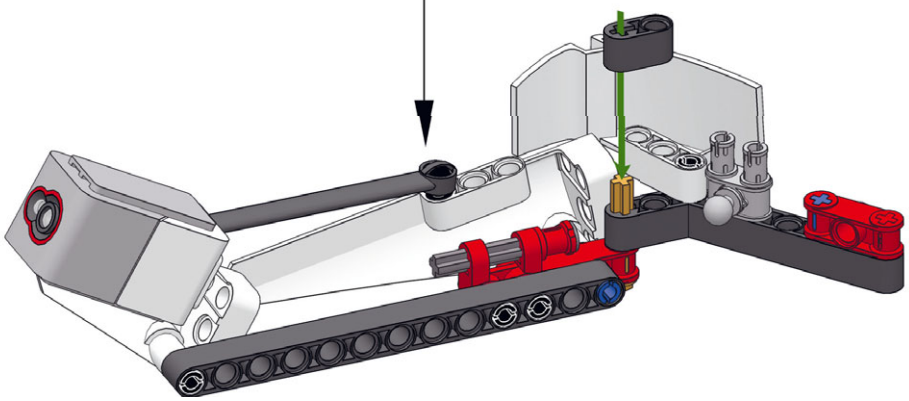


3

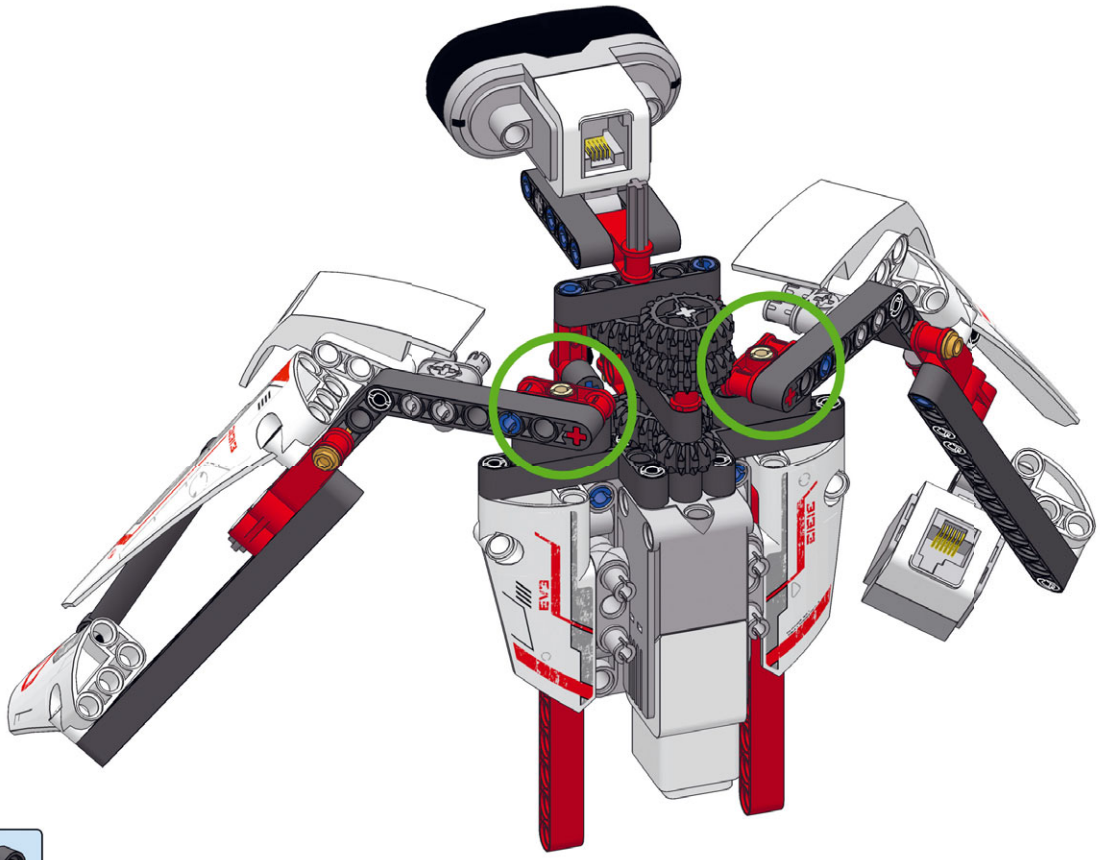


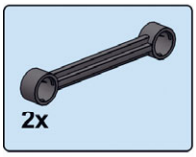

1x
1x
13
1x
3
1x
1x
1x
1x
1x
2x
1x
4
1x
4x
1x


1
4
2
3
1
2
13
3
4

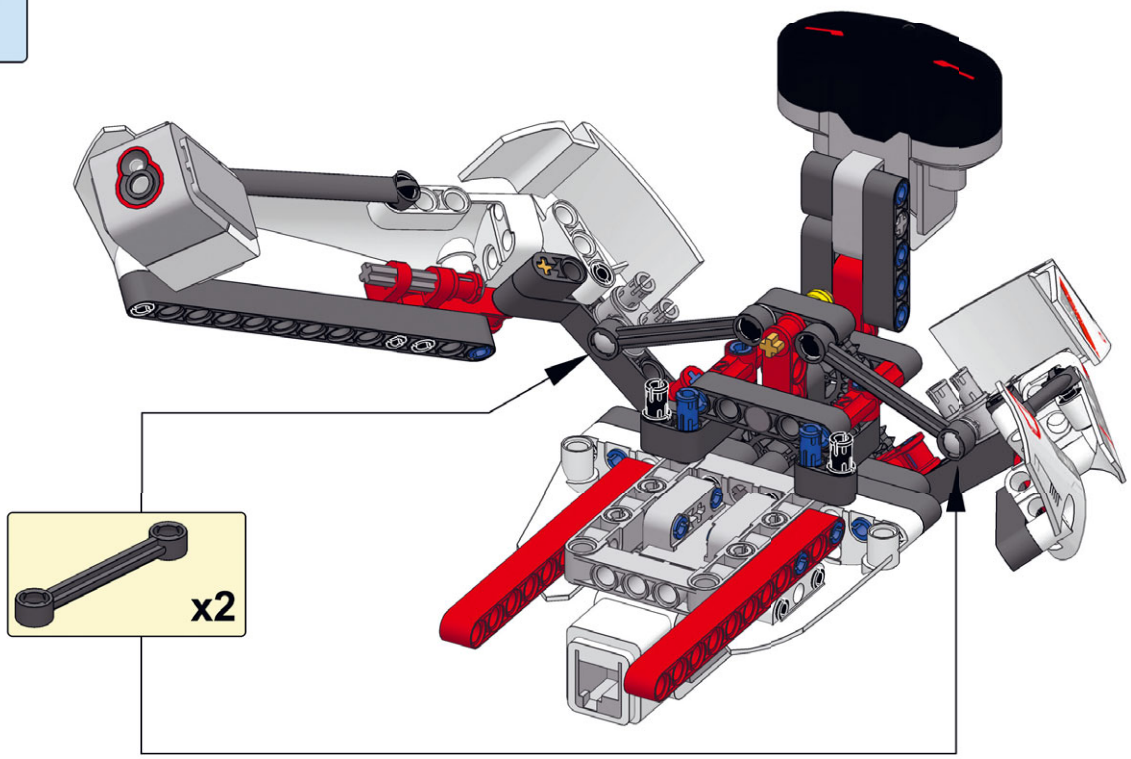

4

5

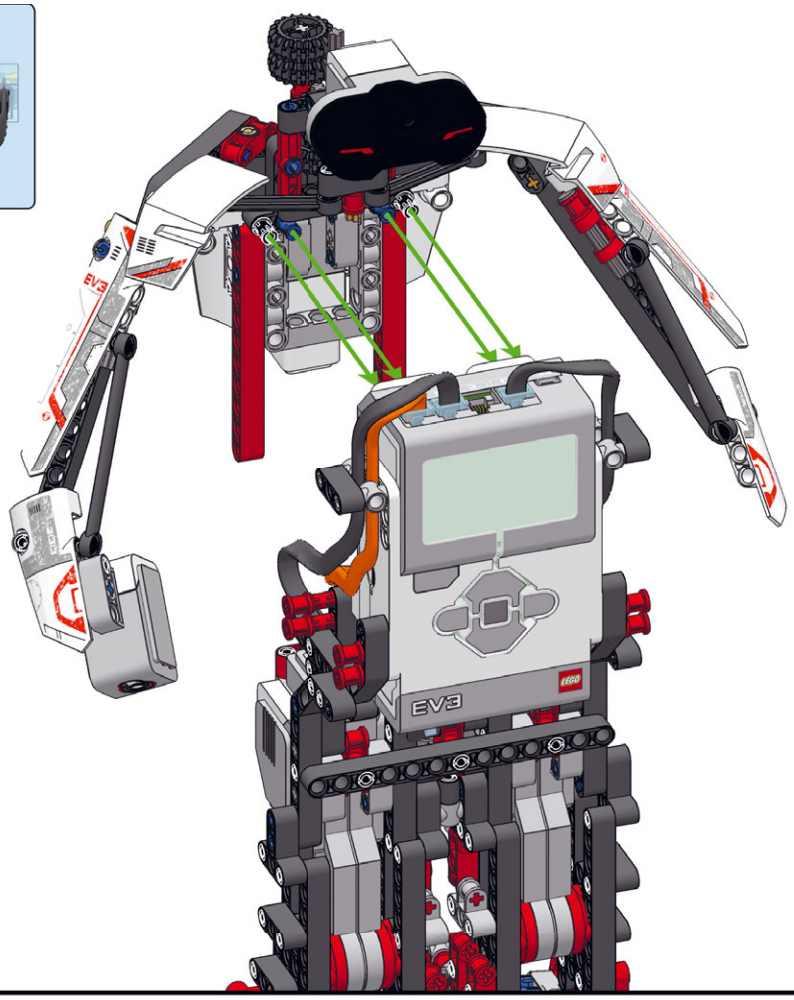




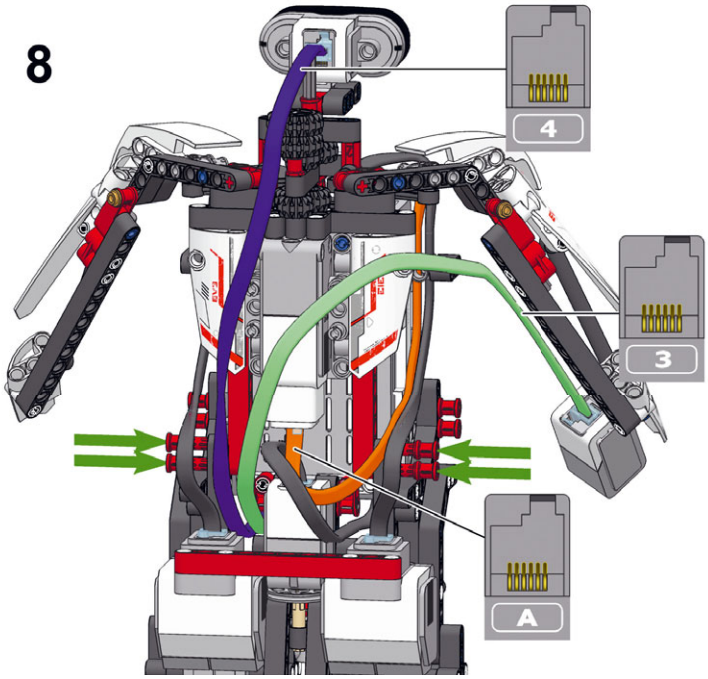
6

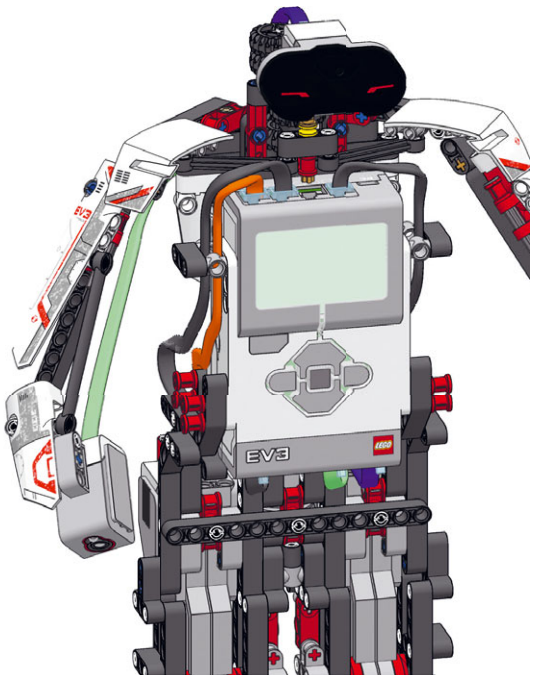


7



8





Управление головой и руками

Построив робота целиком, можно приступать к разработке программы, благодаря которой робот сможет ходить, двигать головой и руками и реагировать на окружающие предметы.

Управление движениями рук и головы осуществляется с помощью среднего мотора: когда мотор крутится вперед, голова и руки движутся вправо; когда мотор крутится назад, они движутся влево.

Контейнер «Мой блок» № 5: Голова

Чтобы было проще управлять движением головы и рук, создайте контейнер «Мой блок» и разместите его параллельно основной программе, отвечающей за ходьбу.

Контейнер **Head** сначала устанавливает средний мотор в заданное положение, повернув голову вправо до упора. Затем он заставляет голову двигаться влево-вправо, как показано на рис. 19.16. Благодаря этому движению инфракрасный датчик фиксирует препятствия не только впереди него, но и по сторонам.

Избегание препятствий и реагирование на рукопожатие

Теперь, когда вы создали контейнеры «Мой блок», можно без труда разрабатывать программы, которые заставят робота ходить и реагировать на датчики. Например, в программе WalkTest можно изменить внутренний блок **Цикл** (Loop) таким образом, чтобы робот не просто шел вперед в течение 15 секунд, а двигался до тех пор, пока инфракрасный датчик не обнаружит препятствие на пути.

Финальная программа научит робота обходить препятствия и отвечать на рукопожатия. Если вы пожмете роботу правую руку, он остановится, скажет: «Привет, доброе утро»*, а затем пойдет дальше. Если робот увидит препятствие, он скажет: «Обнаружено»** и повернет налево.

Перезагрузка ног и движения головы

Программа начинается с того, что ноги ставятся противоположно друг другу с помощью контейнера **Reset**. Затем запускается цикл, заставляющий робота ходить и реагировать на датчики. Во время ходьбы робот поворачивает голову влево-вправо с помощью контейнера **Head**, прикрепленного к собственному блоку **Начало** (Start) (рис. 19.17). Эта конфигурация позволяет управлять головой и ногами независимо друг от друга: можно изменить параметры любой последовательности блоков, не беспокоясь о том, что изменения повлияют на вторую цепочку. Можно изменить движение головы, настроив контейнер **Head**, а можно изменить манеру ходьбы, откорректировав блоки внутри блока **Цикл** (Loop).

Создайте новую программу под названием ObstacleAvoid и добавьте блоки, показанные на рис. 19.17.

Остановка робота после срабатывания любого датчика

Теперь вы добавите в основной цикл блоки, которые заставят робота идти вперед, пока он не обнаружит препятствие или не зафиксирует рукопожатие (рис. 19.18).

Робот идет вперед благодаря тому, что контейнер **OnSync** программирует большие моторы крутиться вперед. Этот блок

* Пункты **Hello**, **Good** и **Morning** в каталоге звуков Lego.

** Пункт **Detected** в каталоге звуков Lego.

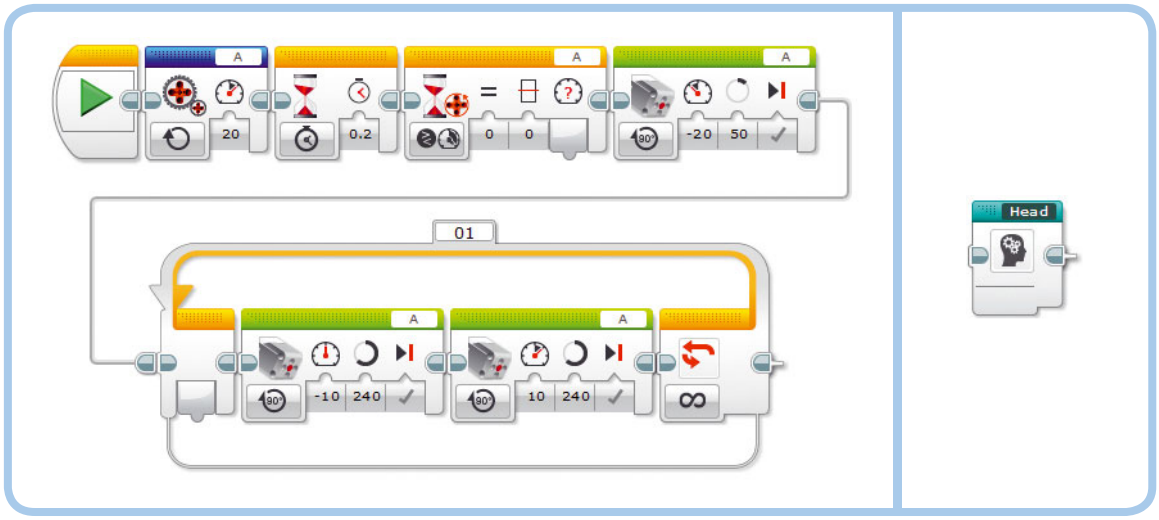


*Рис. 19.16. Конфигурация блоков в контейнере **Head** (слева) и готовый контейнер «Мой блок» (справа)*

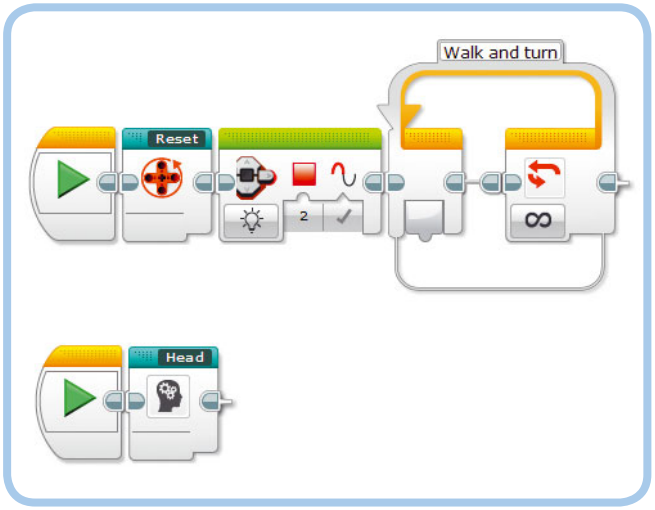


*Рис. 19.17. Шаг 1: эти блоки устанавливают ноги противоположно друг другу и приводят голову робота в движение. Обратите внимание, что выполнение блока **Цикл** в составе контейнера **Head** не ограничено по времени, поэтому голова постоянно поворачивается влево-вправо*

зациклен, поэтому скорость моторов постоянно корректируется и моторы вращаются синхронно. Цикл воспроизводится до тех пор, пока не среагирует инфракрасный датчик (обнаружив препятствие), датчик цвета (зафиксировав рукопожатие) или оба датчика одновременно.

