Практическая работа 18

Пантограф

В этом эксперименте мы вращаем сервопривод на угол, задаваемый потенциометром.

Сервоприводы

Под сервоприводом чаще всего понимают механизм с электромотором, который можно попросить повернуться в заданный угол и удерживать это положение. Однако, это не совсем полное определение.

Если сказать полнее, сервопривод — это привод с управлением через отрицательную обратную связь, позволяющую точно управлять параметрами движения. Сервоприводом является любой тип механического привода, имеющий в составе датчик (положения, скорости, усилия и т.п.) и блок управления приводом, автоматически поддерживающий необходимые параметры на датчике и устройстве согласно заданному внешнему значению.

Иными словами:

Сервопривод получает на вход значение управляющего параметра. Например, угол поворота

Блок управления сравнивает это значение со значением на своём датчике

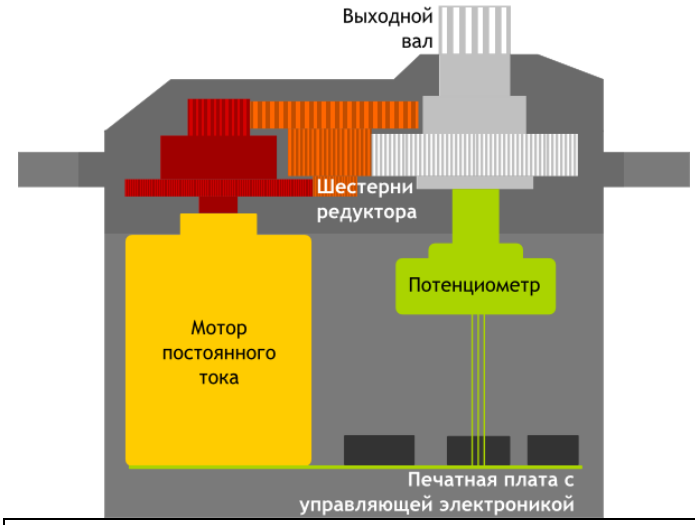
На основе результата сравнения привод производит некоторое действие, например: поворот, ускорение или замедление так, чтобы значение с внутреннего датчика стало как можно ближе к значению внешнего управляющего параметра

Наиболее распространены сервоприводы, которые удерживают заданный угол и сервоприводы, поддерживающие заданную скорость вращения.

Типичный хобби-сервопривод изображён ниже.

Устройство сервопривода

Сервоприводы имеют несколько составных частей.



Привод — электромотор с редуктором. Чтобы преобразовать электричество в механический поворот, необходим электромотор. Однако зачастую скорость вращения мотора бывает слишком большой для практического использования. Для понижения скорости используется редуктор: механизм из шестерней, передающий и преобразующий крутящий момент.

Включая и выключая электромотор, можно вращать выходной вал — конечную шестерню сервопривода, к которой можно прикрепить нечто, чем мы хотим управлять. Однако, для того чтобы положение контролировалось устройством, необходим датчик обратной связи — энкодер, который будет преобразовывать угол поворота обратно в электрический сигнал. Для этого часто используется потенциометр. При повороте бегунка потенциометра происходит изменение его сопротивления, пропорциональное углу поворота.

Таким образом, с его помощью можно установить текущее положение механизма.

