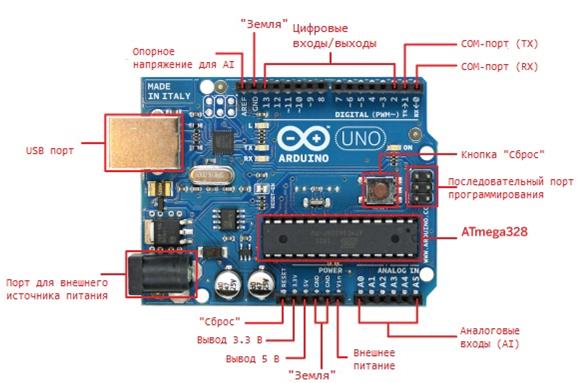
13. Плата и распиновка Arduino Uno

Arduino/Genuino Uno-это плата микроконтроллера на базе ATmega328P (datasheet). Он имеет 14 цифровых входов/выходов (из которых 6 могут использоваться в качестве ШИМ-выходов), 6 аналоговых входов, кварцевый кристалл 16 МГц, USB-соединение, разъем питания, заголовок ICSP и кнопку сброса. Arduino UNO самая функциональная плата со стандартным набором узлов. Очень удобна при подключении к ней уже готовых узлов и плат расширения.



**Микроконтроллер ATmega328P**

Сердцем платформы Arduino Uno является восьмибитный микроконтроллер семейства AVR — ATmega328P.

**Микроконтроллер ATmega16U2**

Микроконтроллер ATmega16U2 обеспечивает связь микроконтроллера ATmega328P с USB-портом компьютера. При подключении к ПК Arduino Uno определяется как виртуальный COM-порт. Прошивка микросхемы 16U2 использует стандартные драйвера USB-COM, поэтому установка внешних драйверов не требуется.

Распиновка выглядит следующим образом☹(КАРТИНКА ВНИЗУ)

1) Последовательный интерфейс использует шины №0 (RX – получение данных), №1 (TX – передача данных).

2) Для внешнего прерывания используются выводы №2, №3.

3) Для ШИМ используются выводы за номерами 3,5, 6, 9, 10, 11. Функция analog Write обеспечивает разрешение в 8 бит.

4) Связь посредством SPI: контакты №10 (SS), №11 (MOSI), №12 (MISO), №13 (SCK).

5) Вывод №13 запитывает светодиод, который загорается при высоком потенциале.

6) Uno оснащена 6 аналоговыми входами (A0 – A5), которые имеют разрешение в 10 бит.

7) Для изменения верхнего предела напряжения используется вывод AREF (функция analog Reference).

8) Связь I2C (TWI, библиотека Wire) осуществляется через выводы №4 (SDA), №5 (SCL).

9) Вывод Reset – перезагрузка микроконтроллера.

**Пины питания**

**VIN**: Напряжение от внешнего источника питания (не связано с 5 В от USB или другим стабилизированным напряжением). Через этот вывод можно как подавать внешнее питание, так и потреблять ток, если к устройству подключён внешний адаптер.

**5V**: На вывод поступает напряжение 5 В от стабилизатора платы. Данный стабилизатор обеспечивает питание микроконтроллера ATmega328. Запитывать устройство через вывод 5V не рекомендуется — в этом случае не используется стабилизатор напряжения, что может привести к выходу платы из строя.

**3.3V**: 3,3 В от стабилизатора платы. Максимальный ток вывода — 50 мА.

**GND**: Выводы земли.

**IOREF:** Вывод предоставляет платам расширения информацию о рабочем напряжении микроконтроллера. В зависимости от напряжения, плата расширения может переключиться на соответствующий источник питания либо задействовать преобразователи уровней, что позволит ей работать как с 5 В, так и с 3,3 В устройствами.

**Порты ввода/вывода**

Цифровые входы/выходы: пины 0–13 – изображены на рисунке 44.5

Логический уровень единицы — 5 В, нуля — 0 В.

Максимальный ток выхода — 40 мА. К контактам подключены подтягивающие резисторы, которые по умолчанию выключены, но могут быть включены программно.