Блок **Датчик цвета** (Color Sensor) может распознать рукопожатие, сравнив текущее значение параметра **Яркость отраженного света** (Reflected Light Intensity) с пороговым значением. Если датчик фиксирует значение, превышающее 10%, робот определяет вашу руку и результат будет — **Истина** (True). Если датчик фиксирует значение 10% или ниже, датчик не определяет вашу руку и результат — **Ложь** (False). Точно так же блок **Инфракрасный датчик** (Infrared Sensor) настроен на вывод результата **Истина** (True), если зафиксированное значение приближения меньше 50%. В противном случае он выдаст результат **Ложь** (False).

Блок **Логические операции** (Logic Operations) сравнивает полученные результаты. Он выдаст результат **Истина** (True), если хотя бы один из полученных результатов — **Истина** (True). В этом случае цикл завершится.

Реакция робота на сигнал датчика

Теперь робот может реагировать на сигнал датчика. Если сигнал поступил от инфракрасного датчика, робот скажет: «Обнаружено» и повернет налево. Если от датчика цвета — робот на время остановится и скажет: «Привет, доброе утро».

После завершения цикла вы сможете определить, какой именно датчик вызвал остановку. Для этого нужно посмотреть на данные блока **Инфракрасный датчик** (Infrared Sensor): если он выдает результат **Истина** (True), то остановку вызвал инфракрасный датчик, а если **Ложь** (False), то датчик цвета. Если цикл по какой-то причине завершился, но при этом инфракрасный датчик не зафиксировал препятствие на пути, можно сделать вывод, что причиной остановки стал датчик цвета.

Блок **Переключатель** (Switch), ориентируясь на полученный результат, отвечает за то, какие блоки запускать дальше, как показано на рис. 19.19.

Хоть это и маловероятно, но цикл может завершиться из-за того, что оба датчика среагировали на что-то одновременно, и тогда оба датчика будут показывать результат **Истина** (True). Поскольку блок **Инфракрасный датчик** (Infrared Sensor) в таком случае выдаст результат **Истина** (True), программа выполнит блоки в верхней части блока **Переключатель** (Switch), т.е. произойдет то же самое, что и в случае, когда сигнал подает только инфракрасный датчик. Другими словами, в подобных ситуациях программа игнорирует данные датчика цвета.

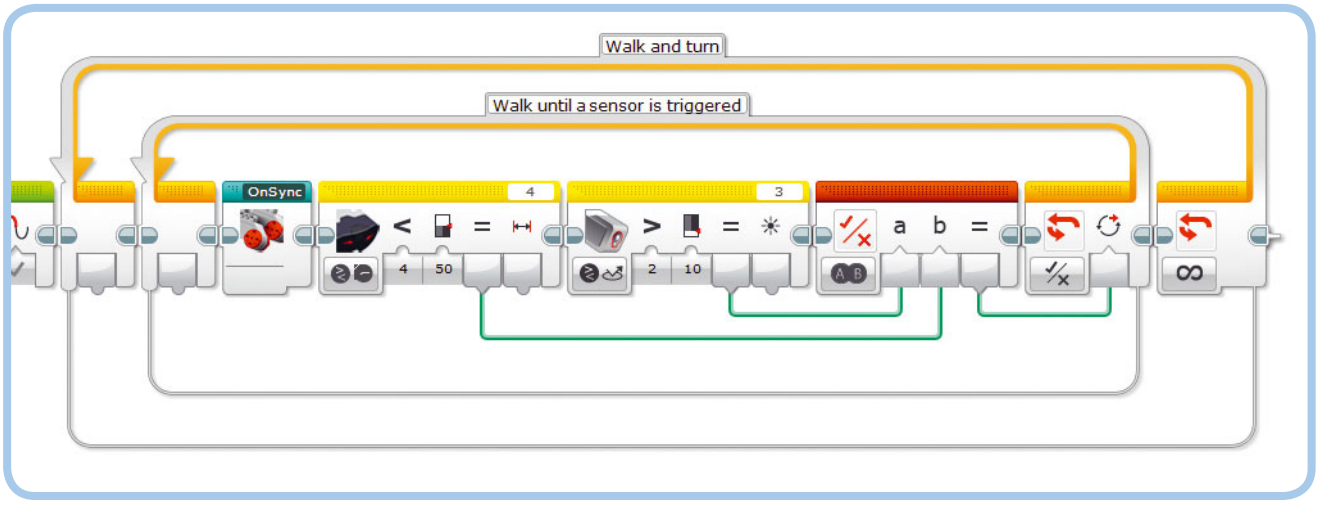


Рис. 19.18. Шаг 2: внутренний цикл заставляет робота идти вперед до тех пор, пока тот не обнаружит препятствие или не зафиксирует рукопожатие

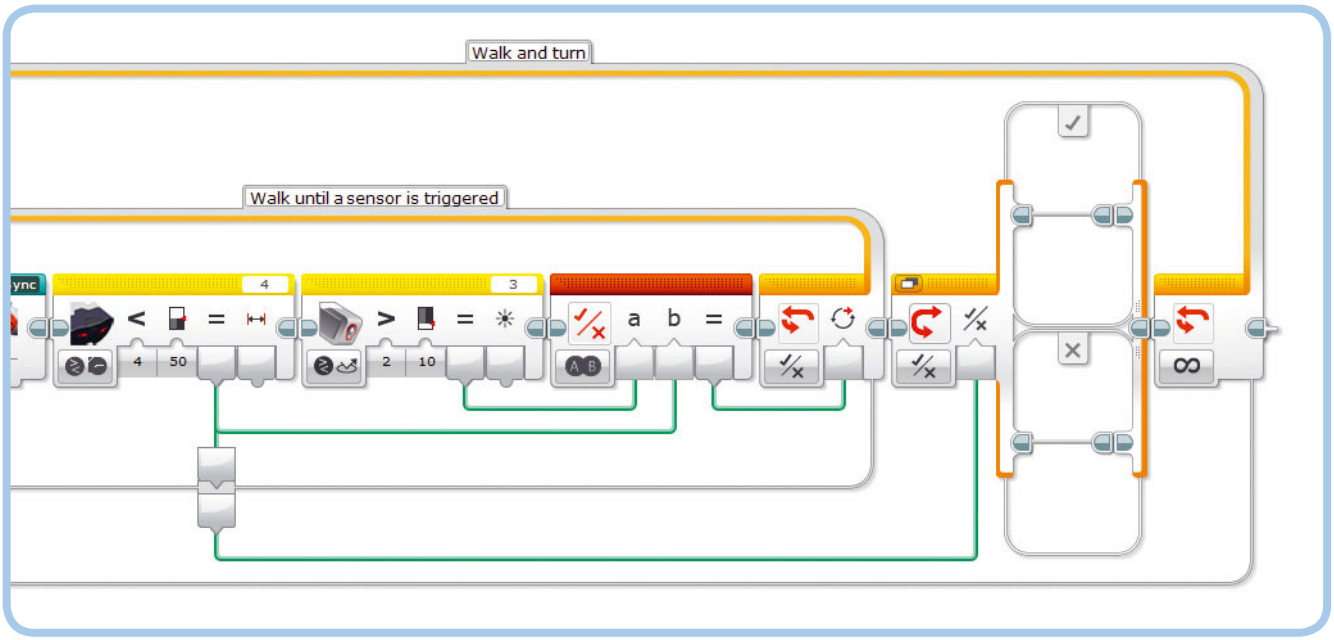


*Рис. 19.19. Шаг 3: блок **Переключатель** определил, какой из датчиков подал сигнал, и остановил цикл. Если сигнал исходил от инфракрасного датчика, будут выполняться блоки в верхней части (**Истина**) блока **Переключатель**; если нет — блоки в нижней части (**Ложь**)*

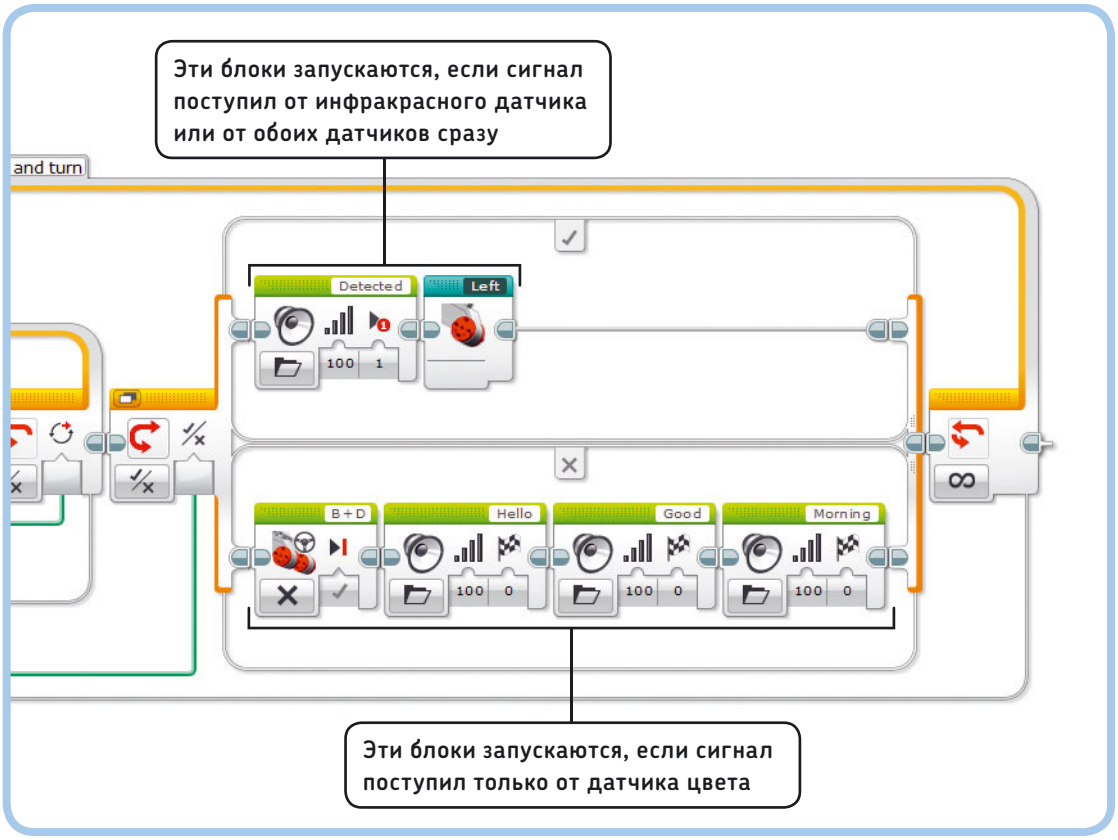


*Рис. 19.20. Шаг 4: робот поворачивает налево, если замечает препятствие (**Истина**), и останавливается, чтобы поздороваться с вами, если чувствует рукопожатие (**Ложь**)*

Добавьте в переключатель блоки, которые запрограммируют робота говорить: «Обнаружено» и поворачивать налево, когда результат — **Истина** (True), и говорить: «Привет, доброе утро», когда результат — **Ложь** (False) (рис. 19.20).

Теперь запустите программу и протестируйте ее. Если для программирования робота вы пользуетесь USB-кабелем, нужно вручную подвинуть голову немного вперед, чтобы освободить пространство для кабеля. Для этого поверните прикрепленные к среднему мотору зубчатые колеса. С помощью кнопки **Загрузить** (Download) перепишите программу в память модуля EV3 робота, отсоедините USB-кабель и запустите программу вручную с помощью кнопок на корпусе модуля EV3.

Теперь робот должен уметь сам ходить по помещению и здороваться с вами, если вы жмете ему руку.

ПРАКТИКУМ № 126: ТАНЦУЮЩИЕ РОБОТЫ!

Сложность: **Время:**

Робот LAVA R3X может ходить, ставя ноги в противоположные позиции и прокручивая оба мотора вперед. Что случится, если поставить ноги одинаково, а затем на скорости 10% сделать пять оборотов моторов в противоположных направлениях? Выполняя это задание, не беспокойтесь о синхронности моторов.

СОВЕТ **Когда вы с помощью контейнера Reset поставили ноги робота в противоположные позиции, можно без труда поставить ноги в одинаковое положение, прокрутив один мотор вперед. На сколько градусов вперед нужно прокрутить мотор?**

Дальнейшее изучение

Вы прочитали книгу до конца. Поздравляю! Надеюсь, вам было интересно изучать основы робототехники с конструктором LEGO MINDSTORMS EV3, а также собирать и программировать роботов, о которых рассказывается в этой книге. Теперь вы можете придумывать собственных роботов и делиться своими идеями с остальными. Конструктор LEGO MINDSTORMS EV3 предоставляет вам безграничные просторы для реализации любых задумок: ваши роботы могут ездить, брать предметы, ходить и говорить!

Но, прежде чем закрыть книгу, попробуйте выполнить задания из последних практикумов и дополнить программу для робота LAVA R3X, сделав его более интерактивным. Когда вы все сделаете, обязательно посмотрите инструкции по сборке и программированию робота, который сортирует детали LEGO по цвету и размеру (как на рис. 19.21), в приложении Г.

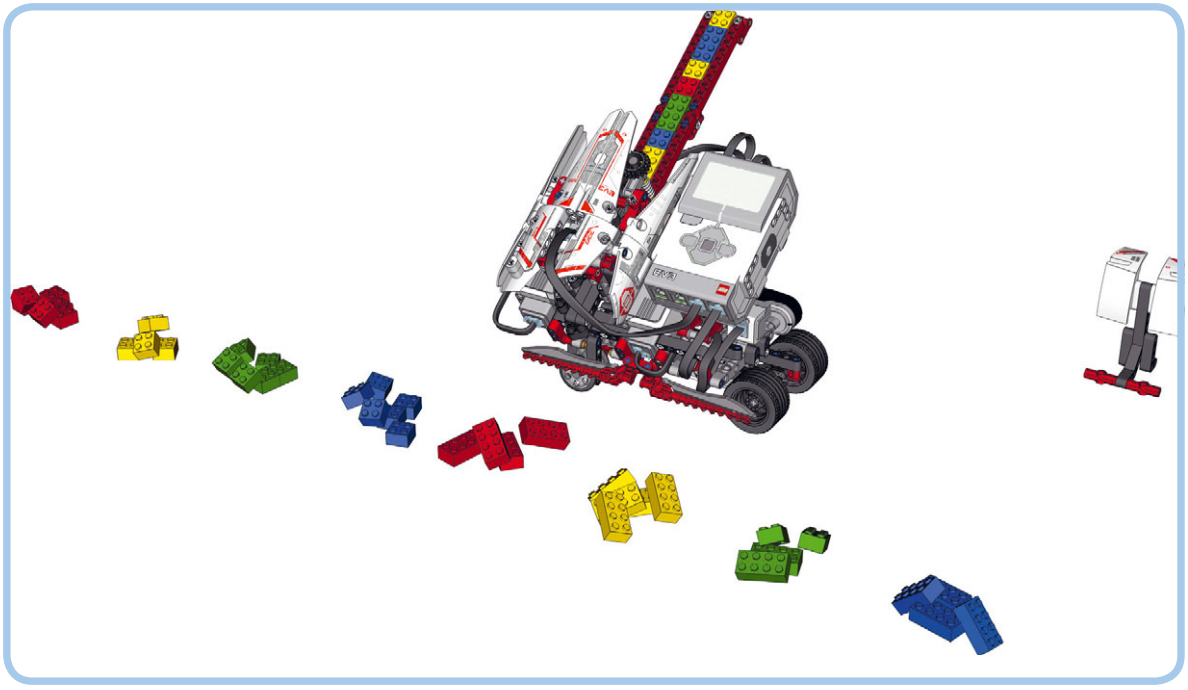


Рис. 19.21. Хотите собрать еще одного робота? Робот BRICK SO RT3R сортирует детали LEGO по цвету (красный, желтый, зеленый и синий) и размеру (2×2 и 2×4). Инструкции по его сборке и программированию приведены в приложении Г

ПРАКТИКУМ № 127: КАКАЯ РАЗНИЦА?

Сложность:  **Время:** 

Для более эффективной работы с контейнером **OnSync** выведите разницу между позициями моторов на экран. Для этого в контейнер **OnSync** поместите блок **Экран** (Screen), который выводит значение последнего блока **Математика** (Math), показанного на рис. 19.9. Затем поместите измененный контейнер **OnSync** в цикл, настроенный на бесконечное воспроизведение. Что случится с разницей, если вы замедлите один мотор, удерживая ногу робота вручную? Как второй мотор будет пытаться это исправить? (Делайте это не более нескольких секунд!)

ПРАКТИКУМ № 128: РОБОТ-ТРЕНЕР!