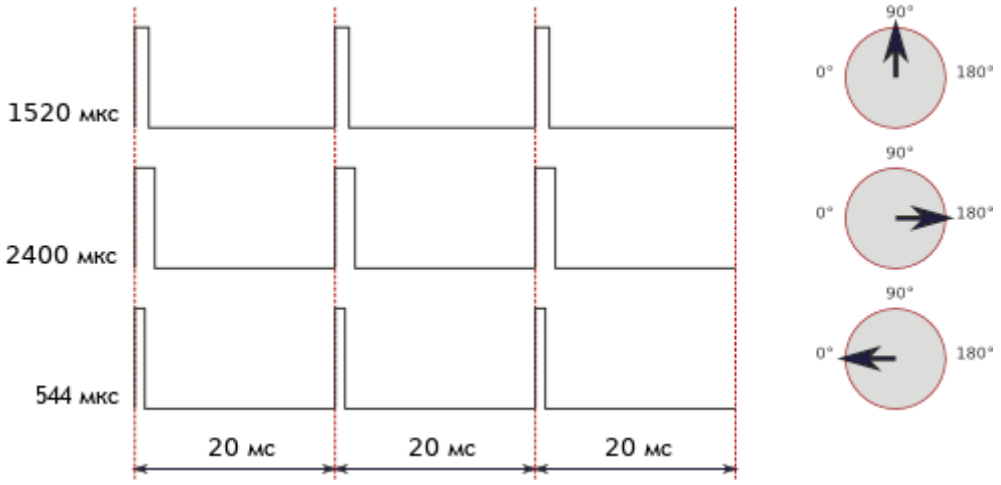
Кроме электромотора, редуктора и потенциометра в сервоприводе имеется электронная начинка, которая отвечает за приём внешнего параметра, считывание значений с потенциометра, их сравнение и включение/выключение мотора. Она-то и отвечает за поддержание отрицательной обратной связи.

К сервоприводу тянется три провода. Два из них отвечают за питание мотора, третий доставляет управляющий сигнал, который используется для выставления положения устройства.

Теперь давайте посмотрим, как управлять сервоприводом извне.

Управление сервоприводом. Интерфейс управляющих сигналов

Чтобы указать сервоприводу желаемое положение, по предназначенному для этого проводу необходимо посылать управляющий сигнал. Управляющий сигнал — импульсы постоянной частоты и переменной ширины.



То, какое положение должен занять сервопривод, зависит от длины импульсов. Когда сигнал поступает в управляющую схему, имеющийся в ней генератор импульсов производит свой импульс, длительность которого определяется через потенциометр. Другая часть схемы сравнивает длительность двух импульсов. Если длительность разная, включается электромотор. Направление вращения определяется тем, какой из импульсов короче. Если длины импульсов равны, электромотор останавливается.

Чаще всего в хобби-сервах импульсы производятся с частотой 50 Гц. Это значит, что импульс испускается и принимается раз в 20 мс. Обычно при этом длительность импульса в 1520 мкс означает, что сервопривод должен занять среднее положение. Увеличение или уменьшение длины импульса заставит сервопривод повернуться по часовой или против часовой стрелки соответственно. При этом существуют верхняя и нижняя границы длительности импульса. В библиотеке Servo для Arduino по умолчанию выставлены следующие значения длин импульса: 544 мкс — для 0° и 2400 мкс — для 180°.

Обратите внимание, что на вашем конкретном устройстве заводские настройки могут оказаться отличными от стандартных. Некоторые сервоприводы используют ширину импульса 760 мкс. Среднее положение при этом соответствует 760 мкс, аналогично тому, как в обычных сервоприводах среднему положению соответствует 1520 мкс.

Также стоит отметить, что это всего лишь общепринятые длины. Даже в рамках одной и той же модели сервопривода может существовать погрешность, допускаемая при производстве, которая приводит к тому, что рабочий диапазон длин импульсов немного отличается. Для точной работы каждый конкретный сервопривод должен быть откалиброван: путём экспериментов необходимо подобрать корректный диапазон, характерный именно для него.

На что ещё стоит обратить внимание, так это на путаницу в терминологии. Часто способ управления сервоприводами называют PWM/ШИМ (Pulse Width Modulation) или PPM (Pulse Position Modulation). Это не так, и использование этих способов может даже повредить привод. Корректный термин — PDM (Pulse Duration Modulation). В нём крайне важна длина импульсов и не так важна частота их появления. 50 Гц — это норма, но сервопривод будет работать корректно и при 40, и при 60 Гц. Единственное, что нужно при этом иметь в виду — это то, что при сильном уменьшении частоты он может

работать рывками и на пониженной мощности, а при сильном завышении частоты (например, 100 Гц) может перегреться и выйти из строя.

