# Лабораторная работа №1

## Управление внешними устройствами с помощью микроконтроллера

**Цель работы**: ознакомиться с основными принципами программирования микроконтроллера (МК) на примере *Arduino*, решая задачи управления светодиодной индикацией и электродвигателем.

**Задачи:**

1. Реализовать аппаратное и программное подключение микроконтроллера (МК) *Arduino* к персональному компьютеру (ПК) для организации обмена данными.
2. Собрать схему подключения светодиодов к МК *Arduino*, написать программу управления ими от ПК.
3. Подключить коллекторный двигатель к МК, реализовать управление от ПК через экранный интерфейс скоростью вращения вала двигателя с помощью широтно-импульсной модуляции (ШИМ).

**Необходимое оборудование:**

1. Микроконтроллер *Arduino Uno*.
2. Персональный компьютер с USB портом.
3. Кабель USB с разъёмом TYPE-B.
4. Провода «папа-папа», 5 штук (+3 запасных).
5. Макетная плата Breadboard 400.
6. Светодиоды разноцветные L-1543S, d=5 мм, 3 штуки (+2 запасных).
7. Резистор 220 Ом, 0,5 Вт, 3 штуки (+2 запасных).
8. Электродвигатель постоянного тока FC-280SA-08600.
9. Драйвер двигателя L298N.

### 1. Микроконтроллер Arduino

*Arduino* – это открытая аппаратная платформа для макетирования электронных устройств, основанная на гибком и простом в использовании аппаратном и программном обеспечении; плата с собственным процессором и памятью для быстрой разработки электронных устройств.

Все платы *Arduino* содержат основные компоненты, необходимые для их программирования и совместной работы с другими схемами (рис. 1):

- микроконтроллер Atmel;
- USB-интерфейс для программирования и передачи данных;
- стабилизатор напряжения и выводы питания;
- контакты портов ввода-вывода;
- индикаторные светодиоды (Debug, Power, Rx, Tx);
- кнопку сброса;
- встроенный последовательный интерфейс программирования (ICSP).

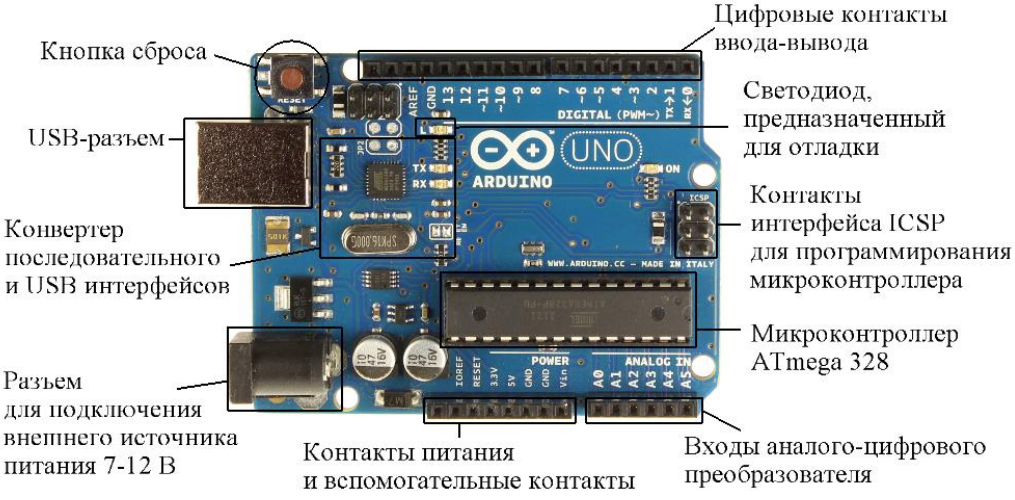



Рис. 1. Компоненты платы *Arduino Uno*

Основной элемент платы *Arduino* – микроконтроллер Atmel. На большинстве плат *Arduino*, включая *Arduino Uno*, установлен микроконтроллер ATmega. На плате *Arduino Uno* (рис. 1), микроконтроллер ATmega 328. Микроконтроллер исполняет весь скомпилированный код программы. Язык *Arduino* предоставляет доступ к периферийным устройствам микроконтроллера: аналого-цифровым преобразователям (ADCs), цифровым портам ввода-вывода, коммуникационным шинам (включая $I^2C$ и SPI) и последовательным интерфейсам. На плате все эти порты выведены на штырьковые контакты.

К тактовым контактам микроконтроллера ATmega подключён кварцевый резонатор на 16 МГц, но имеются версии МК *Arduino* с частотой до 84 МГц.

С помощью кнопки сброса выполнение программы можно перезапустить.

Большинство плат *Arduino* оснащено светодиодом отладки (Debug), подсоединённым к контакту 13, который позволяет реализовать первую программу (мигающий светодиод) без дополнительных компонентов.

Обычно программы микроконтроллера ATmega, написанные на C или Ассемблере, загружаются в микроконтроллер через интерфейс ICSP с помощью программатора. Но одной из важных особенностей *Arduino* является непосредственное программирование через USB-порт, без дополнительного программатора. Эту функцию обеспечивает загрузчик *Arduino*, записанный в микроконтроллер ATmega на заводе-изготовителе, который позволяет загружать пользовательскую программу на плату *Arduino* по последовательному порту USART. В *Arduino Uno* интерфейсом между кабелем USB и контактами USART на основном микроконтроллере служит дополнительный контроллер ATmega 16U2 (или 8U2 в зависимости от версии платы).

В случае *Arduino Uno* интерфейсом между кабелем USB и контактами USART на основном микроконтроллере служит дополнительный контроллер (ATmega 16U2 или 8U2 в зависимости от версии платы).

Загрузчик – это фрагмент программного кода, который записан в зарезервированное пространство памяти программы *Arduino*. Сразу после включения платы *Arduino* запускается загрузчик, который работает в течение нескольких секунд. Если за это время загрузчик получает команду программирования от *IDE* (Integrated Development Environment - интегрированная среда разработки) по последовательному интерфейсу UART, то он загружает программу в свободную область памяти микроконтроллера. Если такая команда не поступает, запускается последняя программа, находящаяся в памяти *Arduino*. При подаче команды загрузки от *Arduino IDE* вспомогательный контроллер сбрасывает основной микроконтроллер, подготавливая его к загрузке. Затем внешний компьютер начинает отправлять код программы, который основной МК получает через соединение UART.

Загрузчики занимают в памяти довольно много места, потому что они реализуют простое программирование через USB без внешних аппаратных средств и имеют два основных недостатка:

– занимают место в памяти (приблизительно 2 Кбайт), которое могло бы пригодиться при написании программ;

– при наличии загрузчика выполнение вашей программы всегда будет задерживаться на несколько секунд при начальной загрузке, поскольку загрузчик обрабатывает запрос на программирование.

С помощью внешнего программатора можно удалить загрузчик из контроллера ATmega и перепрограммировать контроллер.

У контроллеров *Arduino* все цифровые контакты могут служить как входами, так и выходами. Часть контактов *Arduino* может также действовать в качестве аналоговых входов. Многие из контактов работают в режиме мультиплексирования и имеют некоторые дополнительные функции: различные коммуникационные интерфейсы, последовательные интерфейсы, широтно-импульсные модуляторы и внешние прерывания.

Для большинства проектов достаточно 5-вольтового питания, получаемого по кабелю USB. Однако, схема способна работать от внешнего источника на 6-20 В (рекомендуется напряжение 7-12 В). Внешнее питание может подаваться через разъём питания или на контакт $V_{in}$.

У *Arduino* есть встроенные стабилизаторы на 5 и 3,3 В. Напряжение 5В используется для всех логических элементов на плате, уровень напряжения на цифровых контактах ввода-вывода находится в пределах 0-5 В. Напряжение 3,3В выведено на отдельный контакт для подключения внешних устройств.

Для удобства работы с *Arduino* существует бесплатная официальная среда программирования *Arduino IDE* (Arduino Integrated Development Environment, интегрированная среда разработки Arduino) работающая под Windows, Mac OS и Linux. Но возможна работа с Arduino и через другие среды разработки, например, XOD IDE (визуальная среда программирования), Visual Studio, Eclipse, командную строку.

Программное обеспечение *Arduino IDE* загружается в ПК с официального сайта https://www.arduino.cc.

После загрузки и установки можно подключать *Arduino* к компьютеру с помощью USB- кабеля.