Сложность:  **Время:** 

Попробуйте научить робота определять, сколько времени вы проводите за рабочим столом. Запрограммируйте робота LAVA R3X отображать количество времени, проведенное вами за столом, и каждый час советовать вам сделать перерыв. Если он поймет, что вы не следуете его совету, пусть он качает головой и воспроизводит различные звуки, чтобы привлечь ваше внимание.

ПРАКТИКУМ № 129: РОБОТ-КОМПАНЬОН!

Сложность:  **Время:** 

Можете ли вы запрограммировать робота смотреть на вас, пока он идет, поворачивая голову в направлении инфракрасного маяка? Пусть скорость среднего мотора будет пропорциональна значению приближения, как мы делали, собирая робота SNATCH3R (см. рис. 18.20). Примите во внимание, что голова робота не может выполнить полный поворот вокруг своей оси. Как с помощью программы ограничить движение мотора, чтобы он не пытался повернуться дальше, чем может?

ПРАКТИКУМ № 130: В ТАКТ!

Сложность:  **Время:** 

В программе ObstacleAvoid руки робота LAVA R3X раскачиваются влево и вправо, независимо от темпа ходьбы робота. Попробуйте синхронизировать движение рук и ног, чтобы походка робота казалась более естественной и реалистичной.

ПРАКТИКУМ № 131: НА РАССТОЯНИИ!

Сложность:  **Время:** 

Попробуйте разработать программу, с помощью которой можно будет управлять роботом LAVA R3X дистанционно, так чтобы он мог идти в любом направлении. Можно применить методы из главы 8, но, чтобы моторы в ногах робота работали синхронно, вам необходимо использовать знания, полученные из данной главы. Как сделать так, чтобы робот начинал идти или поворачивал, когда вы нажимаете и удерживаете кнопку на маяке, и останавливался, когда вы ее отпускаете?

ПРАКТИКУМ № 132: ТАМАГОЧИ!

Сложность: **Время:**

Попробуйте превратить робота LAVA R3X в «живую» игрушку, у которой может быть разное настроение и поведение. С помощью удаленного инфракрасного маяка приказывайте роботу идти, говорить, есть и спать. С помощью числовых переменных следите за состоянием здоровья робота, контролируя его уровень голода, уровень энергии и счастья.

Пусть уровень энергии падает с каждым шагом робота и увеличивается, когда вы приказываете роботу спать. Точно так же пусть уровень голода увеличивается, пока робот ходит, и падает, когда вы его кормите. Пусть уровень счастья робота с течением времени постепенно уменьшается и увеличивается каждый раз, когда вы жмете роботу руку. Если уровень голода, энергии или счастья достигает определенного предела, робот должен игнорировать новые команды и принимать собственные решения. Например, если робот слишком устал (уровень энергии ниже 10%), он должен засыпать на время, чтобы восстанавливать его. Или, если ему грустно, пусть он говорит «Нет!»* и плачет каждый раз, когда вы даете ему новую команду.

Уровни энергии, голода и счастья должны отображаться на экране модуля EV3, чтобы вы могли следить за здоровьем и настроением робота. Поэкспериментируйте с различными типами поведения робота и добавьте звуковые и световые эффекты, чтобы он мог демонстрировать свои эмоции и казался более реалистичным. Например, улыбающиеся глаза на экране модуля EV3 могут означать, что робот счастлив, а когда робот спит, он может издавать звук храпа.

Создавая свою программу, можете ориентироваться на схему, приведенную на рис. 19.22. Но можете придумать и что-то свое.

* Пункт **No!** в каталоге звуков Lego.

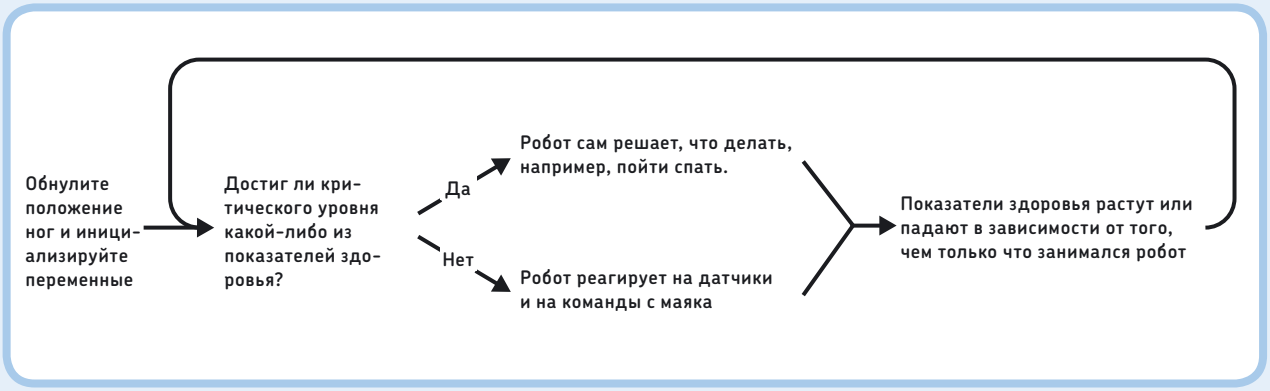


Рис. 19.22. Один из возможных способов выполнить задание из практикума № 132. Обратите внимание, что это только схема, которая поможет вам начать. Каждый из указанных шагов может состоять из множества подобранных вами программных блоков, включая множество блоков Переключатель и Цикл

СДЕЛАЙ САМ № 30: ДВУНОГИЙ РОБОТ!

Сборка: **Программирование:**

Получится ли у вас сделать робота-животное, который будет ходить на двух ногах? Снимите верхнюю часть туловища робота и модуль EV3, чтобы остались только ноги. Теперь вы можете собрать любого робота, который будет ходить на двух ногах. Попробуйте построить страуса или, может быть, динозавра. Пусть движениями головы, хвоста или даже клешней управляет средний мотор. Для управления ногами робота используйте контейнеры «Мой блок», описанные в этой главе.

A

Исправление ошибок в программах, модуле EV3 и беспроводных подключениях

При сборке и программировании роботов, описанных в этой книге, у вас время от времени могут возникнуть сложности с переносом программ в память модуля EV3. Данное приложение призвано помочь вам найти решения подобных проблем. Вы узнаете также, как управлять беспроводными подключениями модуля, как его перезапускать и как обновлять встроенное программное обеспечение.

Решение проблем компиляции

Когда вы загружаете свою программу в модуль EV3, его ПО превращает ваш *исходный код* (блоки программы, которые вы видите на экране) в файл с более компактным программным кодом, представляющим действия, которые будет выполнять модуль EV3. Этот процесс называется *компиляцией*. Иногда она завершается неудачно, и в этом случае вы увидите сообщение об ошибке, подобное приведенному на рис. A.1.

Отсутствующие контейнеры «Мой блок»

Компиляция завершится ошибкой, если программа попытается запустить блок, которого больше нет. Если это происходит, на отсутствующем блоке появится вопросительный знак (рис. A.1). Предположим, у вас есть контейнер **Talk** и программа SoundProgram, которая его использует. Этот проект не удастся скомпилировать, если блок **Talk** отсутствует — например, если вы удалили блок **Talk** на странице **Свойства проекта** (Project Properties).

Обратите внимание на то, что данное сообщение об ошибке не говорит о том, какой именно блок отсутствует (**Talk**); вместо этого оно показывает, какая программа (SoundProgram) содержит ссылки на отсутствующий блок. Чтобы решить эту проблему, можно создать новый контейнер

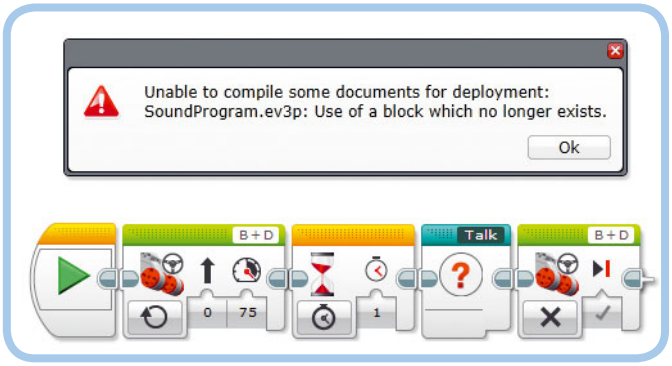


*Рис. A.1. Эту программу невозможно скомпилировать, поскольку отсутствует блок **Talk***

«Мой блок» с тем же именем (**Talk**) или скопировать его из другого проекта в текущий согласно инструкциям, приведенным на рис. 5.13 в главе 5. Либо, если вы желаете продолжить работу без отсутствующих блоков, то просто удалите все блоки, на которых есть вопросительный знак.

Ошибки в программных блоках

Компиляция может завершиться ошибкой, если ваша программа содержит инструкции, которые не поддерживает программное обеспечение. Так, например, компиляция не удастся, если вы введете неизвестный символ в поле **Уравнение** (Equation) блока **Математика** (Math), как показано на рис. A.2.

Программное обеспечение не сообщает вам, какой блок приводит к ошибке, но вы можете отследить ее, выбрав несколько блоков и попробовав запустить их при помощи кнопки **Запустить выбранное** (Run Selected). Если выбранные блоки запускаются, они не содержат ошибок компиляции; если же нет — в них есть ошибка. По мере того как вы методично определите, какие части работают, а какие нет, вы должны в конечном итоге обнаружить блок, содержащий ошибку. После этого вы можете устранить ошибку, если знаете, чем она вызвана, или же удалить такой блок и заменить его новым.

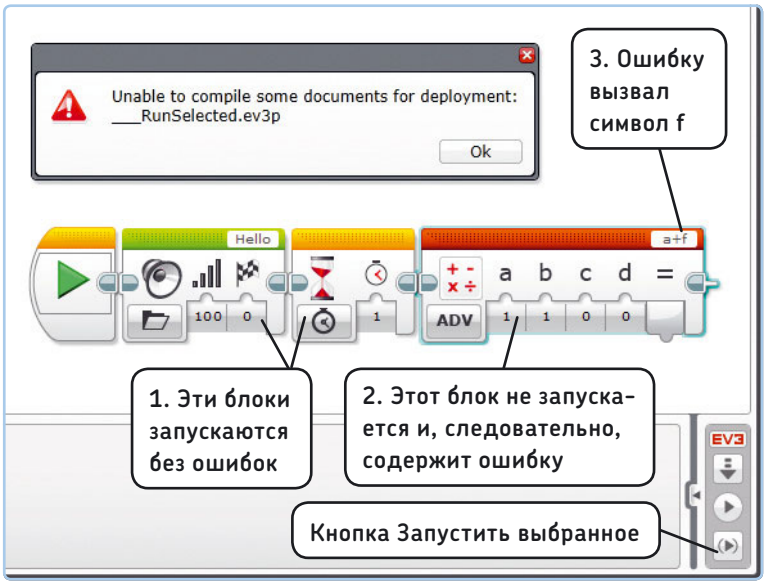


*Рис. А.2. Выявление ошибок компиляции программы путем запуска отдельных блоков. Параметр **Уравнение** в блоке **Математика** может работать только с символами **a**, **b**, **c** и **d**, поэтому символ **f** приводит к ошибке компиляции*

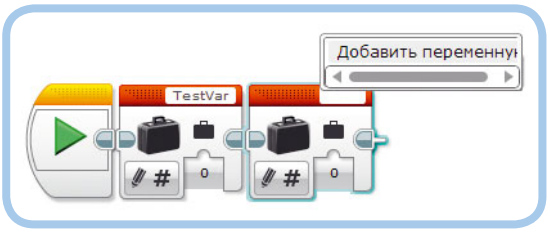


*Рис. А.3. Числовая переменная TestVar недоступна, поскольку она еще не была определена. Чтобы ее определить, выберите пункт **Добавить переменную** и введите **TestVar** в качестве имени переменной*

Неопределенные переменные

Когда вы копируете блок **Переменная** (Variable) из одного проекта и вставляете в другой, с этим блоком не всегда копируется *определение*. Это означает, что, даже несмотря на присутствие имени переменной в скопированном блоке, она недоступна для использования в других блоках **Переменная** (Variable) (рис. А.3). Чтобы решить эту проблему, определите новую переменную того же типа и с тем же именем.

Подобное может произойти, когда вы импортируете в свой проект программу или контейнер «Мой блок», содержащий блоки **Переменная** (Variable). Перечень всех переменных можно увидеть на вкладке **Переменные** (Variables) страницы **Свойства проекта** (Project Properties) (см. рис. 16.3 в главе 16). Эти переменные должны быть доступны для применения всеми программами вашего проекта.

Решение проблем запуска программ

Предыдущий раздел поможет вам справиться с некоторыми техническими проблемами. Но как быть, если программа работает совсем не так, как вы ожидаете? Причин этому может быть множество, и в большинстве случаев это всего лишь ошибка пользователя. Я занимаюсь программированием роботов уже много лет и все равно часто допускаю ошибки. Например, иногда я забываю подключить шину данных, и мой робот не работает.

С другой стороны, благодаря сделанным ошибкам вы можете обнаружить новые методы и решения, которые вы могли бы и не найти, если бы действовали в точности по инструкции. Поиск ошибок не выглядит полезным или забавным занятием, но это существенный элемент робототехники. И когда вам удается самостоятельно решить проблему и сделать так, чтобы робот работал, ценность программирования возрастает.

Приведенные ниже рекомендации могут быть полезны при поиске и предотвращении ошибок в ваших программах.

- *Пишите комментарии к программе.* Комментарии не влияют на работу программы, но могут помочь вам вспомнить, для чего предназначена каждая часть программы, когда вы будете работать с ней впоследствии. **Комментарии** можно добавить в программу либо при помощи инструмента Комментарий (Comment), либо используя блок **Комментарий** (Comment), как показано на рис. А.4. Используйте *инструмент* **Комментарий** (Comment), если вы желаете, чтобы ваш комментарий оставался на одном и том же месте *области программирования*; применяйте *блок* **Комментарий** (Comment), если хотите, чтобы комментарий оставался в одном и том же месте *программы*. Например, если вы вставите еще один программный блок перед блоком **Рулевое управление** (Move Steering), то блок **Комментарий** (Comment) автоматически сдвинется вправо с другими блоками, а обычный комментарий останется на своем месте. Для дополнительного документирования вашего проекта можно добавить текст с иллюстрациями в редактор контента (см. рис. 3.19 в главе 3).
- *Выбирайте описательные имена для программ, контейнеров «Мой блок» и переменных.* Если, например, у вас есть две переменные для подсчета количества нажатий кнопок «Влево» и «Вправо», то разумно назвать их **CountLeft** и **CountRight**, а не **Count1** и **Count2**. Тогда в дальнейшем у вас будет меньше поводов их перепутать.
- *Используйте звуковые, экранные и световые (индикатора состояния модуля) оповещения, чтобы сигнализировать о ходе работы программы.* Поместите, к примеру, блок **Звук** (Sound) после блока, который ожидает срабатывания датчика, или сделайте так, чтобы индикатор состояния светился красным

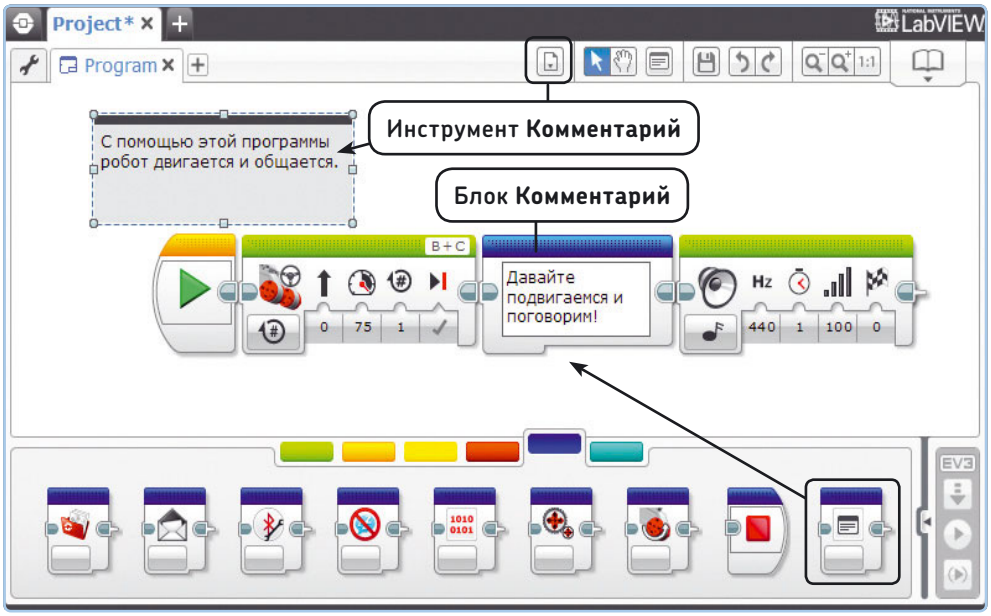


*Рис. А.4. Можно добавлять комментарии к программе либо с помощью блока **Комментарий**, либо применив одноименный инструмент. Блок **Комментарий** недоступен в программе EV3 версии 1.0. Чтобы его использовать, установите версию 1.1 или выше согласно инструкциям, приведенным в главе 1. Если вы не знаете точно, какая у вас версия программы, выберите пункт меню **Справка** ▸ **Информация о LEGO® MINDSTORMS® EV3***

цветом во время работы какого-либо контейнера «Мой блок». Это поможет вам понять, какая часть программы выполняется. Если в программе возникнет сбой, у вас будет приблизительное представление о том, где искать проблему.