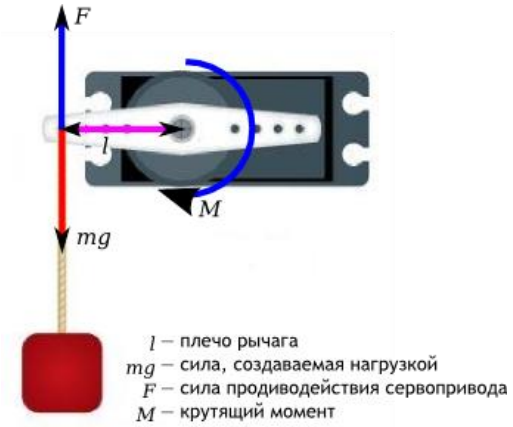
Характеристики сервоприводов

Теперь давайте разберёмся, какие бывают сервоприводы и какими характеристиками они обладают.

Крутящий момент и скорость поворота

Сначала поговорим о двух очень важных характеристиках сервопривода: о крутящем моменте и о скорости поворота.

Момент силы, или крутящий момент — векторная физическая величина, равная произведению радиус-вектора, проведенного от оси вращения к точке приложения силы, на вектор этой силы. Характеризует вращательное действие силы на твёрдое тело.



Проще говоря, эта характеристика показывает, насколько тяжёлый груз сервопривод способен удержать в покое на рычаге заданной длины. Если крутящий момент сервопривода равен 5 кг×см, то это значит, что сервопривод удержит на весу в горизонтальном положении рычаг длины 1 см, на свободный конец которого подвесили 5 кг. Или, что эквивалентно, рычаг длины 5 см, к которому подвесили 1 кг.

Скорость сервопривода измеряется интервалом времени, который требуется рычагу сервопривода, чтобы повернуться на 60°. Характеристика 0,1 с/60° означает, что сервопривод поворачивается на 60° за 0,1 с. Из неё несложно вычислить скорость в более привычной величине, оборотах в минуту, но так

сложилось, что при описании сервоприводов чаще всего используют такую единицу.

Стоит отметить, что иногда приходится искать компромисс между этими двумя характеристиками, так как если мы хотим надёжный, выдерживающий большой вес сервопривод, то мы должны быть готовы, что эта могучая установка будет медленно поворачиваться. А если мы хотим очень быстрый привод, то его будет относительно легко вывести из положения равновесия. При использовании одного и того же мотора баланс определяет конфигурация шестерней в редукторе.

Конечно, мы всегда можем взять установку, потребляющую большую мощность, главное, чтобы её характеристики удовлетворяли нашим потребностям.

Форм-фактор

Сервоприводы различаются по размерам. И хотя официальной классификации не существует, производители давно придерживаются нескольких размеров с общепринятым расположением крепёжных элементов. Их можно разделить на:

- ✓ маленькие
- ✓ стандартные
- ✓ большие

Обладают они при этом следующими характерными габаритами:

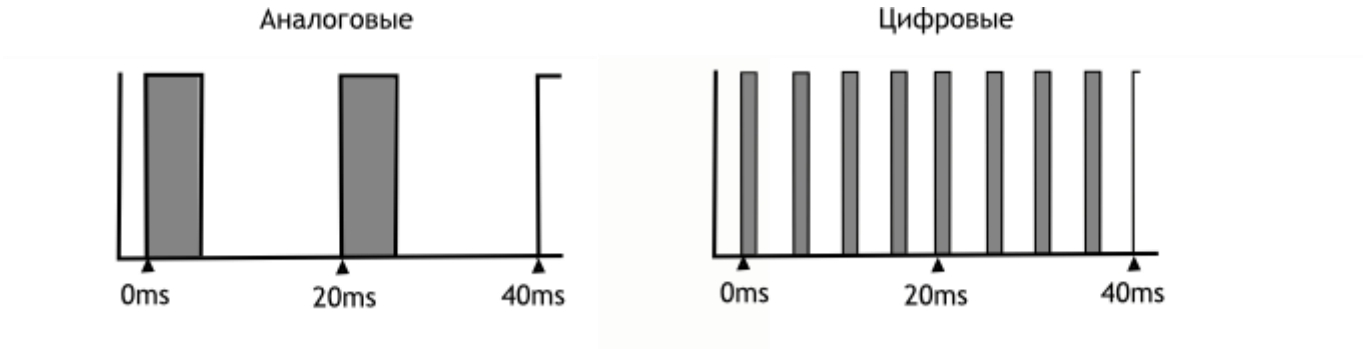
	Вес	Линейные размеры
маленькие	8-25 г	22×15×25 мм
стандартные	40-80 г	40×20×37 мм
большие	50-90 г	49×25×40 мм

