Содержание

Введение

Эта книга создавалась одновременно с набором «Лаборатория электроники и программирования. 77 проектов для Arduino». С этой книгой Вы освоите в теории, а с набором на практике основы программирования, конструирования электронных устройств и робототехники на основе контроллеров – плат Arduino и WiFi модулей ESP8266.

Arduino — это электронный контроллер и удобная платформа быстрой разработки электронных устройств для новичков и профессионалов. Платформа пользуется огромной популярностью во всем мире благодаря удобству и простоте языка программирования, а также открытой архитектуре и программному коду.

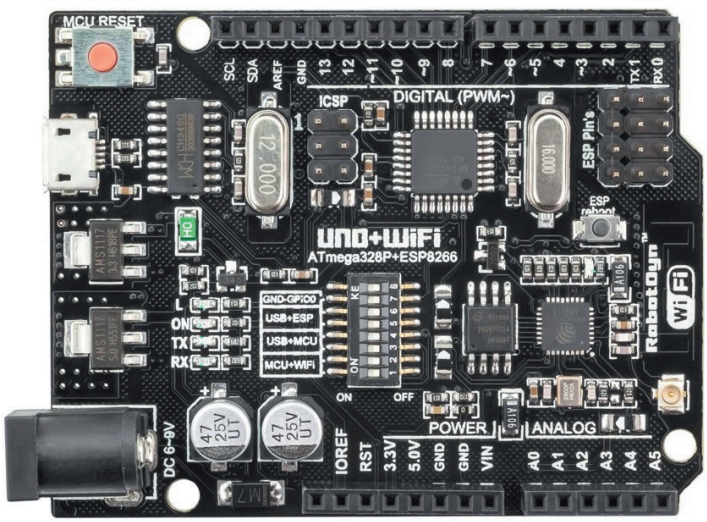


Рис. 1. Плата Arduino+WiFi от компании RobotDyn ESP8266

Появившиеся не так давно платы на основе WiFi модуля ESP8266 и представляющие собой полноценный 32 битный микроконтроллер ESP-8266EX со своим набором GPIO, в том числе SPI, UART, I2C, составляют на данный момент конкуренцию платам Arduino, учитывая низкую цену и возможность программировать устройства ESP8266 в среде Arduino IDE.

Основным элементом набора, является плата Arduino+WiFi (рис. 1), на которой интегрированы контроллер, совместимый с Arduino UNO R3 и WiFi-модуль ESP8266.

Arduino UNO и ESP8266 могут работать вместе или каждый в отдельности, необходимый режим можно установить с помощью находящихся на плате переключателей.

Процесс обучения программированию и конструированию будет проходить посредством создания проектов, начиная с простых, и заканчивая достаточно сложными, требующими достаточного уровня мастерства, которое будет повышаться от проекта к проекту.

В книге описаны, а в набор включены различные датчики, модули, средства отображения информации, источники питания. Для удобства подключения устройств к плате Arduino+WiFi в наборе присутсвует большая плата прототипирования и множество проводов.

Установка Arduino IDE

Установка программного обеспечения

Разработка собственных приложений на базе плат, совместимых с архитектурой Arduino, осуществляется в официальной бесплатной среде программирования Arduino IDE. Среда предназначена для написания, компиляции и загрузки собственных программ в память микроконтроллера, установленного на плате Arduino-совместимого устройства. Основой среды разработки является язык Processing/Wiring — это фактически обычный C++, дополненный простыми и понятными функциями для управления вводом/выводом на контактах. Существуют версии среды для операционых систем Windows, Mac OS и Linux.

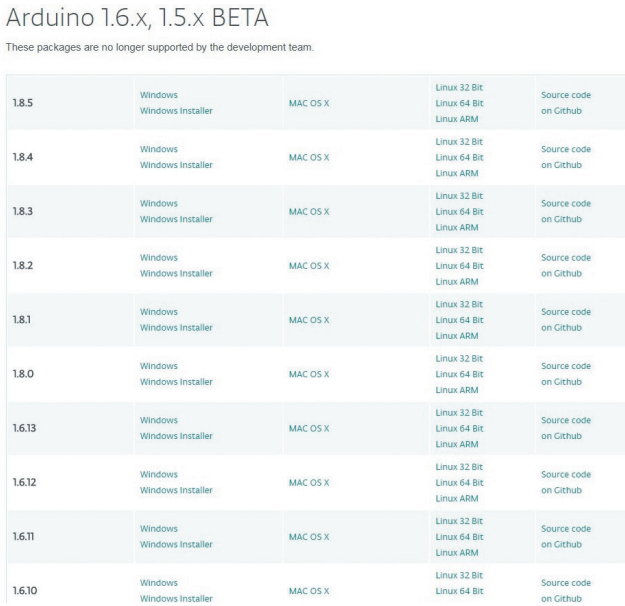
При написании этой книги использовались версии Arduino IDE не ниже 1.6.5. Скачать Arduino IDE можно на официальном сайте **www.arduino.cc.**

Установка Arduino IDE в Windows

Отправляемся на страницу **https://www.arduino.cc/en/Main/OldSoftwareReleases#previous** (рис. 02), выбираем версию для операционной системы Windows и скачиваем архивный файл. Он занимает чуть более 80 Мбайт и содержит все необходимое, в том числе и драйверы. По окончании загрузки распаковываем скачанный файл в удобное для себя место.

