

Документ подписан простой электронной подписью

Информация о владельце:

ФИО: Локтионова Оксана Геннадьевна

Должность: проректор по учебной работе

Дата подписания: 19.08.2017 12:51:29

Уникальный программный ключ:

0b817ca911e6668abb13a5d426d39e5f1c10abb73e943df4a4851fda56d089

Кафедра космического приборостроения и средств связи

УТВЕРЖДАЮ

Проректор

« 15 »



**ВВЕДЕНИЕ В ПРОГРАММИРОВАНИЕ
В СРЕДЕ IDE ARDUINO**

Методические указания по выполнению лабораторных работ
для студентов направления подготовки 11.03.02, 11.03.03

Курек 2017

УДК 681.3

Составитель В.Н. Усенков

Рецензент

Кандидат технических наук, профессор *В.А. Шлыков*

Введение в программирование в среде IDE Arduino: методические указания по выполнению лабораторных работ / Юго-Зап. гос. ун-т; сост.: В.Н. Усенков. - Курск, 2017. - 44 с.: ил. 12, прилож. 5. - Библиогр.: с. 35.

Приводятся краткие сведения о платформе Arduino. Описывается среда программирования IDE Arduino. Приводятся методические указания по проектированию программ на языке среды Arduino, ориентированных для применения в системах, построенных на базе микро-ЭВМ.

Указывается порядок выполнения лабораторных работ. Приводятся рекомендации по оформлению отчетов и контрольные вопросы.

Предназначены для студентов специальностей 11.03.02, 11.03.03 дневной и заочной форм обучения.

Текст печатается в авторской редакции

Подписано в печать . Формат 60×84 1/16.
Усл. печ. л. 2,56. Уч.-изд. л. 2,32. Тираж 100 экз. Заказ. Бесплатно.

Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Содержание

Введение.....	4
1 Развертывание системы программирования IDE Arduino..	5
2 Изучение среды разработки и отладки.....	11
3 Проектирование программ управления объектами с цифровым интерфейсом.....	16
4 Проектирование программ управления объектами с аналоговым интерфейсом.....	21
5 Проектирование программ управления объектами с интерфейсом 1-wire.....	26
6 Проектирование программ управления объектами с интерфейсом I2C.....	31
Литература.....	35
Приложение А - Макетирование устройств при подключении к Arduino.....	36
Приложение Б - стандартные модули для учебных макетов...	38
Приложение В - цифровой датчик температуры с интерфейсом 1-wire.....	40
Приложение Г – цифровые часы с интерфейсом I2C.....	41
Приложение Д – обзор конструкций языка IDE Arduino.....	42

Введение

Платформа Arduino позиционируется как программно-аппаратная система начального обучения программированию микро-ЭВМ. Базовой моделью микро-ЭВМ изначально являлась микросхема ATmega8, однако впоследствии были использованы улучшенные модификации ATmega168 и ATmega328. В настоящее время обозначилась тенденция внедрения в среду разработки IDE Arduino микро-ЭВМ иной архитектуры, в том числе на базе ARM и Intel x86.

Простые программные проекты для встраиваемых систем на базе микроЭВМ пишутся, как правило, на языке ассемблера. Для более сложных проектов более адекватным представляется использование трансляторов с языков высокого уровня, причем в последнее время предпочтение чаще отдается языку C.

Язык C достаточно сложен для освоения, поэтому для платформы Arduino была разработана его упрощенная версия [1], [4]. Программы при этом разрабатываются в среде программирования IDE Arduino, которая пригодна для развертывания в операционных системах Windows и Linux.

При изучении среды программирования Arduino используется IDE Arduino версии 1.0 или более поздней.

Назначением приведенных лабораторных работ являются:

- изучение средств разработки и отладки программного обеспечения для микропроцессорных систем;
- формирование навыков работы в среде программирования Arduino.

1 Развертывание системы программирования IDE Arduino

Цель работы

Подготовка к дальнейшему выполнению цикла лабораторных работ в среде IDE Arduino и приобретение практических навыков в подготовке типичной рабочей среды проектирования программ для микро-ЭВМ для операционной системы Windows.

Подготовка к работе

- изучить основные аппаратные особенности платформы Arduino;
- ознакомиться с инструкцией по подключению платы Arduino к персональному компьютеру;
- уточнить место расположения дистрибутива IDE Arduino;
- уточнить точный тип и место расположения драйвера виртуального COM-порта (если плата не использует интерфейс RS-232 явным образом).
- изучить особенности установки интегрированной среды IDE Arduino [1];
- изучить способы настройки режимов работы последовательных каналов в ОС Windows;
- убедиться в отсутствии уже установленного программного продукта IDE Arduino;
- составить пошаговый план установки среды IDE Arduino с учетом возможностей и особенностей настроек предоставленного персонального компьютера.

Вопросы для самоконтроля

- какую роль играет последовательный канал (COM-port) персонального компьютера в процессе работы в IDE Arduino;
- возможна ли установка и эксплуатация IDE Arduino на персональных компьютерах, не имеющих каналов COM-port.

- какой интерфейс для подключения к персональному компьютеру (ПК) используется в выданной плате микроконтроллера?
- как осуществляется питание платы исследуемого микроконтроллера?
- что такое «виртуальный COM-порт»?

Программа работ

(Для версии платы Arduino с USB интерфейсом)

1. Запустить на исполнение дистрибутив IDE Arduino, выполнить требуемые действия для его установки и убедиться в том, что программа установлена без ошибок.
2. Установить драйвер виртуального COM-порта и убедиться в том, что он установлен без ошибок.
3. Подключить плату микроконтроллера к персональному компьютеру
4. Запустить среду IDE Arduino
5. Осуществить настройку IDE Arduino, указав тип подключенной платы и номер зарегистрированного операционной системой COM-порта.
6. Проверить функционирование системы путем загрузки демонстрационной программы **blink**.

Методические указания

Определить номер виртуального COM-порта можно с использованием Диспетчера устройств ОС Windows (см. рисунок ниже).

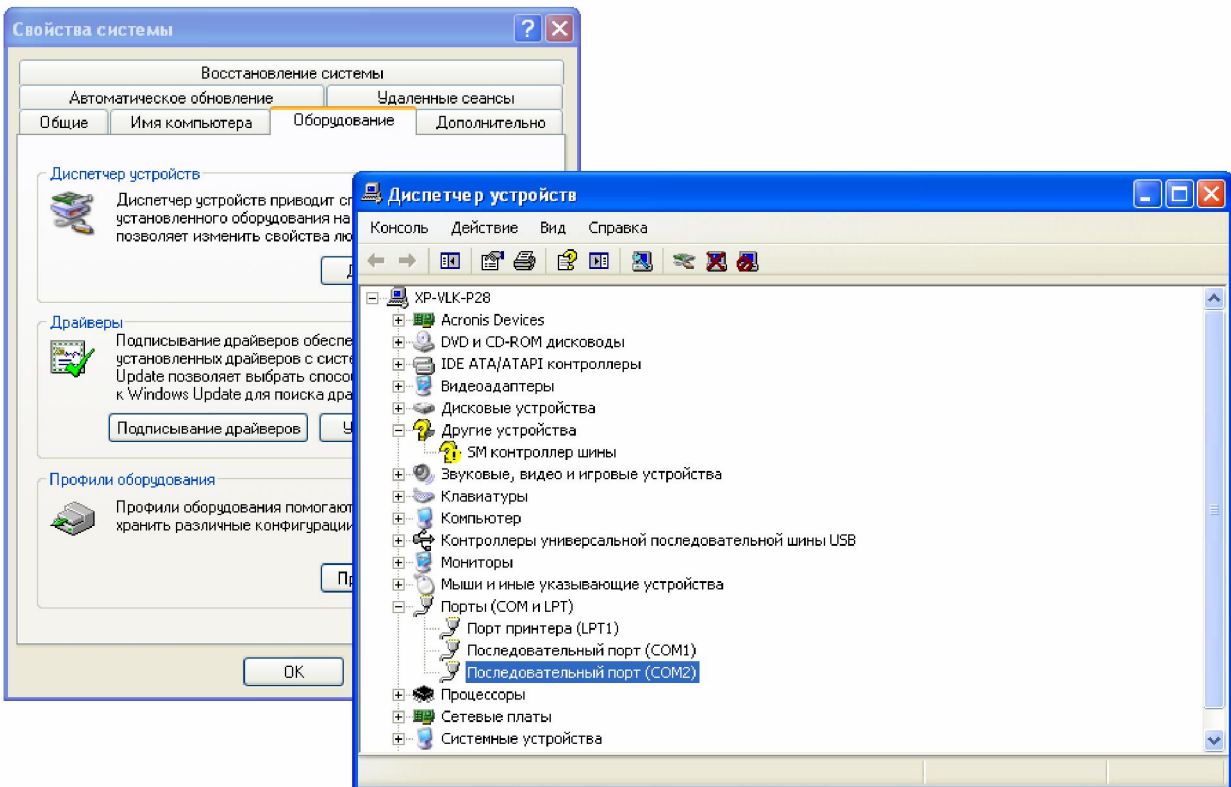


Рисунок 1 – определение номера виртуального COM-порта

Для настройки IDE Arduino (тип подключенной платы и номер зарегистрированного операционной системой COM-порта) необходимо воспользоваться соответствующими вкладками программы, что иллюстрируется следующим рисунком.

