ПРИБОР, КОТОРЫМ УПРАВЛЯЕТ «ПРОГРАММА». КАК РАБОТАЕТ СПИДОМЕТР ДЛЯ ВЕЛОСИПЕДА?

Шеледнев А.Д.

г.Арзамас, МБОУ «Лицей», 4 Б класс

Научный руководитель: Соменкова Н.В., учитель начальных классов, г.Арзамас, МБОУ «Лицей»

В наше время люди используют большое количество электронных устройств, которые помогают нам и облегчают нашу жизнь. И очень часто, каким бы устройством мы не пользовались в семье я очень часто слышу слово «программа». Когда включаю микроволновку - я выбираю программу разогрева пищи. Когда мама использует посудомоечную или стиральную машину она тоже выбирает «программы», согласно которых домашние устройства выполняют свою полезную работу.

Мне захотелось самому понять что-же такое программа и разобраться, смогу ли я сам сделать полезные устройства, которые бы работали с помощью программ.

Пока я разбирался с тем как писать программы, какими они бывают, я сделал много интересных вещей, и в конце концов мне захотелось поставить программу... на мой велосипед!

Точнее я решил сделать управляемый программой стрелочный спидометр, который мог бы показывать скорость моего движения на велосипеде.

Темой моего исследования стал вопрос: Как сделать прибор, которым управляет программа?

Я поставил себе цель: Понять, что такое программа, научиться писать программы и разобраться как программы могут управлять полезными устройствами.

Для достижения цели необходимо ответить на следующие вопросы (задачи исследования):

- Где может работать программа?
- Что такое программа, как написать программу для небольшого устройства?
- Как программы могут управлять устройствами и механизмами?
- Как программа может получать информацию из внешнего мира?
- Как написать программу и заставить ее выполнять полезную работу?

В ходе рассуждений на тему исследования мной были выдвинуты предположения:

Для работы программы необходим компьютер, а для небольшого устройства нужен специальный микрокомпьютер или микроконтроллер.

Для получения программой информации из внешнего мира необходимо использовать специальные устройства для ввода данных или сенсоры, подключенные к компьютеру.

Для отображения информации или воздействия на внешнюю среду необходимы устройства вывода или исполнительные механизмы.

Для самостоятельного создания полезного прибора, управляемого программой необходим компьютер, устройства ввода и устройства вывода.

Где может работать программа?

Всем известно, что программы выполняются в компьютерах. Но компьютеры и даже ноутбуки вещи достаточно большие, а наша конечная цель – сделать спидометр для велосипеда. Поэтому использование обычных компьютеров создаст для нас следующие проблемы:

Закрепить компьютер на велосипеде – задача трудоемкая и выглядеть такой спидометр будет очень смешно и возить его с собой будет тяжело и неудобно.

Как подключить электропитание? Ведь обычный компьютер потребляет очень много электроэнергии, не вести же за собой многокилометровый электро-удлинитель?

Поэтому для создания небольших приборов существуют специальные маленькие компьютеры, которые могут быть даже без экрана, клавиатуры и мыши и которые сделаны специально для того чтобы выполнять строго определенные и относительно простые задачи. Такие компьютеры называются микроконтроллерами или просто контроллерами.

На сегодняшний день самый популярный и недорогой микроконтроллер для самостоятельного создания полезных вещей является контроллер под названием Arduino UNO (рис. 1.)

Микроконтроллер Arduino UNO может хранить программу внутри главной микросхемы, и выполнять ее автономно.

Записать программу в микроконтроллер можно при помощи специальной программы для обычного компьютера, подключив микроконтроллер к компьютеру при помощи интерфейса USB.

Рис. 1. Микроконтроллер Arduino UNO

Вывод: Изучив возможные устройства, которые могут выполнять программы, я пришел к выводу, что для создания небольшого прибора, управляемого программой необходимо использовать микроконтроллер. Чтобы создать спидометр для моего велосипеда, я решил воспользоваться микроконтроллером Arduino UNO.

Что такое программа, как написать программу для небольшого устройства?