Бывают ещё так называемые сервоприводы «специального вида» с габаритами, не попадающими в данную классификацию, однако процент таких сервоприводов весьма мал.

Внутренний интерфейс

Сервоприводы бывают аналоговые и цифровые. Так в чём же их отличия, достоинства и недостатки?

Внешне они ничем не отличаются: электромоторы, редукторы, потенциометры у них одинаковые, различаются они лишь внутренней управляющей электроникой. Вместо специальной микросхемы аналогового сервопривода у цифрового собрата можно заметить на плате микропроцессор, который принимает импульсы, анализирует их и управляет мотором. Таким образом, в физическом исполнении отличие лишь в способе обработки импульсов и управлении мотором.



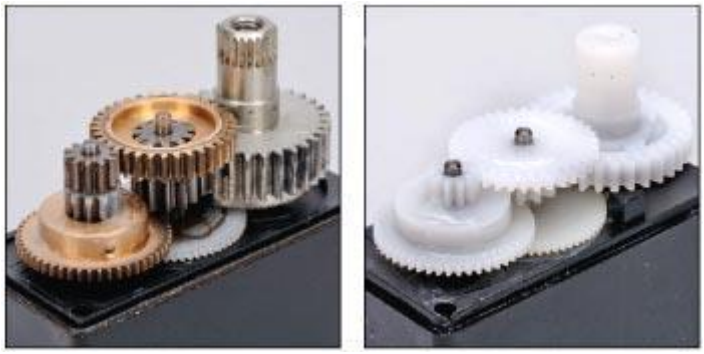
Оба типа сервопривода принимают одинаковые управляющие импульсы. После этого аналоговый сервопривод принимает решение, надо ли изменять положение, и в случае необходимости посылает сигнал на мотор. Происходит это обычно с частотой 50 Гц. Таким образом получаем 20 мс — минимальное время реакции. В это время любое внешнее воздействие способно изменить положение сервопривода. Но это не единственная проблема. В состоянии покоя на электромотор не подаётся напряжение, в случае небольшого отклонения от равновесия на электромотор подаётся короткий сигнал малой мощности. Чем больше отклонение, тем мощнее сигнал. Таким образом, при малых отклонениях сервопривод не сможет быстро вращать мотор или развивать большой момент. Образуются «мёртвые зоны» по времени и расстоянию.

Эти проблемы можно решать за счёт увеличения частоты приёма, обработки сигнала и управления электромотором. Цифровые сервприводы используют

специальный процессор, который получает управляющие импульсы, обрабатывает их и посылает сигналы на мотор с частотой 200 Гц и более. Получается, что цифровой сервопривод способен быстрее реагировать на внешние воздействия, быстрее развивать необходимые скорость и крутящий момент, а значит, лучше удерживать заданную позицию, что хорошо. Конечно, при этом он потребляет больше электроэнергии. Также цифровые сервоприводы сложнее в производстве, а потому стоят заметно дороже. Собственно, эти два недостатка — все минусы, которые есть у цифровых сервоприводов. В техническом плане они безоговорочно побеждают аналоговые сервоприводы.

Материалы шестерней

Шестерни для сервоприводов бывают из разных материалов: пластиковые, карбоновые, металлические. Все они широко используются, выбор зависит от конкретной задачи и от того, какие характеристики требуются в установке.