После запуска программы, открывается окно, показанное на рисунке 2.

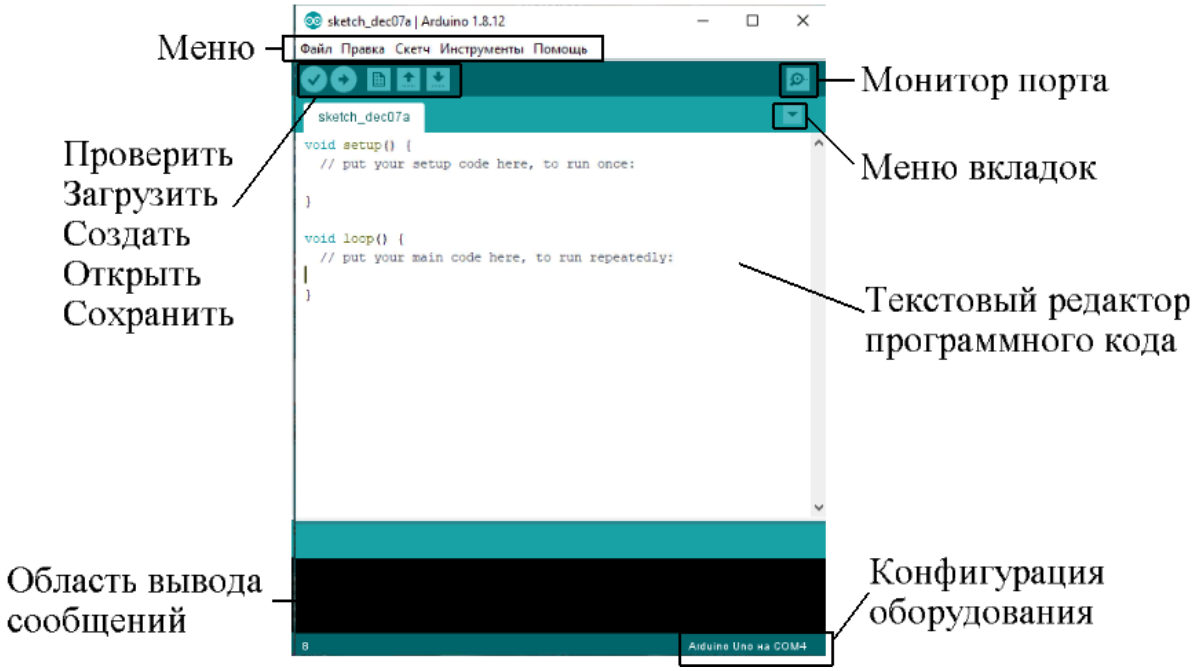



Рис. 2. Среда разработки *Arduino IDE*

Программы, написанные в *Arduino IDE*, имеют следующую базовую структуру:

```
void setup ()          // процедура инициализации, которая вызывается один раз
{                      после запуска программы (подачи питания или сброса
…                      питания платы). Обычно используется для установки
}                      режима портов (вход, выход) и других настроек.


void loop ()           // основной код программы, который выполняется
{                      бесконечное количество раз.
…
}
```

Список основных команд можно посмотреть на сайте http://arduino.ru/Reference.

В основном меню «Инструменты-Плата» следует выбрать плату, в нашем случае, *Arduino Uno*. При подключении *Arduino* к ПК создаётся виртуальный COM-порт. Выбрать данный порт нужно в меню «Инструменты-Порт».

## 2. Управление светодиодом через ПК.

### *2.1 Подключение светодиодов к микроконтроллеру.*

Светодиод – это полупроводниковый прибор (рис. 3). Свет возникает в специальном слое (p-n – переходе), когда через него проходит ток.


Рис. 3. Светодиод

*При подключении* светодиодов важно соблюдать два главных *правила*:

– у светодиода есть положительный (анод) и отрицательный (катод) контакты, поэтому важно соблюдать полярность при подключении, при неправильном подключении светодиод может выйти из строя;

– у светодиодов есть ограничения на максимальный протекающий через них ток, поэтому необходимо обеспечивать правильный режим работы.

Для ограничения тока через светодиод используется подключаемый последовательно со светодиодом резистор, номинал которого вычисляется по формуле:

$$R = \frac{U_{\text{пит}} - U_{\text{пад}}}{I \cdot 0{,}75},$$

где $R$ – сопротивление резистора в омах, $U_{\text{пит}}$ – напряжение источника питания в вольтах; $U_{\text{пад}}$ – прямое падение напряжения на светодиоде в вольтах (обычно в районе 2–2,5 вольт), при последовательном включении нескольких светодиодов величины падений напряжений складываются; $I$ – прямой ток светодиода в амперах (не должен превышать максимально допустимый ток согласно данных паспорта; обычно это 10–20 мА), при последовательном подключении нескольких светодиодов прямой ток не увеличивается; 0,75 – коэффициент надёжности для светодиода.

При использовании резистора, сопротивление которого больше расчётного значения, яркость светодиода будет ниже заявленной.

При использовании резистора сопротивление которого менее расчётного значения, срок жизни светодиода будет меньше.

Рассеиваемая мощность на резисторе – $P$, Ватт:

$$P = I^2 \cdot R.$$

При использовании резистора, мощность которого меньше расчётной, резистор выйдет из строя.

Вариант схемы подключения светодиода к *Arduino* показан на рисунке 4.

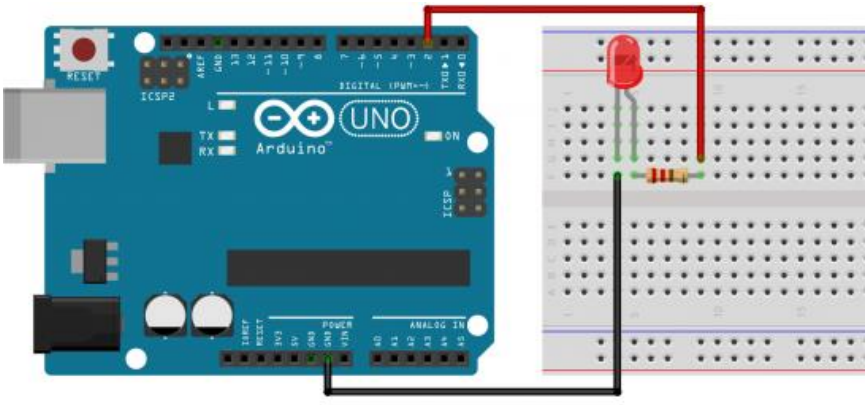



Рис. 4. Подключение светодиода к *Arduino Uno*

Программа для работы со светодиодом:

```
void setup()
{
pinMode(8, OUTPUT);          // задаём работу восьмого пина на выход
}
void loop()
{
digitalWrite(8, HIGH);       // подаём высокое напряжение (5В) на восьмой пин – зажигаем светодиод
delay(1000);                 // ждём 1 секунду
digitalWrite(8, LOW);        // подаём низкое напряжение (5В) на восьмой пина – выключаем светодиод
delay(1000);                 // ждём 1 секунду
}
```

### *2.2 Последовательность выполнения:*

1. Подключить 3 разноцветных светодиода к 3-м «пинам» МК так, как было описано выше.
2. Написать программу управления для ПК с экранным интерфейсом: изображения 3-х разноцветных светодиодов с кнопками под ними для управления их зажиганием/гашением посредством «мыши»: нажатие — «Вкл», нажатие — «Выкл»).
3. Подключить МК к ПК и добиться правильной работы программы. В отчёте описать как это было сделано.

## 3. Управление электродвигателем через ПК.

**Задача**: подключить электрический двигатель к МК, реализовать ШИМ-управление от ПК скоростью и направлением вращения вала двигателя.

### *3.1 Подключение электродвигателя к микроконтроллеру.*

Плата *Arduino Uno* не имеет настоящих аналоговых выходов, но она имеет несколько цифровых выходов, позволяющих использовать ШИМ (Pulse-Width Modulation, PWM).

Широтно-Импульсная Модуляция, или ШИМ – это операция получения изменяющегося аналогового значения сигнала посредством цифровых устройств. Устройства используются для генерирования прямоугольных импульсов с постоянной амплитудой и изменяющейся скважностью (отношение периода импульсов к длительности импульсов). Формируемые импульсы в данном случае имеют амплитуду 5В. С помощью МК выполняется управление частотой импульсов и их шириной (длительностью), т.е. скважностью.