Теперь необходимо установить драйверы. Подключаем Arduino к компьютеру. На контроллере должен загореться индикатор питания — зеленый светодиод. Windows начинает попытку установки драйвера, которая заканчивается сообщением **Программное обеспечение драйвера не было установлено.**

Открываем Диспетчер устройств. В составе устройств находим значок Arduino UNO — устройство отмечено восклицательным знаком. Щелкаем правой кнопкой мыши на значке Arduino UNO и в открывшемся окне выбираем пункт Обновить драйверы и далее пункт **Выполнить поиск драйверов на этом компьютере**. Указываем путь к драйверу — ту папку на компьютере, куда распаковывали скачанный архив. Пусть это будет папка drivers каталога установки Arduino — например, C:\arduino-1.6.5\drivers. Игнорируем все предупреждения Windows и получаем в результате сообщение **Обновление программного обеспечения**



Arduino 1.6.x, 1.5.x BETA

These packages are no longer supported by the development team.

1.8.5	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.8.4	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.8.3	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.8.2	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.8.1	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.8.0	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.6.13	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.6.12	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.6.11	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit Linux ARM	Source code on Github
1.6.10	Windows Windows Installer	MAC OS X	Linux 32 Bit Linux 64 Bit	Source code on Github

Рис. 02. *Страница загрузки всех версий Arduino IDE официального сайта Arduino*

для данного устройства завершено успешно. В заголовке окна будет указан и COM-порт, на который установлено устройство.

Осталось запустить среду разработки Arduino IDE (рис. 03). В списке доступных портов отображается название платы Arduino.

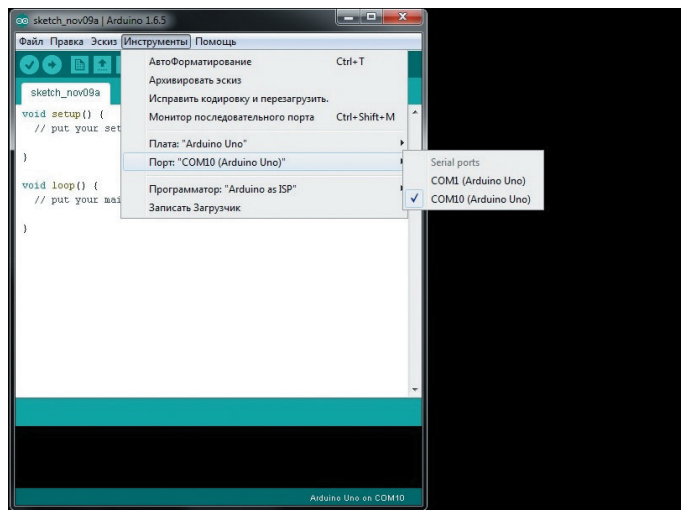


Рис. 03. Arduino IDE — среда разработки

Настройка среды Arduino IDE

Среда разработки Arduino состоит (рис. 04) из:

- редактора программного кода;
- области сообщений;
- окна вывода текста;
- панели инструментов с кнопками часто используемых команд;
- нескольких меню.

