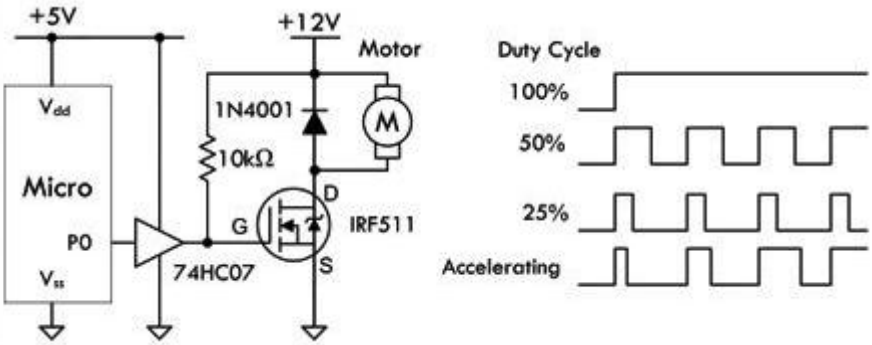
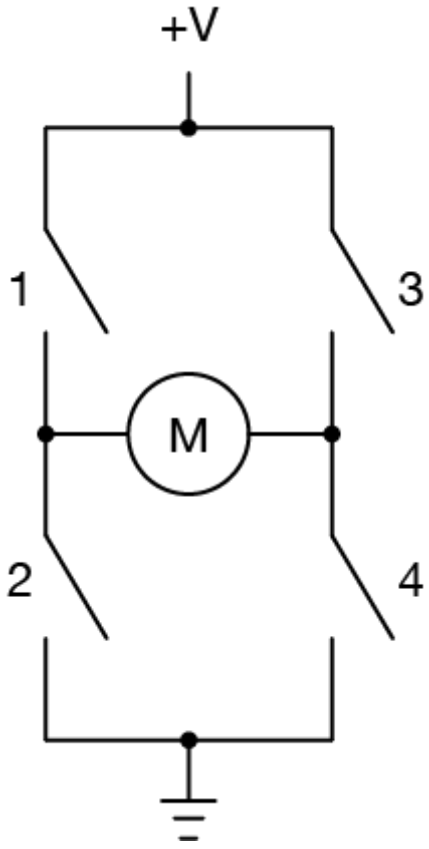
Silnik prądu stałego