- *Отображайте значения на экране.* Например, если вы используете блок **Переключатель** (Switch) для принятия решения на основе значения, полученного от датчика, будет удобно отобразить это значение на экране, чтобы вы могли видеть, какая величина привела к срабатыванию последовательности **Истина** (True) или **Ложь** (False) переключателя. А неожиданное значение, например 0, может сигнализировать о том, что датчик не подключен.
- *Используйте программу EV3, чтобы увидеть, какой блок выполняется в данный момент.* Это поможет вам понять, где программа «спотыкается» (рис. А.5). Например, если программа застопоривается на блоке **Большой мотор** (Large Motor), то, вероятно, что-то препятствует мотору в достижении цели — возможно, он заблокирован.
- *Чаще проверяйте работу программы после внесения изменений.* Если поначалу ваша программа работает прекрасно, но перестает функционировать после добавления нового блока, возможно, этот последний добавленный блок является причиной проблемы. Когда вы вносите несколько существенных изменений в программу, обязательно проверяйте ее работу после каждого изменения.
- *Проверяйте работу программы в различных условиях.* Если ваш робот работает сейчас, то нет никакой гарантии, что он заработает в дальнейшем, если изменятся условия. Программа движения по трассе может прекрасно работать для одного помещения, но некорректно определять контур в другом помещении, где много дополнительных источников света, например на стенде для проведения соревнований между роботами.

ПРИМЕЧАНИЕ **Если какая-либо из программ данной книги не работает так, как описано, вы можете загрузить готовые программы по ссылке eksmo.ru/files/Lego_Mindstorms_Primers.zip и сравнить их с вашими.**

Решение проблем с модулем EV3

В этом разделе рассказано о том, каким образом проверять различные характеристики модуля EV3, такие как уровень заряда аккумулятора, доступная память и USB-подключение между модулем EV3 и компьютером. Описаны также процессы перезагрузки и обновления EV3.

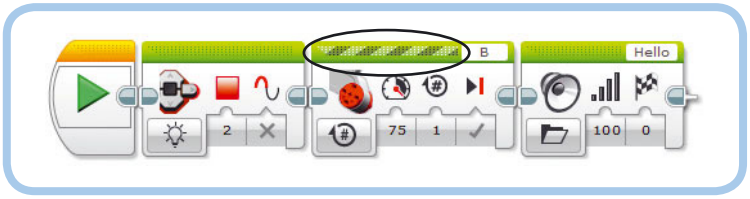


*Рис. А.5. Перемещающиеся полоски указывают на то, что в данный момент работает блок **Большой мотор**. Вы увидите эти полоски, только если вы воспользовались кнопкой **Загрузить и запустить**; они не видны, если вы запустили программу с помощью элементов управления в интерфейсе EV3*

Использование страницы аппаратных средств

Информацию о модуле EV3 и подключенных к нему устройствах можно найти на странице аппаратных средств, как показано на рис. А.6. На вкладке **Информация о модуле** (Brick Information) отображаются название, уровень заряда аккумулятора и версия встроенного ПО для подключенного в данный момент модуля EV3. Чтобы изменить название, введите новый текст в указанном поле и нажмите клавишу **Enter**. Это название должно отобразиться в верхней части экрана модуля EV3. Оно помогает различать модули EV3, если у вас их несколько. Вкладка **Представление порта** (Port View) отображает показания для каждого мотора или датчика из числа подключенных к EV3 (см. рис. 6.5 в главе 6).

Управление подключениями

На вкладке **Доступные модули** (Available Bricks) перечислены названия всех модулей EV3, обнаруженных программным обеспечением EV3. В зависимости от конфигурации EV3 вы можете выбрать подключение через USB, Bluetooth или Wi-Fi, установив флажок в соответствующем столбце. Например, модуль EV3 на рис. А.7 называется EXPLOR3R, он подключен к компьютеру через USB-интерфейс.

Используйте кнопку **Обновить** (Refresh) для обновления этого списка. Используйте кнопку **Отключить** (Disconnect), чтобы разорвать текущее подключение и получить возможность подключить другой модуль EV3.

Управление памятью EV3

Вы можете управлять файлами EV3 с вашего компьютера, открыв диалоговое окно **Обозреватель памяти** (Memory Browser) с помощью кнопки на вкладке **Информация о модуле** (Brick Information). Диалоговое окно **Обозреватель памяти** (Memory Browser) показано на рис. А.8. Вы увидите все программы, которые присутствуют в EV3; каждому проекту соответствует папка.

Скомпилированные программы занимают очень мало места (несколько килобайт) по сравнению с доступным свободным пространством (более 4 Мб) в памяти модуля EV3, и поэтому вы не скоро его израсходуете. Чтобы избежать беспорядка на вкладке **Навигация по файлам** (File Navigation) модуля EV3, можете удалить неиспользуемые проекты, выделив их и нажав кнопку **Удалить** (Delete).

*Рис. А.6. Вкладка **Информация о модуле** на странице аппаратных средств*

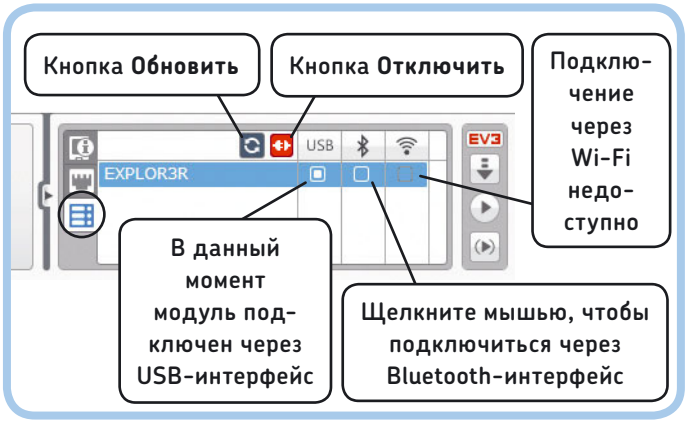


*Рис. А.7. Вкладка **Доступные модули** применяется для управления подключениями к модулю EV3. Если какой-либо флажок затенен, данный тип подключения недоступен*

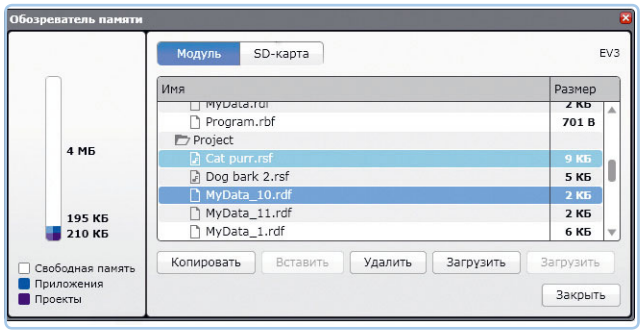


*Рис. А.8. Откройте инструмент **Обозреватель памяти** нажатием кнопки на странице аппаратных средств (см. рис. А.6) или выбрав пункт меню **Инструменты** ▸ **Обозреватель памяти***

Можно отправить файл из *модуля EV3 на компьютер*, выделив его и нажав кнопку **Загрузить** (Upload). Отправить файл *с компьютера в модуль EV3* можно, выделив исходную папку в интерфейсе EV3 и нажав кнопку **Загрузить** (Download)*. Обратите внимание, что при передаче скомпилированных программ обратно в компьютер не появляется исходный код программы, с которым вы обычно работаете в программе EV3. Невозможно «декомпилировать» программу EV3, чтобы продолжить ее разработку, и поэтому всегда следует хранить исходный проект для последующего использования. Программы в модуле являются исключениями из этого правила, как вы увидите в приложении Б.

Решение проблем с USB-подключением

Когда вы подключаете модуль EV3 к компьютеру с помощью USB-кабеля, программное обеспечение EV3 должно автоматически обнаружить его, о чем сообщит красный символ EV3 () на странице аппаратных средств. Если этот символ остается серым (), попробуйте выполнить следующие указания.

1. Убедитесь в том, что модуль EV3 включен.
2. Убедитесь в том, что модуль EV3 подключен к компьютеру через USB-порт с меткой **PC** (см. рис. 2.5 в главе 2). Другой конец кабеля следует подключить в любой USB-порт компьютера.
3. Если вы уверены, что все подключено правильно, попробуйте отключить USB-кабель, а затем подключить его обратно либо используйте другой USB-порт компьютера.
4. Закройте программу EV3 и запустите ее заново или же перезагрузите компьютер.
5. Если это не помогает, отключите USB-кабель, выключите модуль EV3, затем включите его, дождитесь полного запуска и подключите кабель обратно.

У вас могут возникнуть проблемы при попытке подключения EV3 к общедоступному компьютеру, например в школьном кабинете. Если это так, попросите системного администратора войти в систему, запустить программу EV3 и проверить, подключается ли она к модулю EV3. По завершении этого у вас должна появиться возможность подключения к EV3 при использовании вашей учетной записи.

Перезапуск модуля EV3

Если программа EV3 зависла и вам не удается прервать ее нажатием кнопки «Назад», можно перезапустить модуль EV3, нажав и удерживая одновременно центральную кнопку и кнопку «Назад», пока не погаснет индикатор состояния модуля (рис. А.9).

Рис. А.9. Чтобы перезапустить модуль EV3, нажмите одновременно центральную кнопку и кнопку «Назад» и удерживайте их до тех пор, пока не погаснет индикатор состояния модуля

Модуль EV3 должен перезапуститься, когда вы отпустите кнопки. Обратите внимание, что будут удалены все программы и настройки, выполненные вами с момента последнего включения.

Если модуль EV3 не запускается или вы видите, что индикатор состояния всего лишь кратковременно «моргнул», замените батареи и попробуйте обновить встроенное программное обеспечение, как рассказано в следующем разделе.

Обновление встроенного программного обеспечения EV3

Если возникла необходимость обновить встроенное программное обеспечение модуля EV3, это можно сделать при помощи команды меню **Инструменты** ▸ **Обновление встроенного ПО** (Tools ▸ Firmware Update). Подключите модуль EV3 к компьютеру с помощью USB-кабеля, выберите последнюю версию встроенного ПО в списке и нажмите кнопку **Обновить встроенное ПО** (Download). Модуль EV3 должен автоматически перейти в режим обновления, на экране появится надпись **Updating...** (Обновление...). Через несколько минут два индикатора прогресса на компьютере укажут на окончание процесса, и модуль EV3 должен автоматически перезагрузиться.

Если программному обеспечению не удается перевести модуль EV3 в режим обновления, это можно выполнить вручную, одновременно нажав и удерживая кнопки «Назад», «Вправо» и центральную, пока не погаснет индикатор состояния, как показано на рис. А.10. После этого отпустите кнопку «Назад». Как только вы увидите на экране модуля EV3 надпись **Updating...** (Обновление...), можно отпустить и остальные кнопки. После перевода модуля EV3 в режим обновления подключите его

* Обратите внимание, что в русском интерфейсе программы EV3 обе кнопки переведены одинаково. Кнопка **Загрузить** (Upload) — это вторая кнопка от правого края окна **Обозреватель памяти** (Memory Browser). Соответственно, кнопка **Загрузить** (Download) — крайняя правая кнопка в окне.

Рис. А.10. Чтобы вручную перевести модуль EV3 в режим обновления, удерживайте нажатыми кнопки «Назад», «Вправо» и центральную, пока не погаснет индикатор состояния модуля. После этого отпустите только кнопку «Назад». Когда на экране появится надпись ***Updating...****, отпустите остальные кнопки*

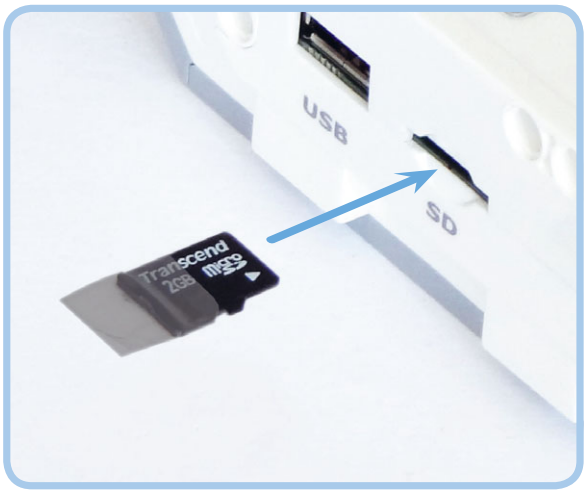


Рис. А.11. Можно использовать карту microSD, чтобы избежать потери данных. Убедитесь в том, что вы вставляете карту металлическими контактами вниз. Прикрепите небольшой кусочек скотча к торцу карты, чтобы впоследствии ее можно было легко вынуть из модуля EV3

к компьютеру через USB-интерфейс и попробуйте обновить встроенное программное обеспечение еще раз.

ПРИМЕЧАНИЕ **При обновлении встроенного программного обеспечения из памяти модуля EV3 будут удалены все программы и файлы. Если вы перешли в режим обновления случайно, можно перезапустить модуль, чтобы продолжить нормальную работу.**

Использование карты microSD во избежание потери данных

Когда вы передаете программу в модуль EV3, она сохраняется во временной памяти. Модуль EV3 помещает файлы и настройки в постоянное хранилище только тогда, когда вы выключаете модуль. Именно поэтому на выключение требуется некоторое время. Если вы перезапустите модуль EV3, не выполнив выключение, или же вынете из него аккумуляторы во время работы, будут утрачены все файлы и настройки, которые изменились с момента последнего включения модуля EV3, поскольку у него не было времени сохранить их. А на деле, если вы вынимаете аккумуляторы во время выключения (когда модуль EV3 занят записью файлов), то можно потерять и старые файлы.

Это может быть неожиданностью, но обычно не является большой проблемой, поскольку на вашем компьютере должны храниться копии исходных проектов. Тем не менее можно избежать таких потерь данных, снабдив модуль EV3 картой microSD, как показано на рис. А.11. Проекты будут сохраняться на этой карте автоматически при каждой загрузке проекта в ваш робот; вам не придется предпринимать дополнительные шаги. Программы будут оставаться на карте, даже если вы перезагрузите модуль EV3 или обновите его встроенное программное обеспечение.

Если вы используете карту microSD, то ваши проекты можно будет найти в папке *SD_Card* на вкладке **Навигация по файлам** (File Navigation) модуля EV3. Даже сложные программы занимают всего несколько килобайт, поэтому карта microSD небольшого объема послужит просторным хранилищем.

Беспроводное подключение к модулю EV3

Вместо использования USB-кабеля, который входит в комплект, можно подключить модуль EV3 к компьютеру через Bluetooth или Wi-Fi интерфейс. Беспроводная передача данных намного упрощает программирование, поскольку вам не нужно постоянно подключать и отключать USB-кабель при каждой загрузке программы.

После настройки беспроводного подключения перенос программ в модуль EV3 управляется кнопкой **Загрузить и запустить** (Download and Run), как и при USB-подключении.

Использование интерфейса Bluetooth для загрузки программ в модуль EV3

В модуле EV3 имеется встроенный интерфейс Bluetooth, который можно использовать для беспроводного подключения и передачи программ, взаимодействия с другим

модулем EV3 или для подключения к смартфону или планшету, чтобы осуществлять дистанционное управление. Обратите внимание: можно использовать только одну из этих возможностей.

Чтобы использовать беспроводное Bluetooth-подключение, вам понадобится либо компьютер со встроенным Bluetooth-модулем, либо Bluetooth-адаптер, который подключается к USB-порту компьютера (рис. A.12).

Выбор Bluetooth-адаптера

Существует множество подходящих Bluetooth-адаптеров, многие из них стоят менее 700 рублей. Как правило, совместимость адаптера зависит не от самого устройства, а от его драйверов в сочетании с операционной системой компьютера. В большинстве случаев вам будет достаточно подключить адаптер к компьютеру, дождаться окончания автоматической установки драйверов, запустить программное обеспечение EV3 и выполнить процедуру соединения, описанную в следующем разделе. Необходимые драйверы зависят от вашей операционной системы и от адаптера Bluetooth.

Если у вас возникают сложности при использовании Bluetooth-устройства, встроенного в компьютер, попробуйте отключить его и воспользоваться внешним адаптером.

Подключение модуля EV3 через Bluetooth-интерфейс

Выполните следующие действия, чтобы настроить Bluetooth-соединение между компьютером и модулем EV3.

1. Вставьте совместимый Bluetooth-адаптер в свободный USB-порт *на компьютере* или убедитесь в том, что включен встроенный Bluetooth-модуль. В зависимости от операционной системы будут автоматически обнаружены и установлены необходимые драйверы. Как правило, нет необходимости в установке дополнительных драйверов, которые предназначены для вашего адаптера.
2. Включите модуль EV3 и соедините его с компьютером посредством USB-кабеля.
3. Активируйте Bluetooth-устройство в модуле EV3, выбрав на вкладке **Settings** (Настройки) пункт **Bluetooth**. Затем установите флажки **Visibility** (Видимость) и **Bluetooth** и сбросьте флажок **iPhone/iPad/iPod**, воспользовавшись центральной кнопкой, как показано на рис. A.13.
4. В программе EV3 на компьютере перейдите на вкладку **Доступные модули** (Available Bricks) на странице аппаратных средств, а затем нажмите кнопку **Обновить** (Refresh), как показано на рис. A.7. Процесс поиска должен занять около 30 секунд. По его завершении список устройств EV3 обновится и в нем появятся EV3-модули, доступные для подключения.
5. В списке EV3-модулей рядом с названием каждого должен быть флажок. Создайте Bluetooth-подключение, установив флажок в столбце под символом Bluetooth (). Если этот флажок нельзя установить, нажмите **Обновить** (Refresh) еще раз. Если это не помогает, сбросьте флажок **Bluetooth** в меню EV3 (рис. A.13), установите его еще раз и попробуйте выполнить заново процедуру подключения.

Вы сможете определить, установлено ли рабочее Bluetooth-соединение с модулем EV3, если посмотрите в верхний левый угол экрана EV3: если подключение к компьютеру есть, будет отображаться символ , если нет — символ . При успешном подключении можно вынуть USB-кабель и начать загрузку программ.

Рис. A.12. Один из вариантов беспроводного подключения через Bluetooth-интерфейс

*Рис. A.13. Перейдите на вкладку **Settings** на экране модуля EV3, выберите пункт **Bluetooth**, а затем установите флажки, как показано на рисунке. Флажок **iPhone/iPad/iPod** следует устанавливать только при дистанционном управлении с помощью устройств под управлением операционной системы iOS. Этот флажок следует сбросить, если Bluetooth-подключение используется для беспроводного соединения с компьютером или при дистанционном управлении с Android-устройства*

При последующем запуске программы следует выполнить только шаги 4 и 5, чтобы установить Bluetooth-подключение, а для его настройки вам не понадобится USB-кабель.