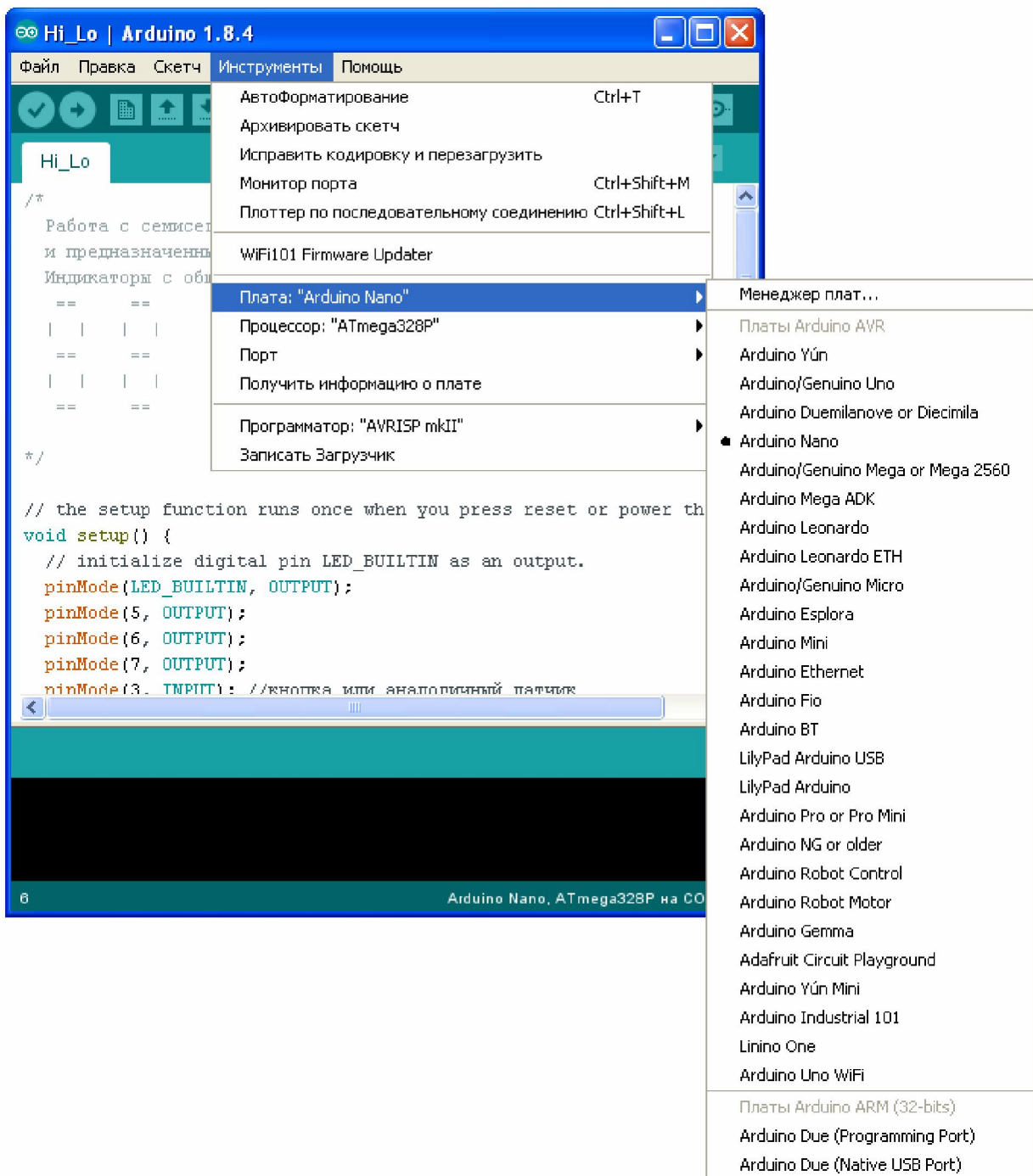


Рисунок 2 – настройка типа платы Arduino

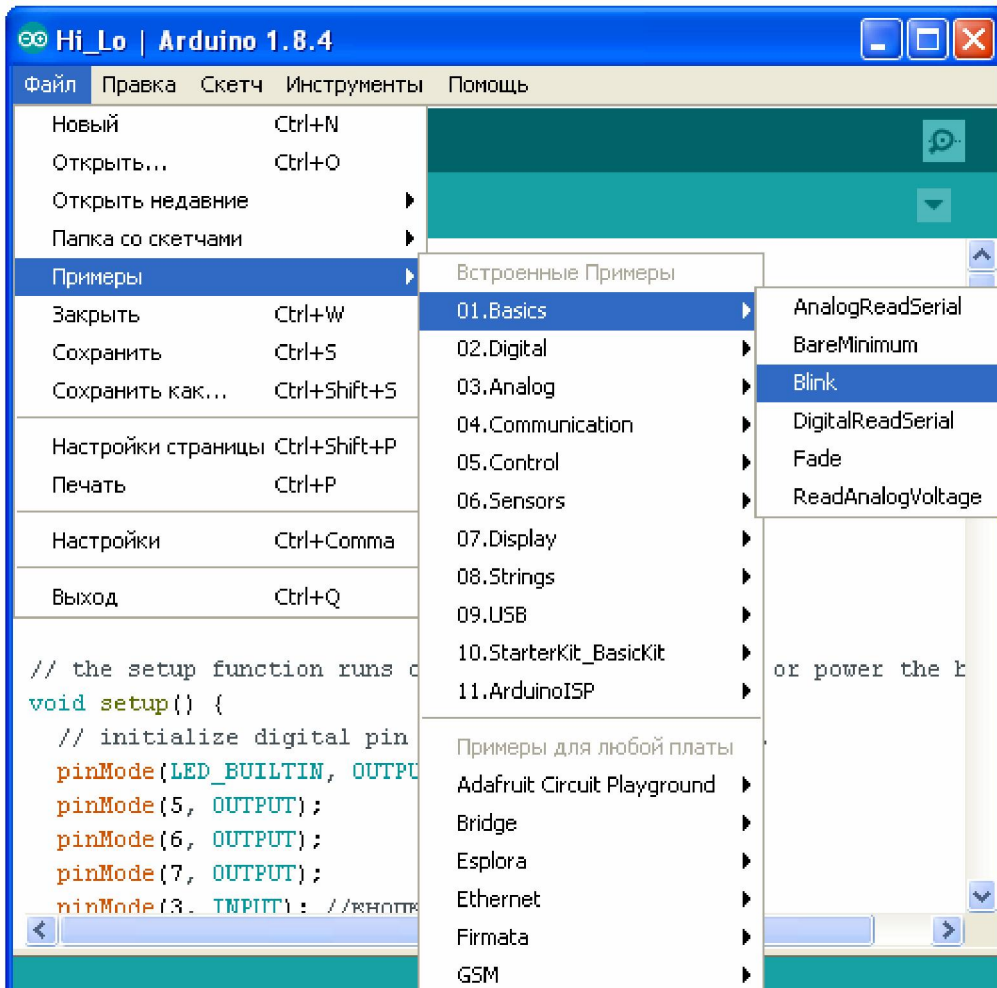


Рисунок 3 – выбор демонстрационной программы

Контрольные вопросы

1. Опишите стандартный модельный ряд плат Arduino с применением микро-ЭВМ серии ATmega.
2. Назовите требования к питанию для плат Arduino UNO и Arduino nano.
3. Каковы уровни логических сигналов на разъемах плат Arduino UNO и Arduino nano?
4. Опишите поведение демонстрационной программы **blink**.
5. Каковы особенности подключения плат, имеющих аппаратный интерфейс RS-232?
6. Какие альтернативные системы программирования разработаны для платформы Arduino?
7. Какие возможности предоставляет среда программирования IDE Arduino?

Содержание отчета

Отчет должен содержать:

- титульный лист с указанием темы работы;
- цель работы;
- описание лабораторного макета платы Arduino;
- протокол выполненных работ, содержащий изображения экрана ПК для основных этапов работ;
- выводы для важнейших этапов работ;
- ответы на контрольные вопросы (если имеются).

2 Изучение среды разработки и отладки

Цель работы

Изучение базовых функций языка Arduino, анализ характеристик подключаемых в цикле лабораторных работ внешних устройств и выбор библиотек для управления этими устройствами.

Подготовка к работе

Изучение функций, необходимых для проведения лабораторных работ (Delay(), средств обмена по последовательному каналу Serial.begin(), Serial.print() и др.).

Вопросы для самоконтроля

- опишите назначение последовательного канала (UART, RS-232);
- опишите программные средства, используемые в программе Arduino для обмена по последовательному каналу;
- опишите средства, используемые в IDE Arduino на персональном компьютере для обмена по последовательному каналу;
- опишите уровни сигналов на цифровых и аналоговых линиях Arduino.

Программа работ

- изучить средства взаимодействия между платой Arduino и программной средой IDE Arduino на персональном компьютере;
- выполнить демонстрационный пример обмена;
- внести изменения в программу обмена в соответствии с рекомендациями преподавателя.

Методические указания

Макетная плата, содержащая микро-ЭВМ, описана в Приложении А. Плата позволяет устанавливать в гнезда изучаемые устройства и осуществлять их подключение к контактам разъема платы Arduino с помощью проводников со штырьками (гнездами).

Для отладки программ удобно обеспечить вывод сообщений на экран системы разработки. Однако плата Arduino не содержит каких-либо средств, пригодных для этой цели. Для решения этой проблемы предложен следующий способ.

Сообщения, формируемые программой, исполняемой на плате Arduino, передаются в особом формате последовательного вида по линии Tx интерфейса UART. Эти сообщения принимаются на персональном компьютере (ПК) и выводятся в текстовом виде в окне *терминала*. Хотя организация такого обмена не является простой, наличие готовых библиотечных функций для программы на плате Arduino и специальные программные средства в среде IDE Arduino на ПК существенно упрощают задачу.