На *Arduino Uno* (рис. 5) PWM-выводы помечены знаком тильда ("~").


Рис. 5. PWM выводы *Arduino*

Управление длительностью импульсов выполняется через управление состоянием данных цифровых выводов (логические «0» и «1») во времени (рис. 6).

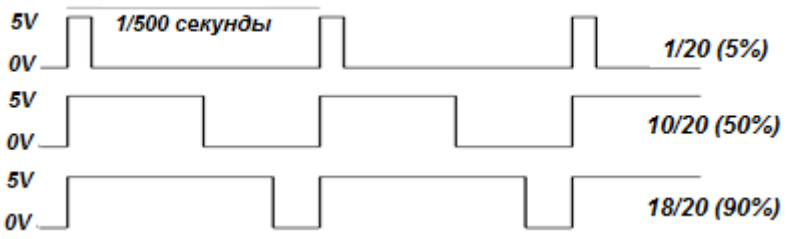



Рис. 6. Широтно-импульсная модуляция

Электродвигатель в данном случае выполняет роль низкочастотного фильтра – он «усредняет» подаваемое на него импульсное напряжение. В этом случае, чем больше длительность импульса при неизменной частоте импульсного напряжения, тем выше среднее напряжение на выходе. Так как импульсы следуют с частотой 500 раз в секунду (см. рис.6), а большинство подключаемых устройств не обладают мгновенной реакцией, возникает эффект плавного изменения напряжения. Изменение этого среднего значения напряжения приводит к изменению скорости вращения двигателя.

*Arduino* имеет существенные ограничения по силе тока присоединённой к ней нагрузки. Контакты ввода-вывода *Arduino* не способны выдавать ток более 40 мА, поэтому управлять двигателем напрямую нельзя. В момент запуска или остановки двигателя создаются пиковые броски тока, превышающие этот предел.

Для работы с двигателями существует Motor Shield – плата расширения для *Arduino*, которая обеспечивает работу двигателей постоянного тока и шаговых двигателей. Самыми популярными платами Motor Shield являются схемы на базе чипов L298N и L293D, которые могут управлять несколькими двигателями. На плате имеется возможность выбора источника напряжения – Motor Shield может питаться как от Arduino, так и от внешнего источника. Также на плате имеется светодиод, который показывает, работает ли устройство. Принцип работы драйвера двигателя основан на принципе работы Н-моста.

Схема подключения двигателя представлена на рисунке 7.

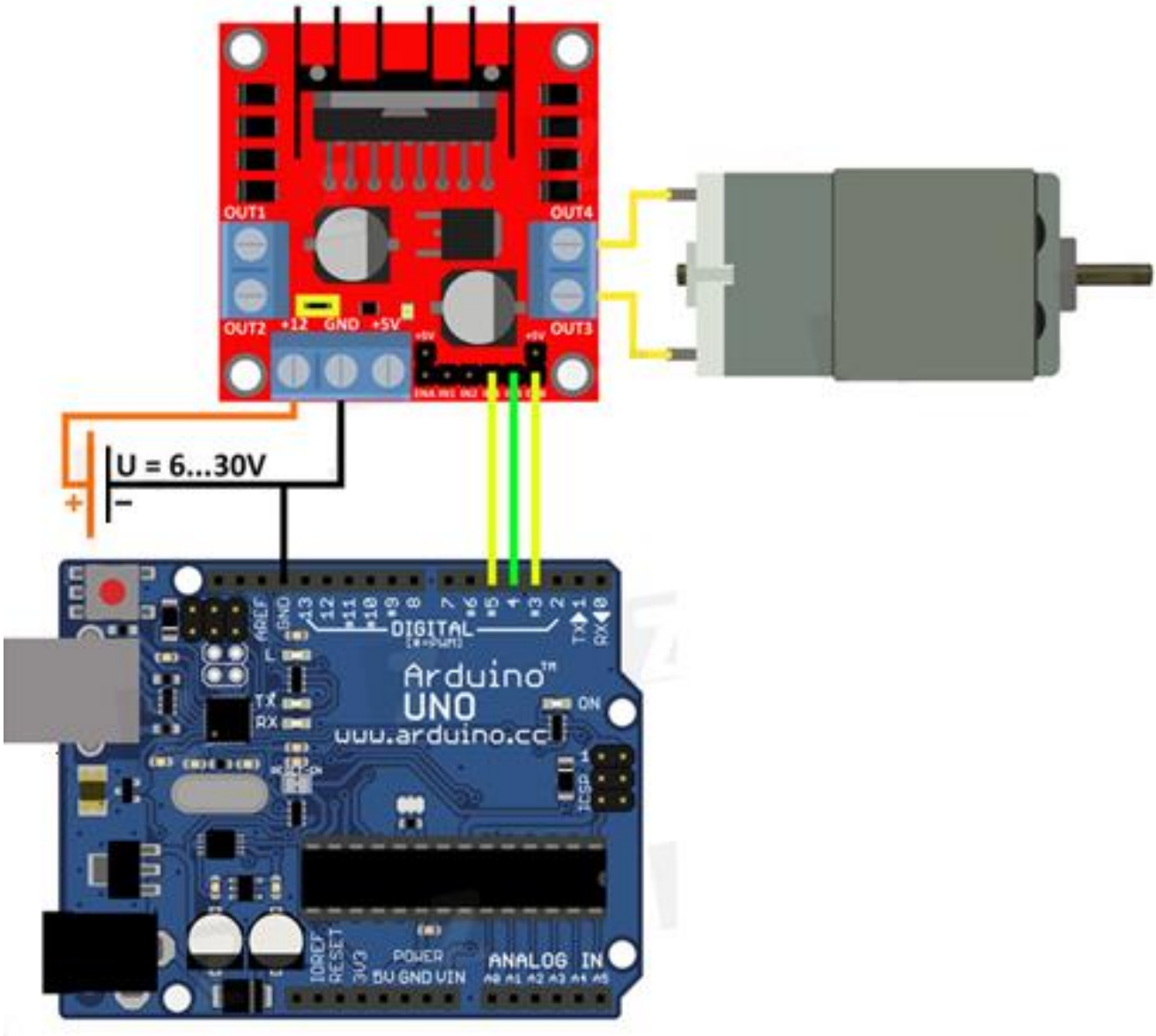



Рис. 7. Подключение мотора к Arduino через драйвер двигателя L298N

Программа для управления вращением вала двигателя:

```
int IN3 = 5;
int IN4 = 4;
int ENB = 3;
void setup()
{
  pinMode (ENB, OUTPUT);
  pinMode (IN3, OUTPUT);
  pinMode (IN4, OUTPUT);
}
void loop()
{
  digitalWrite (IN3, HIGH);
  digitalWrite (IN4, LOW);
  analogWrite (ENB, 55);
  delay (2000);
  analogWrite (ENB, 105);
  delay (2000);
  analogWrite (ENB, 255);
  delay (2000);
  analogWrite (ENB, 0);
  delay (5000);
}
```

Для управления внешними устройствами через поле ввода значения, встроенного в среду программирования *Arduino IDE* монитора последовательного интерфейса (Serial monitor) (рис. 8), используется библиотека *Serial*.