На плате выведены 14 цифровых пинов (контактов), любой из которых может работать как на вывод информации, так и на ввод. Для этого в коде программ применяются специальные функции: ХЗ, надо ли это

pinMode()

Функция pinMode служит для задания режима работы контакта, будет-ли он работать на выход или на вход. В данной функции задается номер контакта, которым мы в дальнейшем собираемся управлять.

digitalRead()

Функция считывает текущее значение с заданного контакта – его значение может быть HIGH или LOW.

digitalWrite()

Функция передает определенное значение на заданный контакт – оно может быть HIGH или LOW.

Все выводы обладают пятивольтовой логикой, то есть выдают логическую единицу как напряжение 5В.

Каждый вывод платы имеет нагрузочный резистор номиналом 20-50 кОм и может пропускать до 40 мА, но по умолчанию все они отключены.

Стандартная конфигурация выводов микроконтроллера позволяет не использовать функцию pinMode(), так как они изначально настроены в качестве портов ввода. Их особенностью является минимальная нагрузка на схему: порты эквивалентны внутреннему резистору, что гарантирует сопротивление в 100 МОм.

Для трансформации порта в один из двух режимов достаточно минимум тока. Благодаря этому, выходы используют для подключения датчиков касания, фотодиодов и других датчиков, имеющих схему аналогичной RC-цепи.

Если к определенному выводу ничего не подключено, то из-за помех, случайных контактов на нем могут возникать случайные величины.

Рекомендуется задавать портам, к которым ничего не подключено, определенное известное значение. Для этого используют подтягивающие резисторы 10 кОм, подключая вход к +5В или к земле.

14. Широтно-импульсная модуляция

Для вывода данных с Arduino традиционно применяется двоичная система с применением логических значений 0 и 1. С этой задачей превосходно справляется управление портами вывода, имеющееся у большинства микроконтроллеров от производителей микросхем. Но не всегда схемотехнику необходимо значение нуля или единицы.

С помощью PWM (ШИМ), изображенного на рисунке 44.6, позволяет создать сигнал, который имеет определенное напряжение и продолжительность. В зависимости от временного промежутка между импульсами на выходе выдается нужное значение, так называемый параметр скважности импульса. **Скважность** – это один из классификационных признаков импульсных систем, определяющий отношение его периода следования (повторения) к длительности импульса.

Микроконтроллер Arduino поддерживает восьмибитную ШИМ, что позволяет выбрать переменную в широком диапазоне значений от 0 до 256.

В качестве примера использования ШИМ можно использовать плавное включение и выключение светодиода, подключенному к 3 пину. При этом постепенно подается напряжение от 0 до 5В. Как видно на картинке ниже, при разном значении переменной мы получаем разное напряжение для светодиода: 5V, 2.5V, 1,25V, 3.75V.

Так же с помощью пьезоизлучателя и ШИМ можно воспроизводить звуки неплохого качества.

**АЦП:** пины A0–A5

АНАЛОГОВЫЕ ВХОДЫ: КОНТАКТЫ A0, A1, A2, A3, A4, A5

6 аналоговых входов, каждый из которых может представить аналоговое напряжение в виде 10-битного числа (1024 значений). Разрядность АЦП — 10 бит.

Arduino Uno имеет на своей платформе 6 аналоговых входов с разрешением 10 Бит на каждый вход. Данное разрешение говорит нам о том, что сигнал, приходящий на него, можно оцифровать в диапазоне от 0 до 1024 условных значений.

Считывать значения с данных контактов можно функцией analogRead(), а передавать значения – функцией analogWrite().

Так как Arduino Uno обладает пятивольтовой логикой, то и значение будет находиться в диапазоне от 0 до 5 вольт, однако при помощи функции analogReference() можно изменять верхний предел.

**TWI/I²C:** пины SDA и SCL

Для общения с периферией по синхронному протоколу, через 2 провода. При помощи данных контактов к Arduino можно подключать внешние цифровые устройства, умеющие общаться по протоколу I2C. Для реализации интерфейса в среде Arduino IDE присутствует библиотека Wire.

**SPI:** пины 10(SS), 11(MOSI), 12(MISO), 13(SCK).

Через эти пины осуществляется связь по интерфейсу SPI. Для работы — используйте библиотеку SPI. С помощью данных контактов происходит подключение периферии, работающей через интерфейс SPI.