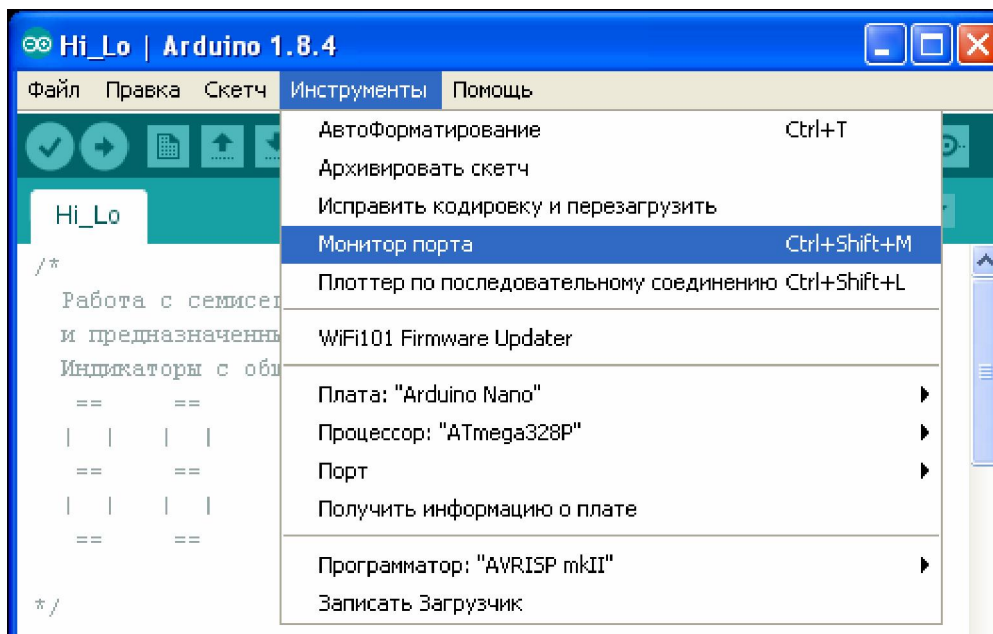


Рисунок 4 – вызов монитора порта из меню

Для работы в режиме обмена по последовательному каналу используется *монитор порта (serial monitor)*, вызов которого осуществляется в соответствии с рисунком выше.

Существует альтернативный способ вызова монитора порта - активацией кнопки справа на панели IDE Arduino:

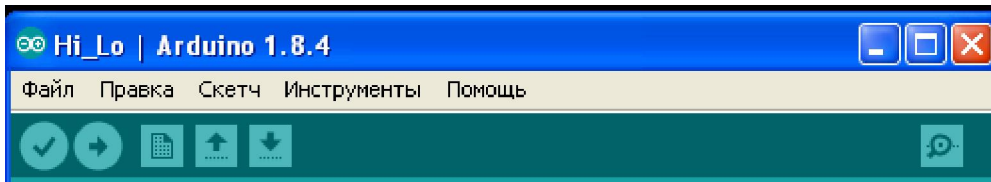


Рисунок 5 - вызов монитора порта кнопкой справа на панели

Выводимые через последовательный канал данные будут отображены в окне терминала:

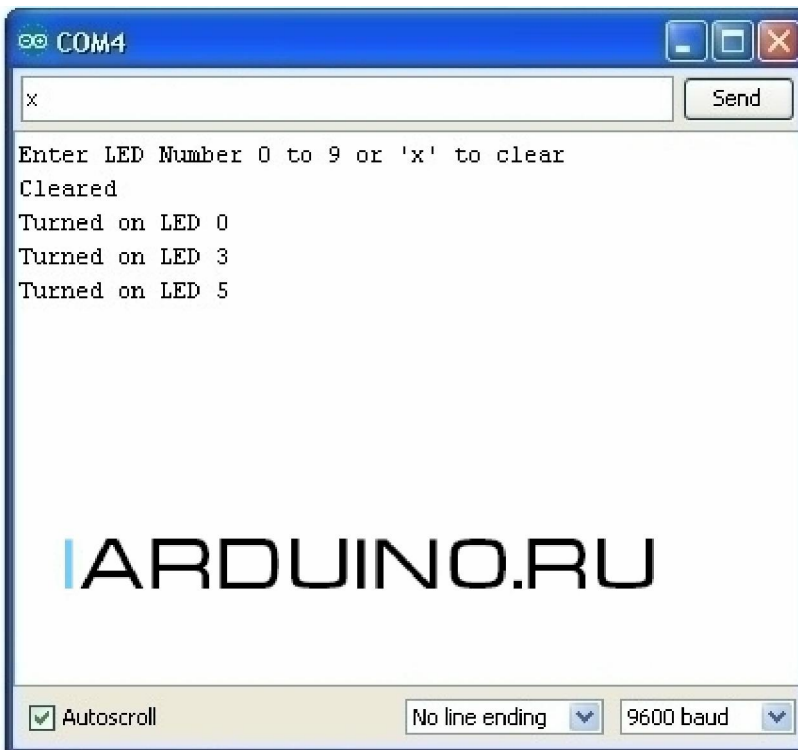


Рисунок 6 – пример вывода сообщений в окно монитора порта

Примеру на рисунке выше соответствует следующий программный код.

```

void setup()
{
    Serial.begin(9600);
    // Ожидание готовности последовательного канала
    while (! Serial);
    // Вывод первого сообщения
    Serial.println("\nEnter LED Number 0 to 7 or \'x\' to
clear\");
}

void loop()
{
    if (Serial.available()) // если приняты данные
    {
        char ch = Serial.read(); // читаем принятые данные
        if (ch >= \'0\' && ch <= \'7\')
        {
            int led = ch - \'0\';
            //выводим принятые данные на экран
            Serial.print("\Turned on LED \");
            Serial.println(led);
        }
        if (ch == \'x\')
        {
            Serial.println("\Cleared\");
        }
    }
}

```

Порядок выполнения лабораторной работы

1. Изучить документацию по подключению платы Arduino и команды языка.
2. Составить схему подключения в соответствии с полученным заданием.
3. Выполнить монтаж в соответствии со схемой устройства на макетной плате.
4. **Предоставить смонтированное устройство для проверки преподавателю.**

5. Выполнить набор, проверку и исполнение демонстрационной программы.
6. Проверить правильности функционирования программы
7. Внести изменения в программу по заданию преподавателя и добиться ее правильного функционирования.

Содержание отчета

Отчет должен содержать:

- титульный лист с указанием темы работы;
- цель работы;
- описание базовых функций языка для ввода-вывода (pinMode(), digitalRead(), digitalWrite(), analogRead(), analogWrite(), Delay() и т.д.);
- описание монтажной схемы подключения устройства;
- протокол выполненных работ, содержащий изображения экрана для основных этапов работ;
- выводы для важнейших этапов работ;
- ответы на контрольные вопросы (вопросы для самоконтроля).

3 Проектирование программ управления объектами с цифровым интерфейсом

Программирование микро-ЭВМ обычно предполагает подключение к ним дополнительных устройств (датчиков, исполнительных механизмов, индикаторов,..). Часть программного кода разрабатываемых программ будет осуществлять взаимодействие с этими устройствами. Для правильного управления внешними устройствами необходимо знать особенности *интерфейса* этих устройств. Примерами интерфейсов являются [2]:

- цифровой интерфейс;
- аналоговый интерфейс;
- интерфейс SPI;
- интерфейс I2C;
- интерфейс 1-wire.

Наиболее простым для реализации и работы является цифровой интерфейс.

Цель работы

Изучение свойств цифровых линий платформы Arduino и приобретение навыков разработки программ для управления цифровыми линиями.

Подготовка к работе

Согласование с преподавателем устройств для исследования, их изучение .

Изучение функций, необходимых для проведения лабораторной работы (`pinMode()`, `digitalRead()`, `digitalWrite()`, `Delay()` и др.).

Вопросы для самоконтроля

- опишите назначение подключаемого устройства;

- опишите уровни сигналов на интерфейсе и сравните их с уровнями сигналов платы Arduino;
- опишите сферы практического применения устройства;
- какие программные функции языка IDE Arduino будет использовать программа?

Программа работ

Для набора предложенных преподавателем модулей:

- изучить документацию, описывающую модуль;
- изучить демонстрационный пример;
- составить схему подключения модуля к плате;
- осуществить подключение устройства к плате;
- выполнить демонстрационный пример;
- внести изменения в программу в соответствии с рекомендациями преподавателя.

Методические указания

Макетная плата, содержащая микро-ЭВМ, описана в Приложении А. Плата позволяет устанавливать в гнезда изучаемые устройства и осуществлять их подключение к контактам разъема платы Arduino с помощью проводников со штырьками (гнездами).

Цифровые линии на плате Arduino имеют порядковые номера (см. Приложение Б). Каждая из этих линий может быть настроена на ввод или вывод. Важно знать, что уровни выходных и входных сигналов на цифровых линиях должны быть согласованы со спецификацией микро-ЭВМ и не превышать напряжения питания микро-ЭВМ. Обычно напряжение питания равно 5 V, а уровни логических сигналов таковы:

- логический 0: 0...0,5V;
- логическая 1: 2,4V...5V.

Для настройки цифровой линии в режим **ввода** или **вывода** используют функцию `pinMode()`.

Линия в режиме *ввода* принимает сигналы от датчиков, которые в данный момент соответствуют одному из логических уровней. Определить конкретное значение уровня можно с использованием функции `digitalRead()`.