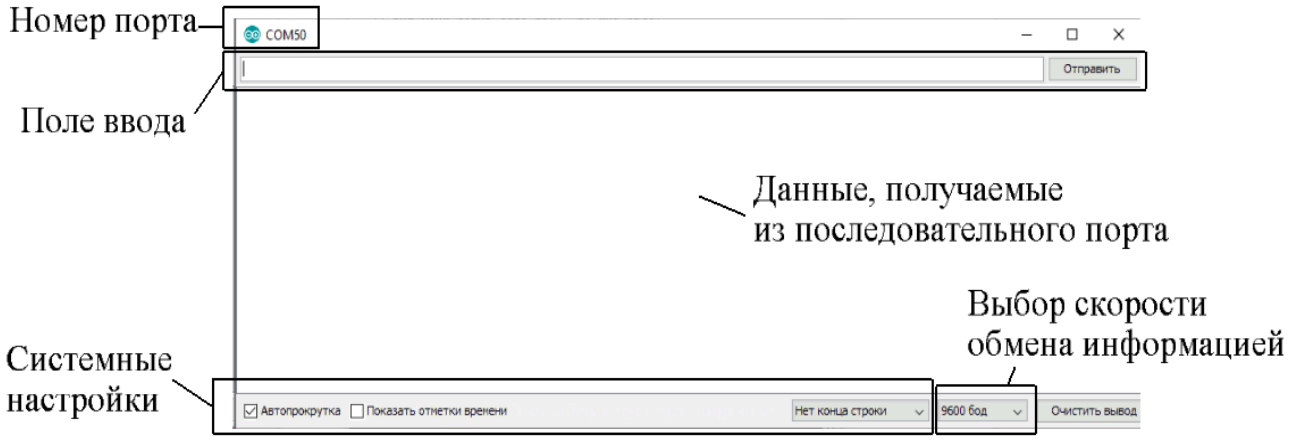



Рис. 8. Монитор последовательного порта *Arduino IDE*

Основные команды для работы с библиотекой *Serial*:

- Serial.begin(); – команда запускает последовательный порт, задаёт скорость передачи данных в бит/с (бод). Для обмена данными с компьютером используются следующие значения: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 или 115200.
- Serial.end(); – останавливает и очищает последовательный порт
- Serial.print(); – отправляет данные в последовательный порт
- Serial.println(); — отправляет данные с переносом строки
- Serial.read(); – принимает данные из последовательного порта
- Serial.parseInt(); – чтение чисел из монитора порта
- Serial.available(); – получает количество байт (символов) доступных для чтения из буфера последовательного интерфейса связи.

Программа для управления скоростью вращения вала двигателя:

```
int IN3 = 5;
int IN4 = 4;
int ENB = 3;
int i;
void setup()
{
Serial.begin(9600);
  pinMode (ENB, OUTPUT);
  pinMode (IN3, OUTPUT);
  pinMode (IN4, OUTPUT);
}
void loop()
{
  digitalWrite (IN3, HIGH);
  digitalWrite (IN4, LOW);
     if (Serial.available() > 0) // есть ли доступные данные
     {
     i=Serial.parseInt();
            analogWrite (ENB, i);
            delay (2000);
     }
 }
```

### *3.2 Задания:*

1. Подключить двигатель к МК, управлять через интерфейс Arduino IDE скоростью вращения вала двигателя с помощью ШИМ. Собрать схему из трёх светодиодов, написать программу для ПК (экранный интерфейс, обеспечивающий управление скоростью вращения вала электродвигателя кнопками клавиатуры или «мышью»), в которой при увеличении скорости вращения вала двигателя светодиоды загораются, а при уменьшении скорости вращения вала – светодиоды гаснут (обеспечить программное ограничение на максимальную скорость).
2. Подключить двигатель к МК, управлять скоростью вращения вала через интерфейс Arduino IDE. Собрать схему из четырёх светодиодов, написать программу для ПК (экранный интерфейс, обеспечивающий управление скоростью вращения вала электродвигателя кнопками клавиатуры или «мышью»), в которой с увеличением скорости вращения от 0 до максимума последовательно загораются 0 (при остановке двигателя) – все 4 светодиода (для максимальной скорости).
3. Подключить двигатель к МК, управлять направлением вращения вала через интерфейс Arduino IDE. Собрать схему из четырёх светодиодов, написать программу для ПК (экранный интерфейс, обеспечивающий управление скоростью вращения вала электродвигателя кнопками клавиатуры или «мышью»), в которой при вращении в одну сторону загорается первый и третий светодиод, при вращении в противоположную сторону – второй и четвёртый.
4. Подготовить отчёт.

# Лабораторная № 2

**Подключение цифровых и аналоговых датчиков к микропроцессору. Фильтрация шума.**

**Цель работы:**

а) исследование аналогового датчика на примере инфракрасного датчика (ИК-датчика) расстояния типа GP2D120 (Sharp);

б) исследование цифрового датчика на примере ультразвукового датчика (УЗ-датчика) расстояния типа HC-SR04.

**Задачи:**

1. Подключить к микроконтроллеру ИК-датчик расстояния типа GP2D120 (Sharp) и обеспечить отображение показаний датчика на экране ПК при изменении расстояния до объекта. Построить характеристику преобразования ИК-датчика расстояния в диапазоне расстояний, указанном в паспортных данных для плоского препятствия, ориентированного перпендикулярно ориентации датчика (идеальный случай), с использованием фильтрации шума по среднему арифметическому значению при заданном количестве измерений N. Найти аппроксимирующую функцию (степенная, экспоненциальная) и вычислить коэффициенты уравнения аппроксимирующей функции. Выполнить фильтрацию шума для заданного расстояния до объекта (для каждой подгруппы студентов задаётся разный набор расстояний) с использованием 5 фильтров: среднее арифметическое, среднее геометрическое, медианный фильтр, среднее гармоническое и фильтр срединной точки. Вычислить отношение сигнал/шум для 5 результатов фильтрации (при едином для всех фильтров окне фильтрации N) и выбрать оптимальный фильтр (максимум отношения сигнал/шум).
2. Подключить к микроконтроллеру УЗ-датчик расстояния типа HC-SR04 и обеспечить отображение показаний датчика на экране ПК при изменении расстояния до объекта. Построить характеристику преобразования УЗ-датчика расстояния в диапазоне расстояний, указанном в паспортных данных для плоского препятствия, ориентированного перпендикулярно ориентации датчика (идеальный случай), с использованием фильтрации шума по среднему арифметическому значению при заданном количестве измерений N. Выполнить фильтрацию шума для 7 значений расстояния до объекта (для каждой подгруппы задаётся разный набор расстояний) с использованием 5 фильтров: среднее арифметическое, среднее геометрическое, медианный фильтр, среднее гармоническое и фильтр срединной точки. Вычислить отношение сигнал/шум для 5 результатов фильтрации (при едином заданном окне фильтрации N) и выбрать оптимальный фильтр (максимум отношения сигнал/шум).

**Необходимое оборудование:**

1. Микроконтроллер Arduino Uno.
2. Персональный компьютер с USB портом.
3. Кабель USB с разъёмом TYPE-B.
4. Провода «мама-папа», 4 штуки (+2 запасных).
5. ИК-датчик расстояния типа Sharp GP2D120.
6. УЗ-датчик HC-SR04.
7. Металлическая линейка (рулетка) на 1000 мм.

Каждый датчик изначально вырабатывает аналоговый сигнал, пропорциональный входному воздействию. Этот сигнал может затем подаваться непосредственно на выходной контакт датчика или обрабатывается в самом датчике, чтобы создать выходной сигнал в виде напряжения, тока или кода, кодированной последовательности импульсов. Если датчик выполнен по интегральной технологии, то микропроцессор, входящий в состав датчика, обрабатывает аналоговый сигнал датчика, чтобы сформировать соответствующий двоичный или цифровой код.

*Аналоговый выход* – датчик выдаёт сигнал, содержащий данные, имеющие непрерывное множество возможных значений от времени.

На выходе аналогового датчика может быть:

- напряжение – самый распространённый тип аналогового выходного сигнала; другие формы аналогового сигнала можно также легко преобразовать в величину напряжения;

- сопротивление;
- ток.

## 1. Исследование ИК-датчиков расстояния типа GP2D120 (Sharp).

### *1.1. Принцип работы ИК-датчиков типа GP2D120 (Sharp).*

Датчик работает в зоне ближнего ИК-излучения – 850нм ± 70нм. Для определения расстояния до объекта используется метод триангуляции (рис. 1).

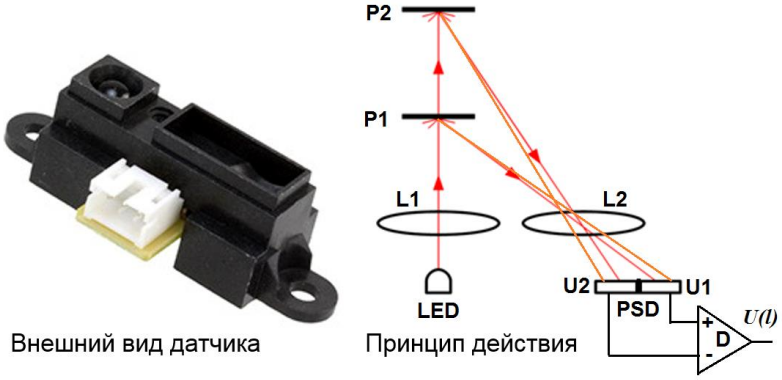



Рис. 1. ИК-датчик GP2D120 (Sharp)

Импульсы ИК-излучения (амплитудно-модулированный сигнал) испускаются излучателем **LED**. Излучение проходит через линзу **L1** и отражается от объектов, находящихся в поле зрения сенсора. Отражённые импульсы собираются линзой **L2** и передаются на позиционно-чувствительный фотоэлемент **PSD**, который поделён на две равные части **U1** и **U2**. Испускаемый и отражённый лучи образуют треугольник «излучатель - объект - приёмник». Угол отражения зависит от расстояния ***l*** до объекта.

