|  |  |  |
| --- | --- | --- |
| Источник питания | Достоинства | Недостатки |
| Комплектное зарядное устройство шуруповёрта. | - несложная переделка;  - используется существующее зарядное устройство;  - не требуется подбирать напряжение блока питания;  - не требуется вмешательство в электрическую схему шуруповёрта. | зарядное устройство занимает место. |
| Готовый блок питания, помещённый в корпус старого аккумулятора. | - несложная переделка;  - красивое инженерное решение — из шуруповёрта выходит только сетевой шнур;  - нет потерь в кабеле с низким напряжением;  - не требуется вмешательство в электрическую схему шуруповёрта. | - поиск готового компактного блока питания на требуемое напряжение;  - блок питания греется в закрытом корпусе, надо делать перерывы в работе. |
| Самодельный блок питания, помещённый в корпус старого аккумулятора. | - красивое инженерное решение — из шуруповёрта выходит только сетевой шнур;  - нет потерь в кабеле с низким напряжением;  - не требуется вмешательство в электрическую схему шуруповёрта. | необходимо опыт пайки, сборки и отладки электрических схем. |
| Внешний блок питания | несложная переделка. | - требуется разбирать шуруповёрт и подключаться к его схеме;  - блок питания занимает место на столе;  - нужно найти походящий блок питания. |
| Блок питания от компьютера | - несложная переделка;  - компьютерный блок питания легко найти;  - подойдёт любой блок питания от 300 Вт. | - требуется разбирать шуруповёрт и подключаться к его схеме;  - блок питания занимает много места на столе. |

Сравнив все представленные варианты, было решено использовать первый предложенный способ, а именно – переделка комплектного зарядного устройства. Предполагаемая последовательность действий:

- припаять или прицепить зажимами «крокодил» к клеммам зарядного устройства два провода;

- разобрать старый аккумулятор и вынуть из него севшие элементы;

- просверлить в корпусе аккумулятора отверстие для кабеля, продеть кабель в отверстие;

- припаять кабель к клеммам бывшего аккумулятора, подключаемым к шуруповёрту;

- собрать корпус аккумулятора.

Для того, чтобы шуруповерт вращался, необходимо подавать на него сигнал. Это будет делать микроконтроллер ATmega2560 на плате Arduino. Необходимо подключить выходы шуруповерта к соответствующим портам микроконтроллера, что может потребовать дополнительных компонентов для согласования уровней напряжения и управления. Написать программу на C++ с использованием библиотек, предоставляемых микроконтроллером. Программа должна предоставлять функции для включения и выключения шуруповерта по запросу. Программа должна быть способна принимать команды на включение и выключение шуруповерта через интерфейс вашего выбора (например, последовательный порт, Wi-Fi, Bluetooth и т. д.) и вызывать соответствующие функции на включение и выключение вращения, загрузить программу на контроллер.

Так как шуруповерт использует для своего питания 12 и более Вольт, то потребуется драйвер мотора или драйвер реле для управления шуруповертом от микроконтроллера. Драйвер предоставит необходимую силу тока и изоляцию между микроконтроллером и шуруповертом. Необходимо подобрать подходящий драйвер мотора или реле, совместимые с микроконтроллером и шуруповертом. Подключить выходы драйвера к шуруповерту и микроконтроллеру. Это обычно включает в себя подключение питания (12 В или более), заземление (GND), и сигнальные пины, которые будут управлять включением и выключением шуруповерта.

Написанный код взаимодействует с драйвером через микроконтроллер, отправляя сигналы на соответствующие пины драйвера. В коде определены функции, которые управляют этими сигналами для включения и выключения шуруповерта или другого мотора. Микроконтроллер ATmega2560, как правило, используется с платформой Arduino Mega. Для управления моторами или другими устройствами с более высокими напряжениями и мощностями, такими как шуруповерты или моторы, часто применяются различные драйверы моторов. Один из популярных драйверов, который совместим с Arduino Mega (ATmega2560), это L298N. Драйвер мотора L298N является распространенным и относительно простым в использовании. Он подходит для управления двигателями постоянного тока, шаговыми моторами и другими моторами. L298N также обеспечивает изоляцию между микроконтроллером и высокими напряжениями мотора. Для подключения драйвера L298N к Arduino Mega (ATmega2560) потребуется:

* подключить питание (обычно 12 В или более) и заземление (GND) драйвера L298N к соответствующим источникам питания;

- подключить выходы моторов шуруповерта к выходам драйвера L298N;

- подключить управляющие пины драйвера L298N к соответствующим пинам на Arduino Mega (ATmega2560);

- использовать программу на Arduino IDE (или другой среде разработки Arduino) с использованием библиотеки, чтобы управлять драйвером и, следовательно, шуруповертом.