Если вы не подключили USB-кабель на втором шаге, чтобы выполнить первичное соединение, модуль EV3 запросит у вас подтверждение соединения и предложит создать пароль, чтобы защитить подключение, когда вы дойдете до пятого шага. После указания пароля программа EV3 предложит вам ввести его. Модуль EV3, в свою очередь, попросит вас подтвердить пароль, и тогда соединение должно быть установлено. Проще будет, если вы воспользуетесь паролем по умолчанию (1234). Если применять USB-кабель для настройки Bluetooth-соединения, то программное обеспечение осуществляет все эти меры безопасности в фоновом режиме.

Использование интерфейса Wi-Fi для загрузки программ в модуль EV3

Можно подключить адаптер Wi-Fi к вашему модулю EV3, чтобы он мог соединяться с беспроводной сетью, как показано на рис. А.14. Когда компьютер и модуль подключены к одной сети, можно выполнить беспроводное программирование вашего робота. На момент написания книги модуль EV3 поддерживал только адаптер *NETGEAR WNA1100 N150 Wi-Fi USB Adapter*.

Для выполнения следующих шагов предполагается, что у вас уже настроена беспроводная сеть, которая защищена паролем WPA2. Я буду рассчитывать на то, что вы знаете имя сети (SSID) и пароль и что компьютер подключен к этой сети. Для установления подключения через Wi-Fi выполните следующее.

1. Включите модуль EV3 и вставьте совместимый адаптер Wi-Fi в USB-порт *модуля EV3* (см. рис. А.14). Подключите модуль EV3 к компьютеру с помощью USB-кабеля (на рисунке не показан).
2. На вкладке **Информация о модуле** (Brick Information) страницы аппаратных средств нажмите кнопку **Настройка беспроводного подключения** (Open Wireless (Wi-Fi) Setup) (см. рис. А.6). Модуль EV3 должен автоматически включить адаптер Wi-Fi и начать поиск беспроводных сетей. По окончании поиска выберите вашу сеть в списке, который появится на экране компьютера, и нажмите кнопку **Подключить** (Connect). Если возникнет сообщение об ошибке, информирующее, что к модулю не подключен адаптер Wi-Fi, это означает, что данный адаптер Wi-Fi не поддерживается модулем.
3. В диалоговом окне введите пароль сети и нажмите кнопку **Подключить** (Connect). При успешном завершении символ [📶] (адаптер Wi-Fi включен) в левой верхней части экрана EV3 должен измениться на [📶⇆] (соединение Wi-Fi установлено). Теперь модуль EV3 подключен к маршрутизатору вашей сети, но еще не подключен к компьютеру.
4. Перейдите на вкладку **Доступные модули** (Available Bricks) страницы аппаратных средств и нажмите кнопку **Обновить** (Refresh), как показано на рис. А.7.
5. Каждому доступному типу подключения должен соответствовать флажок. Создайте Wi-Fi подключение, установив флажок [📶]. Если флажок удалось установить, соединение выполнено, и вы можете отключить USB-кабель. Если не удалось, попробуйте отключить от сети *компьютер*, заново подключите его и нажмите кнопку **Обновить** (Refresh).

Рис. А.14. Схема подключения для беспроводного программирования по сети Wi-Fi

Bluetooth или Wi-Fi?

Если вы хотите подключаться к роботу беспроводным способом, рекомендую использовать Bluetooth. Во-первых, вам не понадобится сеть Wi-Fi и не придется настраивать ее. Во-вторых, существует всего один поддерживаемый адаптер Wi-Fi, который займет много места в вашем роботе. А Bluetooth-адаптер, к слову, уже встроен в модуль EV3. Наконец, начальную настройку подключения Bluetooth можно выполнить всего за несколько щелчков мышью.

С другой стороны, поскольку модуль EV3 фактически является небольшим Linux-компьютером, подключение Wi-Fi можно использовать, чтобы получить доступ к расширенным функциям модуля, которыми нельзя воспользоваться с помощью стандартного программного обеспечения EV3. Тем не менее, если вы не планируете изучение этих функций, вероятно, лучше остановиться на Bluetooth-подключении.

Заключение

Надеюсь, что в этом коротком приложении вам удалось найти решение вашей проблемы. Конечно же, здесь перечислены лишь некоторые проблемы, а у вас могли возникнуть и другие вопросы, касающиеся конструирования или программных инструкций в этой книге. Поищите в интернете сайты, посвященные сборке моделей из конструкторов LEGO MINDSTORMS, и форумы, где вы можете задать волнующие вас вопросы.

Б

Создание программ модуля

Вместо применения программного обеспечения EV3 для разработки программ на компьютере можно создавать простые программы в самом модуле EV3 с помощью приложения **Brick Program**. Этот метод удобен при тестировании робота, если рядом нет компьютера. Иногда достаточно лишь проверить программу с помощью приложения **IR Control**, а также состояние датчиков с помощью инструмента **Port View**, но приложение **Brick Program** можно использовать и для создания программ, в которых задействованы и моторы, и датчики. Так, например, чтобы протестировать механизм с датчиком касания (такой, как манипуляторы робота SNATCH3R), можно выполнить вращение мотора, пока не будет нажата кнопка датчика касания.

В этом приложении рассказано о том, как создавать программы модуля и как импортировать их в программное обеспечение EV3, чтобы продолжить работу с ними на компьютере.

ПРИМЕЧАНИЕ **Создание программ с помощью модуля подобно созданию программ EV3 на компьютере. Ввод команд происходит иначе, но в основе лежат те же принципы. В данном приложении я полагаю, что вы уже освоили приемы, описанные в главах 1–6. Я покажу вам, как использовать такие же методы для создания программы модуля.**

Создание, сохранение и запуск программы модуля

Работу с новой программой модуля можно начать, выбрав пункт меню **Brick Program** на вкладке **Приложения модуля** (Brick Apps) модуля EV3, как показано на рис. Б.1. Программа модуля всегда состоит из одного блока **Цикл** (Loop), внутри

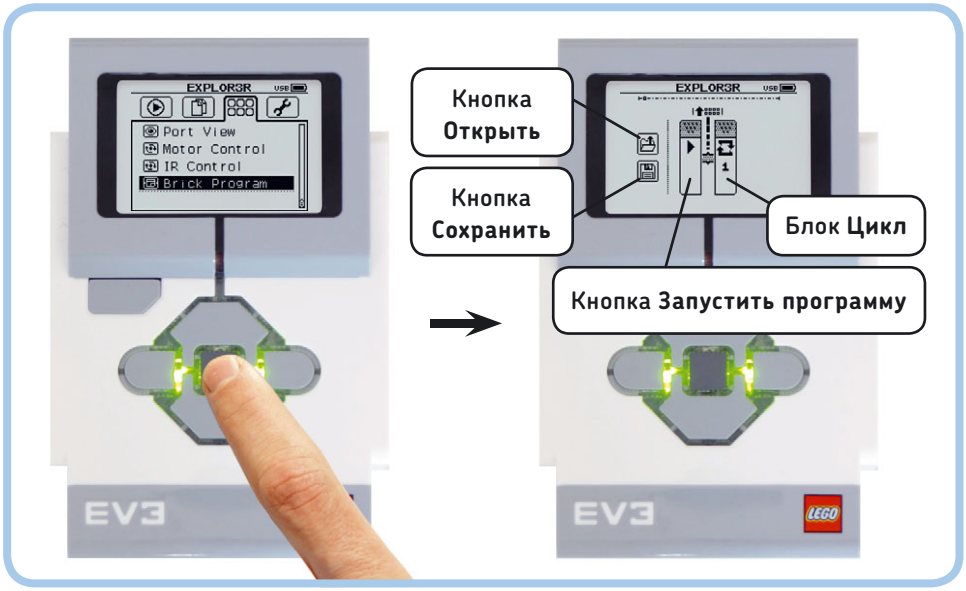


Рис. Б.1. Открытие новой программы модуля

которого вы можете разместить несколько блоков действий и **Ожидание** (Wait).

Добавление блоков в цикл

Перемещение по программе осуществляется при помощи кнопок «Влево» и «Вправо», при этом выбранный элемент всегда располагается в центре экрана. Чтобы вставить новый блок в цикл, перейдите в промежуток между двумя блоками и нажмите кнопку «Вверх» (рис. Б.2). Затем выберите нужный вам блок и поместите его в цикл, нажав центральную кнопку. Выберите, к примеру, блок **Индикатор состояния модуля** (Brick Status Light), чтобы индикатор состояния модуля EV3 стал оранжевым.

В цикле можно разместить до 16 блоков.

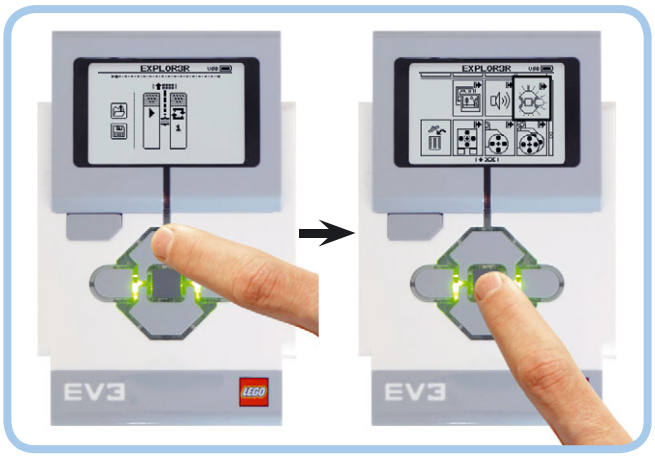


Рис. Б.2. Добавление блока в программу

Замена блока

Заменить один программный блок другим можно, если выбрать его (т.е. расположить в центре экрана при помощи кнопок «Влево» и «Вправо») и нажать кнопку «Вверх». После этого выберите новый блок центральной кнопкой или отмените действие, нажав кнопку «Назад».

Удаление блока

Можно удалить блок из программы, выделив его, нажав кнопку «Вверх», а затем выбрав элемент в виде корзины, как показано на рис. Б.3.

Настройка параметра блока

У каждого блока есть один настраиваемый параметр. Чтобы изменить его, выберите нужный блок и нажмите центральную кнопку, как показано на рис. Б.4. После этого воспользуйтесь кнопками «Вверх» и «Вниз» для изменения значения параметра. Подтвердите действие при помощи центральной кнопки или отмените его кнопкой «Назад».

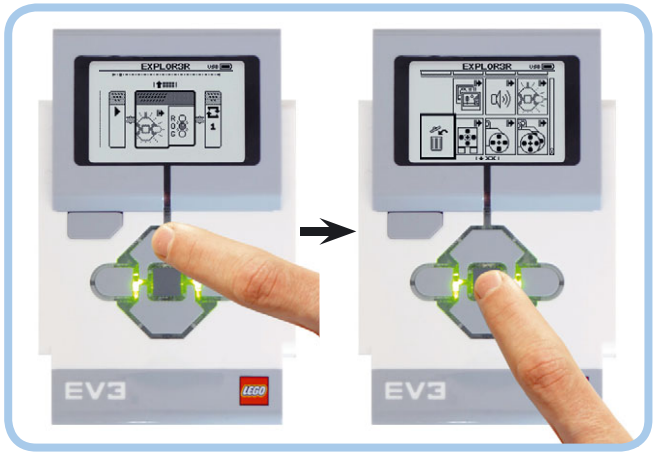


Рис. Б.3. Удаление блока из программы

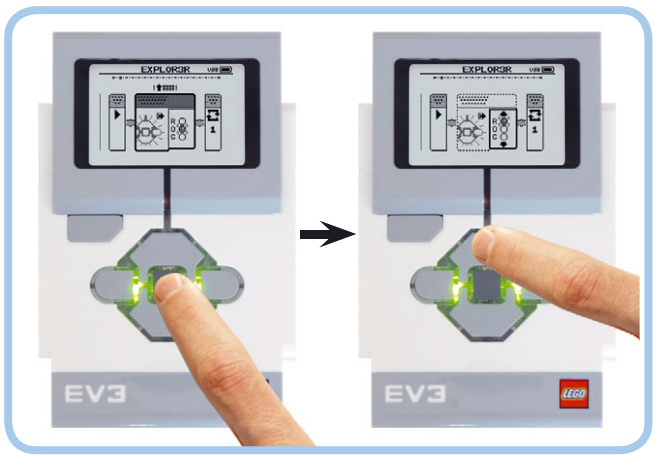


*Рис. Б.4. Изменение параметра блока. В данном примере для цвета индикатора состояния модуля задается значение **R** («красный») вместо **0** («оранжевый»)*

Запуск программы

Создадим теперь программу OnBrickStatus (рис. Б.5)

Поместите в цикл два блока **Индикатор состояния модуля** (Brick Status Light) и два блока **Ожидание** (Wait). После этого настройте программу так, чтобы цвет индикатора менялся на красный, затем следовало двухсекундное ожидание, цвет индикатора менялся на зеленый и следовало бы еще одно двухсекундное ожидание. Наконец, измените параметр блока **Цикл** (Loop) так, чтобы он продолжал повторение находящихся в нем блоков.

Запустите программу, выбрав левую часть блока **Цикл** (Loop) и нажав центральную кнопку, как показано на рис. Б.5. Индикатор состояния модуля должен становиться красным на две секунды, затем зеленым в течение двух секунд, после чего смена цвета продолжится. Можно прервать выполнение программы, нажав кнопку «Назад».

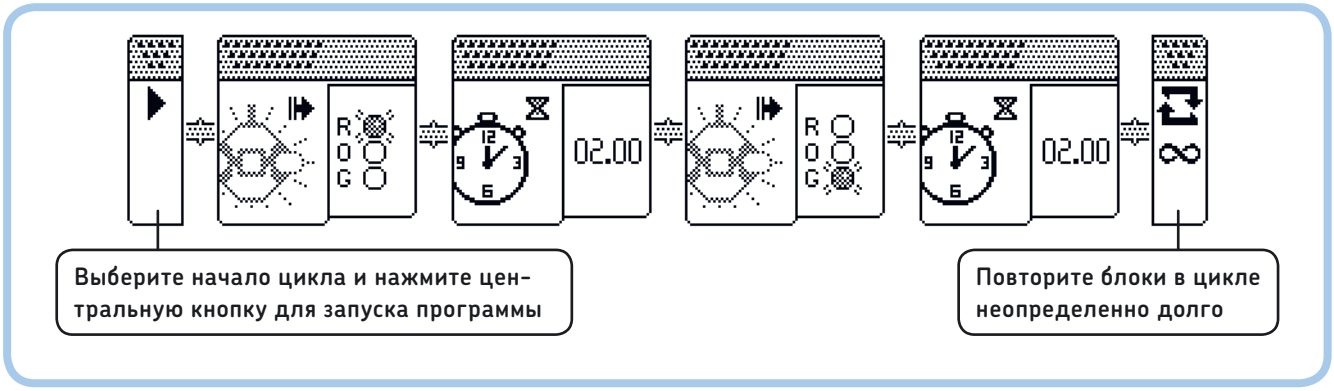


Рис. Б.5. Программа OnBrickStatus каждые две секунды меняет цвет индикатора. Здесь показана вся программа, но на экране EV3 вы увидите лишь ее часть. Воспользуйтесь кнопками «Влево» и «Вправо», чтобы проверить соответствие вашей программы приведенной на рисунке

Сохранение и открытие программы

Выберите значок сохранения в виде дискеты, чтобы сохранить текущую программу, как показано на рис. Б.1. На экране вы увидите клавиатуру, которой можно управлять при помощи кнопок на корпусе модуля EV3. Выберите символ [⇦], чтобы удалить текущее название, ведите OnBrickStatus, выбирая буквы с помощью центральной кнопки, а затем установите флажок.

Чтобы открыть и отредактировать сохраненную ранее программу модуля, выделите значок открытия и выберите программу в появившемся списке. Если вы желаете только запустить сохраненную программу, вы можете отыскать ее в папке *BrkProg_SAVE* на вкладке **Навигация по файлам** (File Navigation) модуля EV3.

При закрытии приложения **Brick Program** при помощи кнопки «Назад» вам будет предложено сохранить программу. Выберите символ **X**, чтобы закрыть ее без сохранения, или установите флажок, чтобы выбрать имя и сохранить программу.

ПРИМЕЧАНИЕ **Вам придется вводить это имя при каждом сохранении программы, поэтому выберите простое и короткое имя, если вы предполагаете, что будете часто изменять свою программу.**

Использование программных блоков на модуле

В табл. Б.1 показаны блоки, которые доступны для программ модуля, и эквивалентные им программные блоки в программном обеспечении EV3. У вас есть выбор из нескольких блоков действий и блока **Ожидание** (Wait) для работы в разных режимах. Номера страниц в таблице помогут найти дополнительную информацию о режиме каждого блока или датчика.

Приложение **Brick Program** позволяет вам изменить лишь один параметр каждого блока. Эквивалентный параметр блока в программе EV3 выделен на рисунке синим овалом. Другие параметры, такие как порты датчика или мотора, изменить нельзя. Если, например, вы желаете использовать датчик касания в программе модуля, он должен быть подключен к порту 1.

Числовые параметры для блоков **Экран** (Display) и **Звук** (Sound) соответствуют различным изображениям и звуковым файлам. Узнать, какой файл соответствует каждому числу, можно в списке «Программные приложения модуля — список ресурсов» на с. 55 руководства пользователя EV3. Щелкните мышью по вкладке **Руководство** (User Guide) в лобби программы EV3, как показано на рис. 3.2 в главе 3.