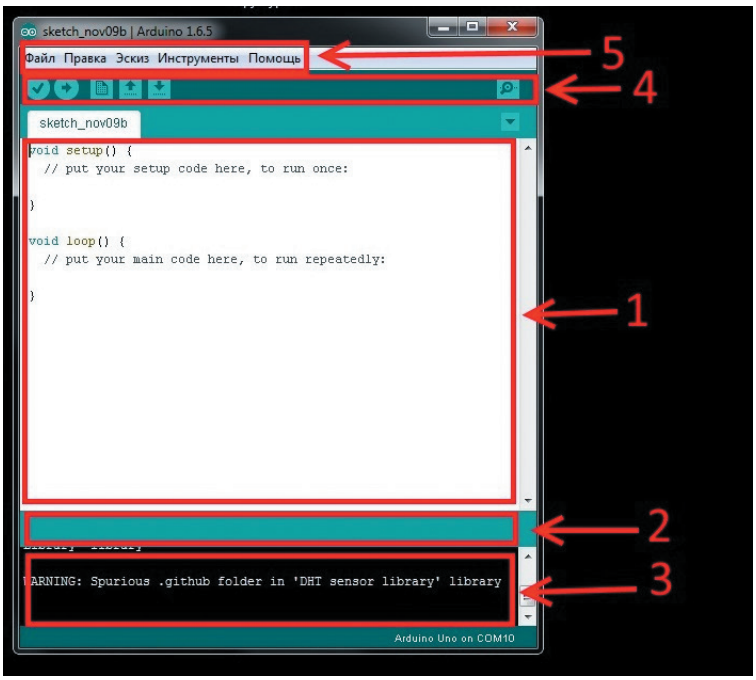


Рис. 04. Окно Arduino IDE

Программа, написанная в среде Arduino, носит название скетч. Скетч пишется в текстовом редакторе, который имеет цветовую подсветку создаваемого программного кода. Во время сохранения и экспорта проекта в области сообщений появляются пояснения и информация об ошибках. Окно вывода текста показывает сообщения Arduino, включающие полные отчеты об ошибках и другую информацию. Кнопки панели инструментов позволяют проверить и записать программу, создать, открыть и сохранить скетч, открыть мониторинг последовательной шины.

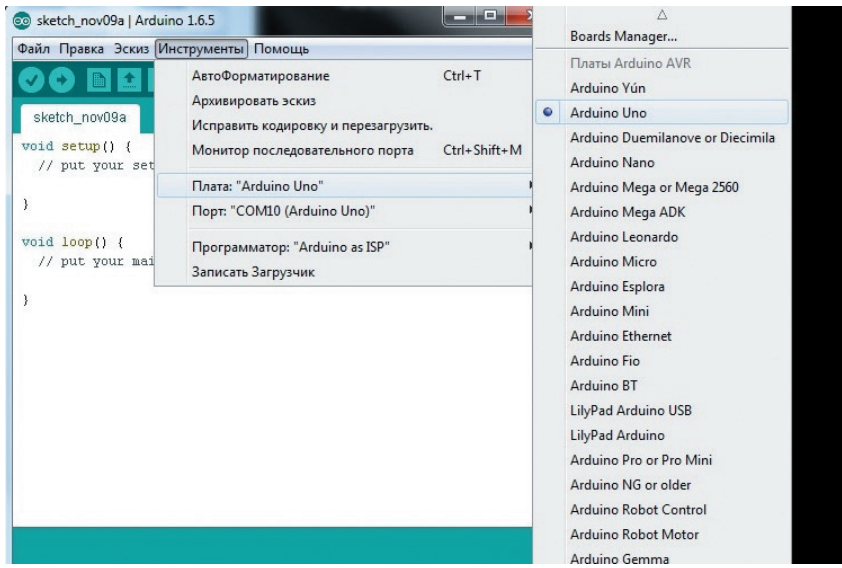


Рис. 05. Arduino IDE — выбор платы

Разрабатываемым скетчам дополнительная функциональность может быть добавлена с помощью библиотек, представляющих собой специальным образом оформленный программный код, реализующий некоторый функционал, который можно подключить к создаваемому проекту. Специализированных библиотек существует множество. Обычно библиотеки пишутся так, чтобы упростить решение той или иной задачи и скрыть от разработчика детали программно-аппаратной реализации. Среда Arduino IDE поставляется с набором стандартных библиотек: Serial, EEPROM, SPI, Wire и др. Они находятся в подкаталоге libraries каталога установки Arduino. Необходимые библиотеки могут быть также загружены с различных ресурсов. Папка библиотеки копируется в каталог стандартных библиотек (подкаталог libraries каталога установки Arduino). Внутри каталога с именем библиотеки находятся файлы *.cpp, *.h. Многие библиотеки снабжаются примерами, расположенными в папке examples. Если библиотека установлена правильно, то она появляется в меню **Эскиз | Импорт библиотек**. Выбор библиотеки в меню приведет к добавлению в исходный код строчки:

#include <имя библиотеки.h>

Эта директива подключает заголовочный файл с описанием объектов, функций и констант библиотеки, которые теперь могут быть использованы в проекте. Среда Arduino будет компилировать создаваемый проект вместе с указанной библиотекой.

Перед загрузкой скетча требуется задать необходимые параметры в меню **Инструменты | Плата** — как показано на рис. 05, и **Инструменты | Последовательный порт**— показано на рис. 3.

