Микроконтроллер – это просто!

Кодрик Виктор Владимирович
учитель информатики гимназии №278, г. Санкт-Петербург.

Робототехника является одним из важнейших направлений научно-технического прогресса, в котором проблемы механики и новых технологий соприкасаются с проблемами искусственного интеллекта. За последние годы успехи в робототехнике и автоматизированных системах позволяют изменить подход при изучении предмета “информатика”, сделать его более практико ориентированным, наглядным. Использование элементов робототехники на уроках позволяют ребятам на “живом примере” увидеть результаты работы того или иного алгоритма, воплотить в жизнь ту или иную идею.

Кроме того приобретение компетенций из области робототехники необходимо для полноценной реализации метода учебных проектов в современных условиях, где конструкторы выступают инструментом, с помощью которого учащийся может реализовать свои идеи. А привнесение соревновательной направленности предмета, позволяет учащемуся оценить те или иные недостатки или преимущества его технического решения в сравнении с решениями сверстников, позволяет зародить между учащимися дискуссию по решению проблемы, выбранному методу решения.

Проблемы внедрения элементов робототехники в школе

Одним из самых доступных на сегодняшний день контроллеров являются контроллеры на базе Arduino. Их основными производителями в России являются такие компании как Амперка или РОББО. Так сложилось, что программное обеспечение для этих микроконтроллеров базируется либо на языке Wiring либо на Scratch-подобных языках программирования. Для различных категорий учащихся следует использовать конструкторы и ПО с

учетом его специфики (преимуществ и недостатков). Обзор существующих языков программирования микроконтроллеров Arduino можно посмотреть в [1]. К примеру, для учащихся первого года обучения – программирование с использованием C++ является чрезвычайно сложным. Для этих целей стоит использовать “интерпретируемые языки”, которые позволяют составить алгоритм в визуальной среде, а затем сгенерировать непосредственно код программы, которую довольно просто через оболочку Arduino IDE перевести в машинные коды и записать в контроллер.

Большинство современных визуальных языков программирования основано на языке Scrach - разработки МИТ [2], которая в свое время, сделала доступным изучение основ алгоритмизации в младшей школе. Однако имеется ряд проблем, которые не позволяют полноценно использовать такие среды при программировании контроллеров. В связи с тем, что при создании языка Scratch не были заложены основные концепции связанные с программированием роботов, последующие надстройки над данным языком программирования привели к тому, что он потерял основную наглядную составляющую. Программа стала представлять из себя смесь блоков в которых прописывались команды роботу на текстовом языке, что делало код сложно читаемым

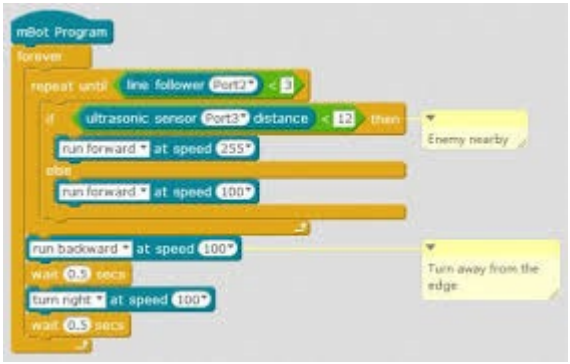

рис.1, а программирование в такой среде становилось еще более сложным чем на обычном текстовом языке.

Рис. 1. Пример кода программ на Scrach-подобном языке программирования роботов

Более того, в связи с теми же проблемами базовой среды, невозможно

провести проверку кода на этапе составления алгоритма. После того как блоки выстроены, на этапе генерации кода оказывается, что допущены синтаксические ошибки в текстовых вставках. Генератор не может связать номер блока со строкой сгенерированного кода, а потому найти блок с допущенной ошибкой вызывает сложности. В связи с этим было принято решение разработать язык программирования микроконтроллеров, который бы не содержал данных недостатком.