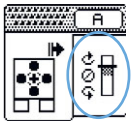
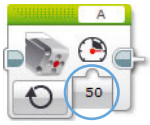
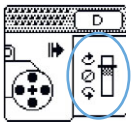
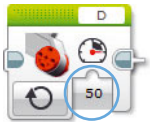
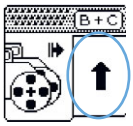
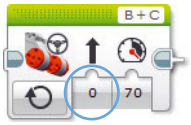
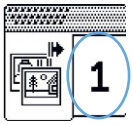
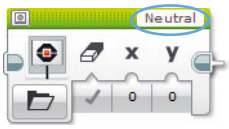
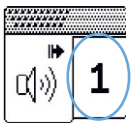
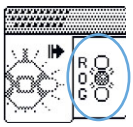
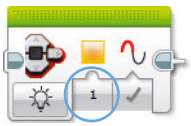
Вы можете использовать эти блоки для создания программ, подобных тем, которые вы разрабатывали при чтении глав 4–6. Например, можно создать программу, которая запускает на 1 секунду большой мотор, когда вы нажимаете кнопку датчика касания. Подключите датчик касания к входному порту 1, подключите большой мотор к выходному порту D, а затем создайте и запустите программу OnBrickTouch (рис. Б.6).

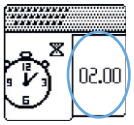
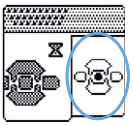
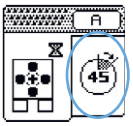
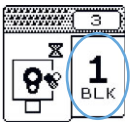
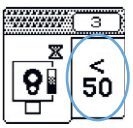
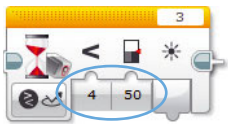
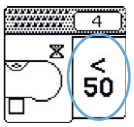
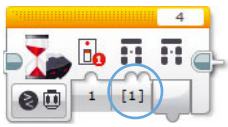
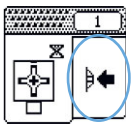
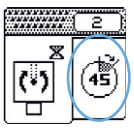
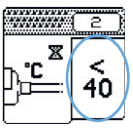
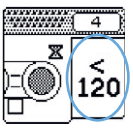
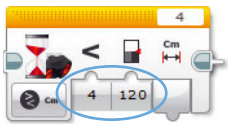
Обратите внимание на то, что в этой программе работа мотора в течение одной секунды задается как включение мотора, секундное ожидание и установка значения 0 в качестве скорости вращения.

Импорт программ модуля

Вы можете импортировать сохраненную программу модуля в проект EV3, подключив модуль EV3 к компьютеру и выбрав

Таблица Б.1. Блоки, доступные в приложении Brick Program, и эквивалентные им блоки в программе EV3

Блоки действий			с.
Средний мотор (Medium Motor)			68
Большой мотор (Large Motor)			68
Рулевое управление (Move Steering)			57
Экран (Display)			64
Звук (Sound)			62
Индикатор состояния модуля (Brick Status Light)			66

Блок Ожидание (режим)			с.
Время (Time)			71
Кнопки управления модулем (Brick Buttons)			120
Вращение мотора (Motor Rotation) **Градусы** (Degrees)			121
Датчик цвета (Color Sensor) **Цвет** (Color)			100
Датчик цвета (Color Sensor) **Яркость отраженного света** (Reflected Light Intensity)			103
Инфракрасный датчик (Infrared Sensor) **Приближение** (Proximity)			112
Инфракрасный датчик (Infrared Sensor) **Удаленный** (Remote)			114
Датчик касания (Touch Sensor) **Состояние** (State)			88
Гироскопический датчик (Gyro Sensor)* **Угол** (Angle)			нет
Датчик температуры (Temperature Sensor)* **Цельсий** (Celsius)			нет
Ультразвуковой датчик (Ultrasonic Sensor)* **Расстояние, сантиметры** (Distance Centimeters)			нет

* Эти датчики не входят в набор EV3 с артикулом 313131. Дополнительную информацию о них можно узнать на сайте **www.lego.com/ru-ru/mindstorms/support**.

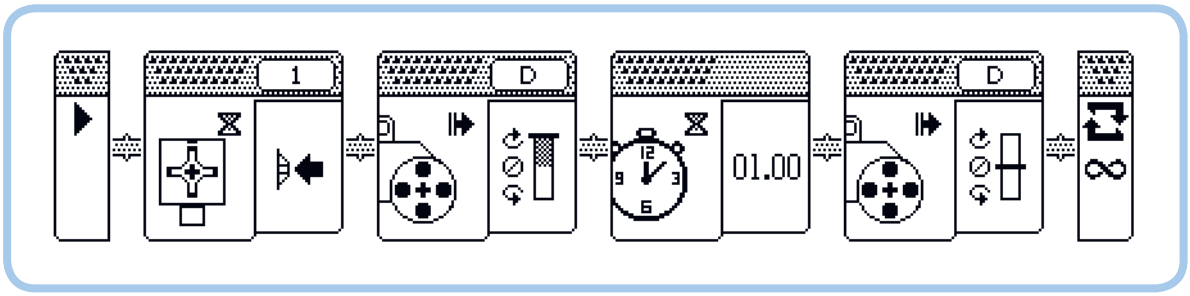


Рис. Б.6. Программа OnBrickTouch запускает большой мотор в части D на 1 секунду после нажатия кнопки датчика касания

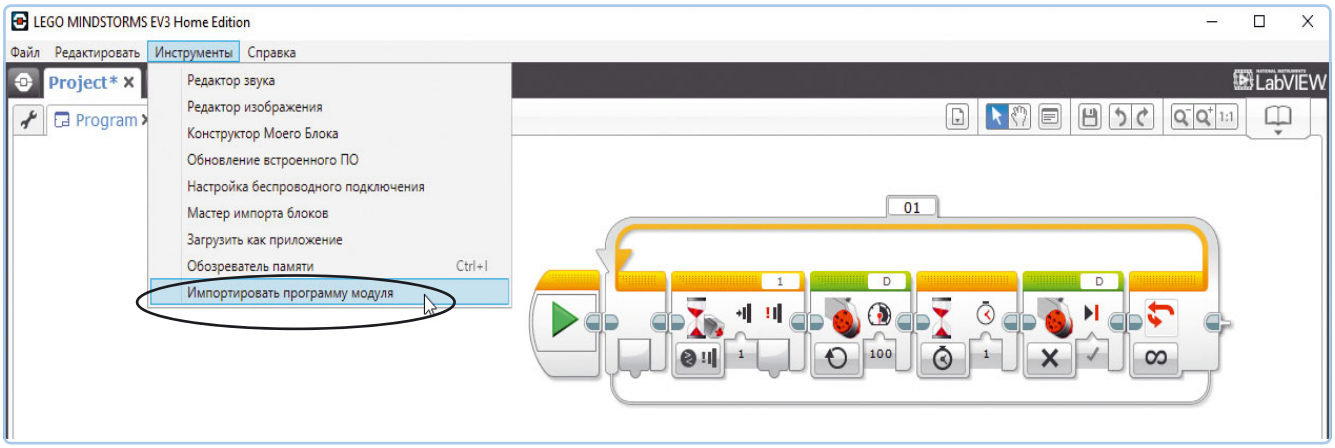


Рис. Б.7. Импорт программы модуля в программе EV3

команду меню **Инструменты ▸ Импортировать программу модуля** (Tools ▸ Import Brick Program), как показано на рис. Б.7. Выберите нужную программу в диалоговом окне и нажмите кнопку **Импорт** (Import). После этого программа должна появиться на собственной вкладке внутри текущего проекта.

Когда ваша программа будет импортирована в ПО EV3, вы сможете изменять ее, как любую другую программу EV3. Программные блоки OnBrickTouch превратятся в эквивалентный набор блоков EV3 в соответствии с табл. Б.1. При запуске OnBrickTouch она должна работать так же, как любая другая программа, созданная в модуле EV3.

После импорта в программное обеспечение EV3 исходная копия программы модуля остается в памяти модуля EV3, но, если вы измените эту программу на компьютере, вы не сможете превратить ее обратно в программу модуля.

ПРИМЕЧАНИЕ **В табл. Б.1 есть одно исключение: для программы модуля блок Ожидание (Wait) в режиме Вращение мотора ▸ Градусы (Motor Rotations ▸ Degrees) всегда присваивает значение 0 датчику вращения перед запуском, но после импорта программы это выполняться не будет. Можно при необходимости перезапустить датчик вращения в вашей программе с помощью дополнительного блока (см. рис. 9.4 в главе 9).**

Заключение

Вы научились создавать простые программы в модуле EV3 без помощи компьютера, используя приложение **Brick Program**. Этот вариант может оказаться удобным при тестировании робота, когда рядом нет компьютера. Программы модуля всегда состоят из последовательности блоков действий и блока **Ожидание** (Wait), помещенных в один блок **Цикл** (Loop). После импорта программы модуля в программное обеспечение EV3 вы можете изменять и запускать ее, подобно обычной программе EV3.

B

Различия между наборами LEGO MINDSTORMS EV3 (артикул 31313) и LEGO MINDSTORMS Образовательный набор EV3 (артикул 45544)

В этом приложении описаны различия между домашней и образовательной версиями набора LEGO MINDSTORMS EV3.

Робототехническая платформа LEGO MINDSTORMS EV3 выпускается в двух вариантах, которые рассчитаны на различные целевые аудитории. Пользователей можно разделить на две категории: домашние пользователи (дети и любители робототехники) и образовательные учреждения (учащиеся и преподаватели). Для каждой из этих групп компания LEGO разработала базовый набор и несколько дополнительных.

Домашняя версия LEGO MINDSTORMS EV3

Набор LEGO MINDSTORMS EV3 (арт. 31313) можно купить в магазинах игрушек и в интернет-магазинах. Его часто называют домашней версией (Home Edition или Retail Edition). С помощью данного набора можно собрать пять основных (EV3RSTORM, GRIPP3R, TRACK3R, R3PTAR и SPIK3R) и двенадцать бонусных моделей роботов EV3.

Для программирования этих роботов используется программное обеспечение, которое можно бесплатно скачать с сайта **education.lego.com/ru-ru/downloads**. В состав набора входит инфракрасный маяк, с помощью которого можно управлять роботом дистанционно. Кроме того, есть возможность управлять роботом через смартфон.

Набор включает в себя: 1 программируемый микрокомпьютер EV3, 2 больших мотора, 1 средний мотор, 1 датчик касания, 1 датчик цвета, 1 инфракрасный датчик и удаленный инфракрасный маяк, как показано на рис. B.1. В наборе содержатся 7 соединительных кабелей и 1 USB-кабель для подключения к компьютеру.

Для питания микрокомпьютера используются 6 батареек типа AA и 2 батарейки AAA для маяка. Вместо 6 батареек AAA можно приобрести аккумуляторную батарею LEGO EV3 и зарядное устройство к ней. (См. раздел «Аккумуляторная батарея EV3» далее.)

В комплекте содержатся около 600 строительных деталей, среди которых балки, зубчатые колеса, фиксаторы, оси и колеса, как показано на рис. B.1. Если разрезать коробку с набором, вы получите трассу для робота.

Базовая образовательная версия LEGO MINDSTORMS EV3

Базовая образовательная версия набора LEGO MINDSTORMS EV3 (арт. 45544) поставляется через официальных дистрибьюторов и дилеров LEGO. В комплект входит инструкция по сборке одной модели с различными вариантами датчиков. Остальные инструкции можно

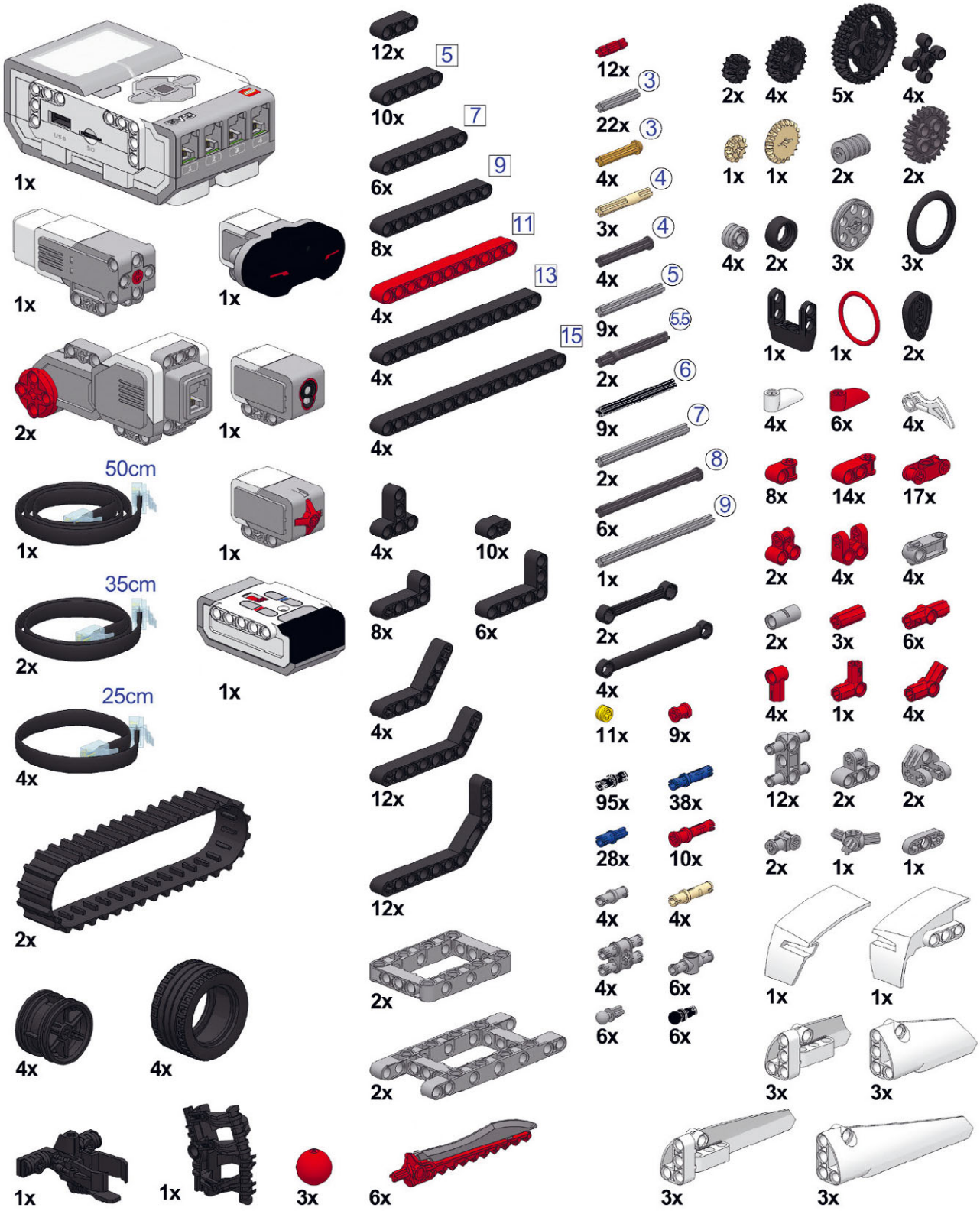


Рис. B.1. Состав набора LEGO MINDSTORMS EV3 (31313)

посмотреть на странице **goo.gl/wElhd3**. Учебные материалы предоставляются вместе с программным обеспечением для образовательной версии LEGO MINDSTORMS EV3, которое приобретается отдельно.

Набор включает в себя: 1 программируемый микрокомпьютер EV3, 2 больших мотора, 1 средний мотор, 2 датчика касания, 1 датчик цвета, 1 ультразвуковой датчик и 1 гироскопический датчик, как показано на рис. B.2. Набор также содержит 7 соединительных кабелей и 1 USB-кабель для подключения к компьютеру. В комплект входит аккумулятор для модуля EV3 и зарядное устройство к нему.

В набор входят порядка 540 деталей, включая балки, оси, зубчатые колеса и фиксаторы, как показано на рис. B.2. Прочная пластиковая коробка содержит дополнительные отсеки для хранения деталей из серии Technic.

Модернизация образовательной версии набора до домашней

Если у вас есть образовательная версия набора LEGO MINDSTORMS EV3 (арт. 45544), возможно, вы захотите построить роботов по инструкциям из домашней версии набора LEGO MINDSTORMS EV3 (арт. 31313), а также 12 бонусных моделей или роботов из книг с описанием домашней версии. Для этого необходимо отдельно приобрести инфракрасный датчик и удаленный инфракрасный маяк, а также ряд строительных деталей из серии Technic.

Чтобы найти необходимые детали, ознакомьтесь со схемой выше. Возможно, у вас уже и так есть многие детали из серии LEGO Technic. Ниже вы увидите несколько списков с деталями, необходимыми для сборки роботов из домашней версии конструктора и роботов, описанных в данной книге.

Мэтт О'Брин составил список деталей, которые нужны, чтобы превратить базовый образовательный набор (45544) и ресурсный набор (45560) в домашнюю версию (31313), и опубликовал его по адресу **goo.gl/QoOw9g**.

Питер Биттнер (Peter Bittner) составил документ Excel (**goo.gl/CY4llG**), он послужит вам ориентиром, когда вы будете искать детали, с помощью которых образовательная версия набора (45544) превратится в домашнюю (31313) или же образовательная версия + ресурсный набор (45544+45560) превратятся в домашнюю версию конструктора (31313).

Большинство деталей можно заказать в различных интернет-магазинах LEGO, торгующих ими поштучно (например, **ebricks.ru**, **homebricks.ru** или **educube.ru**). В качестве альтернативы можно приобрести большой набор LEGO Technic или ресурсный набор MINDSTORMS EV3 (см. раздел «Ресурсный образовательный набор LEGO MINDSTORMS EV3» далее). Это отличный способ сэкономить деньги и время, но, возможно, некоторые детали вам все же придется приобретать отдельно.

Модернизация домашней версии до образовательной

Аналогично, если у вас есть домашняя версия набора LEGO MINDSTORMS EV3 (арт. 31313), вы можете построить роботов из образовательной версии (арт. 45544). Для этого необходимо отдельно приобрести ультразвуковой и гироскопический датчики, а также ряд деталей из серии Technic. Следует отметить, что инфракрасный датчик имеет точно такую же форму, что и ультразвуковой. Оба датчика могут измерять расстояние, но ультразвуковой датчик отличается большей точностью при определении дистанции. С другой стороны, инфракрасный датчик служит еще и приемником для удаленного инфракрасного маяка. Вы сами должны решить, нужны ли вам оба датчика.

