Arduino krokohrátky

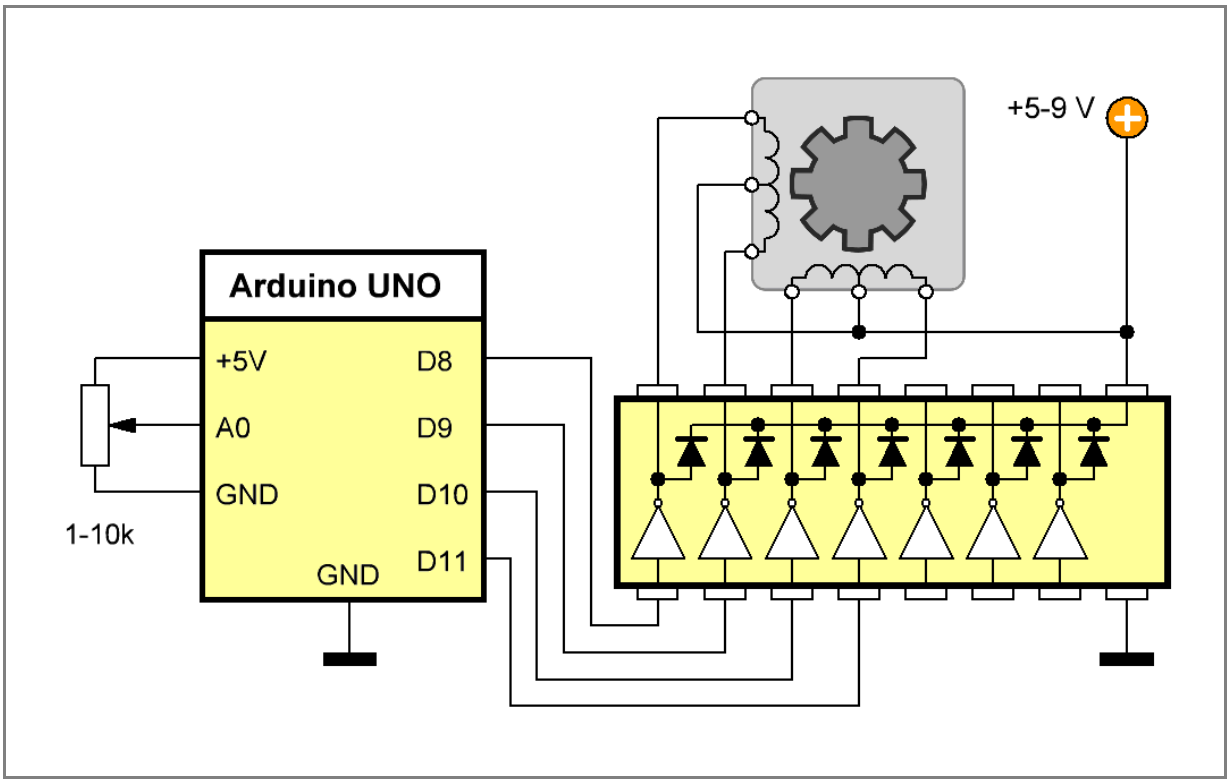
Začneme tím nejjednodušším, tedy řízením rychlosti otáčení krokového motoru potenciometrem.

Pro jednoduchost je v tomto příkladu použit v současnosti velmi populární miniaturní krokový motorek 28BYJ-48.

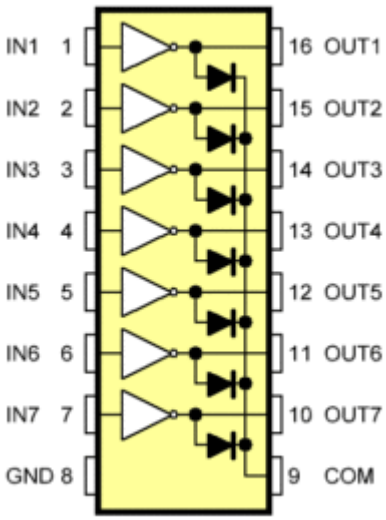
Motorek se dodává i s jednoduchým budičem, takže ho můžeme připojit přímo k Arduinu.

Motorek s rotorem z permanentního magnetu má jednu otáčku o 360 st. rozdělenu na 32 kroků. K motoru je pevně připojena převodovka s převodovým poměrem 64:1, k otočení výstupního hřídele o 360 st. tedy potřebujeme (32×64) 2048 kroků. Díky převodovce běží motorek hladce a přes miniaturní rozměry má celkem slušný točivý moment.