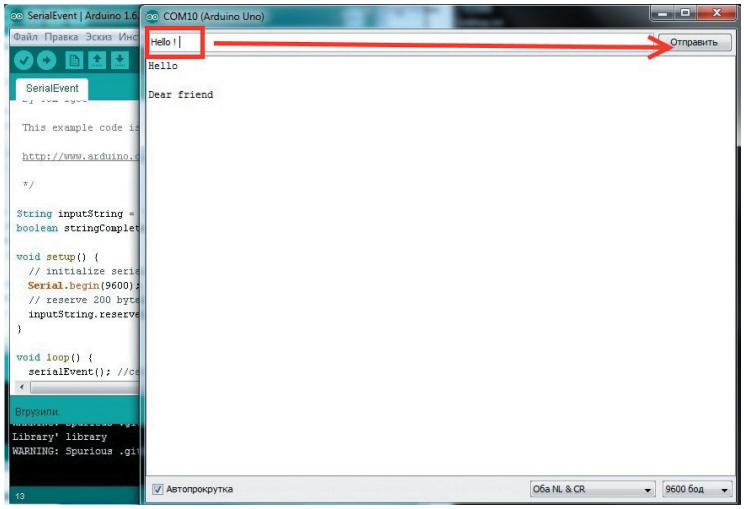


Рис. 06. Arduino IDE — монитор последовательного порта

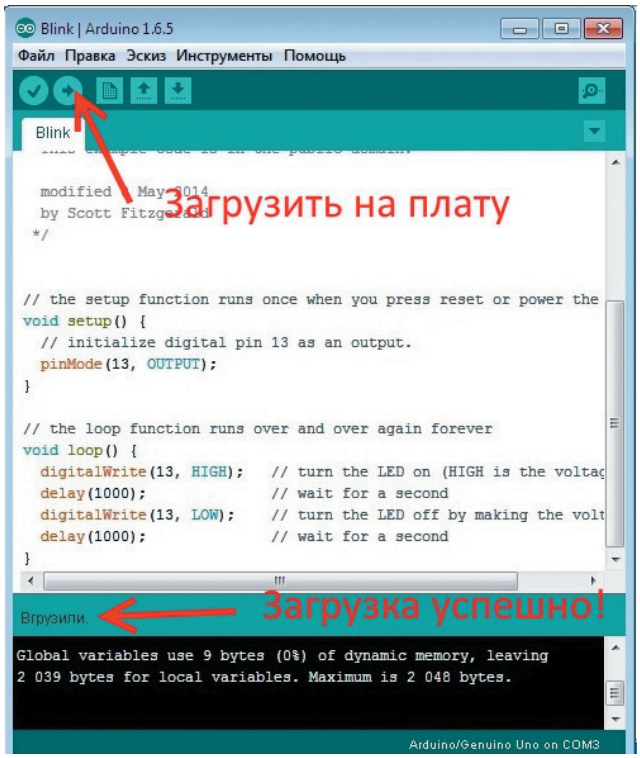


Рис. 07. v Загрузка скетча на плату Arduino

Современные платформы Arduino перезагружаются автоматически перед загрузкой. На старых платформах необходимо нажать кнопку перезагрузки. На большинстве плат во время процесса загрузки будут мигать светодиоды RX и TX.

При загрузке скетча используется загрузчик (bootloader) Arduino — небольшая программа, загружаемая в микроконтроллер на плате. Она позволяет загружать программный код без использования дополнительных аппаратных средств. Работа загрузчика распознается по миганию светодиода на цифровом выводе D13.

Монитор последовательного порта (Serial Monitor) отображает данные, посылаемые в платформу Arduino (плату USB или плату последовательной шины). Для отправки данных необходимо ввести в соответствующее поле текст и нажать кнопку **Послать** (Send) или клавишу <Enter> (рис. 06). Затем следует из выпадающего списка выбрать скорость передачи, соответствующую значению Serial.begin в скетче. На ОС Mac или Linux при подключении мониторинга последовательной шины платформа Arduino будет перезагружена (скетч начнется сначала).

Текст программы (скетч) пишется в окне редактора программного кода. В программе обязательно должны быть две записи, void setup() и void loop() (см. рис. 07) – это так называемые функции, первая выполняется единоразово, при подаче питания на Arduino, а вторая выполняется циклически до тех пор, пока присутствует питание микроконтроллера. В функцию setup() записываются различные настройки микроконтроллера для дальнейшей работы — например, это может быть конфигурация портов ввода/вывода, либо инициализация подключенного вами дисплея или датчика. Главное, что нужно запомнить, с этой функции начинается работа микроконтроллера и все, что в ней написано, выполняется только один раз. Функция loop() выполняется сразу же после функции setup(), и после этого микроконтроллер постоянно работает в ней.

Теперь загрузим на плату Arduino какой-нибудь скетч. Мы можем найти примеры скетчей в пункте меню Файл → Образцы, например **Файл → Образцы → Basics → Blink**. Для загрузки скетча на плату Arduino нажимаем на значок загрузки в панели инструментов (рис. 1.20) и в случае успешной загрузки скетча на плату в окне сообщений появится надпись **Вгрузили** (рис. 07).

Результат работы программы – мигание светодиода, подключенного к цифровому выводу 13.

Установка Arduino IDE для ESP8266

Arduino IDE для ESP8266 позволяет писать скетчи и загружать их одним кликом в ESP8266 в знакомой среде Arduino IDE. Рассмотрим установку Arduino IDE для ESP8266.