Программное обеспечение EV3

Роботы LEGO MINDSTORMS EV3 программируются с компьютера или планшета (под управлением операционной системы iOS или Android). Приложение для планшета представляет собой упрощенную версию настольной программы EV3, предлагающую ограниченные возможности программирования. Программное обеспечение для компьютера представляет собой полную версию, которая точно так же проста в применении.

Программное обеспечение как для компьютера, так и для планшета поставляется в двух вариантах: для домашней версии и для образовательной (для учителей и учеников).

Программное обеспечение для домашней версии

Домашняя версия набора LEGO MINDSTORMS EV3 (арт. 31313) поставляется без компакт-диска с программным обеспечением. Вместо этого необходимо загрузить бесплатное программное обеспечение с веб-сайта LEGO MINDSTORMS (**goo.gl/45qm2u**). Программное обеспечение включает в себя

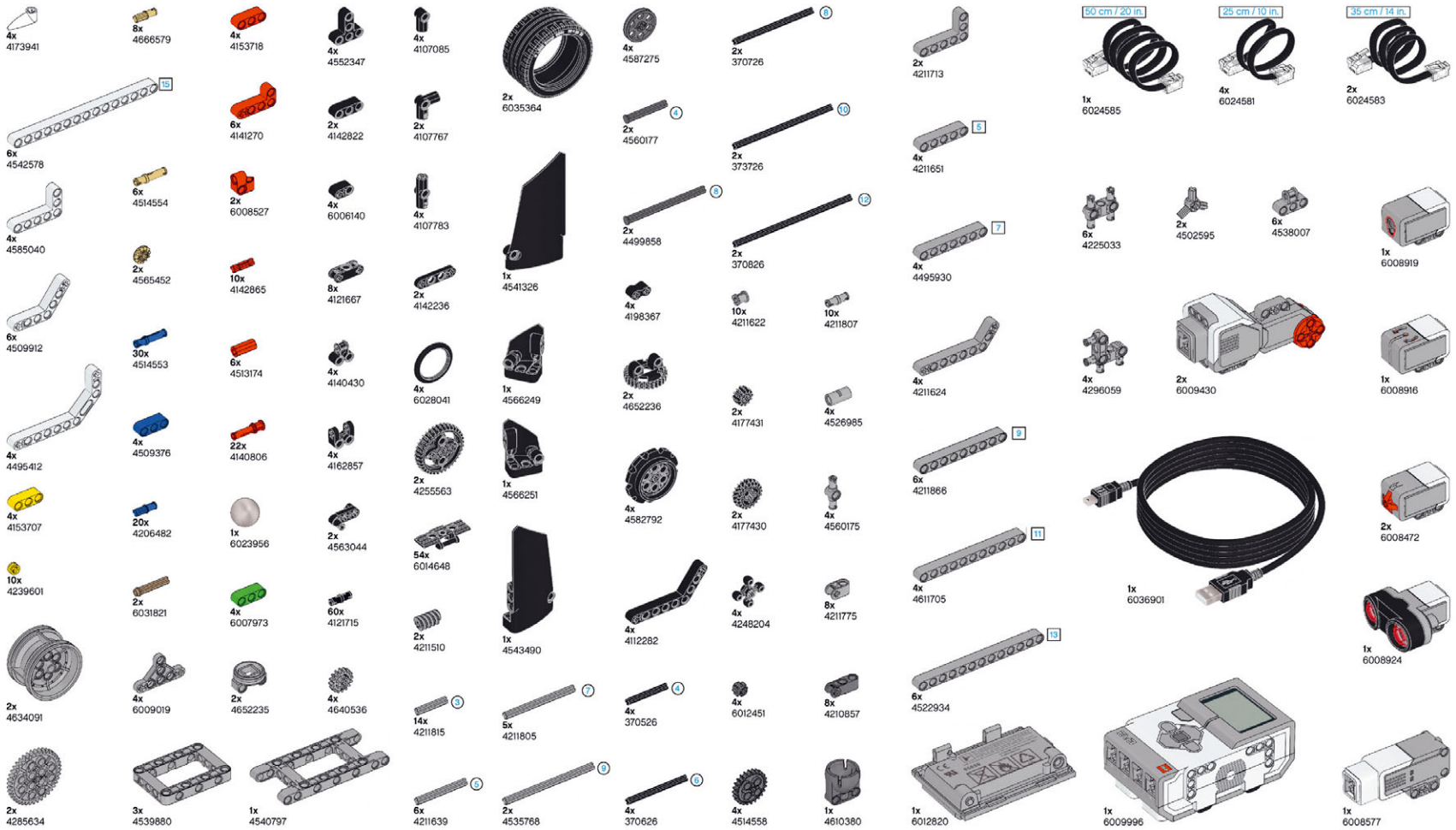


Рис. В.2. Состав набора LEGO MINDSTORMS Образовательный набор EV3 (45544)

среду программирования и инструкции по сборке и программированию всех моделей роботов, которых можно построить с помощью этого набора.

Программное обеспечение для образовательной версии

Программное обеспечение для образовательной версии набора LEGO MINDSTORMS EV3 (ар. 45544) приобретается отдельно. Оно включает в себя среду программирования и инструкции по сборке и программированию роботов, которых можно построить с помощью этого набора (арт. 45544) и ресурсного набора (арт. 45560). Кроме того, программное обеспечение содержит функционал регистрации данных для научных экспериментов. Например, вы можете построить график уровня освещенности в комнате и увидеть, как его значения меняются со временем.

Использование программного обеспечения для домашней версии применительно к образовательному набору

В домашнюю версию набора входит тот же самый программируемый модуль EV3, что и в образовательную версию, так что вы можете использовать любую версию программного обеспечения, чтобы управлять роботом. На самом деле, вы можете установить обе версии программного обеспечения на одном компьютере, если хотите.

Таким образом, программное обеспечение для домашней версии можно применять для работы с образовательным набором и наоборот. Нужно будет просто установить дополнительное программное обеспечение для гироскопического и ультразвукового датчика. На странице **goo.gl/UXLRDE** вы найдете инструкции по загрузке и установке необходимого программного обеспечения.

Ресурсный образовательный набор LEGO MINDSTORMS EV3

Вы можете расширить возможности образовательной версии набора LEGO MINDSTORMS EV3 (арт. 45544) с помощью ресурсного набора (арт. 45560). Так вы сможете строить более крупных роботов, например слона или тележку для подъема по ступеням. Набор включает в себя более 850 строительных деталей, таких как балки, оси и зубчатые колеса, как показано на рис. B.3. Он не содержит никаких электронных компонентов.

Хотя этот набор был разработан для образовательной версии, он отлично подходит и для домашней версии EV3 (арт. 31313). Вы не сможете построить все модели из этого дополнительного набора, которые нарисованы на упаковке, потому что вам может не хватить некоторых элементов из образовательного набора, но благодаря дополнительным деталям можно создать собственных роботов. Например, набор содержит 9 O-образных рамок, 6 H-образных рамок, 13 балок 15M и различные зубчатые колеса, включая дифференциал и вращающуюся платформу.

Аккумуляторная батарея EV3

Домашняя версия набора LEGO MINDSTORMS EV3 не включает в себя аккумулятор. Вместо него вы можете использовать 6 батареек типоразмера AA. Также можно приобрести 6 (или лучше 12) аккумуляторных батарей AA с зарядным устройством, чтобы использовать один комплект, пока заряжается другой.

Вы также можете приобрести аккумулятор LEGO MINDSTORMS EV3 (арт. 45501) и зарядное устройство (арт. 8887). Так вы сможете заряжать своего робота во время работы. Когда батарея разрядится, вам не нужно будет разбирать робота. Аккумулятор немного увеличит размер модуля EV3.

Приложения для смартфона и планшета

Вы можете управлять EV3 с помощью приложения для устройств под управлением операционной системы iOS или Android. Официальное приложение ориентировано на пользователей домашней версии и предлагает элементы управления, рассчитанные на роботов домашней версии, но оно совместимо и с образовательной версией. Например, если у вас есть образовательная версия базового набора, вы можете управлять моделями из этого набора с помощью элементов управления TRACK3R. Кроме того, вы можете изготовить собственное дистанционное управление для каждого мотора, подключенного к модулю EV3. На планшете можно создавать программы для EV3. Все эти приложения доступны на сайте по адресу **goo.gl/fnGSI8**.

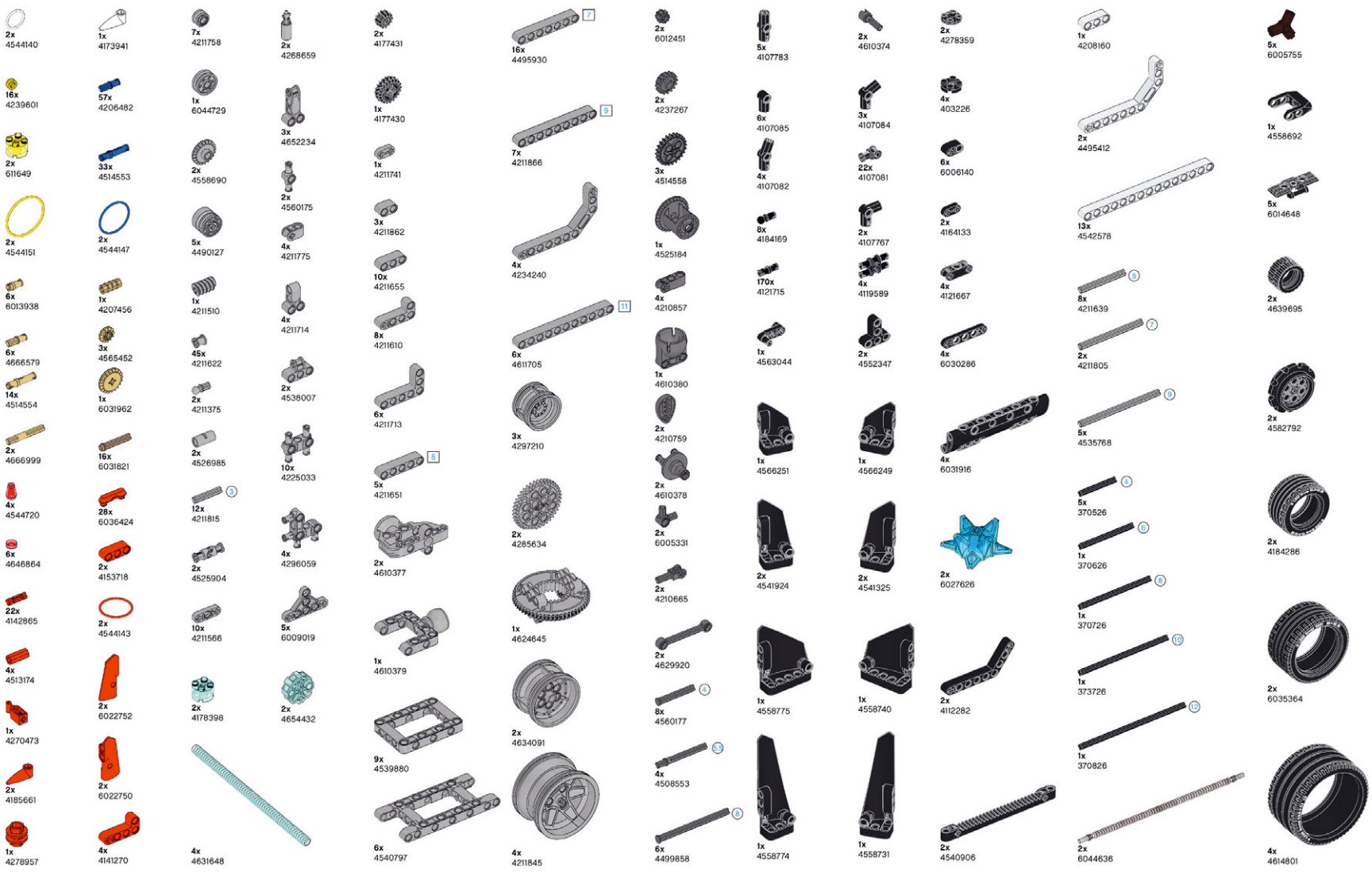


Рис. В.3. Состав набора LEGO MINDSTORMS ресурсный набор EV3 (45560)

Подключение через Bluetooth- и Wi-Fi-интерфейсы

После того как вы создали программу с помощью программного обеспечения LEGO MINDSTORMS EV3, ее можно перенести на модуль EV3 с помощью кабеля USB, включенного в комплект. После этого робот будет сам выполнять программу. Кроме того, можно передать программу в модуль робота по беспроводной сети с помощью Bluetooth- или Wi-Fi адаптера. Тогда вам не придется подключать USB-кабель при каждом изменении программы.

В домашних условиях это отличный вариант, но для учебных заведений он может не подойти, поскольку учебные компьютеры могут быть лишены администраторских прав (для подключения Bluetooth) или могут не обладать доступом к настройке параметров беспроводной сети (для Wi-Fi). Если вы планируете пользоваться возможностями беспроводной связи в учебном заведении, предварительно уточните эти нюансы.

Г

BRICK SORT3R: сортировщик деталей LEGO

Робот BRICK SORT3R, показанный на рис. 19.21 в главе 19, может сортировать детали LEGO по цвету и размеру.

Вам когда-нибудь хотелось, чтобы детали LEGO сортировались автоматически? Теперь вы можете собрать робота, который будет делать это за вас! BRICK SORT3R различает детали LEGO по цвету и размеру. В этом приложении представлены инструкции по сборке и программированию робота BRICK SORT3R на основе набора EV3.

Механизм сортировщика покрывается белыми декоративными панелями. Чтобы понять, как робот будет определять цвет и размер деталей LEGO, прежде чем приступать к строительству, посмотрите видеоролик, посвященный работе робота, доступный по адресу **goo.gl/8cUn6u**.

Необходимые элементы

- Один набор LEGO MINDSTORMS EV3 (№ 31313), необходимый для сборки робота BRICK SORT3R.
- Несколько деталей LEGO для сортировки: детали-кирпичики размером 2×4 и 2×2 красного, желтого, зеленого и синего цвета.

После того как вы протестируете программу, вы можете настроить ее для определения деталей других цветов и размеров.

При желании можно использовать набор маленьких емкостей для сортировки деталей LEGO. Вы можете изменить программу, чтобы робот учитывал размеры используемых емкостей (см. практикум № 133). Вы также можете собрать емкости из деталей LEGO, как показано в видеоролике в начале этой главы. Если у вас нет никаких емкостей, которые можно использовать для сортировки, пусть робот просто раскладывает детали на полу.

Сборка робота BRICK SORT3R

Для начала выберите нужные детали из вашего набора EV3, которые показаны на рис. Г.1.

Инструкцию по сборке в формате PDF вы найдете в папке *BRICK-SORT3R* архива, который доступен для скачивания по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip**. В этой же папке находится программа для робота BRICK SORT3R.

Разработка программы для робота BRICK SORT3R

Скачайте архив с файлами, прилагаемыми к книге, по ссылке **eksmo.ru/files/Lego_Mindstorms_Primers.zip** и распакуйте в любой каталог. Найдите в нем файл *BRICK-SORT3R.ev3* и откройте его в программе EV3.

Программа для робота BRICK SORT3R показана на рис. Г.2.