Углубляясь в изучение вопроса о том, как написать программу для выбранного мной контроллера Arduino UNO, оказалось, что для этого необходимо установить на обычный стационарный компьютер специальную среду для разработки, которую можно скачать с сайта http://arduino.cc/ (рис.2)

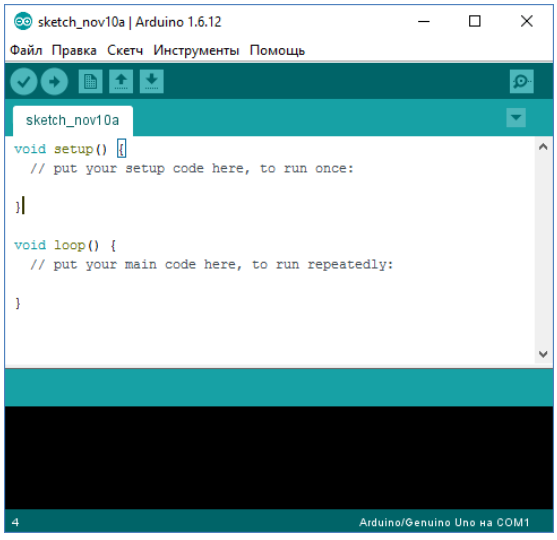

Рис. 2 Среда программирования для Arduino

В этой среде, используя специальный язык программирования, можно писать последовательность действий который должен совершать контроллер. Эта последовательность действий и называется программой. Программу для контроллера Arduino еще называют скетчем (sketch).

Контроллер будет в точности следовать написанному скетчу, если загрузить его при помощи USB-кабеля.

Для того чтобы начать писать сскетчи для микроконтроллера Arduino я разобрался со следующими основными элементами программ: функции; переменные; математические операторы; операторы ветвления; циклы.

Вывод: Программа - это последовательность действий, описанных на специальном языке программирования, которые в точности будет выполнять компьютер или контроллер. Для создания простой программы для контроллера **Arduino** необходимо знать**,** что такое функции, переменные, операторы, команды ветвления и циклы.

Как программа может управлять устройствами и механизмами?

На рисунке 3 изображен контроллер Arduino, на котором можно увидеть 13 цифровых входов-выходов.

Рис. 3. Цифровые входы и выходы и аналоговые входы Arduino

Цифровые входы-выходы - это контакты которые можно использовать и как входы, и как выходы. Как будет работать каждый контакт можно задать в скетче, разместив специальную команду в функции setup().

Когда, цифровые контакты используются в качестве выходов, они действуют подобно маломощным источникам электропитания, которые при помощи специальных команд можно включать или выключать. Таким образом, подавая и отключая электропитания на контакты мы можем управлять разными устройствами.

Я, изучая Arduino подключал к нему и мог управлять следующими устройствами: светодиоды; Электромотор; сервопривод;

Автомат для запуска мыльных пузырей:

Воспользовавшись полученными знаниями по управлению электродвигателем и сервоприводом, я создал автомат для запуска мыльных пузырей и написал программу, которая заставляет его работать (Рис.4, приложение 1). Сделанный мной автомат

очень весело работал, самостоятельно пуская мыльные пузыри, что очень нравилось моему маленькому двоюродному братику Платону.

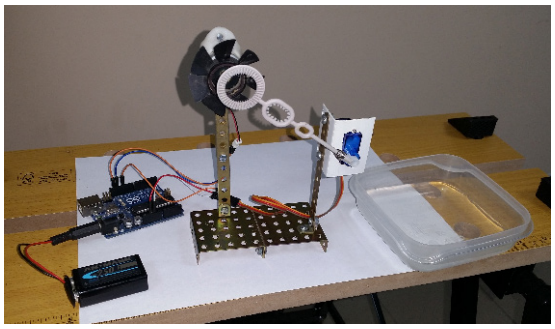

Рис. 4 Автомат для мыльных пузырей.

Вывод: Контроллер Arduino может управлять различными устройствами при помощи цифровых выходов, такими как светодиодами, моторами и сервоприводами. Указанные механизмы можно использовать для создания различных полезных вещей и роботов.