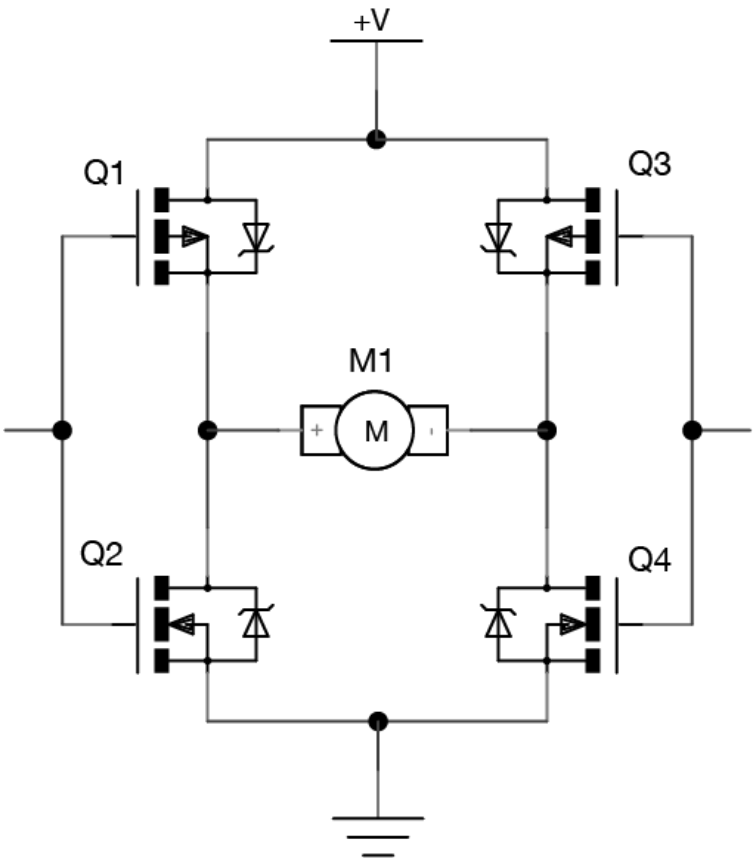
Sterowanie silnika prądu stałego



Specyfikacja silnika MT68

- Napięcie zasilania: od 3 V do 6 V
- Prąd na biegu jałowym: 45 mA
- Obroty: 12100 obr/min dla 3 V
- Wymiary: 10 x 15 mm
- długość silnika (bez wrzeciona): 16.5mm
- wielkość silnika: 10 x 12mm
- napięcie zasilania:3-6V
- długość wrzeciona: 3mm
- średnica wrzeciona: 1mm

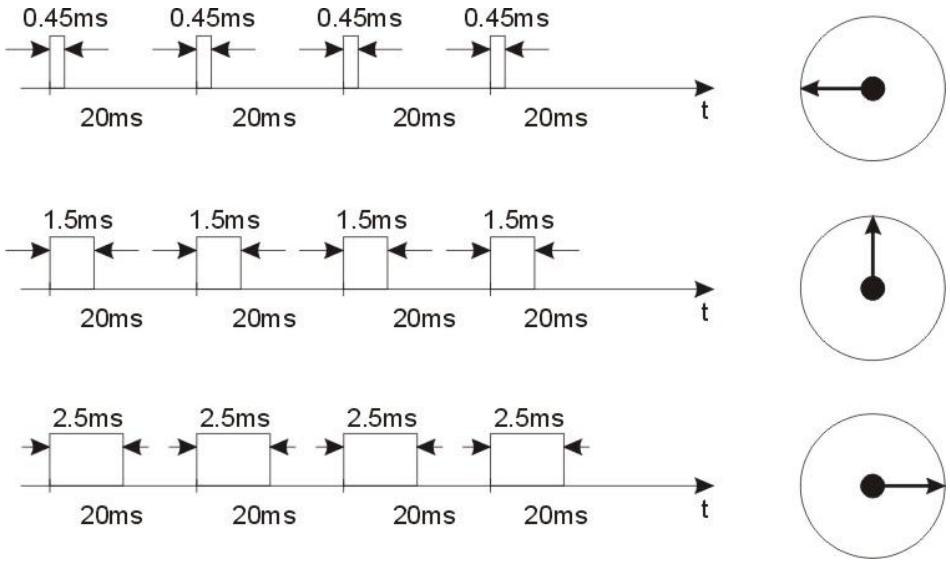




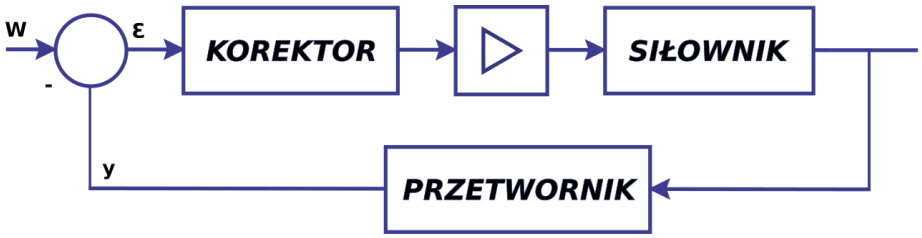
Serwomechanizm

Serwomechanizm jest małym urządzeniem posiadające na zewnątrz mały wał. Wał ten może być ustawiony w określonej pozycji kątowej poprzez wysłanie do serwomechanizmu odpowiedniego sygnału. Kiedy na linii sygnałowej utrzymuje się stały sygnał wał jest utrzymywany w stałej pozycji. Gdy sygnał się zmienia, wał zmienia swoją pozycję.