Визуальная среда программирования BlockCode

Разработанная среда позволяет составлять алгоритм программы в виде блок-схемы. Блоки команд сконструированы так, что не допускают ошибки,

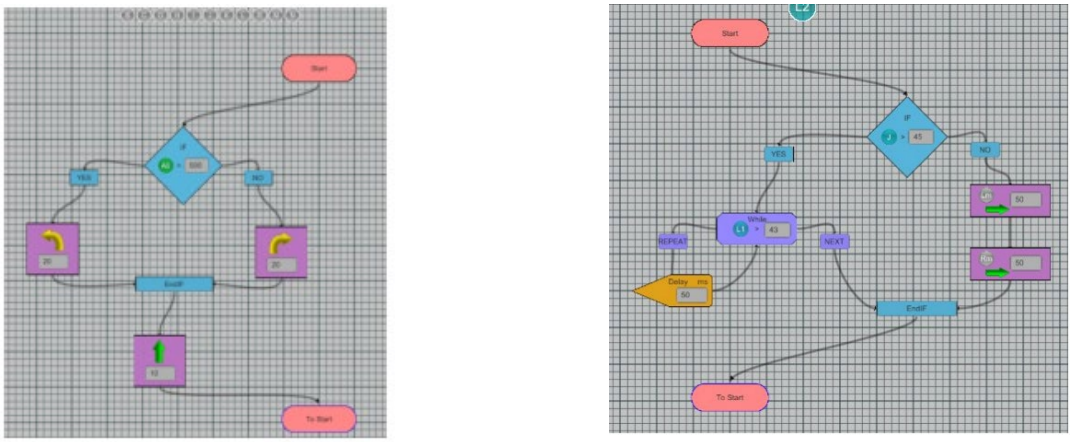

Рис. 2. Пример кода программ на в среде BlockCode

синтаксиса, позволяя концентрировать внимание ребят на логике работы алгоритма выявлении логических ошибок программы. Программирование робота разбито на два этапа. Сначала необходимо сконфигурировать его модель: указать типы подключаемых устройств, а также порты к которым они подсоединены (среда инкапсулирует работу связанную с получением данных с различных устройств, позволяя работать с ними как с переменными, не задумываясь о методе их получения). Затем нужно составить алгоритм управления роботом. Этот алгоритм может быть запущен в виртуальной модели, либо на его основе может быть сгенерирован

код для загрузки в плату. Модель позволяет проверить алгоритмы и служит для поиска ошибок программы. Генераторы кода, позволяют создавать программу для запуска ее на роботе под управлением LEGO Mindstorms Ev3 или Arduino Uno R3. Для компиляции программы под Lego, в настройках указывается расположение Microsoft Framework 4, а для работы с UNO, расположение Arduino IDE.

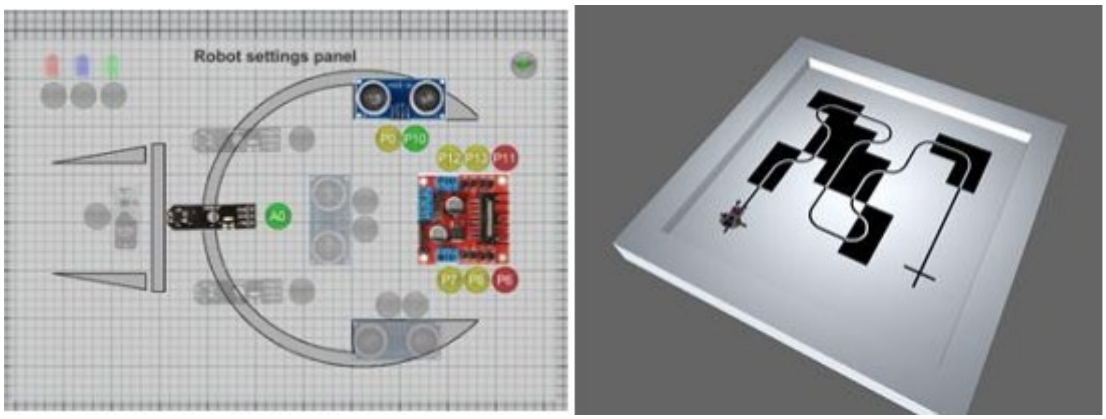

Рис. 3. Окно конфигурирования робота(слева) и виртуальная модель(слева)