(И ИНФА В ВОПРОСЕ 4)

15. Среда разработки Arduino IDE

Arduino – контроллер является перепрограммируемым и настраиваемым на конкретную задачу пользователя.

Среда разработки Arduino IDE состоит из набора специфичных библиотек и драйверов, необходимых для корректной работы с контроллером. Установив программу и запустив ее, будет интерфейс, похожий на простой текстовый блокнот, но с подсветкой синтаксиса. В этот текстовый блокнот и пишется программа для контроллера.

С помощью встроенных библиотек возможно подключить дополнительный датчик или дисплей к контроллеру. Имеется возможность проверить код на наличие ошибок. Программа укажет точное место ошибки в коде или где были введены некорректные данные. Если их не исправить, то программа просто не сможет скомпилировать код, и не удастся залить скетч в контроллер.

Прежде чем заливать программу в контроллер, необходимо выбрать порт подключения и модель платы Arduino. Если используется программатор, то на вкладке нужно выбрать способ закачки программы через данное устройство.

Arduino IDE имеет предустановленные скетчи для демонстрации возможности устройства. Это помогает удостовериться, что плата не повреждена и что на нее можно залить программу. Кроме того, базовые примеры позволят понять принцип программирования. В базовые примеры входят простые скетчи управления выводами.

Самый элементарный скетч позволяет управлять светодиодом, расположенным на плате. Можно подкорректировать данные и заставить светодиод моргать чаще или медленнее, или задать, например, такт мигания каждые две секунды и при этом сделать само горение светодиода на полсекунды. Также можно посмотреть работу программы аналоговых и цифровых выводов, простых и сенсорных дисплеев, сделать проверку различных звуковых элементов и т.д.

Среда разработки Arduino состоит из встроенного текстового редактора программного кода, области сообщений, окна вывода текста(консоли), панели инструментов с кнопками часто используемых команд и нескольких меню. Для загрузки программ и связи среда разработки подключается к аппаратной части Arduino.

Программа, написанная в среде Arduino, называется скетч. Скетч пишется в текстовом редакторе, имеющем инструменты вырезки/вставки, поиска/замены текста. Во время сохранения и экспорта проекта в области сообщений появляются пояснения, также могут отображаться возникшие ошибки. Окно вывода текста(консоль) 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

16.Программирование в Arduino. Цифровые выводы. Аналоговые входы. Широтно-импульсная модуляция

Цифровые выводы

Выводы платформы Arduino могут работать как входы или как выходы. Также ана­логовые входы Arduino (ATmega) могут конфигурироваться и работать так же, как и цифровые порты ввода/вывода.

Выводы Arduino настроены как порты ввода, поэтому не требуется декларации в функции pinMode(). Сконфигурированные порты ввода находятся в высокоимпедансном состоянии. Это означает, что порт ввода дает слишком малую нагрузку на схему, в которую он включен. Для перевода порта ввода из одного состояния в дру­гое требуется маленькое значение тока. Если к выводу ничего не подключено, то значения на нем будут принимать случайные величины, наводимые электрически­ми помехами.

Если на порт ввода не поступает сигнал, то рекомендуется задать порту известное состояние. Это делается добавлением подтягивающих резисторов 10 кОм, подклю­чающих вход либо к питанию +5 В, либо к земле.

Микроконтроллер ATmega имеет программируемые встроенные подтягивающие резисторы 20 кОм. Программирование данных резисторов осуществляется так:

pinMode(pin, INPUT); // назначить выводу порт ввода

digitalWrite(pin, HIGH); // включить подтягивающий резистор

Выводы, сконфигурированные как порты вывода находятся в низкоимпедансном состоянии. Данные выводы могут пропускать через себя достаточно большой ток. Выводы микросхемы ATmega могут быть источником тока до 40 мА. Такого значе­ния тока недостаточно для большинства реле, соленоидов и двигателей.

Короткие замыкания выводов Arduino или попытки подключить энергоемкие уст­ройства могут повредить выходные транзисторы вывода или весь микроконтроллер ATmega.

**Аналоговые входы**

Микроконтроллеры ATmega, используемые в Arduino, содержат шестиканальный аналого-цифровой преобразователь (АЦП). Разрешение преобразователя составляет 10 битов, что позволяет на выходе получать значения от 0 до 1023.