После этого мы сможем управлять шуруповертом с помощью кода на Arduino, используя функции для включения и выключения моторов через драйвер L298N и библеотеку AFMotor.

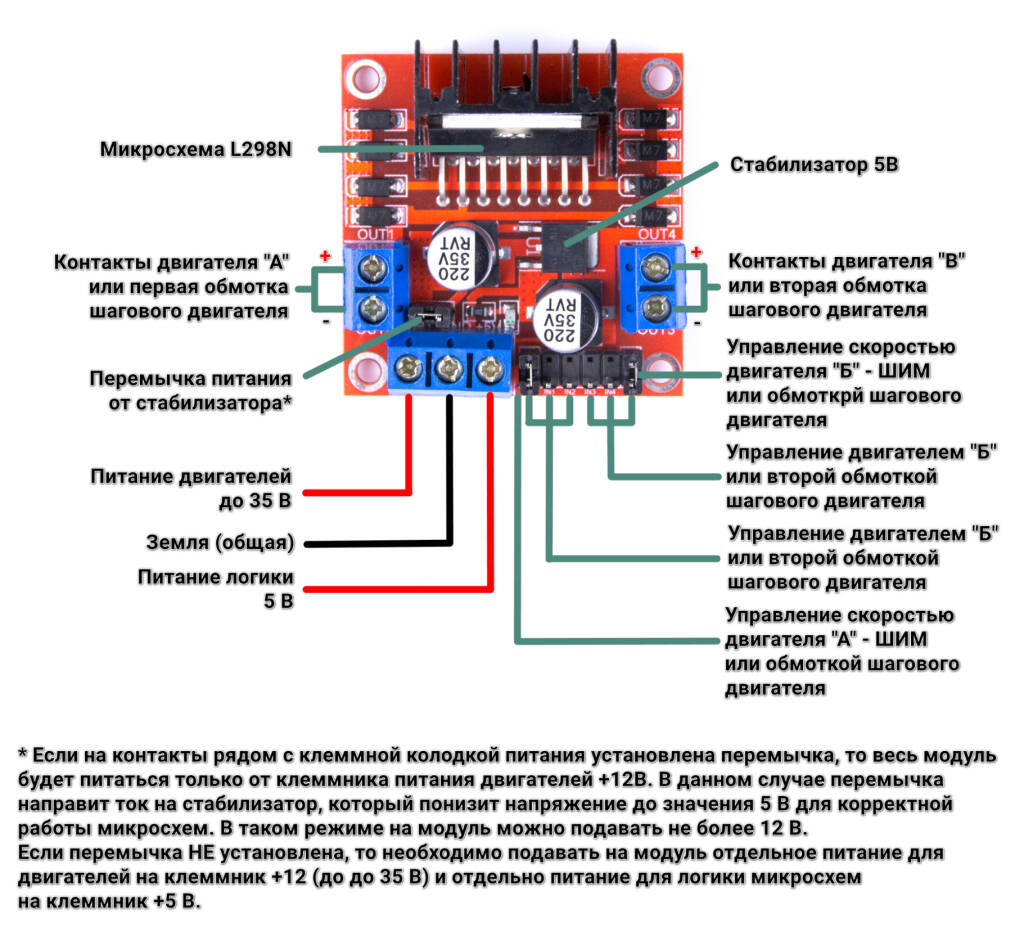


Рисунок 1 – драйвер L298N.