Пластиковые, чаще всего нейлоновые, шестерни очень лёгкие, не подвержены износу, более всего распространены в сервоприводах. Они не выдерживают больших нагрузок, однако если нагрузки предполагаются небольшие, то нейлоновые шестерни — лучший выбор.

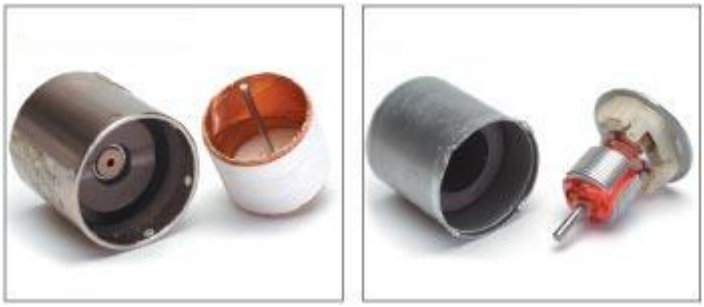
Карбоновые шестерни более долговечны, практически не изнашиваются, в несколько раз прочнее нейлоновых. Основной недостатой — дороговизна.

Металлические шестерни являются самыми тяжёлыми, однако они выдерживают максимальные нагрузки. Достаточно быстро изнашиваются, так что придётся менять шестерни практически каждый сезон. Шестерни из титана — фавориты среди металлических шестерней, причём как по

техническим характеристикам, так и по цене. К сожалению, они обойдутся вам достаточно дорого.

Коллекторные и бесколлекторные моторы

Существует три типа моторов сервоприводов: обычный мотор с сердечником, мотор без сердечника и бесколлекторный мотор.



Обычный мотор с сердечником (справа) обладает плотным железным ротором с проволочной обмоткой и магнитами вокруг него. Ротор имеет несколько секций, поэтому когда мотор вращается, ротор вызывает небольшие колебания мотора при прохождении секций мимо магнитов, а в результате получается сервопривод, который вибрирует и является менее точным, чем сервопривод с мотором без сердечника. Мотор с полым ротором (слева) обладает единым магнитным сердечником с обмоткой в форме цилиндра или колокола вокруг магнита. Конструкция без сердечника легче по весу и не имеет секций, что приводит к более быстрому отклику и ровной работе без вибраций. Такие моторы дороже, но они обеспечивают более высокий уровень контроля, вращающего момента и скорости по сравнения со стандартными.

Сервоприводы с бесколлекторным мотором появились сравнительно недавно. Преимущества те же что и у остальных бесколлекторных моторов: нет щёток, а значит они не создают сопротивление вращению и не изнашиваются,

скорость и момент выше при токопотреблении равном коллекторным моторам. Сервоприводы с бесколлекторным мотором — самые дорогие сервоприводы, однако при этом они обладают лучшими характеристиками по сравнению с сервоприводами с другими типами моторов.

Подключение к Arduino

Многие сервоприводы могут быть подключены к Arduino непосредственно. Для этого от них идёт шлейф из трёх проводов:

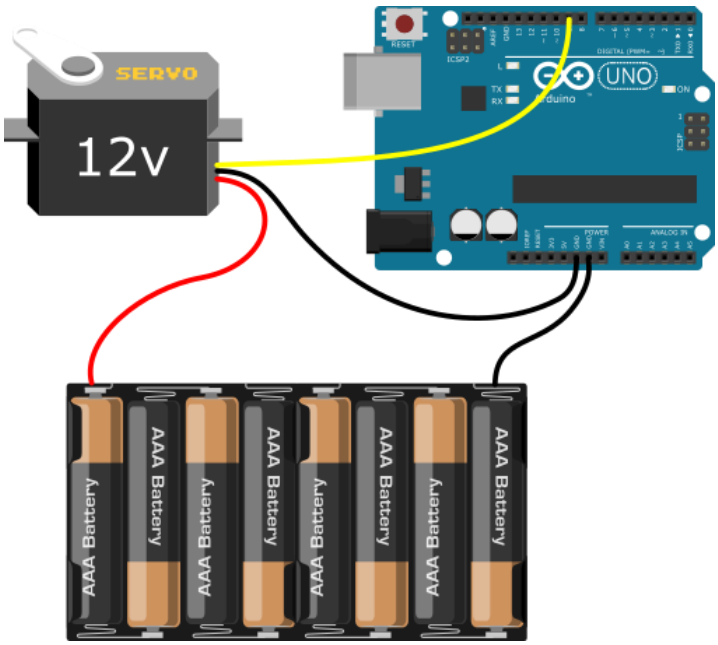
- ✓ красный — питание; подключается к контакту 5V или напрямую к источнику питания
- ✓ коричневый или чёрный — земля
- ✓ жёлтый или белый — сигнал; подключается к цифровому выходу Arduino.