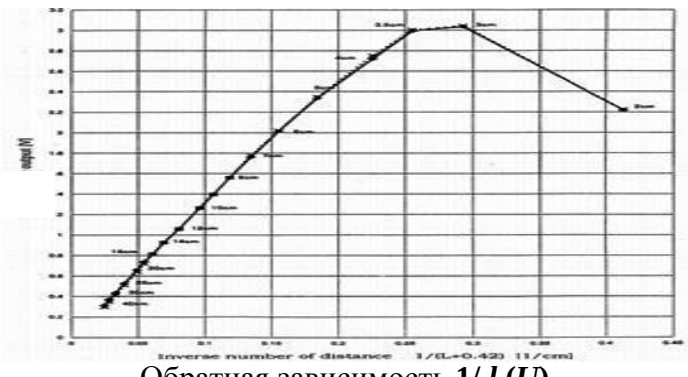
Отражённый луч (вследствие диффузии) на поверхности **PSD** образует световое пятно. Когда луч находится в центральной части, оба сегмента фотоэлемента будут освещены одинаково. В положении объекта **P1** отражённый свет будет падать на часть **U1 PSD**, в положении **P2** – на часть **U2**. Смещение луча света приводит к большей доле освещённости одного элемента по сравнению с другим. В результате на выходе дифференциального усилителя **D** возникает аналоговый электрический сигнал ***U*(*l*)**, пропорциональный углу отклонения светового луча от центральной части фотоэлемента.

Величина угла, в свою очередь, зависит от расстояния до объекта ***l***. Диапазон измерения перемещений для такого инструмента составляет несколько миллиметров. Такие приборы могут определять изменение смещения порядка 1мкм и обладают хорошей стабильностью.

На левом графике (рис. 2) приведена типовая зависимость напряжения на выходе датчика Sharp GP2D120 от расстояния до объекта. Эта характеристика даётся в техническом описании датчика (паспорте).

На правом графике (рис. 2) приведена обратная зависимость значения сигнала на выходе датчика от расстояния (**1/ *l*(*U*)**). Для определения расстояния до объекта можно использовать этот график, потому что на нём есть небольшой линейный участок.


Типовая зависимость ***U*(*l*)**.



Обратная зависимость **1/ *l* (*U*)**.

Рис. 2. Зависимость между выходным напряжением и расстоянием до цели

Более точно зависимость расстояния ***l*** от напряжения на выходе датчика ***U*** вычисляется с помощью аппроксимирующей функции:

$$l = a \cdot U^{-b} \quad . \tag{1}$$

ИК-датчики расстояния обладают разбросом параметров ***a*** и ***b***. Поэтому перед установкой в устройство необходимо выполнять их *калибровку*. В процессе калибровки датчика определяются параметры ***a*** и ***b***, зная которые можно вычислить расстояние ***l*** по формуле (1).

**Достоинства датчика**:

- Сенсор способен не только обнаруживать объект, но и с достаточно высокой точностью измерять расстояние до него.
- Сенсоры SHARP излучают амплитудно-модулированный ИК-сигнал; это позволяет практически полностью исключить влияние помех от внешнего освещения.
- Датчики показывают почти полное безразличие к цвету объекта обнаружения (датчик способен обнаруживать чёрные стены при солнечном свете).
- Высокая точность измерения расстояния.
- Компактность.

**Недостатки:**

- На малых расстояниях одному значению выходного напряжения соответствует два расстояния. Для предотвращения проблемы необходимо избегать слишком близкого расстояния до объектов (ближе, чем на 3 см).
- Функция преобразования дальности до объекта в выходное напряжение нелинейна.
- Сенсор не работает с прозрачными и светопоглощающими объектами.
- Зона чувствительности по углу не велика, так как при больших углах плоскости отражения относительно плоскости фотоприёмника отражённый луч «уходит» из зоны чувствительности.

### *1.2. Подключение ИК-датчиков типа GP2D120 (Sharp) к микропроцессору.*

Поскольку на выходе датчика формируется аналоговое напряжение, пропорциональное расстоянию до объекта, для его обработки цифровыми устройствами этот сигнал необходимо преобразовать в цифровую форму. Для этого используется аналого-цифровой преобразователь микроконтроллера Arduino UNO. Схема подключения ИК-датчика расстояния Sharp GP2D120 к микроконтроллеру Arduino UNO приведена на рисунке 3.

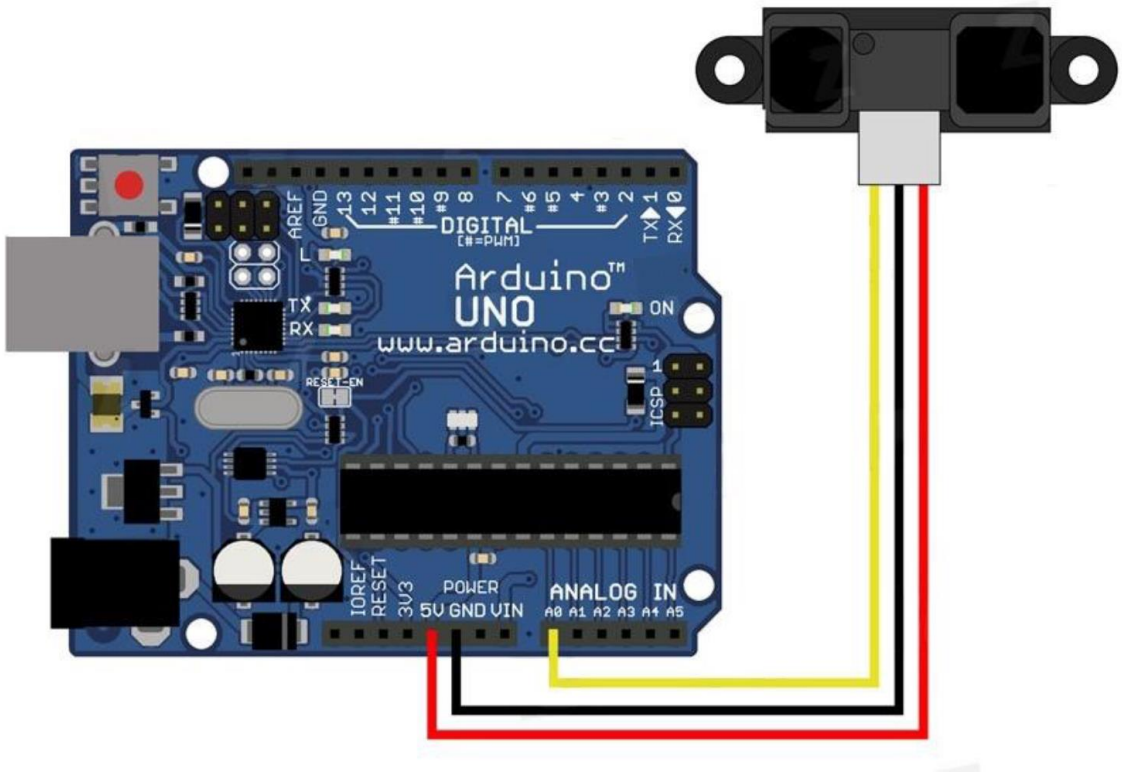



Рис. 3. Схема подключения ИК-датчика

Программа для получения показаний ИК-датчика:

```
float IRpin = A0;                                   // аналоговый вход для подключения выхода Vo сенсора

void setup()
{
Serial.begin(9600);                                 // запуск последовательного порта
}
void loop()
{
// 5V/1024 = 0.0048828125                           // считываем значение сенсора и
float volts = analogRead(IRpin)*0.0048828125;       переводим в напряжение в диапазоне 0-5В
Serial.println(volts);                              // выдаём значение в порт
delay(50);
}
```

### *1.3. Задание 1. Калибровка датчика.*

**Цель**: построить зависимость показаний (в цифровой форме в диапазоне 10-разрядного АЦП микропроцессора) на выходе датчика Sharp GP2D120 от расстояния до объекта. Определить параметры $a$ и $b$ аппроксимирующей функции (1).