Сначала необходимо установить Arduino IDE с официального сайта версии не ниже 1.6.5. Запускаем Arduino IDE. Выбираем пункт **Файл → Настройки** и в поле

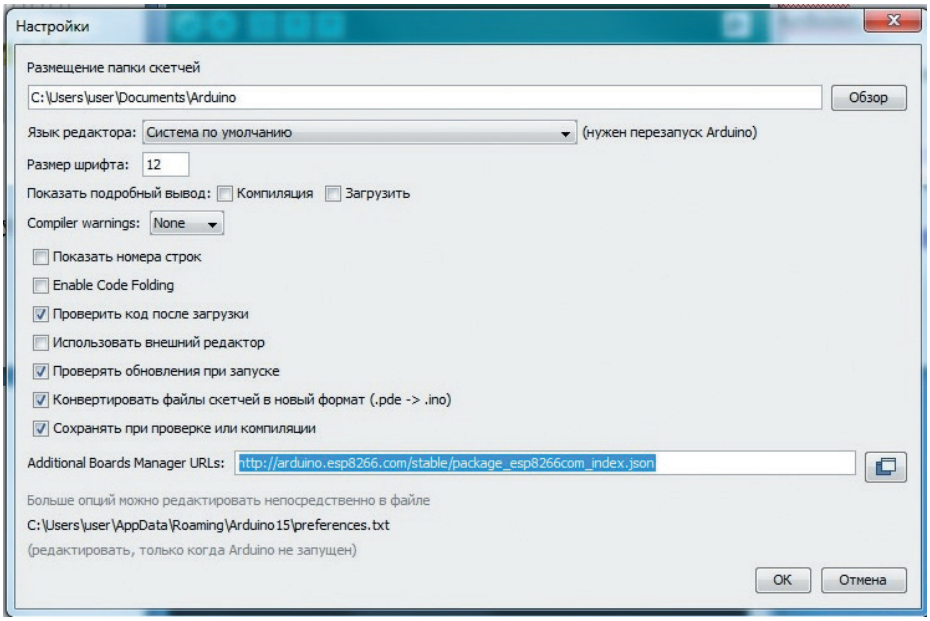


Рис. 08. Ввод адреса для скачивания Arduino IDE для ESP8266

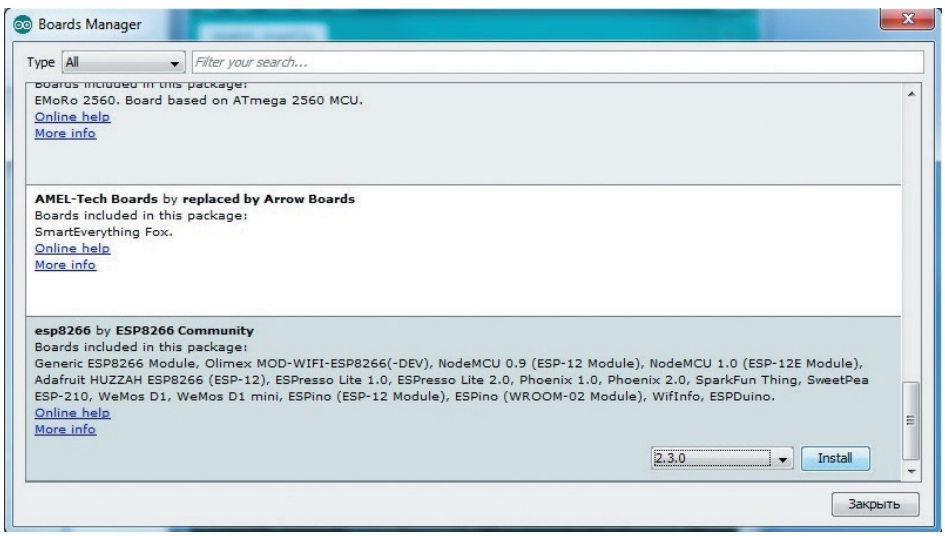


Рис. 09. Загрузка Arduino IDE для ESP8266

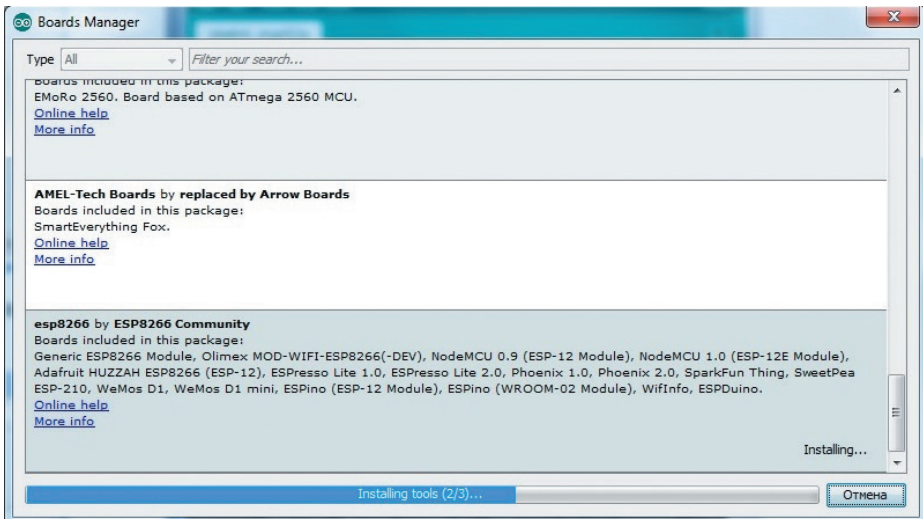


Рис. 010. Загрузка Arduino IDE для ESP8266

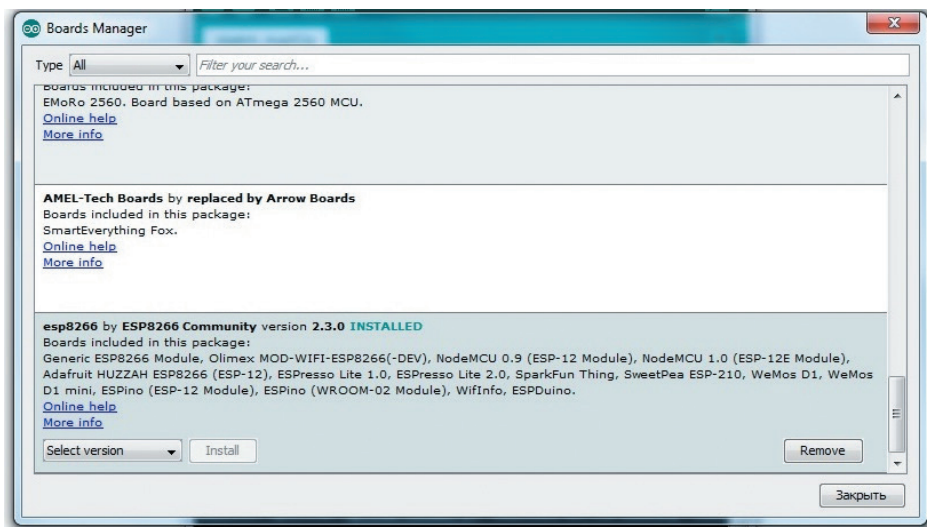


Рис. 011. Загрузка Arduino IDE для ESP8266

Additional Boards Manager URLs вводим **http://arduino.esp8266.com/stable/package_esp8266com_index.json.** Нажимаем **OK** (см. рис. 08).