На рисунке ниже приводится пример подключения кнопки к одному из цифровых входов.

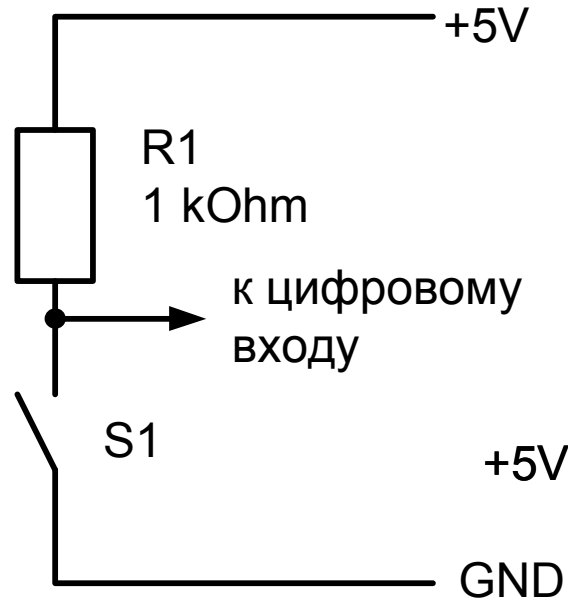


Рисунок 7 – подключение кнопки к цифровому входу

При отпущенной (не замкнутой) кнопке на цифровом входе будет присутствовать уровень логической 1. При нажатии (замыкании) кнопки этот уровень станет равен логическому 0.

Линия в режиме *вывода* формирует сигналы уровня логического 0 или логической 1, что достигается применением функции `digitalWrite()`.

На рисунке ниже приводится пример подключения светодиодного индикатора к одному из цифровых выходов.

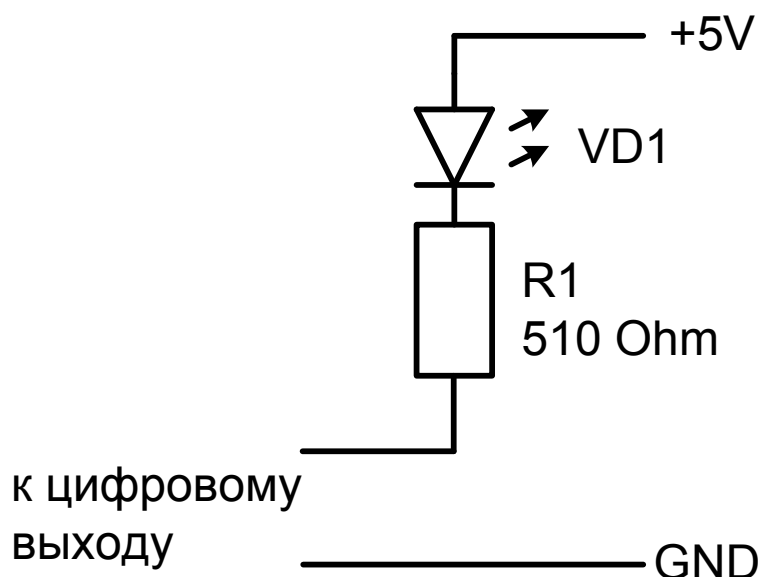


Рисунок 8 – подключение светодиода к цифровому выходу

Если на цифровом выходе сформирован уровень логического 0, замыкается цепь *источник питания–светодиод–резистор* и светодиодный индикатор светится.

Если на цифровом выходе сформирован уровень логической 1, уровень выходного сигнала близок к напряжению питания, а при этом цепь *источник питания–светодиод–резистор* будет разорвана (закрыт светодиод) и индикатор будет погашен.

Замечание:

использование ограничительного резистора R1 совместно со светодиодом обязательно!

Порядок выполнения лабораторной работы

Порядок работ зависит от того, какое устройство будет подключаться. Ниже перечислены типичные шаги выполнения работы.

1. Изучение документации.
2. Составление схемы подключения.
3. Монтаж в соответствии со схемой устройства на макетной плате.
4. Набор, проверка и исполнение демонстрационной программы.

5. Проверка правильности функционирования программы.
6. Внесение изменений в программу по заданию преподавателя и доведение ее до правильного функционирования.

Замечание: перед подключением необходимо предоставить смонтированное устройство для проверки преподавателю.

Содержание отчета

Отчет должен содержать:

- титульный лист с указанием темы работы;
- цель работы;
- описание использованных функций языка для ввода-вывода;
- описание монтажной схемы подключения устройства;
- протокол выполненных работ, содержащий изображения экрана для основных этапов работ;
- выводы для важнейших этапов работ;
- ответы на контрольные вопросы (вопросы для самоконтроля).

4 Проектирование программ управления объектами с аналоговым интерфейсом

Цель работы

Изучение свойств аналоговых линий платформы Arduino и приобретение навыков разработки программ для управления аналоговыми линиями.

Подготовка к работе

Согласование с преподавателем устройств для исследования, их изучение .

Изучение функций, необходимых для проведения лабораторной работы (analogRead(), analogWrite(), Delay() и др.).

Вопросы для самоконтроля

- опишите назначение подключаемого устройства;
- опишите уровни сигналов на интерфейсе и сравните их с уровнями сигналов платы Arduino;
- опишите сферы практического применения устройства;
- какие программные функции языка IDE Arduino будет использовать программа?

Программа работ

Для набора предложенных преподавателем модулей:

- изучить документацию, описывающую модуль;
- изучить демонстрационный пример;
- составить схему подключения модуля к плате;
- осуществить подключение устройства к плате;
- выполнить демонстрационный пример;
- внести изменения в программу в соответствии с рекомендациями преподавателя.

Методические указания

Аналоговые линии на плате Arduino имеют порядковые номера, начинающиеся с буквы А (см. Приложение Б). Аналоговые линии Arduino обычно имеют predetermined настройку на ввод и подключены к входам аналого-цифрового преобразователя (АЦП). Важно знать, что уровни выходных и входных сигналов на цифровых линиях должны быть согласованы со спецификацией микро-ЭВМ и не превышать напряжения питания микро-ЭВМ.

Обычно напряжение питания равно 5 V, а уровни аналоговых сигналов могут принимать любое значение в диапазоне 0...5V.

АЦП распознает эти сигналы, но воспринять бесконечное число возможных значений он не способен. Упрощенно говоря, АЦП способен представить измеренное значение с точностью, определяемой его *разрядностью*. Например, 10-разрядный АЦП выдает 10-битовый код, разбивая весь интервал измерений на 1024 части. Если номинальное напряжение на входе АЦП соответствует величине 5V, то минимальное воспринимаемое значение сигнала примерно равно $5V/1024=5mV$.

Функция `analogRead (pin)` считывает значение из заданного аналогового входа (pin) с 10-битовым разрешением. Эта функция работает только на аналоговых портах (0-5). Результирующее целое значение находится в диапазоне от 0 до 1023.

Для типичной платы Arduino цифровые выходы работают в режиме широтно-импульсной модуляции (ШИМ, PWM). Плата Arduino с ATmega8 поддерживает только выводы 9, 10 и 11. Для плат Arduino с ATmega168 (328) функция ШИМ работает на выводах 3, 5, 6, 9, 10 и 11.

Функция `analogWrite(pin, value)` записывает псевдо-аналоговое значение, используя схему с широтно-импульсной модуляцией (PWM), на выходной вывод, помеченный как PWM. Значение может быть задано как переменная или константа в диапазоне 0-255.

На рисунке ниже приводится пример формирования аналогового сигнала на одном из аналоговых входов.

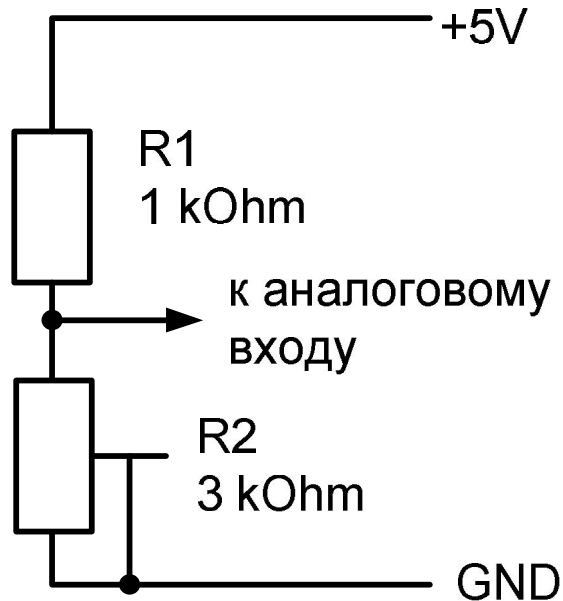


Рисунок 9 – подключение кнопки к цифровому входу

Переменный (подстроечный) резистор R2 в схеме может изменять свое сопротивление при регулировке. Совместно с резистором R1 он образует делитель напряжения, напряжение на выходе которого может принимать произвольное значение в некотором диапазоне. Минимальное напряжение при верхнем положении движка резистора R2 равно 0, максимальное при нижнем положении движка равно $\frac{3}{4}$ от напряжения питания (для указанных на схеме номиналах резисторов).

Линия в режиме ШИМ формирует сигналы уровня, который имеет 256 градаций в некотором диапазоне. Для этого используется функция `analogWrite()`.

На рисунке ниже приводится пример подключения светодиодного индикатора к одному из аналоговых выходов.