Для калибровки сенсора показания ИК-датчика считываются в определённом диапазоне расстояний до объекта с некоторым шагом. По полученным данным, например, в Microsoft Excel, строится график зависимости напряжения на выходе от расстояния до объекта. По данному графику находится аппроксимирующая функция (в Excel – линия тренда), формула которой и есть формула для вычисления расстояния до объекта в зависимости от выдаваемых ИК-датчиком напряжений. Подставив данную формулу в программу можно начать измерения.

**Постановка эксперимента:**

1. Прочно закрепить датчик на столе.
2. На расстояниях $l$ от 20 мм до 500 мм с шагом 20 мм (рис. 4) установить объект (устойчиво стоящий объект с ***плоскими гранями***, например, деревянный кубик или картонную коробку).

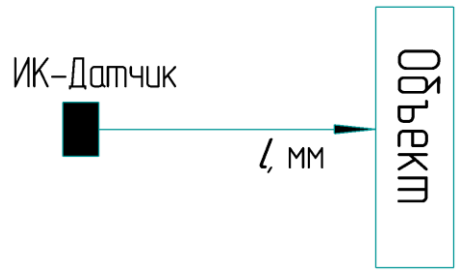



Рис. 4. Схема эксперимента

3. Для каждой позиции $l$ выполнять по N=100 замеров $a_k$ расстояния. Вычислить математическое ожидание $M_l$:

$$M_l = \frac{\sum a_k}{N}, \tag{2}$$

где $a_k$ – показание датчика (значение напряжения, полученное с помощью микропроцессора для $k = 1,2,\dots,N$);

$N$ – количество отсчётов в одном замере.

Вычислить дисперсию $D_l$:

$$D_l = \frac{\sum(a_k - M_l)^2}{N-1}. \tag{3}$$

4. Заполнить таблицу 1 (не вручную, выполнить программно на ПК).

Таблица 1. Значения ИК-датчика

| $l$, мм | $M_l$ | $D_l$ |
|---|---|---|
| **20** | | |
| **40** | | |
| **60** | | |
| **…** | | |
| **500** | | |

5. Построить в Excel график зависимости $U(l)$, где $U$ – это среднее значение полученного замера ИК-датчика (значения $M_l$). Для построения графика необходимо выполнить Вставка-Диаграммы-Вставить точечную (X,Y) или пузырьковую диаграмму-Точечная с гладкими кривыми и маркерами, выбрать области данных.

6. Аппроксимировать полученную зависимость с помощью линии тренда: щёлкнуть по диаграмме, Конструктор-Добавить элемент диаграммы-Линия тренда-Дополнительные параметры линии тренда. Поставить галочку на «показывать уравнение на диаграмме». Путём перебора выбрать вид кривой, наиболее близко прилегающей к графику. Полученное уравнение аппроксимирующей функции вида $l = a \cdot U^{-b}$ позволит рассчитывать расстояние $l$ в зависимости от полученного с датчика значения $U$ на всём диапазоне изменения выходного напряжения. Это существенно повысит точность измерения расстояния.

### *1.4. Задание 2. Фильтрация шума.*

**Цель**: определить оптимальный фильтр по критерию отношения сигнал/шум.

Измерения, производимые реальными датчиками, всегда имеют некоторый разброс значений, присутствует погрешность измерения – **шум**. Для сглаживания шумовой составляющей используются фильтры. Ниже перечислены пять основных фильтров:

1. Фильтр, основанный на вычислении среднего арифметического:

$$f_1 = \frac{1}{N}\sum_{k=1}^{N} a_k. \quad (4)$$

2. Фильтр, основанный на вычислении среднего геометрического:

$$f_2 = \left[\prod_{k=1}^{N} a_k\right]^{\frac{1}{N}}. \quad (5)$$

3. Медианный фильтр:

$$f_3 = med\{a_1, \dots, a_N\}. \quad (6)$$

4. Фильтр, основанный на вычислении среднего гармонического:

$$f_4 = \frac{N}{\sum_{k=1}^{N}\frac{1}{a_k}}. \quad (7)$$

5. Фильтр срединной точки:

$$f_5 = \frac{1}{2}[\max\{a_1 \dots a_N\} + \min\{a_1 \dots a_N\}]. \quad (8)$$

В приведённых формулах $a_k$ – значение входного сигнала для $k$-го отсчёта в окне фильтрации размером $N$ ($k$ = 1, 2, …, $n$).

**Постановка эксперимента:**

1. Согласно выполненной калибровке ИК-датчика (Задание 1) получить параметры $a$ и $b$ аппроксимирующей функции.

2. Получить по $N$=100 замеров на следующих расстояниях до объекта: $L$ = 50, 100, 150, 200, 250мм (для расчёта расстояния использовать уравнение аппроксимирующей функции).

3. Вычислить $f_m$ ($m$=1, 2, … 5) каждого фильтра для каждого расстояния $L$ и сравнить с показаниями линейки.

4. Вычислить абсолютную погрешность измерений $\Delta y_L$ для каждого из 4-х значений $L$:

$$\Delta y_L = |f_m - L|, \quad (9)$$

где $f_m$ – значение расстояния, полученное в результате фильтрации.

5. Вычислить относительную погрешность измерений:

$$\delta_L = \frac{\Delta y_L}{f_m}. \quad (10)$$

6. Вычислить дисперсию:

$$D_L = \frac{\sum(a_k - f_m)^2}{N-1}. \quad (11)$$

7. Вычислить среднеквадратичное отклонение:

$$\sigma_L = \sqrt{D_L}. \quad (12)$$

8. Вычислить отношение сигнал/шум (ОСШ):

$$\text{ОСШ}_L = \frac{f_m}{\sigma_L}. \quad (13)$$

9. Заполнить таблицу 2.

Таблица 2. Экспериментальные показания

| | $L$, мм (линейка) | $L_f$, мм (датчик) | $\Delta y_L$, мм | δ | $D_L$ | $\sigma_L$ | $\text{ОСШ}_L$ |
|---|---|---|---|---|---|---|---|
| $f_1$ | 50 | | | | | | |
| | 100 | | | | | | |
| | 150 | | | | | | |
| | 200 | | | | | | |
| | 250 | | | | | | |
| $f_2$ | 50 | | | | | | |
| | 100 | | | | | | |
| | 150 | | | | | | |
| | 200 | | | | | | |
| | 250 | | | | | | |
| $f_3$ | 50 | | | | | | |
| | 100 | | | | | | |
| | 150 | | | | | | |
| | 200 | | | | | | |
| | 250 | | | | | | |
| $f_4$ | 50 | | | | | | |
| | 100 | | | | | | |
| | 150 | | | | | | |
| | 200 | | | | | | |
| | 250 | | | | | | |
| $f_5$ | 50 | | | | | | |
| | 100 | | | | | | |
| | 150 | | | | | | |
| | 200 | | | | | | |
| | 250 | | | | | | |

**Вывод**: Оптимальным по значению отношения сигнал/шум является фильтр _____

## 2. Исследование ультразвукового датчика расстояния HC-SR04.

### *2.1 Принцип работы датчика HC-SR04.*

Работа УЗ-датчика основана на пьезоэлектрическом эффекте. Существует прямой и обратный пьезоэлектрический эффект.

Сущность *прямого эффекта* заключается в следующем. Под механическим воздействием на пьезоэлектрическую пластину (кристалл) происходит её деформация, в результате чего на её поверхности образуются электростатические заряды, а внутри – электрическая поляризация. В результате между гранями кристалла возникает разность потенциалов – напряжение, которое можно «снимать» с металлизированного покрытия граней.

*Обратный пьезоэлектрический* эффект состоит в том, что при приложении к пьезоэлектрической пластине переменного электрического напряжения она начинает изменять геометрические размеры с частотой прикладываемого напряжения, т.е. при воздействии электрического поля на элементарные заряды в ячейках кристалла, происходит изменение средних расстояний между ними; в результате возникает деформация всего кристалла.

Прямой пьезоэлектрический эффект применяется для приёма ультразвуковых колебаний, обратный – для излучения УЗ-волн.