Выбираем пункт **Инструменты → Плата → BoardsManager** и в списке ищем плату ESP8266. Выбираем этот пункт, версию и нажимаем на **Install** (см. рис. 09, 010, 011).

После загрузки программного обеспечения в списке плат (**Инструменты → Плата →**) появятся платы ESP8266 (см. рис. 012).

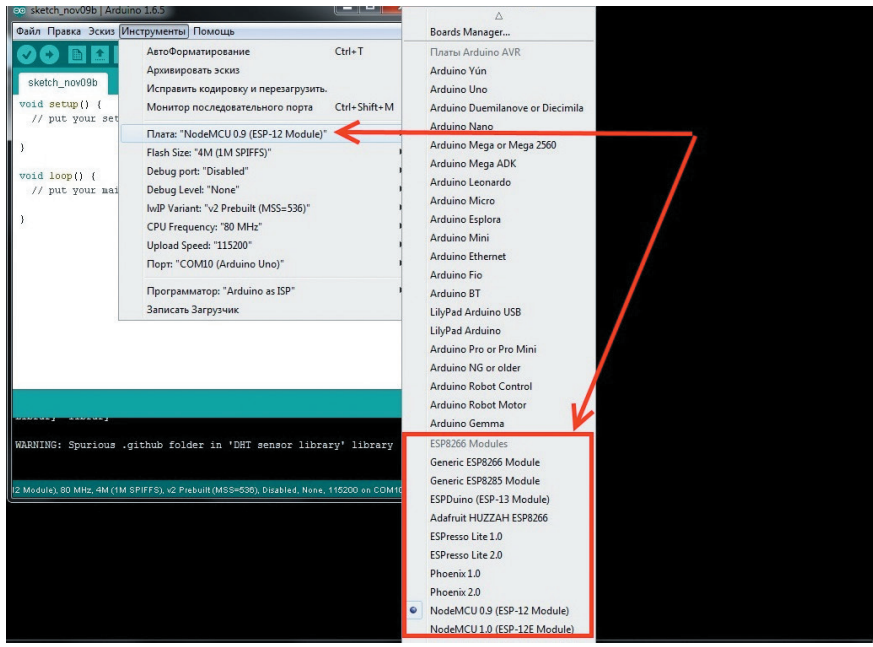


Рис. 012. Выбор платы ESP8266

Эксперимент 1.
Светодиодный маячок на 4 светодиодах

Здесь мы познакомимся с платой Arduino,
узнаем о режимах работы цифровых контактов,
научимся правильно подключать к Arduino светодиоды
и создадим первую программу
светодиодного маячка на 4 светодиодах.