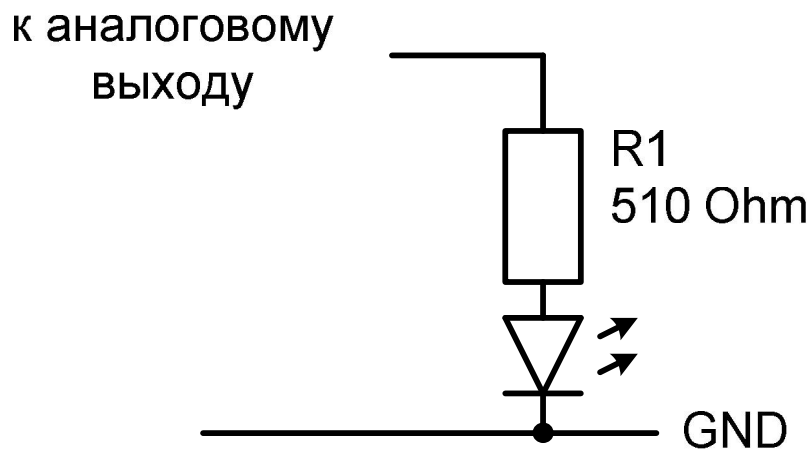


Рисунок 10 – подключение светодиода к аналоговому выходу

Средняя величина тока, протекающего через светодиод (а следовательно, и яркость свечения индикатора) определяется уровнем выходного напряжения аналогового выхода.

Замечание:

использование ограничительного резистора R1 совместно со светодиодом обязательно!

Порядок выполнения лабораторной работы

Порядок работ зависит от того, какое устройство будет подключаться. Ниже перечислены типичные шаги выполнения работы.

1. Изучение документации.
2. Составление схемы подключения.
3. Монтаж в соответствии со схемой устройства на макетной плате.
4. Набор, проверка и исполнение демонстрационной программы.
5. Проверка правильности функционирования программы.
6. Внесение изменений в программу по заданию преподавателя и доведение ее до правильного функционирования.

Замечание: перед подключением необходимо предоставить смонтированное устройство для проверки преподавателю.

Содержание отчета

Отчет должен содержать:

- титульный лист с указанием темы работы;
- цель работы;
- описание использованных функций языка для ввода-вывода;
- описание монтажной схемы подключения устройства;
- протокол выполненных работ, содержащий изображения экрана для основных этапов работ;
- выводы для важнейших этапов работ;
- ответы на контрольные вопросы (вопросы для самоконтроля).

5 Проектирование программ управления объектами с интерфейсом 1-wire

Программирование микро-ЭВМ обычно предполагает подключение к ним дополнительных устройств (датчиков, исполнительных механизмов, индикаторов,..). Часть программного кода разрабатываемых программ будет осуществлять взаимодействие с этими устройствами. Для правильного управления внешними устройствами необходимо знать особенности *интерфейса* этих устройств. Примерами интерфейсов являются:

- цифровой интерфейс;
- аналоговый интерфейс;
- интерфейс SPI [2];
- интерфейс I2C [2];
- интерфейс 1-wire [3].

Многие интерфейсы сложны в использовании. Для облегчения процесса подключения и управления разрабатываются специальные программные библиотеки, использование которых позволяет абстрагироваться от аппаратных и программных особенностей интерфейса и взаимодействовать с устройствами через простые и понятные программные функции. Среда IDE Arduino поддерживает множество таких библиотек.

Цель работы

Изучение свойств интерфейса 1-wire и приобретение навыков разработки программ для управления периферийным оборудованием, использующим данный интерфейс в проектах с применением платформы Arduino.

Подготовка к работе

Согласование с преподавателем устройств для исследования.
Изучение документации на заданное устройство.

Изучение функций, необходимых для проведения лабораторной работы

Вопросы для самоконтроля

- опишите назначение подключаемого устройства;
- опишите интерфейс, используемый устройством;
- опишите сферы практического применения устройства;
- какие программные функции языка IDE Arduino будет использовать программа?

Программа работ

Для набора предложенных преподавателем модулей:

- изучить документацию, описывающую модуль;
- изучить демонстрационный пример;
- составить схему подключения модуля к плате;
- осуществить подключение устройства к плате;
- **предоставить собранную систему для проверки преподавателю;**
- если ошибок в соединениях нет, исполнить демонстрационный пример.

Методические указания

Макетная плата, содержащая микро-ЭВМ, описана в Приложении А. Плата позволяет устанавливать в гнезда изучаемые устройства и осуществлять их подключение к контактам разъема платы Arduino с помощью проводников со штырьками (гнездами).

Интерфейс 1-wire позволяет передавать и принимать информацию по одному сигнальному проводу, что обуславливает сферу его применения. Встречается интерфейс чаще всего в изделиях фирмы Dallas Semiconductor, которая и представила его в конце 90-х годов прошлого века. Системы 1-Wire привлекательны благодаря легкости монтажа, низкой стоимости устройств,

возможности распознавать устройство при подключении к функционирующей сети, большому числу устройств в сети и т.д.

Типичная система 1-Wire состоит из управляющего контроллера (мастера или ведущего) и одного или нескольких устройств (ведомых), присоединенных к общей шине (см. рисунок ниже).

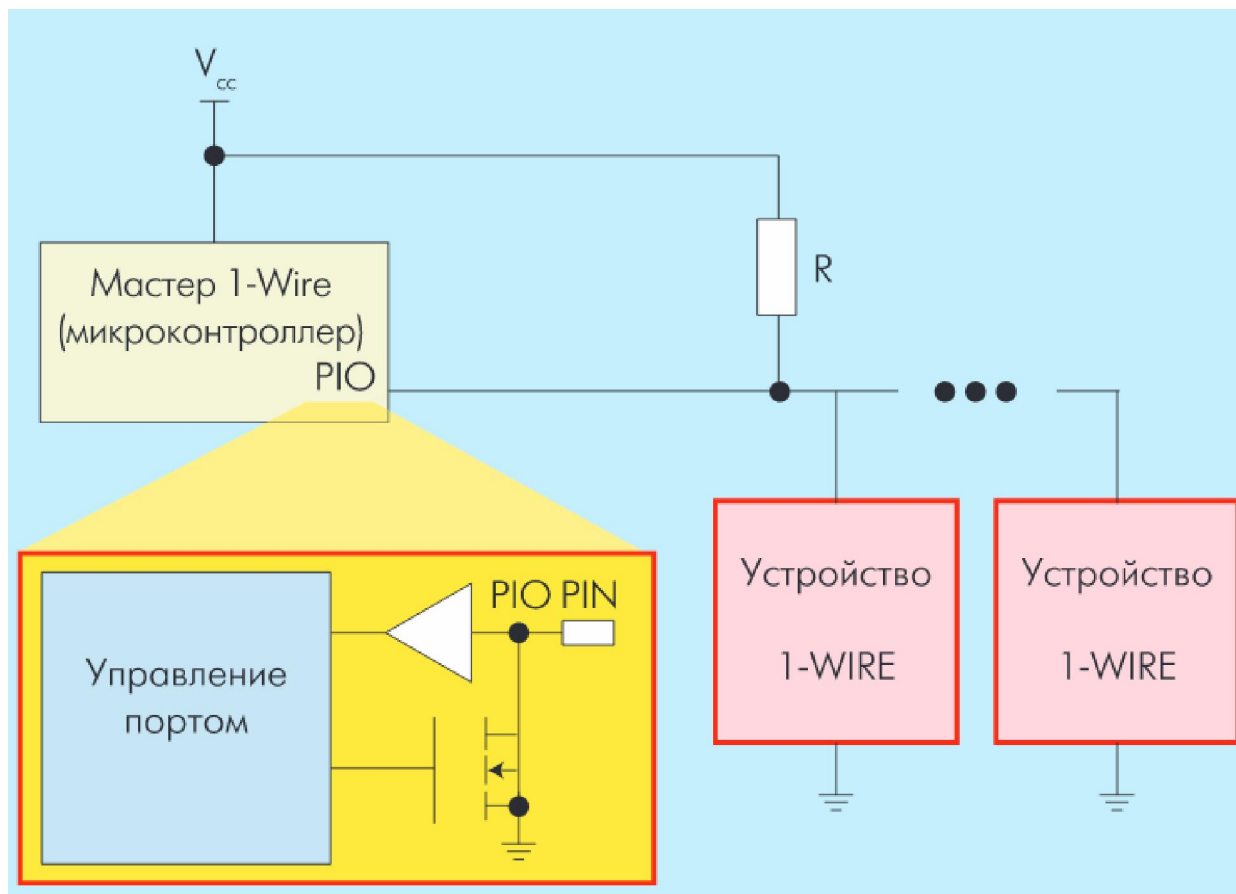


Рисунок 11 - подключение устройств по интерфейсу 1-wire

Весь обмен на шине 1-Wire происходит посредством специальных команд. Их число для каждого типа устройств различно. Но есть и минимальный набор стандартных команд, которые поддерживают все 1-Wire-устройства – так называемые ROM-команды.

У каждого устройства 1-Wire есть 64-разрядный идентификатор (ID). Он состоит из 8-разрядного кода семейства, который идентифицирует тип устройства и поддерживаемые им функции, 48-разрядного серийного номера и 8-битного поля кода циклического избыточного контроля (CRC-8). ID вводится при

изготовлении устройства и хранится в ПЗУ. Фирма Maxim гарантирует, что один раз использованный адрес никогда не повторится в другом устройстве. Для обмена с устройством необходимо этот адрес знать. С этой целью разработан специальный алгоритм, позволяющий осуществить определение номера каждого устройства в сети 1-wire.

Для Arduino существуют библиотеки, существенно упрощающие разработку программ при использовании интерфейса 1-wire.