Для управления реле с помощью микроконтроллера Arduino или другой платформы, не требуется специализированный драйвер, как в случае с моторными драйверами. Однако можно использовать обычные цифровые выходы микроконтроллера и библиотеки для управления реле. Подключить реле к микроконтроллеру: подключить контакт управления (обычно это третий контакт реле) к выходу микроконтроллера. Подключить контакты реле, которые управляют нагрузкой, к вашей нагрузке (например, лампе, мотору и т. д.). Подключить шуруповерт к источнику питания. Настроить пин микроконтроллера: в коде Arduino определите пин, который будет использоваться для управления реле, как выходной пин. Использовать библиотеку, такую как "Arduino Relay" или "Adafruit Relay", чтобы упростить управление реле. Эти библиотеки предоставляют функции для включения и выключения реле. Можно использовать функции из библиотеки или просто установить пин в HIGH (HIGH - для включения) и LOW (LOW - для выключения) в соответствии с вашими потребностями.

Подключение – необходимо загрузить программу на ардуино, подключить драйвер к ардуино, реле подключить к драйверу, шуруповерт подключить к реле.

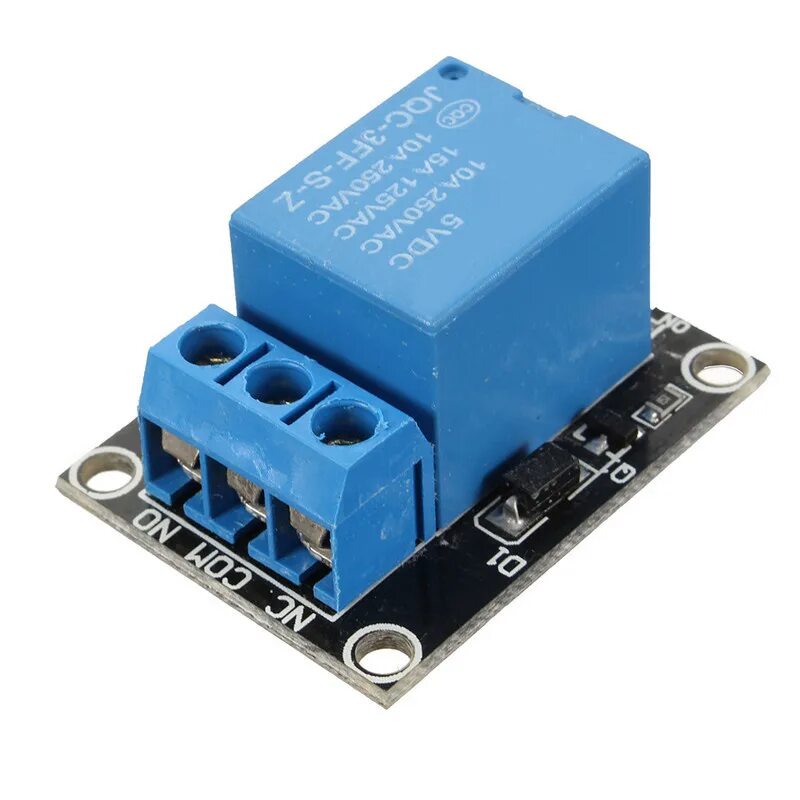


Рисунок 2 - Модуль реле одноканальный 5В (KY-019).

В качестве модуля реле используем Модуль реле одноканальный 5В (KY-019).

Для подачи теста мы воспользуемся ШАРОВОЙ КРАН С ЭЛЕКТРОПРИВОДОМ 3/4" 12В (С РУЧНЫМ УПРАВЛЕНИЕМ И ИНДИКАТОРОМ). Как понятно из названия, питается он с помощью 12-тивольтовой линии. Для управления им воспользуемся реле SSR-10DA Твердотельное реле 10A Однофазный модуль SSR 10DA DC-AC 3-32V 12V DC до 24-380V 220V AC.



Рисунок 3 – шаровой электрический кран

Для проверки уровня жидкости воспользуемся ﻿– “ПОПЛАВКОВЫЙ ВЫКЛЮЧАТЕЛЬ (ДАТЧИК УРОВНЯ ВОДЫ)”. Подключим его напрямую через плату ардуино.



Рисунок 4 - датчик уровня жидкости

Приложение А – код программы.