**Факторы, влияющие на показания ультразвуковых датчиков:**

– изменение скорости звука в зависимости от температуры и свойств окружающей среды (в основном, воздуха); в некоторых УЗ-датчиках расстояния установлен датчик температуры, с помощью которого выполняется компенсация влияния изменения температуры окружающей среды;

– уровень внешних шумов, возникающих в результате отражений УЗ-волны от различных объектов сцены;

– изменения амплитуды отражённого эха в зависимости от расстояния до объекта, размеров и геометрии поверхности.

**Особенно хорошо отражают звуковую волну следующие объекты:**

– все гладкие и твёрдые объекты, расположенные перпендикулярно направлению звуковой волны;

– все расположенные случайным образом твёрдые объекты с шероховатой поверхностью, вызывающей рассеянное отражение.

**Следующие материалы плохо отражают звуковую волну:**

– материалы, поглощающие ультразвуковые сигналы, например, фетр, хлопок, грубые ткани и пеноматериалы, шерсть животных;

– материалы с температурой более 100 °C.

На дальность распространения сигнала влияет затухание. *Затухание* – это некоторая интегральная характеристика, которая определяет потерю мощности колебаний в среде, выражающуюся в ослабления амплитуды и интенсивности сигнала.

Затухание УЗ-сигнала вызывается:

– уменьшением амплитуды УЗ-сигнала с увеличением расстояния от излучателя до объекта, обусловленное формой и волновыми размерами источника; это связано с тем, что по мере распространения волны от точечного или сферического источника, энергия, излучаемая источником, распределяется на все увеличивающуюся поверхность волнового фронта и, соответственно, уменьшается поток энергии через единицу поверхности, т.е. интенсивность звука;

– поглощением, т.е. необратимым переходом энергии звуковой волны в другие формы, в частности в тепло; поглощение может быть обусловлено различными механизмами, большую роль в которых играет вязкость и теплопроводность среды, взаимодействие волны с различными молекулярными процессами вещества, с тепловыми колебаниями кристаллической решётки и др.;

– рассеванием на неоднородностях среды, в результате чего уменьшается поток энергии в первоначальном направлении распространения; рассеивание происходит из-за резкого изменения свойств среды – её плотности и модулей упругости – на границе неоднородностей, размеры которых сравнимы с длиной волны (в газах это могут быть, например, жидкие капли, в

водной среде – пузырьки воздуха, в твёрдых телах – различные инородные включения или отдельные кристаллиты в поликристаллах и т.п.).

Технические характеристики УЗ-датчика HC-SR04 (рис. 5):

-напряжение питания 5В;
-диапазон расстояний: 2-400 см;
-потребление в режиме тишины 2 мА;
-потребление при работе: 15 мА;
-эффективный угол наблюдения: $15^{O}$;
-рабочий угол наблюдения: $30^{O}$.


TRIG – триггер: пин, на который подаётся запускающий импульс.
ECHO – эхо: пин, с которого считывается выходной сигнал – импульс, длительность которого пропорциональна измеренному расстоянию.
VCC – питание, +5В.
GND – заземление.

Рис. 5. Выводы УЗ-датчика

Принцип работы УЗ-датчика расстояния типа HC-SR04 следующий (рис. 6).

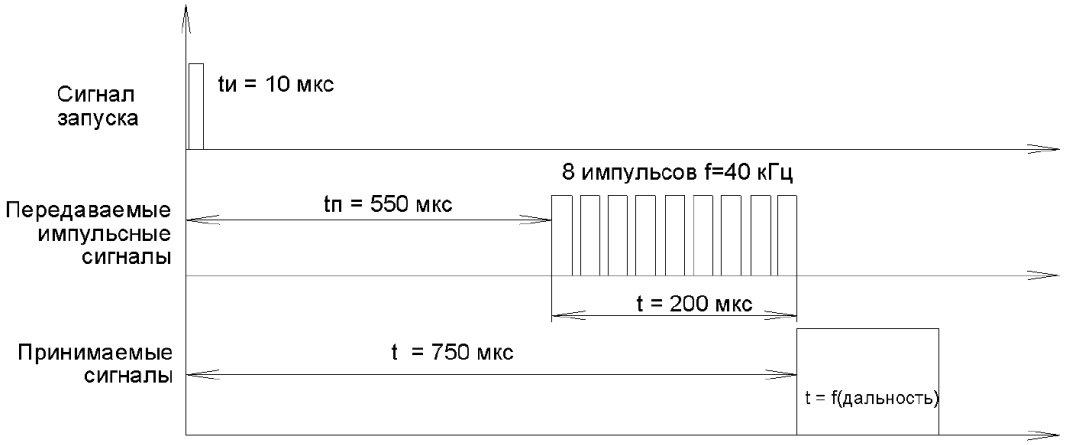



Рис. 6. Временная диаграмма УЗ-датчика

На вход (Trigger Input) подаётся запускающий импульс ($t_и$ = 10÷15мкс). Для возбуждения УЗ излучателя (пьезоэлемента) микроконтроллер, входящий в состав датчика, с некоторой задержкой относительно запускающего импульса формирует пакет из 8 импульсов (Ultrasonic burst) с амплитудой ±5В и резонансной (для пьезоэлемента) частотой 40 кГц. Пакет из 8 импульсов необходим для увеличения «мощности» излучения (аналогия с качелями – несколько раз толкнуть для увеличения амплитуды колебаний). Затем происходит приём отражённого от объекта сигнала.

Приём отражённой звуковой волны выполняется другим пьезоэлементом, который имеет такую же резонансную частоту, как и излучающий пьезоэлемент. Микроконтроллер измеряет время задержки от передачи до приёма звуковой волны, и на выходе (Echo Output) датчика формирует импульс, длительность которого ($t_u \leq 24$ мс), пропорциональна измеренному расстоянию.

Расстояние *S*, в сантиметрах, до препятствия вычисляется по формуле:

$$S = v \cdot t; t = \frac{t_u}{2} \rightarrow S = \frac{v \cdot t_u}{2}, \qquad (14)$$

где $t_u$ – длительность сформированного импульса (мксек),

$v$ – скорость звука в воздушной среде при температуре $20^{O}$C ($\approx 343$ м/сек, $\approx 0{,}0343$ см/мксек).

Деление на два необходимо из-за того, что сигнал проходит расстояние до объекта и обратно, тогда как требуется только расстояние до объекта. Следовательно, окончательная форма будет выглядеть следующим образом:

$$S = \frac{v \cdot t_u}{2} = \frac{0{,}0343 \cdot t_u}{2} = t_u \cdot \frac{0{,}0343}{2} = t_u / \left(\frac{2}{0{,}0343}\right) \approx t_u/58.$$

Следующий замер может быть выполнен только после исчезновения эха от предыдущего. Это время называется периодом цикла (cycle period). Рекомендованный период между запускающими импульсами не менее 30 мс.

### *2.2 Подключение УЗ-датчика HC-SR04 к микропроцессору.*

Схема подключения УЗ-датчика показан на рисунке 7. Поскольку на входе и выходе сенсора формируется импульс с ТТЛ-уровнями (логическая «1» – импульс амплитудой < 5В), то запускающий импульс подаётся с одного из цифровых выходов МК, а выходной – подаётся на один из цифровых входов МК.