Как программа может получать информацию из внешнего мира?

На рисунке 3 можно увидеть, что у контроллера Arduino кроме 13 цифровых входов-выходов есть еще 6 аналоговых входов, которые отмечены как входы A0, A1, A2, A3, A4, A5, A6.

Цифровые и аналоговые входы, могут быть использованы для получения разнообразной информации из внешнего мира при помощи специальных устройств, которые называются сенсоры.

Цифровые входы могут сообщить нам о наступлении какого-либо события, измеренного подключенным сенсором (например, нажата кнопка или нет, сработал датчик присутствия или нет). Аналоговые входы могут дать более расширенную числовую информацию от разных сенсоров (например, температуру, положение вала потенциометра или расстояние до препятствия от дальномера)

Изучая цифровые и аналоговые входы Arduino, я подключал к нему и мог получать информацию со следующих сенсоров: кнопка или геркон; датчик присутствия; потенциометр; датчик температуры и датчик освещения; ультразвуковой дальномер.

Вывод: Контроллер **Arduino** может получать информацию о внешней среде и команды управления при помощи цифровых и аналоговых входов и подключенных к ним сенсоров. При помощи сенсоров поведения программы может изменяться, подстраиваться под пользователя прибора или внешнюю среду. Использование различных сенсоров позволяет создавать «разумные» автоматические устройства, которые становятся очень похожими на роботов.

Создание стрелочного спидометра для велосипеда

После того, как я разобрался с тем, как можно использовать входы и выходы контроллера Arduino, мне захотелось сделать действительно полезную вещь, которая управлялась бы написанной мной программой. Такой идеей стал спидометр для моего велосипеда.

Для изготовления такого прибора мне понадобилось минимальное количество материалов, это: контроллер Arduino; магнит; геркон; светодиод; сервопривод; два резистора. Устройство было собрано как изображено на рисунке 5.

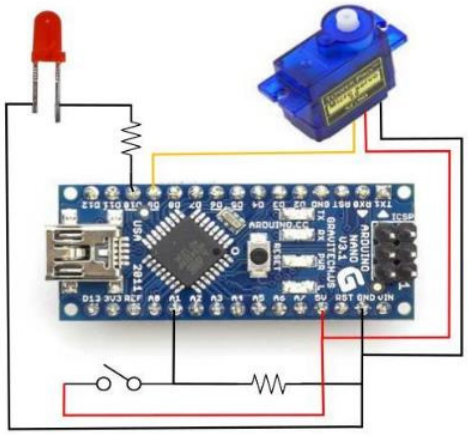

Рис. 5. Электрическая схема стрелочного спидометра

Подключение светодиода.

Мы подключили светодиод к цифровому выходу 10, и будем подавать на него напряжение командой **digitaWrite**, чтобы светодиод включался, когда геркон окажется около магнита и выключался когда магнит «отъедет» от геркона. В результате при вращении колеса мы будем видеть, как мигает светодиод.

Подключение геркона.

Геркон мы подключим одним контактом к выводу питания 5Вольт, а другим контактом к аналоговому входу A1. В результате, когда геркон будет замкнут на вход A1 потечет электрический ток от вывода питания и мы сможем замерить появление тока командой **analogRead**.

Чтобы при размыкании геркона, команда **analogRead** выдавала нам нулевое напряжение соединим контакт геркона, со-

единенный с выходом A1 еще из контактом GND (земля) на контроллере при помощи резистора.

Таким образом, при размыкании геркона выход A1 будет соединен через резистор с землей и analogRead должно на выдать значение равное 0.

Но, проверяя работу Геркона с магнитом я выяснил, что с имеющимся у меня резистором при замкнутом герконе при помощи функции analogRead мы получаем значение около 1000 (однозначно больше, чем 100). А при разомкнутом герконе, полученное значение лежит в промежутке от 0 до 6 (однозначно меньше чем 100).

Поэтому, при написании программы мы будем считать геркон разомкнутым, если полученное при помощи analogRead значение меньше чем 100, и замкнутым если это значение больше, чем 100.