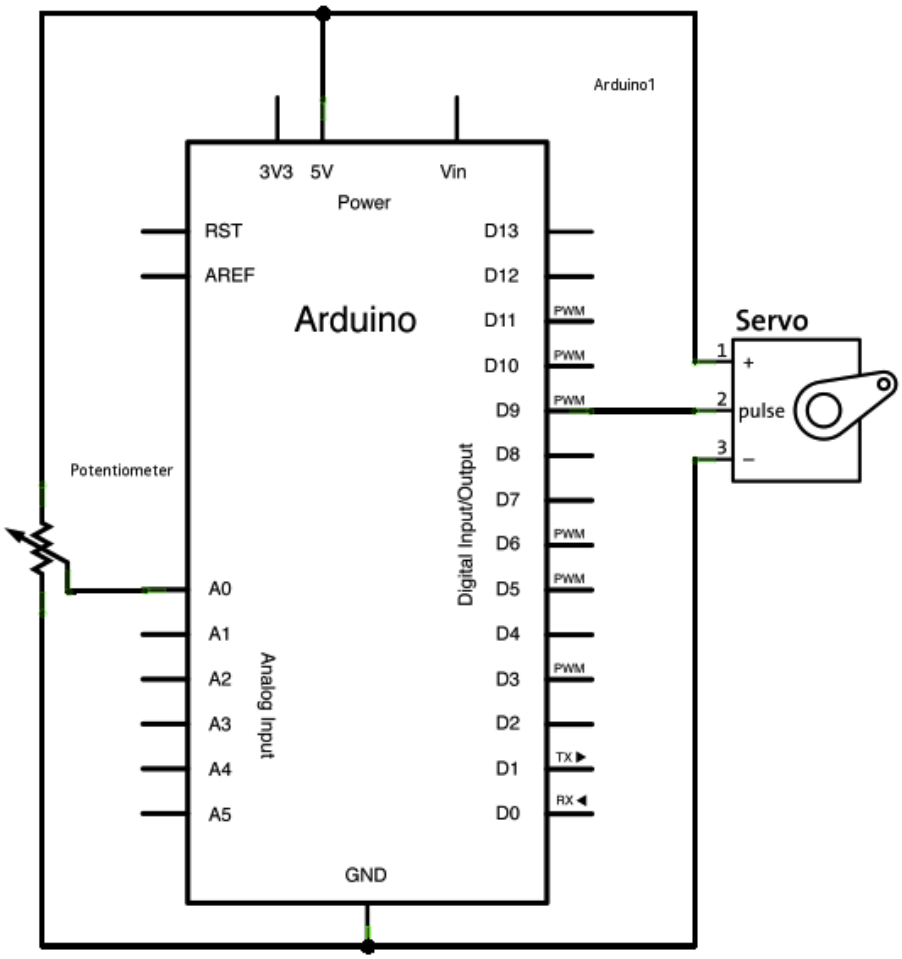
Budowa serwomechanizmu opiera się o zwykły silnik prądu stałego z dodatkowymi przekładniami oraz układem sterującym. Układ sterujący może dokładnie stwierdzić w jakiej pozycji znajduje się wał, gdyż wał obraca potencjometrem obrotowym. Dzięki temu można z dużą dokładnością pozycjonować wał. Układ sterujący ustawia wał w zależności od długości impulsu na linii sterującej próbkowanego co 20ms. W większości przypadków wał serwomechanizmu może przyjmować pozycje kątowe z zakresu 0-180 stopni. Większość mechanizmów posiada ograniczniki nie pozwalające na większe wychylenia wału. W zasadzie serwomechanizmy są głównie wykorzystywane w układach pozycjonowania np. kamer, w sprzęcie elektronicznym wyposażonym w mechaniczne układy pozycjonowania, w modelach np. samolotów do pozycjonowania lotek, no i oczywiście w różnego rodzaju robotach. Więc w większości przypadków nie jest potrzebny pełny obrót. Bardziej potrzebna jest dokładność, łatwość sterowania oraz mały pobór prądu. Tam gdzie potrzebny jest pełny obrót stosuje się silniki krokowe, lub silniki prądu stałego. Ze światem zewnętrznym serwomechanizm jest połączony za pomocą trzech przewodów: +zasilania, masy i sygnału sterującego.