В эксперименте мы будем использовать следующие компоненты:

- ❑ Плата Arduino + WiFi – 1;
- ❑ Кабель USB;
- ❑ Плата прототипирования – 1;
- ❑ Светодиод красный - 4;
- ❑ Резистор 220 Ом – 4;
- ❑ Провода ММ – 5.

ПЕРЕКЛЮЧАТЕЛЬ						
1	2	3	4	5	6	7
OFF	*OFF*	*ON*	*ON*	*OFF*	*OFF*	*OFF*

Итак, установите переключатели на плате Arduino+WiFi следующим образом:

В результате получаем обычную Arduino UNO. Данное положение переключателей мы и будем использовать в большинстве экспериментов.

Arduino UNO представляет собой плату, с размещенными на ней компонентами, главным из которых является микроконтроллер Atmel AVR. Он является основной вычислительной системой этой платформы, поскольку именно для него и создается программное обеспечение, с помощью которого микроконтроллер взаимодействует с внешним миром посредством специальных портов ввода/вывода данных. Плата Arduino UNO имеет 14 цифровых вход/выходов (6 из которых могут использоваться как выходы ШИМ), 6 аналоговых входов, кварцевый генератор 16 МГц, разъем USB, силовой разъем, разъем ICSP и кнопку перезагрузки. Питание платы Arduino UNO при помощи адаптера AC/DC (рекомендуемое напряжение 7-12В), либо от компьютера посредством кабеля USB. Микроконтроллер ATmega328 располагает 32 кБ флэш-памяти, из которых 0.5 кБ используется для хранения загрузчика, а также 2 кБ ОЗУ (SRAM) и 1 Кб энергонезависимой памяти EEPROM.

Цифровые выводы на платах Arduino позволяют подключать к Arduino датчики, приводы и другие микросхемы. Изучение того, как использовать их, позволит вам использовать Arduino для выполнения практических полезных вещей.

Цифровые сигналы имеют только два отдельных значения: высокий (HIGH, 1) и низкий (LOW, 0) уровни. Вы можете использовать цифровые сигналы в ситуациях, где вход или выход будет принимать одно из этих двух значений. Поскольку цифровые выводы Arduino могут использоваться в качестве и входа, и выхода, сначала необходимо их настроить. Для настройки цифровых выводов в Arduino используется встроенная функция pimMode(), которая имеет следующий синтаксис:

pinMode(pin, mode)

где

- pin – номер вывода Arduino;
- mode – устанавливаемый режим для вывода pin:
  - INPUT – pin в режиме входа;
  - OUTPUT – pin в режиме выхода;
  - INPUT_PULLUP – в этом режиме к выводу подключается внутренний подтягивающий резистор 20 кОм, чтобы привести уровень на выводе к значению HIGH, если к нему ничего не подключено.

В данном эксперименте мы будем использовать выводы Arduino в режимы выходов (OUTPUT), для включения и выключения светодиодов. Светодиод – это полупроводниковый прибор, преобразующий электрический ток непосредственно в световое излучение. Цветовые характеристики светодиодов зависят от химического состава использованного в нем полупроводника. Светодиод излучает в узкой части спектра, его цвет чист, что особенно ценят дизайнеры. Светодиоды поляризованы, имеет значение, в каком направлении подключать их. Положительный вывод светодиода (более длинный) называется анодом, отрицательный – катодом. Как и все диоды, светодиоды позволяют току течь только в одном направлении – от анода к катоду. Поскольку ток протекает от положительного к отрицательному, анод светодиода должен быть подключен к цифровому сигналу, а катод должен быть подключен к земле.

Собираем схему согласно рис. 1.1.

В схеме подключения светодиодов к цифровым выходам мы используем ограничительный резистор номиналом 220 Ом. Рассмотрим, как подобрать ограничительный резистор и как будет влиять номинал резистора на яркость светодиода.