На рисунках 6-8 видно, как я закрепил на колесе велосипеда магнит и геркон, чтобы обеспечить замыкание геркона при каждом повороте колеса.

Рис. 6

Рис. 7

Рис. 8

Рис. 6-8. Подключение геркона

Подключение сервопривода

Сервопривод я подключил к контактам питания и GND, а также к цифровому выходу 9, при помощи которого мы будем задавать положение сервопривода.

На рисунке 9 видно какое табло спидометра мы сделали, чтобы показывать скорость передвижения велосипеда. На вал сервопривода я прикрепил красную стрелку, которая будет указывать на скорость передвижения велосипеда.

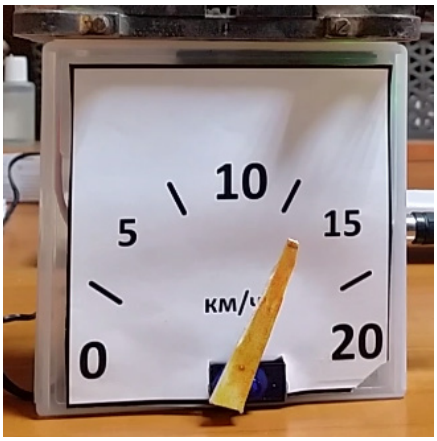

Рис. 9. Спидометр

Проверив работу сервопривода оказалось, что при скорости равной 0, угол отклонения вала сервопривода должен быть равен 180^0.

Таким образом при изменении скорости от 0 до 20 км/час мне будет необходимо изменять положение угла стрелки от 180 до 0 градусов.

Вычисления скорости.

Для того чтобы вычислить скорость передвижения велосипеда, мне нужно разделить пройденное велосипедом расстояние в километрах на время, за которое это расстояние было пройдено.

$$\text{скорость} = \frac{\text{расстояние}}{\text{время}}$$

Т.е. чтобы узнать какая скорость была у велосипеда во время одного поворота колеса нам нужно узнать расстояние, которое прошел велосипед за один оборот колеса и узнать время за которое этот оборот был совершен.

Расстояние величина, постоянная и просто равна длине колеса, которое я измерил, как показано на рисунках 10-11.

Расстояние оказалось равным 1,43 метрам. Но так как мы считаем скорость в километрах в час, то переведем это расстояние в километры:

$$\text{расстояние} = \frac{1{,}43}{1000}$$

Рис. 10 *Рис. 11*

Рис. 10-11. Измерение длины окружности колеса

Время, за которое, выполняется один оборот колеса я вычислил внутри моей программы по формуле:

$$t\_ms = \text{millis}() - ms,$$

где millis() – функция, которое выдает нам текущее время (т.е. когда было замыкание контакта геркона), а ms – это время когда было предыдущее срабатывание геркона. Значение ms мы каждый раз запоминаем при срабатывании геркона и используем его, при следующем замыкании.

Но, t_ms – это время поворота колеса в миллисекундах, а нам нужно в часах, поэтому переведем это время сначала в секунды $t_{сек}$, затем в минуты $t_{мин}$ и потом в часы $t_{час}$, итак:

$$t_{сек} = \frac{t\_ms}{1000},\ t_{мин} = \frac{t_{сек}}{60},\ t_{час} = \frac{t_{мин}}{60},$$

а значит:

$$t_{час} = \frac{t\_ms}{1000 \cdot 60 \cdot 60} = \frac{t\_ms}{1000 \cdot 3600}$$

Ну и скорость в километрах в час будет равна:

$$\text{скорость} = \frac{\text{расстояние}}{\text{время}} = \frac{1{,}43}{1000} \quad \frac{t\_ms}{1000 \cdot 3600} = \frac{1{,}43}{1000} \cdot \frac{1000 \cdot 3600}{t\_ms} = \frac{1{,}43 \cdot 3600}{t\_ms}.$$

Поэтому в программе, для вычисления скорости, я буду использовать следующее выражение:

speedometr = wheel_m·3600/t_ms.

Вычисление угла поворота вала сервопривода.