Пример устройства с интерфейсом 1-wire (DS18B20 – цифровой термометр с программируемым разрешением, от 9 до 12-bit) приведен в Приложении В.

Порядок выполнения лабораторной работы

Порядок работ зависит от того, какое устройство будет подключаться. Ниже перечислены типичные шаги выполнения работы.

1. Изучение документации.
2. Составление схемы подключения.
3. Монтаж в соответствии со схемой устройства на макетной плате.
4. Набор, проверка и исполнение демонстрационной программы.
5. Проверка правильности функционирования программы.
6. Внесение изменений в программу по заданию преподавателя и доведение ее до правильного функционирования.

Замечание: перед подключением необходимо предоставить смонтированное устройство для проверки преподавателю.

Содержание отчета

Отчет должен содержать:

- титульный лист с указанием темы работы;
- цель работы;
- описание использованных функций языка для ввода-вывода;

- описание монтажной схемы подключения устройства;
- протокол выполненных работ, содержащий изображения экрана для основных этапов работ;
- выводы для важнейших этапов работ;
- ответы на контрольные вопросы (вопросы для самоконтроля).

6 Проектирование программ управления объектами с интерфейсом I2C

Изучение свойств интерфейса 1-wire [3] и приобретение навыков разработки программ для управления периферийным оборудованием, использующим данный интерфейс в проектах с применением платформы Arduino.

Подготовка к работе

Согласование с преподавателем устройств для исследования.

Изучение документации на заданное устройство.

Изучение функций, необходимых для проведения лабораторной работы

Вопросы для самоконтроля

- опишите назначение подключаемого устройства;
- опишите интерфейс, используемый устройством;
- опишите сферы практического применения устройства;
- какие программные функции языка IDE Arduino будет использовать программа?

Программа работ

Для набора предложенных преподавателем модулей:

- изучить документацию, описывающую модуль;
- изучить демонстрационный пример;
- составить схему подключения модуля к плате;
- осуществить подключение устройства к плате;
- **предоставить собранную систему для проверки преподавателю;**
- если ошибок в соединениях нет, исполнить демонстрационный пример.

Методические указания

Макетная плата, содержащая микро-ЭВМ, описана в Приложении А. Плата позволяет устанавливать в гнезда изучаемые устройства и осуществлять их подключение к контактам разъема платы Arduino с помощью проводников со штырьками (гнездами).

Интерфейс I2C (I I C) был разработан компанией Philips и зарегистрирован под запатентованным названием "I2C". Существуют аналоги этого интерфейса, которые используют иные названия (TWI, 2 line interface). Все они работают по единому принципу.

Интерфейс I2C использует для обмена информацией 2 линии и позволяет организовать сеть, в которую включается большое (128) количество различных устройств - от датчиков температуры до микроконтроллеров.

Линии интерфейса:

- SDA - отвечает за передачу информации(начало передачи, адрес, данные);
- SCL - тактирование шины.

Интерфейс I2C предусматривает устройства двух типов: Master (главное, ведущее) и Slave (ведомое).

Структурная схема подключения устройств с интерфейсом I2C приведена на рисунке ниже. Стандарт предусматривает наличие подтягивающих резисторов R1, R2. Эти резисторы при отсутствии активности на линиях обеспечивают уровни логической 1.

Передача/прием сигналов осуществляется подтягиванием уровней на линиях к общему проводу (уровни логического 0). Взаимодействие устройств описывается стандартом.

Для упрощения программирования устройств с интерфейсом I2C для платформы Arduino разработаны специальные библиотеки.

Пример устройства с интерфейсом 1-wire (DS18B20 – цифровой термометр с программируемым разрешением, от 9 до 12-bit) приведен в Приложении Г.

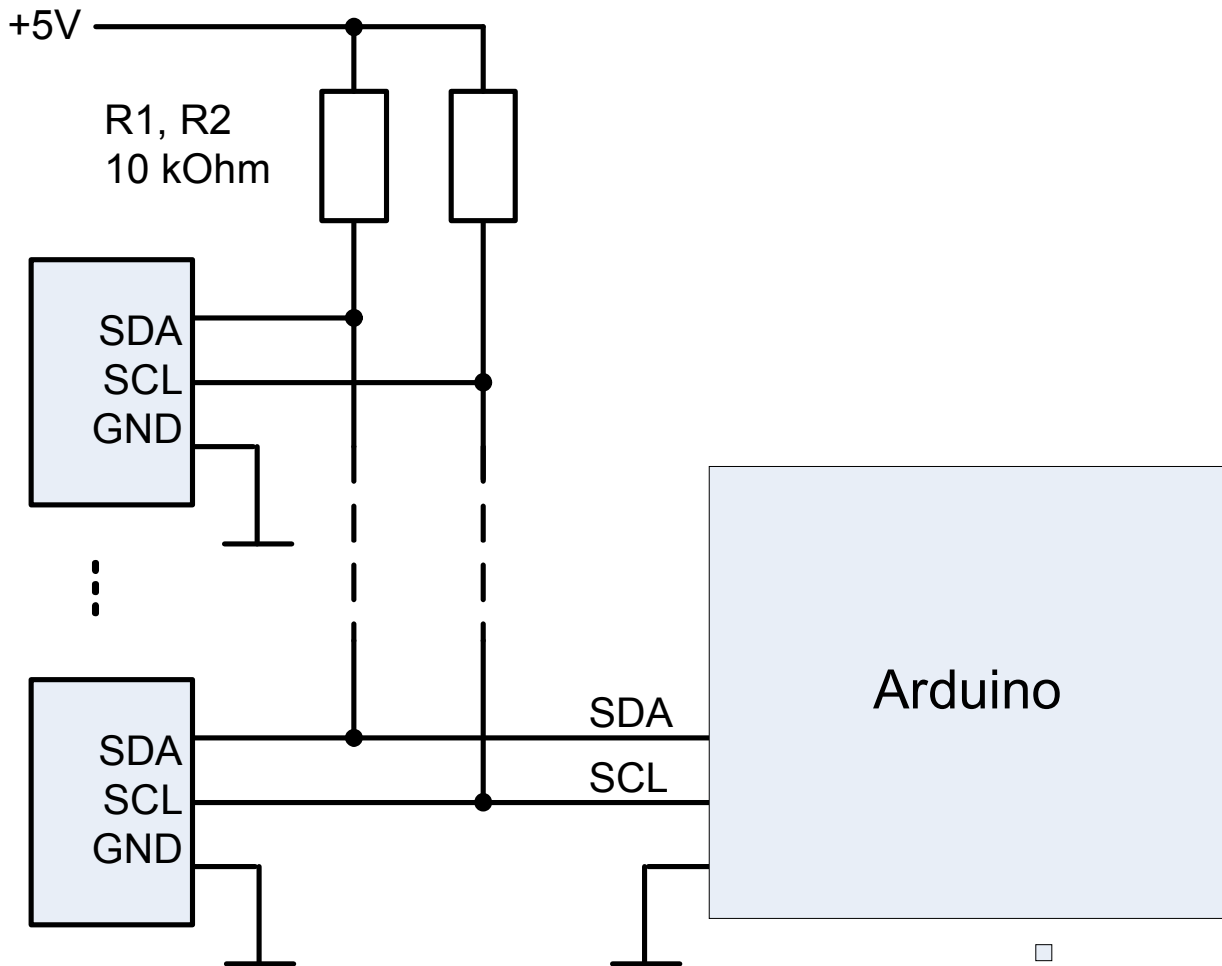


Рисунок 12 – подключение устройств с интерфейсом I2C

Порядок выполнения лабораторной работы

Порядок работ зависит от того, какое устройство будет подключаться. Ниже перечислены типичные шаги выполнения работы.

1. Изучение документации.
2. Составление схемы подключения.
3. Монтаж в соответствии со схемой устройства на макетной плате.
4. Набор, проверка и исполнение демонстрационной программы.
5. Проверка правильности функционирования программы.
6. Внесение изменений в программу по заданию преподавателя и доведение ее до правильного функционирования.

Замечание: перед подключением необходимо предоставить смонтированное устройство для проверки преподавателю.

Содержание отчета

Отчет должен содержать:

- титульный лист с указанием темы работы;
- цель работы;
- описание использованных функций языка для ввода-вывода;
- описание монтажной схемы подключения устройства;
- протокол выполненных работ, содержащий изображения экрана для основных этапов работ;
- выводы для важнейших этапов работ;
- ответы на контрольные вопросы (вопросы для самоконтроля).

Литература

1. Монк, С. Програмируем Arduino. Профессиональная работа со скетчами[Текст]: пер. с английского. - СПб.: Питер, 2017.

2. Лапин, А. А. Интерфейсы. Выбор и реализация [Текст] / А. А. Лапин. - Москва : Техносфера, 2005. - 168 с.

3. Интерфейс 1-Wire: устройство и применение
http://www.electronics.ru/files/article_pdf/0/article_668_639.pdf

4. Arduino - блокнот программиста - Brian W. Evans (русский перевод)
http://robocraft.ru/files/books/arduino_notebook_rus_v1-1.pdf

Приложение А - Макетирование устройств при подключении к Arduino

Макетная плата (breadboard) обеспечивает безопасный способ монтажа плат Arduino и исследуемых устройств. Конструкция одного из вариантов платы приведена на рисунке ниже.