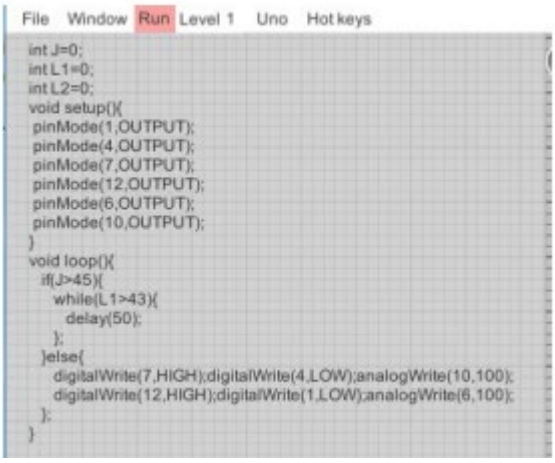

Рис. 4. Окно генератора кода

Робот как “исполнитель”

Программирование ведется путем манипулирования блоками команд. Они составлены таким образом, что позволяют управлять как отдельными элементами робота (моторами, сервоприводами, датчиками), так и роботом, как единым механизмом. Благодаря чему, становиться возможным

использование конструкторов для демонстрации работы исполнителей на уроках информатики. Рассмотрим следующую задачу: на поле имеется три лужи. Необходимо написать программу, в результате выполнения которой, робот сможет посетить их все [см. Рис.5].

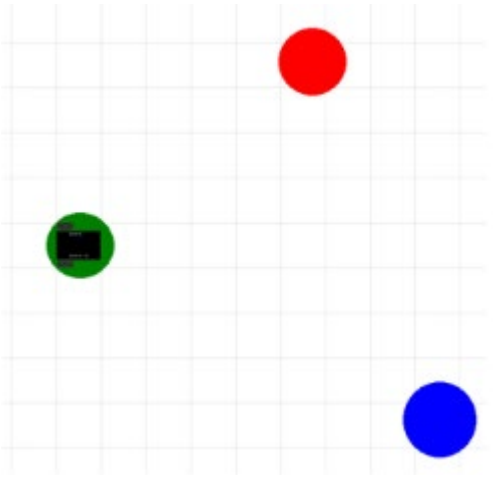

Рис 5. Поле

Будем двигаться от простого к сложному. Для начала решим эту задачу следующим образом, используя блоки - “Двигаться вперед на ..”, “повернуться вправо на ..”, “повернуться влево на..”. Понятно, что исполнителем в данном случае является робот, четко представлена его систему команд. Легко можно составить алгоритм решающий задачу:

Повернуться влево на 45

Двигаться вперед на 100

Повернуться вправо на 30

Двигаться вперед на 150

Однако, как правило, в робототехнике исполнителем является не сам робот целиком, а контроллер, который управляет отдельными агрегатами. Предположим он умеет выполнять следующие команды: “Левый мотор включить вперед”,“левый мотор включить назад”, “правый мотор включить вперед”, “правый мотор включить назад”, ждать n миллисекунд. В такой системе команд алгоритм решающий данную задачу будет выглядеть иначе:

Левый мотор включить назад

правый мотор включить вперед

ждать 450 миллисекунд

левый мотор включить вперед

ждать 1000 миллисекунд

правый мотор включить назад

ждать 300 миллисекунд

правый мотор включить вперед

ждать 1500 миллисекунд

Оба эти алгоритма возможно составить в BlockCode и продемонстрировать их работу “вживую”.

Заключение

Разрабатываемый визуальный язык программирования для микроконтроллера Arduino основан на визуальном представлении алгоритма, который интерпретирует блоки языка в программы на языке Wiring. Идеология предложенного языка позволяет защитить алгоритм от неверной записи команд и сосредоточиться на записи алгоритма. Код программы является простым для восприятия учащихся.

Источники:

[1]. Обзор визуальных языков программирования под Arduino https://novator.team/post/566

[2]. Визуальный язык программирования Scrach https://scratch.mit.edu