Так как я решил измерять скорость от 0 до 40 километров в час (быстрее велосипед не разгонится), то если бы 0км час соответствовал угол отклонения стрелки 0 градусов, а скорости 40 км/час угол 180 градусов, то скорости 1 км/час соответствовал бы угол равный $\frac{180}{40}$ градусов.

Поэтому, произвольной скорости V км/час, соответствовал бы угол, равный $V \cdot \frac{180}{40}$ градусов. Значит в программе угол отклонения стрелки мы вычисляем как:

angle = speedometer * 180 /40.

Но, так как у нашего сервопривода крайнее левое положение вала соответствует 180 градусам, а крайнее правое 0 градусам, то мне пришлось вычислить правильное значение для вала по следующей формуле:

180 – angle.

Программа спидометр

В результате моих размышлений я написал программу (приложение 2), которая показывает правильную скорость движения велосипеда на основании измеренных данных (длины колеса и времени одного его поворота).

Проверка точности показаний изготовленного спидометра.

Для проверки точности показаний созданного мной спидометра был приобретен велокомпьютер **Cyclotech i6** промышленного производства и также установлен на велосипед.

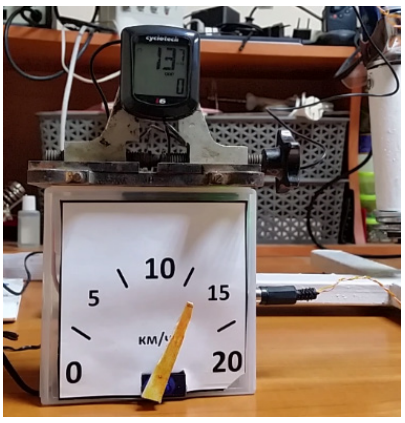

Рис. 12 Сравнение работы спидометра с эталоном

Во время движения колеса показания двух спидометров, работающих одновременно совпадали, что подтверждает правильность работы программы и моего устройства (рис. 12).

Вывод: используя полученные во время исследования знания у меня получилось создать полезное устройство, которое управляется программой. Я понял, что такое программа и создал много программ-скетчей, которые обеспечивали работу автоматических устройств.

Приложение 1

Автомат для мыльных пузырей

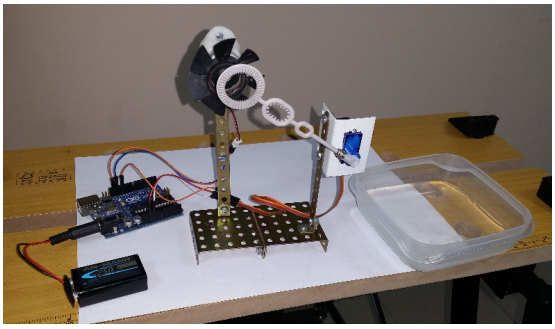

Рис. 1 Автомат для мыльных пузырей

```
#include <Servo.h>
Servo s;

void setup() {
s.attach(10);// Подключаем сервопривод
s.write(0);  // Поворачиваем серво на 0 градусов
pinMode(3,OUTPUT);        // Подключаем мотор
}

void loop() {
analogWrite(3,0);    // Выключаем мотор
to_mylo();  //    Запускаем    функцию для «макания» в раствор
delay(2000);          // Ждем 2 секунды пока «намылится»
to_vent();    // Перемещаем к вентилятору
analogWrite(3,200); // Включсаем вентилятор
delay(5000);          // Ждем 5 секунд
}

void to_mylo() {
int i = 180;  // Начальное значение угла 180гр (около вент)
while (i>0) {          // В цикле пока сервопривод не вернется к 0гр
i = i - 2;       // уменьшаем угол на 2 градуса
s.write(i);   // и отклонять на этот угол сервопривод
delay(20);  // каждый раз ждем 20 мс, чтобы поплавней
}
}

void to_vent() {
int i = 0;      // Начальное значение угла 0 гр (около мыла)
while (i<180) {{      // В цикле пока сервопривод не повернется на 180 гр
i = i + 2;       // увеличиваем угол на 2 градуса
s.write(i);   // и отклонять на этот угол сервопривод
delay(20);  // каждый раз ждем 20 мс, чтобы поплавней
if (i==44)    // Когда отклонимся на 45 гр, ждем чтобы излишки
{delay(2000);}        // мыльного раствора стекли
}
}
```