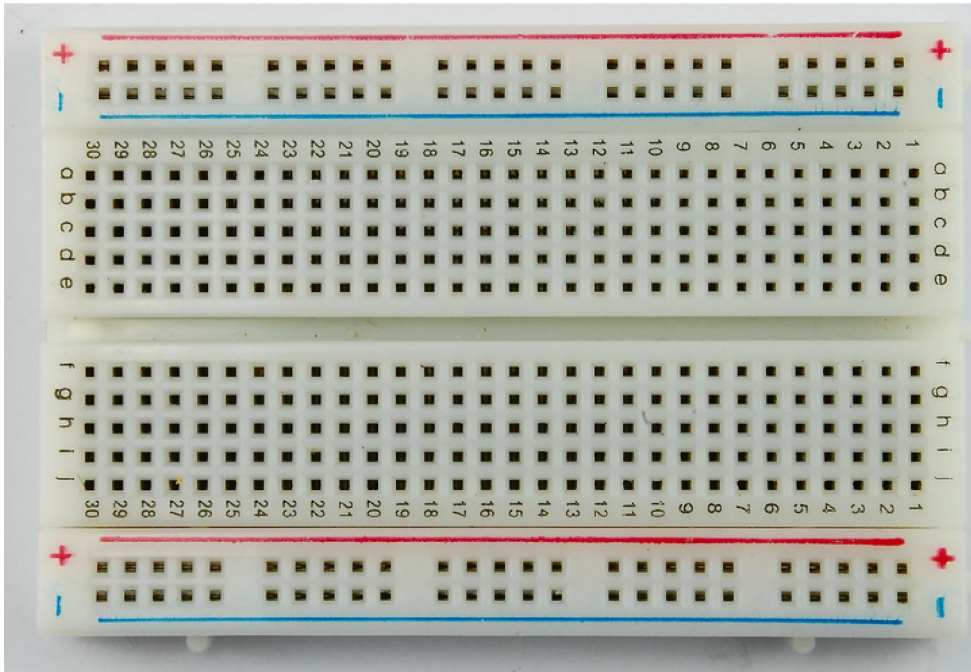


Рисунок А.1 - Макетная плата (breadboard)

Контакты, имеющие один номер в пределах верхней или нижней части платы, соединены друг с другом и изолированы от прочих элементов платы. Например, соединены контакты a30, b30, c30, d30, e30 в верхней части платы и контакты f30, g30, h30, i30, j30 в нижней части.

Цепи для подводки питания обозначены знаками «+» и «-», а также выделены цветом: синий (минус) и красный (плюс). Эти контакты соединены по горизонтали и изолированы друг от друга.

Пример установки модулей на монтажную плату показан на следующем рисунке и включает плату формата Arduino nano и дополнительный модуль питания. Заметим, что питание Arduino nano может осуществляться по линиям USB, а дополнительная плата необходима только при наличии модулей с большим потреблением или питанием 3,3V.

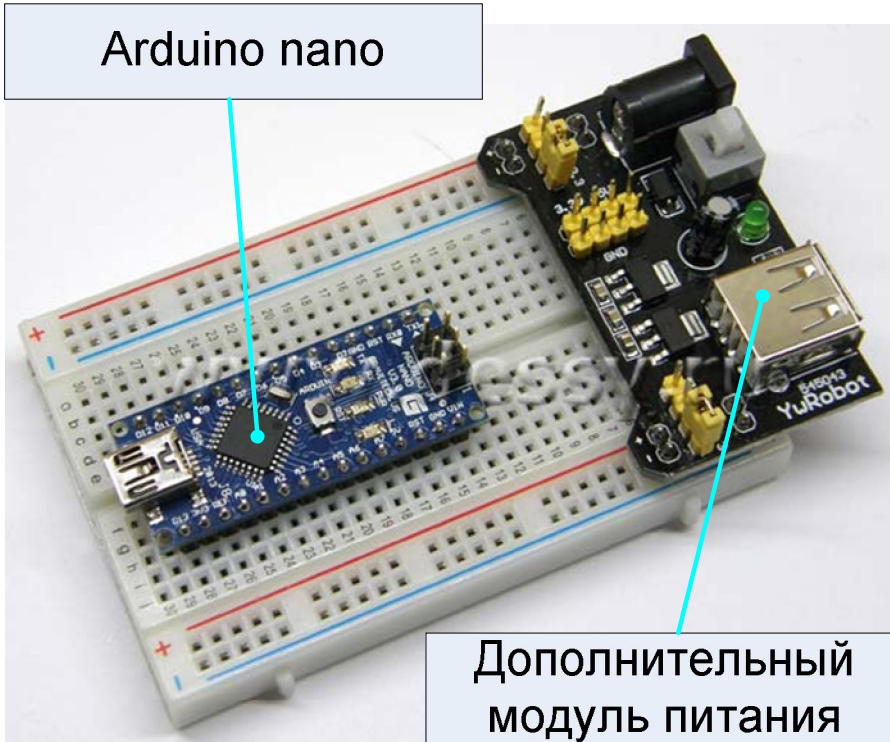


Рисунок А.2 – установка модулей на печатной плате

Дополнительные соединения осуществляются проводниками со штырьками (гнездами), что показано на рисунке ниже (https://ahrameev.ru/wp-content/uploads/2015/09/IMG_5015.jpg)

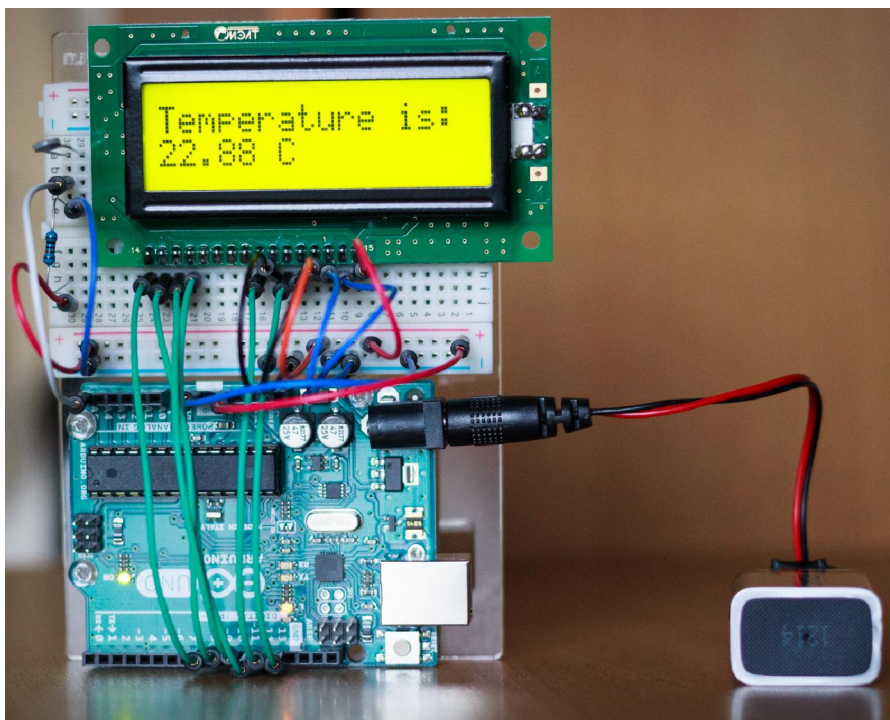
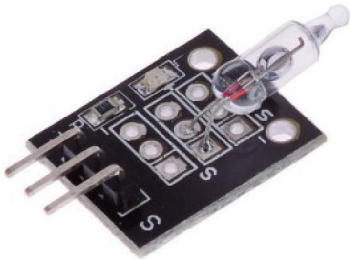
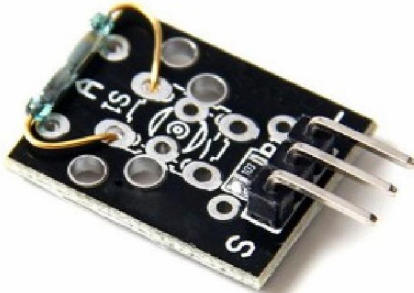


Рисунок А.3 – пример монтажа

Приложение Б – стандартные модули для учебных макетов

Для учебных целей выпускается большое количество модулей, содержащих типичные элементы, применяемые при проектировании простых систем: кнопки, индикаторы, датчики и т.п. Их описание можно найти в сети Интернет, но нет гарантии, что они одинаковы для изделий разных производителей. По этой причине перед подключением необходимо изучение этих модулей и составление схемы их коммутации.

Таблица 1. Замыкатели-размыкатели

Внешний вид модуля	Описание
	<p>Модуль KY-004 позволяет смонтировать кнопку на передней панели небольшого прибора благодаря отверстиям в плате..</p>
	<p>Модуль датчика наклона KY-017 (Mercury open optical module) осуществляет замыкание/размыкание контактов при наклоне платы. Механизм замыкания обеспечивается положением ртутного шарика в баллоне датчика.</p>
	<p>Модуль на основе геркона KY-021 (Mini magnetic reed modules) осуществляет замыкание контактов при воздействии магнитного поля.</p>

Датчики, приведенные в таблице 1, подключаются по схеме на рисунке 7.

Таблица 2. Светодиодные индикаторы

Внешний вид модуля	Описание
	<p>Двухцветный светодиодный индикатор (2color LED module 3MM) KY-029</p> <p>Светодиод светится красным или зеленым светом. При включении обоих источников света светодиод излучает свет оранжевого оттенка.</p>
	<p>Модуля KY-009 содержит: 3-х цветный светодиод , с излучателями света красного, зеленого и синего цветов. Они могут светиться одновременно или поочередно.</p>

Каждый светодиодный элемент устройств из таблицы 2 управляется индивидуально, как показано на рисунках 8 и 10. Подобные модули могут содержать светодиоды с общим анодом или общим катодом, что требует различных вариантов включения.