Аналоговые входы могут использоваться как цифровые выводы портов ввода/вывода, при этом они имеют номера от 14 до 19:

pinMode(14,OUTPUT); digitalWrite(14, HIGH);

Для вывода, работавшего ранее как цифровой порт вывода, команда analogRead будет работать некорректно. В этом случае рекомендуется сконфигурировать его как аналоговый вход.

**Широтно-импульсная модуляция**

Широтно-импульсная модуляция (ШИМ) — это операция получения изменяюще­гося аналогового значения посредством цифровых устройств. Подавая на выход сигнал, состоящий из высоких и низких уровней, мы моделируем напряжение между максимальным значением (5 В) и минимальным (0 В). Длительность включения максимального значения называется шириной импульса. Для получения различных аналоговых величин изменяется ширина импульса. В результате на выходе будет величина напряжения, равная площади под импульсами, показанные на рисунке

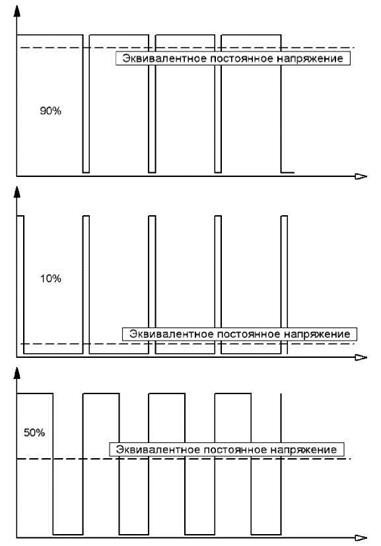


Рисунок 46.1 – Широтно-импульсная модуляция

Вызов функции analogWrite() с масштабом 0-255 означает, что значение analogWrite(255) будет соответствовать 5 В (100 % рабочий цикл — постоянное включение 5 В), а значение analogWrite(127) — 2,5 В (50 % рабочий цикл).

17.Память в Arduino

В микроконтроллерах ATmega168, ATmega328, ATmega1280, ATmega2560, исполь­зуемых на платформах Arduino, существует три вида памяти:

- флеш-память — используется для хранения скетчей;

- ОЗУ (статическая оперативная память) — служит для хранения и работы пере­менных;

- EEPROM (энергонезависимая память) — применяется для хранения постоянной информации.

Флеш-память и EEPROM являются энергонезависимыми видами памяти (дан­ные сохраняются при отключении питания). ОЗУ является энергозависимой па­мятью.

Микроконтроллер ATmega168 имеет:

-  16 Кбайт флеш-памяти (2 Кбайт используется для хранения загрузчика);

-  1024 байта ОЗУ;

-  512 байт EEPROM.

Для ATmega328 эти показатели следующие:

-  32 Кбайт флеш-памяти (2 Кбайт используется для хранения загрузчика);

-  2 Кбайт ОЗУ;

-  1024 байт EEPROM.

Для ATmega1280 эти показатели следующие:

-  128 Кбайт флеш-памяти (2 Кбайт используется для хранения загрузчика);

- 8 Кбайт ОЗУ;

-  4096 байт EEPROM.

Для ATmega2560 эти показатели следующие:

-  256 Кбайт флеш-памяти (2 Кбайт используется для хранения загрузчика);

-  16 Кбайт ОЗУ;

-  9182 байт EEPROM.

При отсутствии свободного места в ОЗУ могут произойти сбои программы.

18.Структура программы Ардуино.

Структура программы Ардуино состоит из двух частей setup() и loop().

**void setup()** {

// код выполняется один раз при запуске программы

**}**

**void loop() {**

// основной код, выполняется в цикле

**}**

Функция setup() выполняется один раз, при включении питания или сбросе контроллера. Функция должна присутствовать в программе, даже если в ней ничего нет. Перед функцией setup() идет объяв­ление переменных, подключение библиотек. Она используется для инициализации переменных, установки режима работы портов и прочих подгото­вительных для основного цикла программы действий.

После завершения setup() управление переходит к функции loop(). Она в бесконечном цикле выполняет команды, записанные в ее теле (между фигурными скобками). Собственно эти команды и совершают все алгоритмические действия контроллера.

Пример простейшей программы.

**Void setup()**

{

Serial.begin(9600);

}

**Void loop()**

{

Serial.println(millis()); delay(1000);

}