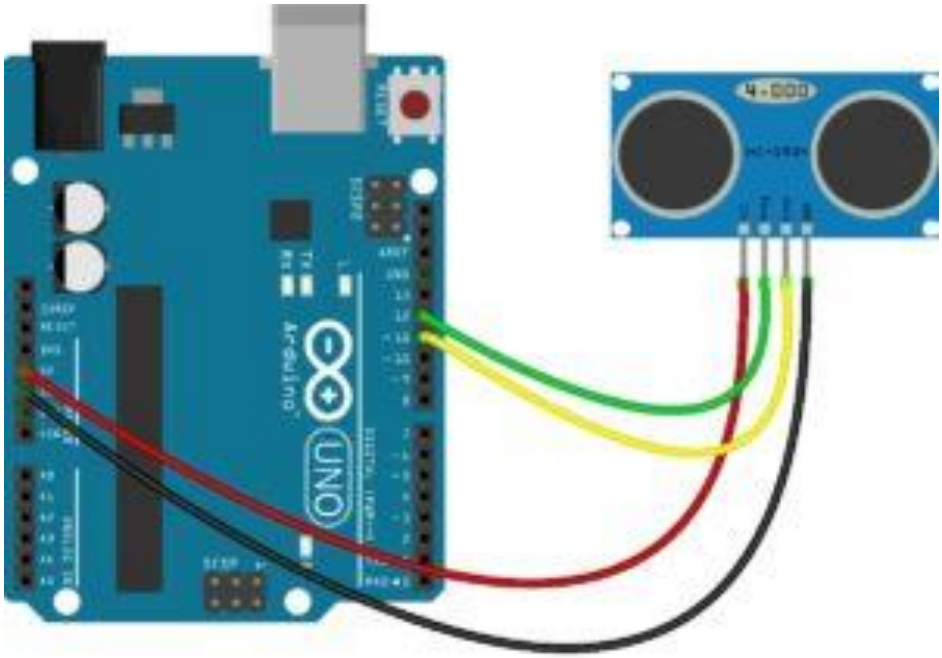


Рис. 7. Схема подключения УЗ-датчика

Программа для получения показаний УЗ-датчика:

```
int echoPin = 8;
int trigPin = 9;
void setup()
{
  Serial.begin (9600);                  // инициализация последовательного порта
  pinMode(trigPin, OUTPUT);             // задаёт пин (триггер) на выход
  pinMode(echoPin, INPUT);              // задаёт пин (эхо) на вход
}
void loop()
{
int duration, cm;
digitalWrite(trigPin, LOW);             // подаётся низкий сигнал в течение 2-х
delayMicroseconds(2);                   микросекунд
digitalWrite(trigPin, HIGH);            // подаётся высокий сигнал в течение 10-ти
delayMicroseconds(10);                  микросекунд
digitalWrite(trigPin, LOW);             // опять подаётся низкий сигнал
duration = pulseIn(echoPin, HIGH);      // считывается длина принятого импульса
cm = duration / 58;                     // определяется дистанция в сантиметрах
```

```
Serial.println(cm);                    // выводятся данные в последовательный порт
delay (50);                            // задержка между импульсами
}
```

УЗ-датчик HC-SR04 не требует калибровки.

### *2.3 Задание. Фильтрация шума.*

**Цель:** определить оптимальный фильтр по критерию отношения сигнал/шум.

**Постановка эксперимента:**

1. Получить по N=100 замеров на следующих расстояниях до объекта: $L$ =2, 4, 8, 16, 32, 64, 128 см.
2. Вычислить $f_m$ ($m$=1, 2, … 5) каждого фильтра (по формулам 4-8) для каждого расстояния $L$ и сравнить с показаниями линейки.
3. Вычислить абсолютную погрешность измерений $\Delta y_L$ для каждого из 7 значений $L$ по формуле 9.
4. Вычислить относительную погрешность измерений $\delta_L$ по формуле 10.
5. Вычислить дисперсию $D_L$ по формуле 11.
6. Вычислить среднеквадратичное отклонение $\sigma_L$ по формуле 12.
7. Вычислить отношение сигнал/шум $\text{ОСШ}_L$ по формуле 13.
8. Заполнить таблицу 3.

Таблица 3. Экспериментальные показания

| | $L$, см (линейка) | $L_f$, см (датчик) | $\Delta y_L$, см | $\delta_L$ | $D_L$ | $\sigma_L$ | $\text{ОСШ}_L$ |
|---|---|---|---|---|---|---|---|
| $f_1$ | 2 | | | | | | |
| | 4 | | | | | | |
| | 8 | | | | | | |
| | 16 | | | | | | |
| | 32 | | | | | | |
| | 64 | | | | | | |
| | 128 | | | | | | |
| $f_2$ | 2 | | | | | | |
| | 4 | | | | | | |
| | 8 | | | | | | |
| | 16 | | | | | | |
| | 32 | | | | | | |
| | 64 | | | | | | |
| | 128 | | | | | | |
| $f_3$ | 2 | | | | | | |
| | 4 | | | | | | |
| | 8 | | | | | | |
| | 16 | | | | | | |
| | 32 | | | | | | |
| | 64 | | | | | | |
| | 128 | | | | | | |
| $f_4$ | 2 | | | | | | |
| | 4 | | | | | | |
| | 8 | | | | | | |
| | 16 | | | | | | |
| | 32 | | | | | | |
| | 64 | | | | | | |
| | 128 | | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $f_5$ | **2** | | | | | | |
| | **4** | | | | | | |
| | **8** | | | | | | |
| | **16** | | | | | | |
| | **32** | | | | | | |
| | **64** | | | | | | |
| | **128** | | | | | | |

**Вывод**: Оптимальным по значению отношения сигнал/шум является фильтр _____

**Задание:**

Построить функцию преобразования УЗ-датчика расстояния в диапазоне расстояний, указанном в паспортных данных для плоского препятствия, ориентированного перпендикулярно ориентации датчика.

## 3. Отчёт

По каждой лабораторной работе студентами формируется **отчёт**, *который должен содержать*:

1. Титульный лист с указанием организации (МГТУ «СТАНКИН»), кафедры, названия учебного курса, темы лабораторной работы, авторов, даты проведения лабораторной работы.
2. Описать цели экспериментов; перечислить используемое оборудование; приложить схемы, фотографии экспериментальных установок; блок-схемы алгоритмов управления двигателем, светодиодами; программный код, графики.
3. Провести необходимые расчёты по формулам, заполнить таблицы, сделать графики.
4. Описать полученные результаты, сделать выводы.
5. Привести список использованных источников.

***Список рекомендованной литературы.***


1. Arduino Uno:
   1.1. http://arduino.ru/Hardware/ArduinoBoardUno (10.12.2020),
   1.2. https://alexgyver.ru/lessons/serial/ (10.12.2020).
2. Светодиоды:
   2.1. https://svetodiode.blogspot.com/2012/01/blog-post_19.html (10.12.2020),
   2.2. http://wiki.amperka.ru/%D0%BA%D0%BE%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%82-arduino:%D1%81%D0%B2%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D0%BE%D0%B4 (10.12.2020),
   2.3. https://xn--18-6kcdusowgbt1a4b.xn--p1ai/%D1%81%D0%B2%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D0%BE%D0%B4-%D0%B0%D1%80%D0%B4%D1%83%D0%B8%D0%BD%D0%BE/ (10.12.2020).
3. Двигатели:
   3.1. https://arduino-diy.com/arduino-dvigateli (10.12.2020),
   3.2. https://3d-diy.ru/wiki/arduino-moduli/drayver-dvigatelya-l298n/ (10.12.2020),
   3.3. http://arduino.ru/Tutorial/PWM (10.12.2020).
4. ИК-датчик:
   4.1. http://robocraft.ru/blog/electronics/783.html (10.12.2020),
   4.2. https://3d-diy.ru/wiki/arduino-datchiki/infrakrasnyj-datchik-rasstojanija/ (10.12.2020),
   4.3. http://zelectro.cc/SHARP_GP2Y0A02YK0F/ (10.12.2020).
5. УЗ-датчик:
   5.1. https://xn--18-6kcdusowgbt1a4b.xn--p1ai/%D1%83%D0%BB%D1%8C%D1%82%D1%80%D0%B0%D0%B7%D0%B2%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B9-

%D0%B4%D0%B0%D1%82%D1%87%D0%B8%D0%BA-%D0%B0%D1%80%D0%B4%D1%83%D0%B8%D0%BD%D0%BE/ (10.12.2020),
5.2. https://arduino-diy.com/arduino-ultrazvukovoy-datchik-rasstoyaniya (10.12.2020),
5.3. https://helpduino.ru/podklychenie_dalnometra_%20HC-SR04.html (10.12.2020).
6. Джереми Блум, «Изучаем Arduino. Инструменты и методы технического волшебства», Санкт-Петербург, Наука, 2013.
7. Виктор Петин, «Проекты с использованием контроллера Arduino», Санкт-Петербург, БВХ-Петербург, 2014
8. Саймон Монк, «Программируем Arduino», Санкт-Петербург, Питер, 2017
9. Даташит на ИК-датчик: https://www.pololu.com/file/0J157/GP2D120-DATA-SHEET.pdf (6.12.2020).
10. Даташит на УЗ-датчик: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf (6.12.2020).
11. Андреев В.П., Ким В.Л., Плетенев П.Ф., Тарасова В.Э. и др. Проектно-исследовательская деятельность учащихся в области интеллектуальной роботроники с использованием модульных робототехнических систем. Учебно-методическое пособие для организации дополнительного образования. Курс 3. Интеллектуальная модульная роботроника: http://proxy.uniar.ru/rggu/ .

Преподаватель:
д.т.н. Андреев В.П.