Можно генерировать управляющие импульсы самостоятельно, но это настолько распространённая задача, что для её упрощения существует стандартная библиотека Servo.

Ограничение по питанию

Обычный хобби-сервопривод во время работы потребляет более 100 мА. При этом Arduino способно выдавать до 500 мА. Поэтому, если вам в проекте необходимо использовать мощный сервопривод, есть смысл задуматься о выделении его в контур с дополнительным питанием.

Рассмотрим на примере подключения 12V сервопривода:



Ограничение по количеству подключаемых сервоприводов

На большинстве плат Arduino библиотека Servo поддерживает управление не более 12 сервоприводами, на Arduino Mega это число вырастает до значения 48. При этом есть небольшой побочный эффект использования этой библиотеки: если вы работаете не с Arduino Mega, то становится невозможным использовать функцию analogWrite() на 9 и 10 контактах независимо от того, подключены сервоприводы к этим контактам или нет. На Arduino Mega можно подключить до 12 сервоприводов без нарушения функционирования ШИМ/PWM, при использовании большего количества сервоприводов мы не сможем использовать analogWrite() на 11 и 12 контактах.

Функционал библиотеки Servo

Библиотека Servo позволяет осуществлять программное управление сервоприводами. Для этого заводится переменная типа Servo. Управление осуществляется следующими функциями:

✓ attach() — присоединяет переменную к конкретному пину. Возможны два варианта синтаксиса для этой функции: servo.attach(pin) и servo.attach(pin, min, max). При этом pin — номер пина, к которому присоединяют сервопривод, min и max — длины импульсов в микросекундах, отвечающих за

углы поворота 0° и 180°. По умолчанию выставляются равными 544 мкс и 2400 мкс соответственно.

✓ write() — отдаёт команду сервоприводу принять некоторое значение параметра. Синтаксис следующий: servo.write(angle), где angle — угол, на который должен повернуться сервопривод.

✓ writeMicroseconds() — отдаёт команду послать на сервоприводимульс определённой длины, является низкоуровневым аналогом предыдущей команды. Синтаксис следующий: servo.writeMicroseconds(uS), где uS — длина импульса в микросекундах.

✓ read() — читает текущее значение угла, в котором находится сервопривод. Синтаксис следующий: servo.read(), возвращается целое значение от 0 до 180.

✓ attached() — проверка, была ли присоединена переменная к конкретному пину. Синтаксис следующий: servo.attached(), возвращается логическая истина, если переменная была присоединена к какому-либо пину, или ложь в обратном случае.

✓ detach() — производит действие, обратное действию attach(), то есть отсоединяет переменную от пина, к которому она была приписана. Синтаксис следующий: servo.detach().