Приложение В – цифровой датчик температуры с интерфейсом 1-wire

DS18B20 – это цифровой термометр с программируемым разрешением, от 9 до 12-bit. DS18B20 обменивается данными по 1-Wire шине и при этом может быть как единственным устройством на линии, так и работать в группе. Все процессы на шине управляются центральным микропроцессором.

Диапазон измерений от -55°C до $+125^{\circ}\text{C}$ и точностью 0.5°C в диапазоне от -10°C до $+85^{\circ}\text{C}$.

Каждая микросхема DS18B20 имеет уникальный 64-битный код, который позволяет идентифицировать каждый датчик на шине.

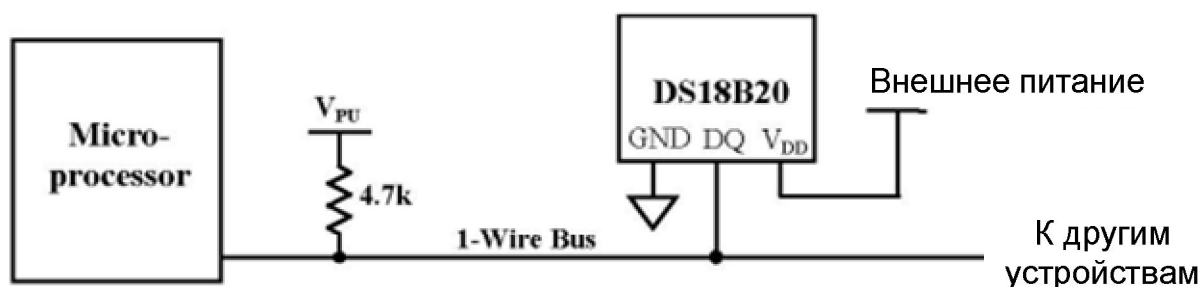


Рисунок В.1 - Схема подключения микросхемы DS18B20

На основе микросхемы DS18B20 изготавливается модуль KY-001, удобный для исследования на макетных платах с использованием Arduino.

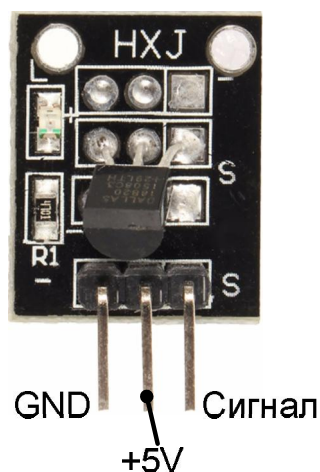


Рисунок В.2 – Модуль KY-001

Приложение Г – цифровые часы с интерфейсом I2C

В некоторых технических задачах требуется знать точное текущее время. Для этого удобно использовать специальные модули часов реального времени. Одной из самых распространённых систем для использования в модулях для Arduino используется микросхема DS1307, для которой существует большое количество библиотек под различные платформы.



Рисунок Г.1 - модуль часов реального времени на микросхеме DS1307

Стандартная схема включения микросхемы DS1307 приведена на рисунке ниже.

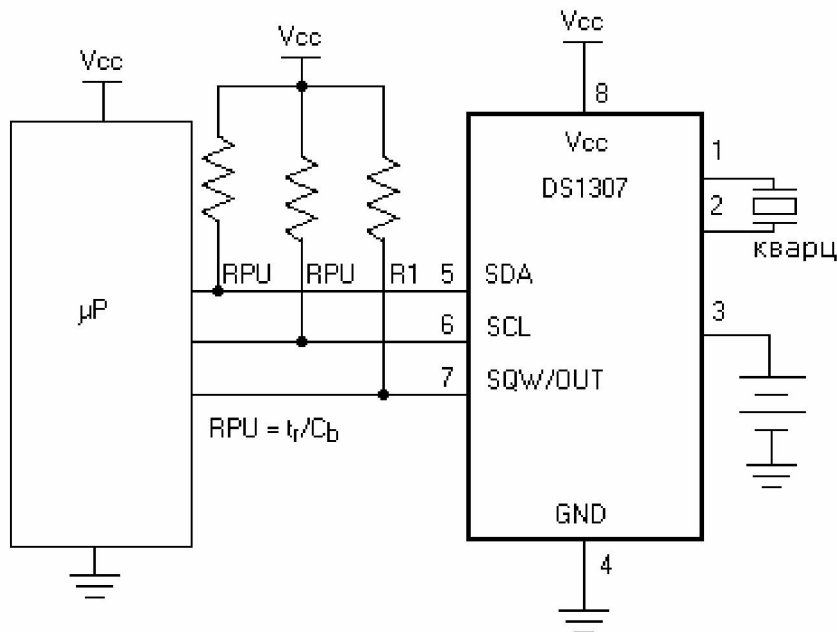


Рисунок Г.2 – схема подключения DS1307

Приложение Д – обзор конструкций языка IDE Arduino

Конструкция языка	Пример	Описание
Обязательные конструкции		
setup()	<pre>void setup() { pinMode(3, INPUT); }</pre>	<p>Конструкция используется для инициализации переменных, настройки режимов работы линий и т.д. Функция запускается только один раз при рестарте микроконтроллера.</p>
loop()	<pre>void loop() { digitalWrite(3, HIGH); delay(1000); digitalWrite(3, LOW); delay(1000); }</pre>	<p>Конструкция обеспечивает бесконечный цикл включенных в нее действий.</p> <p>Функции setup() и loop() должны присутствовать в каждом скетче, даже если эти функции не используются.</p>
Управляющие операторы		
if	<pre>... if (x > 100) digitalWrite(13, HIGH); if (x < 100) digitalWrite(13, LOW); ...</pre>	<p>Оператор if используется в сочетании с операторами сравнения (==, !=, <, >) и проверяет, достигнута ли истинность условия. Пример: если значение переменной x больше 100, то включается светодиод на выходе 13, если меньше — выключается.</p>
if..else	<pre>... if (x > 100) digitalWrite(13, HIGH); else digitalWrite(13, LOW); ...</pre>	<p>Оператор else позволяет сделать проверку отличную от указанной в if, чтобы осуществлять несколько взаимоисключающих проверок. Если ни одна из проверок не получила результат ИСТИНА, то выполняется блок операторов в else.</p>
switch...case	<pre>... switch (x) { case 1: digitalWrite(3, HIGH); case 2: digitalWrite(3, LOW); case 3: break; default: digitalWrite(4, HIGH); } ...</pre>	<p>Оператор switch управляет программой, позволяя задавать действия, которые будут выполняться при разных условиях. Break является командой выхода из оператора, default выполняется, если не выбрана ни одна альтернатива.</p>
for	<pre>void setup()</pre>	<p>Конструкция for используется для</p>

	<pre>{ pinMode(3, OUTPUT); } void loop() { for (int i=0; i <= 255; i++){ analogWrite(3, i); delay(10); } }</pre>	<p>повторения при известном числе повторений.</p> <p>Пример: плавное затемнение светодиода. Заголовок цикла for состоит из трех частей: for (initialization; condition; increment) — initialization выполняется один раз, далее проверяется условие condition, если условие верно, то выполняется приращение increment и цикл повторяется.</p>
while	<pre>void loop() { while (x < 10) { x = x + 1; Serial.println(x); delay(200); } }</pre>	<p>Оператор while используется, как цикл, который будет выполняться, пока условие в круглых скобках является истиной.</p> <p>Пример: оператор цикла while будет повторять код в скобках бесконечно до тех пор, пока x будет меньше 10.</p>
do...while	<pre>void loop() { do { x = x + 1; delay(100); Serial.println(x); } while (x < 10); delay(900); }</pre>	<p>Оператор цикла do...while работает так же, как и цикл while. Однако, при истинности выражения в круглых скобках происходит продолжение работы цикла, а не выход из цикла.</p> <p>Пример: при x больше 10 операция сложения будет продолжаться, но с большей паузой (1000 мс).</p>
break continue	<pre>switch (x) { case 1: digitalWrite(3, HIGH); case 2: digitalWrite(3, LOW); case 3: break; case 4: continue; default: digitalWrite(4, HIGH); }</pre>	<p>Break используется для принудительного выхода из циклов switch, do, for и while, не дожидаясь завершения цикла.</p> <p>Оператор continue пропускает оставшиеся операторы в текущем шаге цикла.</p>
Синтаксис		
; (точка с запятой)	<pre>... digitalWrite(3, HIGH); ...</pre>	<p>Обязательная точка с запятой используется для обозначения конца оператора.</p>
{ (фигурные скобки)	<pre>void setup() { pinMode(3, OUTPUT); digitalWrite(3, HIGH); }</pre>	<p>Фигурные скобки образуют составной оператор, исполняющий включенные действия как единое целое.</p>

	}	
// (комментарий)	x = 5; // комментарий	Комментарии используются для напоминания, как работает программа. Они игнорируются компилятором и не экспортируются в процессор, не занимая место в памяти.
#define	#define ledPin 3	Директива #define позволяет задать имя константе. Директива служит исключительно для удобства и улучшения читаемости программы.
#include	// библиотека для серво #include <Servo.h>	Директива #include используется для включения сторонних библиотек в скетч. Обратите внимание, что директивы #include и #define не требуют точки с запятой в конце строки.

За основу взята информация из источника:

<http://роботехника18.рф/%D1%8F%D0%B7%D1%8B%D0%BA-%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F-%D0%B0%D1%80%D0%B4%D1%83%D0%B8%D0%BD%D0%BE/>