Самым главным уравнением для любого инженера-электрика является закон Ома. Закон Ома определяет отношения между напряжением, током и сопротивлением в цепи. Закон Ома определяется следующим образом:

$$V = I \times R,$$

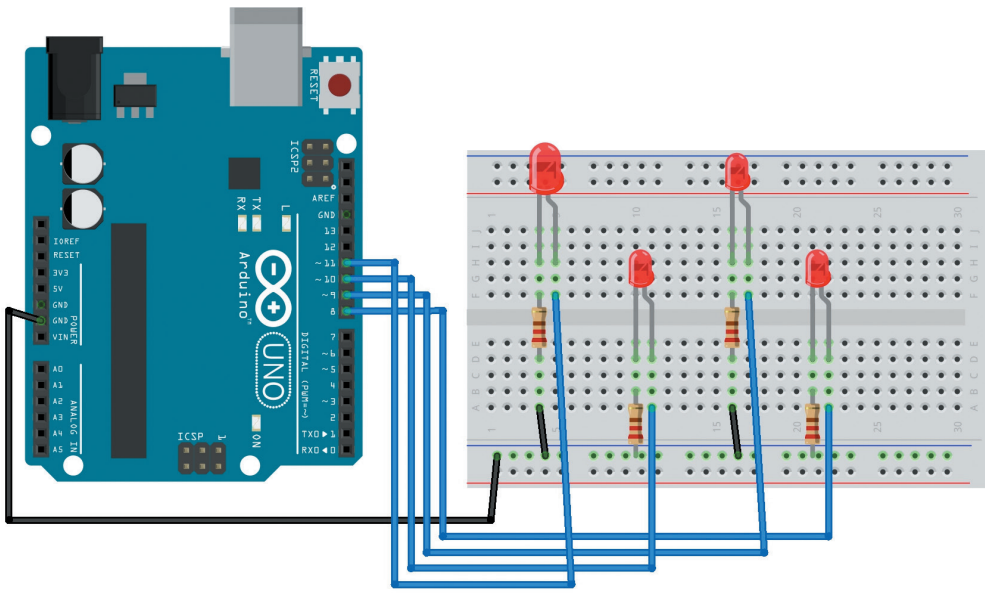


Рис. 1.1. Схема соединений для эксперимента

где V – напряжение в вольтах; I – ток в амперах; R – сопротивление в омах.

В электрической схеме каждый компонент имеет некоторое сопротивление, что снижает напряжение. Светодиоды имеют предопределенное падение напряжения на них и предназначены для работы в определенном значении тока. Чем больше ток через светодиод, тем ярче светодиод светится, до предельного значения. Для наиболее распространенных светодиодов максимальный ток составляет 20 мА. Обычное значение падения напряжения для светодиода – около 2 В. Напряжение питания 5 В должно упасть на светодиоде и резисторе, поскольку доля светодиода 2 В оставшиеся 3 В должны упасть на резисторе. Зная максимальное значение прямого тока через светодиод (20 мА), можете найти номинал резистора.

$$R = V/I = 3/0{,}02 = 150 \text{ Ом}$$

Таким образом, со значением резистора 150 Ом ток 20 мА протекает через резистор и светодиод. По мере увеличения значения сопротивления ток будет уменьшаться. 220 Ом немного более, чем 150 Ом, но все же позволяет светиться светодиоду достаточно ярко, и резистор такого номинала очень распространен.

Приступим к написанию программы (скетча).

Светодиоды должны одновременно мигать с определенной частотой. В процедуре setup() настроим режим работы контактов (пинов), к которым подключены светодиоды, как OUTPUT (выход)

```
void setup() {
   pinMode(8,OUTPUT);
   pinMode(9,OUTPUT);
   pinMode(10,OUTPUT);
   pinMode(11,OUTPUT);
}
```

В процедуре loop() сначала зажигаем все светодиоды (подаем на выводы 8,9,10,11 сигнал HIGH), ждем некоторое время (время "горения"), затем "тушим" светодиоды (подаем на выводы 8,9,10,11 сигнал LOW) и опять ждем некоторое время и т.д. по кругу. Для выполнения операции подождать "некоторое время" мы будем использовать встроенную Arduino функцию delay(). Эта функция просто останавливает выполнение программы на заданное в параметре количество миллисекунд (1000 миллисекунд в 1 секунде). Например:

delay(2000); // останавливает выполнение программы на 2000 мсек (

И весь код программы в листинге 1.1. Скачать данный скетч можно на сайте Arduino-kit по ссылке https://arduino-kit.ru/scetches/exp_01_01.

Листинг 1.1

```
void setup() {
   // настроить выводы 8, 9, 10, 11 Arduino как OUTPUT
   pinMode(8,OUTPUT);
   pinMode(9,OUTPUT);
   pinMode(10,OUTPUT);
   pinMode(11,OUTPUT);
}

void loop() {
  // включить светодиоды
  digitalWrite(8, HIGH);
  digitalWrite(9, HIGH);
  digitalWrite(10, HIGH);
  digitalWrite(11, HIGH);
  // пауза 1000 мсек (1 сек)
  delay(1000);
  // выключить светодиоды
  digitalWrite(8, LOW);
  digitalWrite(9, LOW);
  digitalWrite(10, LOW);
  digitalWrite(11, LOW);
  // пауза 1000 мсек (1 сек)
  delay(1000);
}
```

Загрузим данный скетч на плату Arduino. Вы должны наблюдать включение/выключение светодиодов с частотой 2 секунды.