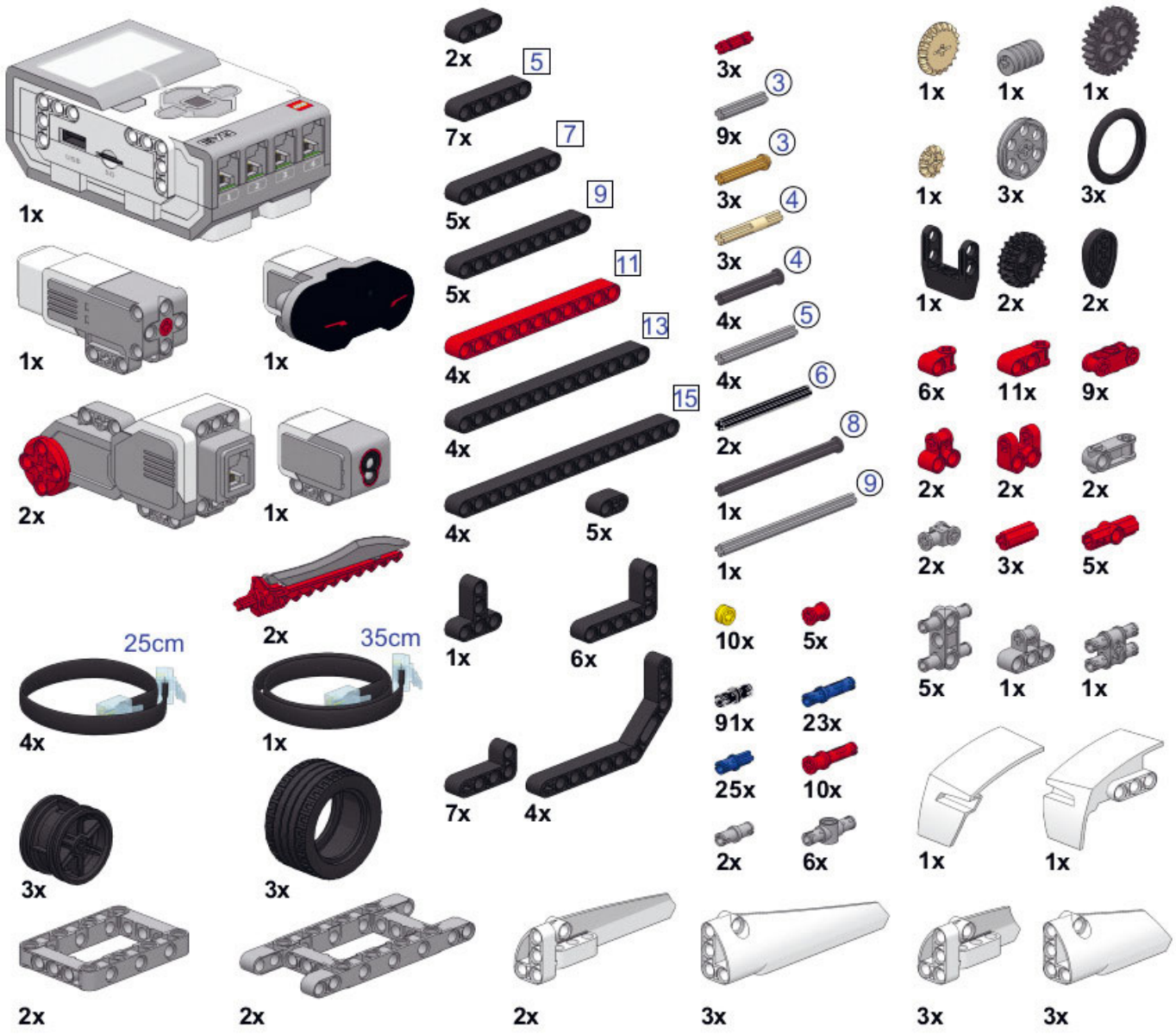


Рис. Г.1. Детали, необходимые для сборки робота BRICK SORT3R

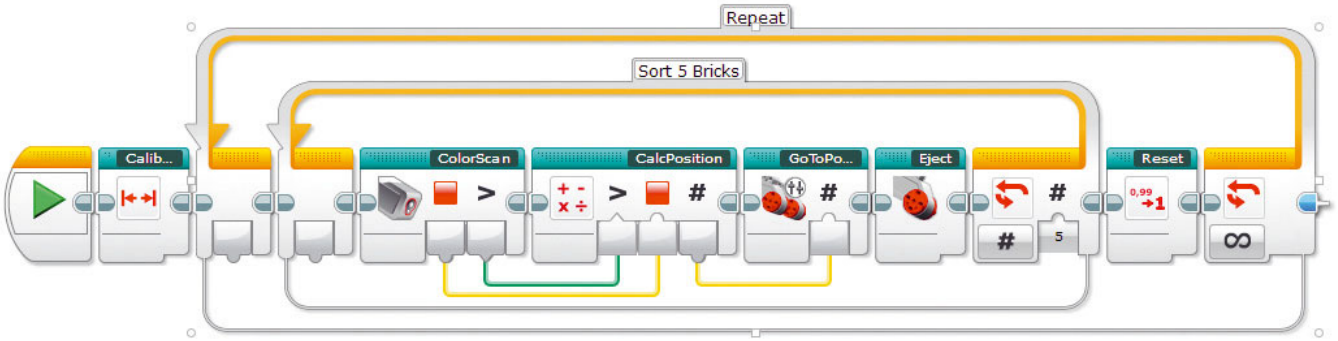


Рис. Г.2. Программа для робота BRICK SORT3R

Проверка работоспособности робота

После того как робот собран, а программа загружена в память модуля EV3, проверим работу сортировщика. Для этого выполните следующие действия.

1. Поместите несколько деталей LEGO в подающий желоб, как показано на рис. Г.3.
2. Установите отдельную конструкцию-ориентир напротив емкости с большими синими деталями, как показано на рис. Г.3. Благодаря этому робот «знает», где находится каждая емкость с деталями LEGO.
3. Поместите робота между емкостями с деталями LEGO и ориентиром, как показано на рис. Г.3.
4. Запустите программу.

Робот сначала будет двигаться в противоположную от ориентира сторону, а затем обратно, пока инфракрасный датчик робота не обнаружит ориентир. После этого робот начнет сортировать детали. После сортировки пяти деталей робот выполнит сброс настроек и вернется к ориентиру, прежде чем продолжить сортировку.

Дальнейшее изучение

Поздравляем, вы успешно собрали и запрограммировали робота для сортировки деталей LEGO! Вам интересно больше узнать о конструкции этого робота? Попробуйте выполнить практикумы, приведенные далее.

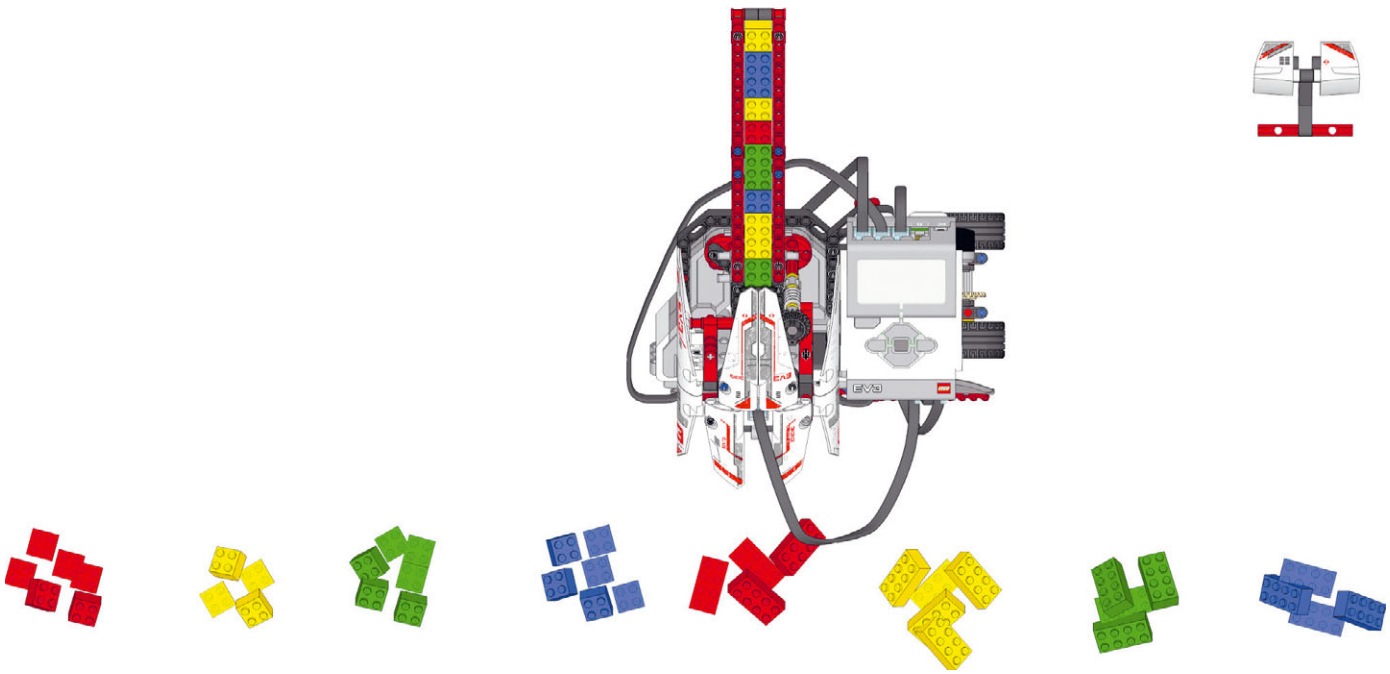


Рис. Г.3. Установка робота BRICK SORT3R в исходную позицию

ПРАКТИКУМ № 133: РАЗЛИЧНЫЕ ПРИЕМНИКИ!

Вы можете использовать емкости для отсортированных деталей другого размера, чем те, которые показаны в видеоролике. Чтобы убедиться, что ваш робот работает правильно, вам понадобится настроить программу соответствующим образом. В видеоролике робот должен поворачивать средний мотор на 387 градусов для перемещения от одной емкости к другой. Вы можете измерить необходимое количество градусов с помощью приложения **Port View** (см. рис. 6.4 в главе 6) и повернуть колеса вручную. После того как вычислите правильное значение для вашего случая, укажите его в программе. Для этого дважды щелкните мышью по контейнеру **CalcPosition** и замените значение 387 градусов в крайнем правом блоке **Математика** (Math).

ПРАКТИКУМ № 134: БОЛЬШЕ ЦВЕТОВ!

В моей версии программы можно сортировать только детали красного, желтого, зеленого и синего цветов. Но датчик цвета определяет еще и черный, белый и коричневый цвета. Можете ли вы дополнить программу, чтобы робот сортировал детали и этих цветов?

ПРАКТИКУМ № 135: ДРУГИЕ РАЗМЕРЫ!

Исходная программа различает только два размера деталей (2×4 и 2×2). Можете ли вы дополнить ее, чтобы робот мог сортировать детали и других размеров? Попробуйте добавить детали размером 2×3 и 2×6 к тем, которые уже используются.

ПРАКТИКУМ № 136: ПУСТОЙ ЖЕЛОБ!

Исходная программа не предпринимает никаких действий, если у робота кончились детали для сортировки. Можете ли вы дополнить программу, чтобы робот реагировал, если желоб становится пустым? Робот должен прекратить сортировку и воспроизвести некоторый звук.

СДЕЛАЙ САМ № 31: СОРТИРОВКА С ВРАЩЕНИЕМ!

Текущая конструкция робота-сортировщика позволяет помещать детали в разные емкости, перемещаясь влево и вправо. Попробуйте собрать и запрограммировать для BRICK SORT3R поворотную платформу с использованием среднего мотора. Это позволит роботу раскладывать детали LEGO в различные емкости, расположенные вокруг него. Такая конструкция будет более эффективной, так как занимает меньше места.

СОВЕТ **Используйте поворотную платформу, которую вы разработали в практикуме «Сделай сам № 18».**

Предметный указатель

Э

Научно-популярное издание

ПОДАРОЧНЫЕ ИЗДАНИЯ. КОМПЬЮТЕР

Валк Лоренс

БОЛЬШАЯ КНИГА LEGO MINDSTORMS EV3

Директор редакции *Е. Капьёв*
Ответственный редактор *В. Обручев*
Художественный редактор *Е. Мишина*

ООО «Издательство «Э»
123308, Москва, ул. Зорге, д. 1. Тел. 8 (495) 411-68-86.

Өндіруші: «Э» АҚБ Баспасы, 123308, Мәскеу, Ресей, Зорге көшесі, 1 үй.
Тел. 8 (495) 411-68-86.
Тауар белгісі: «Э»
Қазақстан Республикасында дистрибьютор және өнім бойынша арыз-талаптарды қабылдаушының
өкілі «РДЦ-Алматы» ЖШС, Алматы қ., Домбровский көш., 3«а», литер Б, офис 1.
Тел.: 8 (727) 251-59-89/90/91/92, факс: 8 (727) 251 58 12 вн. 107.
Өнімнің жарамдылық мерзімі шектелмеген.
Сертификация туралы ақпарат сайтта Өндіруші «Э»

Сведения о подтверждении соответствия издания согласно законодательству РФ
о техническом регулировании можно получить на сайте Издательства «Э»

Өндірген мемлекет: Ресей
Сертификация қарастырылмаған

Подписано в печать 14.07.2017. Формат 60x84$^{1}/_{8}$.
Печать офсетная. Усл. печ. л. 47,6.
Тираж экз. Заказ

ISBN 978-5-699-94356-2

Оптовая торговля книгами Издательства «Э»:
142700, Московская обл., Ленинский р-н, г. Видное,
Белокаменное ш., д. 1, многоканальный тел.: 411-50-74.

По вопросам приобретения книг Издательства «Э» зарубежными оптовыми покупателями обращаться в отдел зарубежных продаж
International Sales: International wholesale customers should contact Foreign Sales Department for their orders.

По вопросам заказа книг корпоративным клиентам, в том числе в специальном оформлении, *обращаться по тел.: +7 (495) 411-68-59, доб. 2261.*

Оптовая торговля бумажно-беловыми и канцелярскими товарами для школы и офиса:
142702, Московская обл., Ленинский р-н, г. Видное-2,
Белокаменное ш., д. 1, а/я 5. Тел./факс: +7 (495) 745-28-87 (многоканальный).

Полный ассортимент книг издательства для оптовых покупателей:
Москва. Адрес: 142701, Московская область, Ленинский р-н,
г. Видное, Белокаменное шоссе, д. 1. Телефон: +7 (495) 411-50-74.
Нижний Новгород. Филиал в Нижнем Новгороде. Адрес: 603094,
г. Нижний Новгород, улица Карпинского, дом 29, бизнес-парк «Грин Плаза».
Телефон: +7 (831) 216-15-91 (92, 93, 94).
Санкт-Петербург. ООО «СЗКО». Адрес: 192029, г. Санкт-Петербург, пр. Обуховской Обороны,
д. 84, лит. «Е». Телефон: +7 (812) 365-46-03 / 04. **E-mail:** server@szko.ru
Екатеринбург. Филиал в г. Екатеринбурге. Адрес: 620024,
г. Екатеринбург, ул. Новинская, д. 2щ. Телефон: +7 (343) 272-72-01 (02/03/04/05/06/08).
Самара. Филиал в г. Самаре. Адрес: 443052, г. Самара, пр-т Кирова, д. 75/1, лит. «Е».
Телефон: +7 (846) 269-66-70 (71…73). **E-mail:** RDC-samara@mail.ru
Ростов-на-Дону. Филиал в г. Ростове-на-Дону. Адрес: 344023,
г. Ростов-на-Дону, ул. Страны Советов, 44 А. Телефон: +7(863) 303-62-10.
Центр оптово-розничных продаж Cash&Carry в г. Ростове-на-Дону. Адрес: 344023,
г. Ростов-на-Дону, ул. Страны Советов, д.44 В. Телефон: (863) 303-62-10. Режим работы: с 9-00 до 19-00.
Новосибирск. Филиал в г. Новосибирске. Адрес: 630015,
г. Новосибирск, Комбинатский пер., д. 3. Телефон: +7(383) 289-91-42.
Хабаровск. Филиал РДЦ Новосибирск в Хабаровске. Адрес: 680000, г. Хабаровск,
пер.Дзержинского, д.24, литера Б, офис 1. Телефон: +7(4212) 910-120.
Тюмень. Филиал в г. Тюмени. Центр оптово-розничных продаж Cash&Carry в г. Тюмени.
Адрес: 625022, г. Тюмень, ул. Алебашевская, 9А (ТЦ Перестройка+).
Телефон: +7 (3452) 21-53-96/ 97/ 98.
Краснодар. Обособленное подразделение в г. Краснодаре
Центр оптово-розничных продаж Cash&Carry в г. Краснодаре
Адрес: 350018, г. Краснодар, ул. Сормовская, д. 7, лит. «Г». Телефон: (861) 234-43-01(02).
Республика Беларусь. Центр оптово-розничных продаж Cash&Carry в г.Минске. Адрес: 220014,
Республика Беларусь, г. Минск, проспект Жукова, 44, пом. 1-17, ТЦ «Outleto».
Телефон: +375 17 251-40-23; +375 44 581-81-92. Режим работы: с 10-00 до 22-00.
Казахстан. РДЦ Алматы. Адрес: 050039, г. Алматы, ул.Домбровского, 3 «А».
Телефон: +7 (727) 251-58-12, 251-59-90 (91,92,99).
Украина. ООО «Форс Украина». Адрес: 04073, г.Киев, Московский пр-т, д.9.
Телефон: +38 (044) 290-99-44. **E-mail:** sales@forsukraine.com

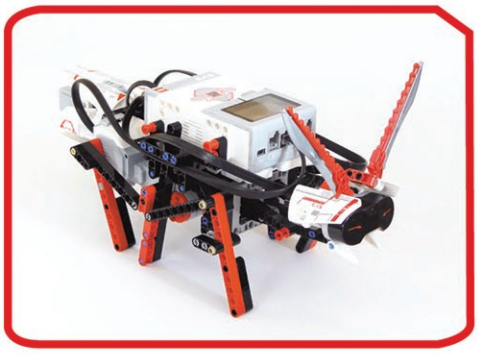
Полный ассортимент продукции Издательства «Э» можно приобрести в магазинах «Новый книжный» и «Читай-город».
Телефон единой справочной: 8 (800) 444-8-444. Звонок по России бесплатный.

В Санкт-Петербурге: в магазине «Парк Культуры и Чтения БУКВОЕД», Невский пр-т, д.46.
Тел.: +7(812)601-0-601, www.bookvoed.ru

Розничная продажа книг с доставкой по всему миру. Тел.: +7 (495) 745-89-14.

Воображай! Создавай! Изобретай!

Открой увлекательный мир LEGO® MINDSTORMS® EV3!



С «Большой книгой LEGO MINDSTORMS EV3» вы научитесь собирать и программировать роботов – от самых простых до очень сложных. Вместе с этим вы научитесь основам программирования с использованием шин данных, переменных и контейнеров «Мой блок».

Вы научитесь собирать и программировать:

- колесные транспортные средства, которые смогут перемещаться по комнате и следовать по трассе;
- обтекаемый гоночный автомобиль на дистанционном управлении;
- шестиногого шагающего робота-муравья;
- роботизированную руку, которая может самостоятельно находить, поднимать и перемещать предметы;
- говорящего и ходящего человекоподобного робота.

Более **150** практикумов по сборке и программированию научат вас творчески мыслить и изобретать собственных роботов, используя новые знания. С помощью **«Большой книги LEGO MINDSTORMS EV3»** вы очень быстро научитесь конструировать собственные модели!

Все, что нужно для постройки собственных моделей, вы найдете в наборе **LEGO MINDSTORMS EV3** (#31313)

Примеры к книге можно скачать по адресу **https://eksmo.ru/files/Lego_Mindstorms_Primers.zip**

Лоренс Валк (Laurens Valk) состоит в международном сообществе фанатов серии MINDSTORMS, помогающих тестировать и разрабатывать новые продукты в этой линейке. Его книги помогли десяткам тысяч начинающих любителей роботов по всему миру приобщиться к интереснейшему миру робототехники. Лоренс ведет свой блог о роботах на сайте robotsquare.com.

ISBN 978-5-699-94356-2

Очень подробная инструкция по Mindstorms EV3 с разбором программирования и детальными сборками. Прочитав эту книгу, вы можете смело приступать к созданию и программированию своего робота и экспериментам с разными задачами.

Степан Плохотнов, Директор Лиги Роботов, Москва