Sygnał sterujący pozycją serwomechanizmu



Sterowanie w układzie ze sprzężeniem zwrotnym



Budowa serwomechanizmu

Przykład

```
// Sterowanie serwomechanizmem za pomocą potencjometru

#include <Servo.h>
/*******************************************************
Podłączenie serva:
//  czerwony +5V
//  brązowy 0V
//  pomarańczowy sterowanie z pinu 9
*******************************************************/
Servo  myservo;        // utworzenie obiektu do sterowania serwomechanizmem

int  potpin = 0;       // wejście analogowe do przyłączenia suwaka potencjometru
int  val;              // zmienna przechowująca odczytaną z wejścia analogowego wartość

void setup()
{
  myservo.attach(9);   // serwomechanizm sterowany z wyjścia 9
}

void loop()
{
  val = analogRead(potpin);    //odczytanie wartości z potencjometru (zakres 0 do 1023)
  val = map(val, 0, 1023, 0, 179);
 // przeskalowanie wartości do zakresu stosowanego przez serwomechanizm (0 do 180)
  myservo.write(val);          // ustawienie zgodnie z przeskalowaną wartością
  delay(15);                   // oczekiwanie na wykonanie polecenia
}
```

Zadanie

Napisać program, który pozwala sterować serwomechanizmem za pomocą wartości przesyłanych przez monitor portu szeregowego.

Silnik krokowy

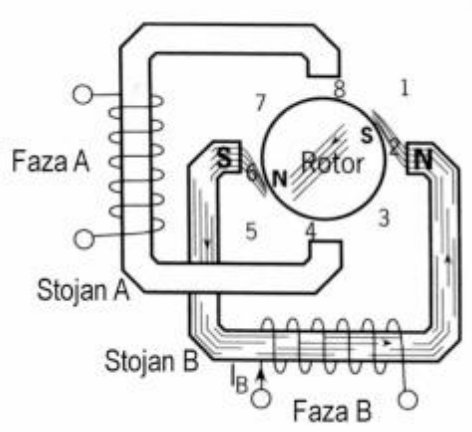
Silnik krokowy jest urządzeniem elektromechanicznym, które przekształca impulsy elektryczne w dyskretne ruchy mechaniczne. Oś silnika krokowego obraca się o niewielkie przyrosty kąta pod wpływem impulsów elektrycznych podawanych w odpowiedniej kolejności. Kierunek obrotów osi jest ściśle związany z sekwencją podawanych impulsów, prędkość obrotów zależy od częstotliwości tych impulsów, a kąt obrotu – od ich ilości.

Podstawową zaletą silników krokowych jest możliwość ich precyzyjnego pozycjonowania bez pętli sprzężenia zwrotnego (nie potrzeba mierzyć aktualnej pozycji silnika – jest ona określona ilością wykonanych kroków), możliwość szybkiego rozbiegu, hamowania i zmiany kierunku, duża niezawodność ze względu na brak szczotek.

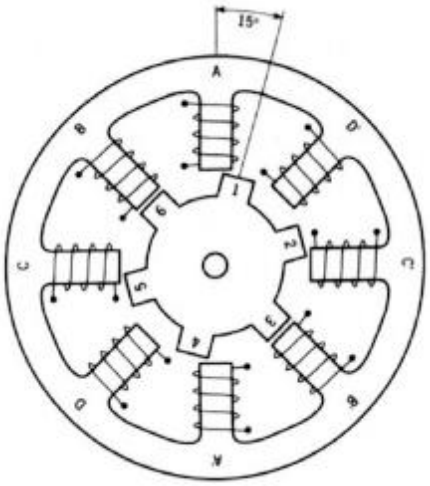
Do wad silników krokowych zaliczamy małą moc, niską szybkość obrotową i możliwość wystąpienia zjawisk rezonansowych.