Задание 1. Ответить на вопросы

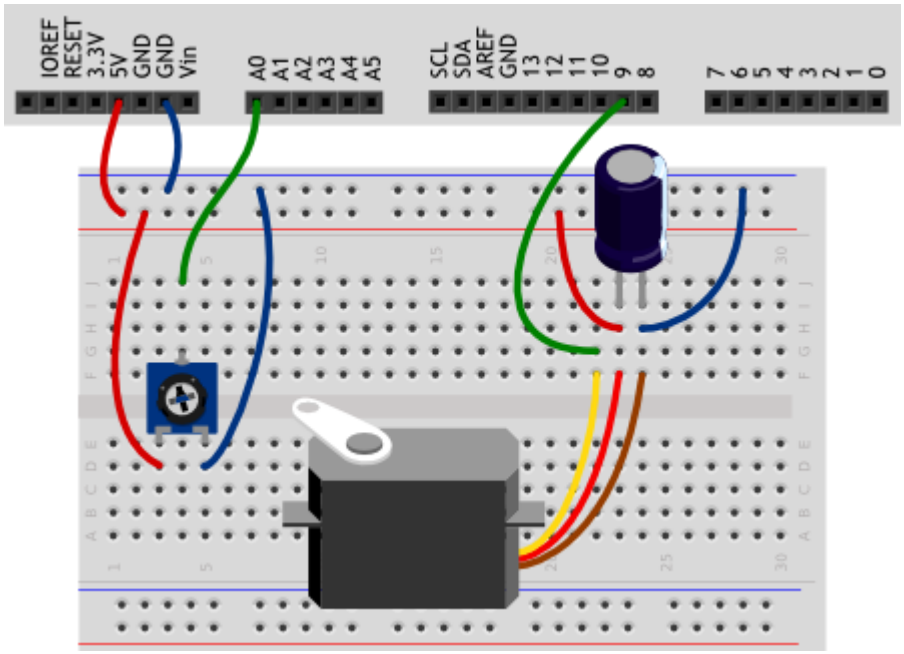
1. Что такое сервопривод?
2. Зачем нужно подключение библиотеки при работе с сервоприводом?
3. Опишите принцип работы конденсатора.

Задание 2. Список деталей для эксперимента

1. 1 плата Arduino Uno
2. 1 беспаечная макетная плата
3. 1 сервопривод
4. 1 конденсатор емкостью 220 мкФ

5. 11 проводов «папа-папа»
6. 1 потенциометр

Схема на макетной плате:



1. Зарисуйте принципиальную схему установки.

Обратите внимание

Конденсатор в данной схеме нам нужен для того, чтобы при включении сервопривода избежать просадки питания платы.

Не забывайте про то, что нужно соблюдать полярность электролитического конденсатора. Короткая ножка (со стороны белой полосы на корпусе) — «минус».

Вы можете соединить провод сервопривода с макетной платой проводами «папа-папа»: коричневый это земля, красный — питание, оранжевый — сигнал.

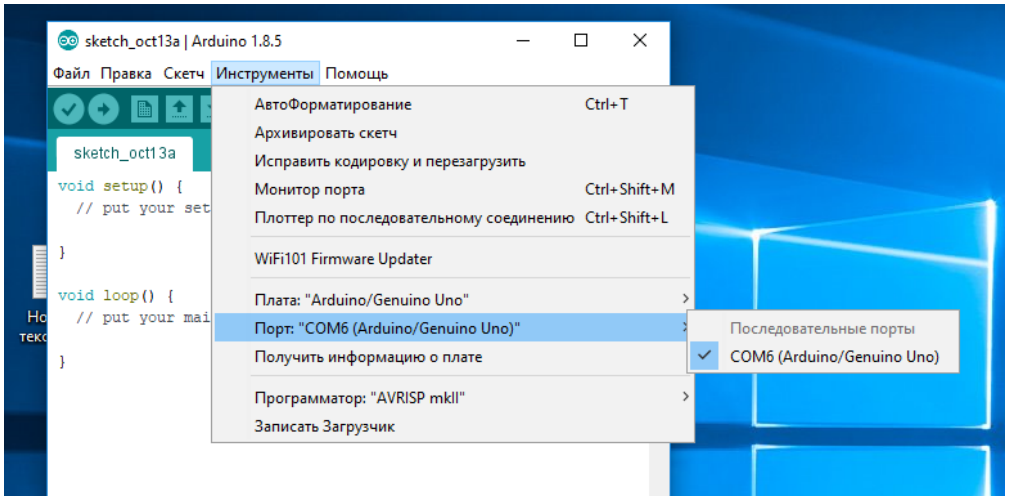
В данном эксперименте мы подключаем питание сервопривода к 5V-выходу Arduino. С одним сервоприводом плата справится, но если в каком-либо

проекте вам нужно больше серв, используйте специальные платы-драйвера с отдельным источником питания для серв.