Приложение 2

Программа управления спидометром

```
#include <Servo.h> // Подключаем библиотеку для работы с серво
Servo s;       // создаем переменную s для управления серво
int led = 10;                // светодиод подключаем к цифровому выходу 10
int gerkon = A1;        // геркон подключаем к аналоговому входу A1
int servo_pin = 9;     // серво будет управляться через цифровой выход 9
float wheel_m = 1.43;          // измеренная длина колеса 1,43 метра
int gerkonstate;        // переменная для запоминания состояния геркона
int check;     // переменная для исключения повторного опроса уже замкнутого геркона
long ms;      // переменная для запоминания времени срабатывания геркона
long t_ms;  // переменная для вычисления времени одного оборота колеса
float speedometr;    // переменная для вычисления скорости
int angle = 0;          // переменная для вычисления угла поворота спидометра

void setup() {
 pinMode(led, OUTPUT);    // настраиваем выход для светодиода
 pinMode(gerkon, INPUT); // настраиваем вход для геркона
 s.attach(servo_pin);          // подключаем серво
 s.write(180);          // отклоняем серво на 180гр, что
      // соответствует нулю на табло
}

void loop() {
gerkonstate = analogRead(gerkon);  //считываем состояние геркона
```

```
    if (gerkonstate > 100 ) {     // если геркон замкнут то
    if ( check == 0 ) {  // если предыдущее состояние геркона
          // “разомкнут“
    digitalWrite(led, HIGH);     // тогда включаем светодиод
    t_ms = millis()-ms; // вычисляем время от предыдущего
          // срабатывания геркона
    speedometr = wheel_m*3600/t_ms;
// вычисляем скорость в км/час
    ms = millis();          // запоминаем время срабатывания
          // геркона
    check = 1;  // запоминаем что геркон замкнут
    };
    } else {      // иначе, т.е. когда геркон разомкнут
    digitalWrite(led, LOW);     // выключаем светодиод
    check = 0;  // запоминаем что геркон разомкнут
    }
    if ((millis()-ms)>1000)      // если геркон не срабатывал больше
    { speedometr = 0;};          // секунды, то скорость = 0
    angle = speedometr * (180/40);      // вычисляем угол сервопривода
    s.write(180-angle); // поворачиваем сервопривод
    }
```

Выводы

Для создания небольшого прибора, управляемого программой необходимо использовать микроконтроллер, например, Arduino UNO.

Программа – это последовательность действий, описанных на специальном языке программирования, которые в точности будет выполнять компьютер или контроллер. Для создания простой программы необходимо знать, что такое функции, переменные, операторы, команды ветвления и циклы.

Контроллеры могут управлять различными устройствами при помощи цифровых выходов, такими как светодиодами, моторами и сервоприводами и т.д. Указанные механизмы можно использовать для создания различных полезных вещей и роботов.

Контроллеры могут получать информацию о внешней среде и команды управления при помощи цифровых и аналоговых входов и подключенных к ним сенсоров. При помощи сенсоров поведение программы может изменяться, подстраиваться под пользователя прибора или внешнюю среду. Использование различных сенсоров позволяет создавать «разумные» автоматические устройства, которые становятся очень похожими на роботов.

Используя полученные во время исследования знания у меня получилось создать полезное устройство, которое управляется программой. Я понял, что такое программа и создал много программ-скетчей, которые обеспечивали работу автоматических устройств.

Список литературы

1. Саймон Монк. Программируем Arduino. Основы работы со скетчами. 2016.

2. Массимо Банци. Arduino для начинающих волшебников. 2012.

3. Бахметьев А.А. Играем и учимся. Электронный конструктор Знаток.

4. Формула расчёта скорости. Видеоуроки физики. (http://www.radostmoya.ru/project/akademiya_zanimatelnyh_nauk_fizika/video/?watch=skorost)