Podstawowe rodzaje silników krokowych to:

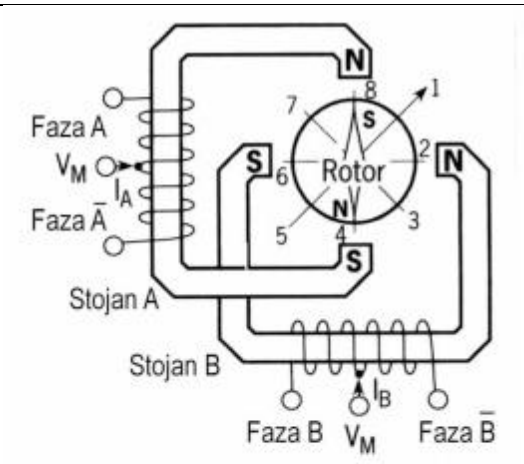
1. silniki o zmiennej reluktancji VR
2. silniki z magnesem trwałym PM (silniki kubkowe)
3. silniki hybrydowe

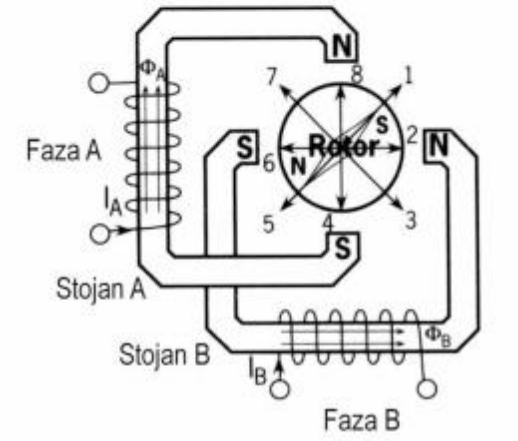


Zasada działania silnika krokowego

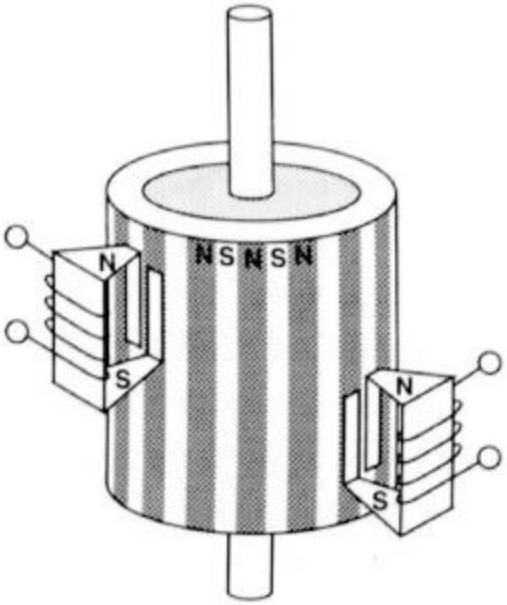


Silnik o zmiennej reluktancji

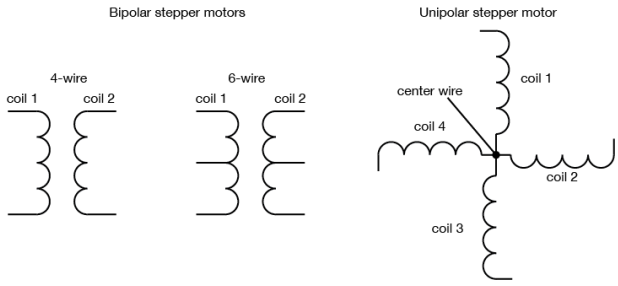




Silniki krokowe o uzwojeniu unipolarnym i bipolarnym



Zasada działania silnika kubkowego z magnesami trwałymi



Przykład 1

```
/*
 Sterowanie silnikiem krokowym - jeden obrót w lewo/prawo
 Silnik podłączony jest do wyjść cyfrowych Arduino    Pin1-Pin0, in2-Pin1, in3-Pin2, in4-Pin3
 Należy również podłączyć do sterownika napięcie +5V oraz masę
 */

#include <Stepper.h>

const int stepsPerRevolution = 2048;  // zmienić w zależności od ilości kroków na obrót danego silnika
                //  64*32 (64 kroki po 5,625 stopnia oraz przekładnia 1:32 - w dokumentacji 1:64)

// utworzenie obiektu do sterowania silnikiem
Stepper myStepper(stepsPerRevolution, 8,10,9,11);     //in1-8, in2-9, in3-10, in4-11
void setup() {
  // ustawienie szybkości obrotowej na wartość 12 rpm:
  myStepper.setSpeed(10);
  // inicjalizacja monitora portu szeregowego:
  Serial.begin(9600);
}

void loop() {
  // wykonanie obrotu w jednym kierunku:
    Serial.println(" zgodnie z ruchem zegara");
    myStepper.step(stepsPerRevolution);
    delay(500);

   // obrót w kierunku przeciwnym:
    Serial.println("i przeciwnie");
    myStepper.step(-stepsPerRevolution);
    delay(500);
}
```

Przykład 2

```
/*
 Sterowanie silnikiem krokowym - skok o jeden krok - bardzo powoli.
 Układ można wykorzystać do śledzenia zmian na stykach sterujących oraz do określenia
 czy zostały połączone we właściwej kolejności (zmiana kolejności świecenia w jedną stronę).