Задание 3. Программирование микроконтроллера

1. Запустите приложение

2. Убедитесь, что выбран нужный порт



3. Наберите в редакторе кода следующий код программы:

```
// управлять сервоприводами (англ. servo motor) самостоятельно
// не так то просто, но в стандартной библиотеке уже всё
// заготовлено, что делает задачу тривиальной
#include <Servo.h>

#define POT_MAX_ANGLE 270.0 // макс. угол поворота потенциометра

// объявляем объект типа Servo с именем myServo. Ранее мы
// использовали int, boolean, float, а теперь точно также
// используем тип Servo, предоставляемый библиотекой. В случае
// Serial мы использовали объект сразу же: он уже был создан
// для нас, но в случае с Servo, мы должны сделать это явно.
// Ведь в нашем проекте могут быть одновременно несколько
// приводов, и нам понадобится различать их по именам
Servo myServo;

void setup()
{
  // прикрепляем (англ. attach) нашу серву к 9-му пину. Явный
  // вызов pinMode не нужен: функция attach сделает всё за нас
  myServo.attach(9);
}

void loop()
{
  int val = analogRead(A0);
  // на основе сигнала понимаем реальный угол поворота движка.
  // Используем вещественные числа в расчётах, но полученный
  // результат округляем обратно до целого числа
  int angle = int(val / 1024.0 * POT_MAX_ANGLE);
  // обычная серва не сможет повторить угол потенциометра на
  // всём диапазоне углов. Она умеет вставать в углы от 0° до
  // 180°. Ограничиваем угол соответствующе
  angle = constrain(angle, 0, 180);
  // и, наконец, подаём серве команду встать в указанный угол
  myServo.write(angle);
}
```

Пояснения к коду

В данном эксперименте мы также имеем дело с объектом, на этот раз он нужен для простого управления сервоприводом. Как отмечено в комментариях, в отличие от объекта Serial, объекты типа Servo нам нужно явно создать: Servo myServo, предварительно подключив библиотеку <Servo.h>.

Далее мы используем два метода для работы с ним:

✓ myServo.attach(pin) — сначала «подключаем» серву к порту, с которым физически соединен его сигнальный провод. pinMode() не нужна, метод attach() займется этим.

✓ myServo.write(angle) — задаем угол, т.е. позицию, которую должен принять вал сервопривода. Обычно это 0—180°.

myServo здесь это имя объекта, идентификатор, который мы придумываем так же, как названия переменных. Например, если вы хотите управлять двумя захватами, у вас могут быть объекты leftGripи rightGrip.

Мы использовали функцию int() для явного преобразования числа с плавающей точкой в целочисленное значение. Она принимает в качестве параметра значение любого типа, а возвращает целое число. Когда в одном выражении мы имеем дело с различными типами данных, нужно позаботиться о том, чтобы не получить непредсказуемый ошибочный результат.

Задание 4. Ответьте на следующие вопросы

1. Зачем нужен конденсатор при включении в схему сервопривода?
2. Каким образом библиотека <Servo.h> позволяет нам работать с сервоприводом?
3. Зачем мы ограничиваем область допустимых значений для angle?
4. Как быть уверенным в том, что в переменную типа int после вычислений попадет корректное значение?

Задание 5. Самостоятельно измените существующую программу и схему

1. Измените программу так, чтобы по мере поворота ручки потенциометра, сервопривод последовательно занимал 8 положений: 45, 135, 87, 0, 65, 90, 180, 150°.
2. Предположим, что сервопривод управляет шторкой, и нам нужно поддерживать постоянное количество света в помещении. Создайте такой механизм.