#include <Arduino.h> // Библиотека для работы с Arduino

const int floatSwitchPin = 2; // Пин, к которому подключен поплавковый выключатель (замените на свой пин)

const int relayPin1 = 12; // Пин для первого реле (открытие и закрытие крана) (замените на свой пин)

const int relayPin2 = 13; // Пин для второго реле (открытие и закрытие крана) (замените на свой пин)

const int buttonPin = 3; // Пин, к которому подключена кнопка (замените на свой пин)

const int motor1A = 4; // Пин для управления мотором 1 (направление A)

const int motor1B = 5; // Пин для управления мотором 1 (направление B)

const int motor2A = 6; // Пин для управления мотором 2 (направление A)

const int motor2B = 7; // Пин для управления мотором 2 (направление B)

bool isRelay1On = false; // Флаг для состояния первого реле (открыто или закрыто)

bool isRelay2On = false; // Флаг для состояния второго реле (открыто или закрыто)

unsigned long startTime = 0; // Время начала программы

unsigned long openTime = 0; // Время следующего открытия крана

unsigned long closeTime = 0; // Время следующего закрытия крана

bool isOpening = false; // Флаг для состояния открытия крана

void setup() {

pinMode(floatSwitchPin, INPUT\_PULLUP); // Пин поплавкового выключателя как вход с подтягивающим резистором

pinMode(relayPin1, OUTPUT); // Настроить пин первого реле как выход (открытие и закрытие крана)

pinMode(relayPin2, OUTPUT); // Настроить пин второго реле как выход (открытие и закрытие крана)

pinMode(buttonPin, INPUT\_PULLUP); // Настроить пин кнопки как вход с подтягивающим резистором

// Настроить пины моторов как выходы

pinMode(motor1A, OUTPUT);

pinMode(motor1B, OUTPUT);

pinMode(motor2A, OUTPUT);

pinMode(motor2B, OUTPUT);

// Установить начальное время

startTime = millis();

openTime = startTime + 240000; // Открыть кран через 240 секунд

}

void loop() {

// Проверить состояние поплавкового выключателя

if (digitalRead(floatSwitchPin) == LOW) {

// Поплавок опущен - есть жидкость

// Проверить состояние кнопки

if (digitalRead(buttonPin) == LOW) {

// Кнопка нажата, переключить состояние первого реле (открытие и закрытие крана)

isRelay1On = !isRelay1On;

digitalWrite(relayPin1, isRelay1On ? HIGH : LOW);

// Подождать, пока кнопка отпущена

while (digitalRead(buttonPin) == LOW) {

delay(10);

}

}

// Проверить состояние первого реле (открытие и закрытие крана)

if (isRelay1On) {

// Первое реле включено, подождать 240 секунд (4 минуты)

delay(240000);

// Выключить первое реле (закрыть кран) и установить флаг

digitalWrite(relayPin1, LOW);

isRelay1On = false;

}

// Проверить состояние второго реле (открытие и закрытие крана)

if (isRelay2On) {

// Второе реле включено, подождать 240 секунд (4 минуты)

delay(240000);

// Выключить второе реле (закрыть кран) и установить флаг

digitalWrite(relayPin2, LOW);

isRelay2On = false;

}

// Проверить время и открыть кран после 240 секунд

unsigned long currentTime = millis();

if (currentTime >= openTime && !isOpening) {

// Открыть кран (включить второе реле)

digitalWrite(relayPin2, HIGH);

delay(5000); // Подождать

// 5 секунд (держим кран открытым)

digitalWrite(relayPin2, LOW); // Закрыть кран

isOpening = true;

closeTime = currentTime + 140000; // Закрыть кран через 140 секунд

}

// Закрыть кран после 140 секунд с момента открытия

if (currentTime >= closeTime) {

digitalWrite(relayPin2, HIGH); // Открыть кран

delay(5000); // Подождать 5 секунд (держим кран открытым)

digitalWrite(relayPin2, LOW); // Закрыть кран

isOpening = false;

openTime = currentTime + 140000; // Открыть кран через 140 секунд

}

}

else {

// Поплавок поднят - нет жидкости.

// Выключить реле (открытие и закрытие крана)

digitalWrite(relayPin1, LOW);

digitalWrite(relayPin2, LOW);

// Выключить моторы

digitalWrite(motor1A, LOW);

digitalWrite(motor1B, LOW);

digitalWrite(motor2A, LOW);

digitalWrite(motor2B, LOW);

// Остановить программу в бесконечном цикле

while (true) {

// Программа остановлена

}

}

}