 Za pomocą tego programu można określić liczbę kroków wykonywanych na jeden obrót.
 Silnik podłączony jest do wyjść cyfrowych  Arduino (in1-Pin0, in2-Pin1, in3-Pin2, in4-Pin3
  Należy również podłączyć do sterownika napięcie +5V oraz masę
  */

#include <Stepper.h>

const int stepsPerRevolution = 2048;  // zmienić w zależności od ilości kroków na obrót danego silnika
                  //  64*32 (64 kroki po 5,625 stopnia oraz przekładnia 1:32 - w dokumentacji 1:64)

// utworzenie obiektu do sterowania silnikiem
Stepper myStepper(stepsPerRevolution, 8,10,9,11);     //in1-8, in2-9, in3-10, in4-11


int stepCount = 0;         // liczba wykonanych kroków

void setup() {
  Serial.begin(9600);
}

void loop() {
  // wykonanie kroku
  myStepper.step(1);

  Serial.print("krok:" );
  Serial.println(stepCount);
  stepCount++;
  delay(200);
}
```

Przykład 3

```
/*
 Sterowanie silnikiem krokowym – regulacja szybkości

 Program steruje unipolarnym lub bipolarnym silnikiem krokowym
przyłączonym do wyjść cyfrowych  8 - 11 Arduino.
 Potencjometr służący do regulacji przyłączony jest do wejścia analogowego A0.

 Regulacja prędkości polega na zmianie odstępu czasu (opóźnienia) pomiędzy impulsami do silnika za pomocą
funkcji setSpeed().

 Created 30 Nov. 2009
 Modified 28 Oct 2010
 by Tom Igoe
  */

#include <Stepper.h>

const int stepsPerRevolution = 2048;


Stepper myStepper(stepsPerRevolution, 8,10,9,11);

int stepCount = 0;  // number of steps the motor has taken

void setup() {
  // nothing to do inside the setup
}

void loop() {
  // read the sensor value:
  int sensorReading = analogRead(A0);
  // map it to a range from 0 to 100:
  int motorSpeed = map(sensorReading, 0, 1023, 0, 100);
  // set the motor speed:
  if (motorSpeed > 0) {
    myStepper.setSpeed(motorSpeed);
    // step 1/100 of a revolution:
    myStepper.step(stepsPerRevolution/100);
  }
}
```