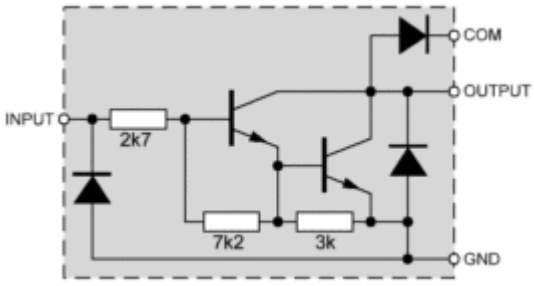
Schéma zapojení:



Krokový motor je připojen k řídící desce s ULN2003 a musí být napájen externím zdrojem napětí v rozsahu 5 až 9 V. Při pokusu o napájení přímo z Arduina hrozí nebezpečí jeho poškození! Obvod ULN2003 je dobře znám, takže si jen připomeňme rozložení pinů a zapojení jednoho kanálu ve vnitřní struktuře.



Rozložení pinů obvodu ULN2003



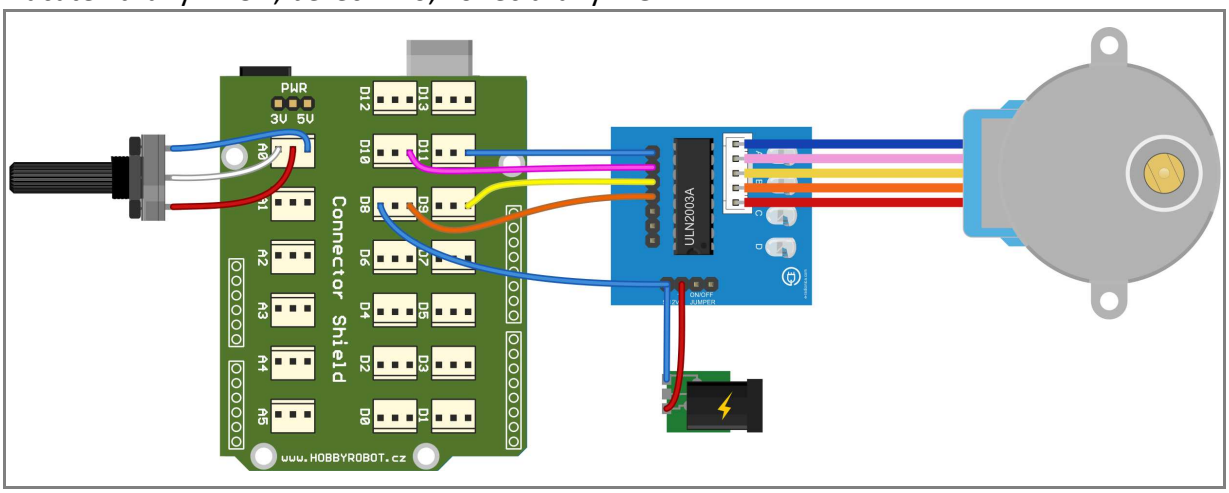
Vnitřní zapojení jednoho kanálu obvodu

Propojení s Arduinem:

IN1 – D11, IN2 – D10, IN3 – D9, IN4 – D8

Potenciometr je připojen na pin A0 Arduina podle obrázku, tedy:

Začátek dráhy – +5V, běžec – A0, konec dráhy – GND



Pro snadné a přehledné propojení s Arduinem je vhodné použít Connector shield.

Program

V programu je použita knihovna stepper, která je součástí instalace vývojového prostředí Arduina.

Její funkce jsou značně omezené:

Konstruktor

vytvoří instanci třídy, (v příkladu pojmenovanou stepper), předá tím programu počet kroků motoru na jednu otáčku a čísla pinů, které budou použity pro řízení motoru.

```
Stepper stepper(STEPS, IN4, IN2, IN3, IN1)
```

Knihovna obsahuje jen dvě funkce:

setSpeed(rpm)

Funkce nastavuje rychlost v otáčkách za minutu (ot/min nebo RPM), jakou se motor bude otáčet při volání funkce step().

step(steps)

Funkce otáčí motorem o parametrem určený počet kroků, rychlostí, určenou parametrem funkce setSpeed().

Zdrojový kód:

```
// připoj knihovnu Stepper motor Arduina
#include <Stepper.h>

// počet kroků motoru na otáčku
#define STEPS 32

// definice řídících pinů pro motor
#define IN1  11
#define IN2  10
#define IN3   9
#define IN4   8

// inicializace knihovny stepper
Stepper stepper(STEPS, IN4, IN2, IN3, IN1);

// potenciometr (1 - 10k) je připojen na pin A0
#define pot A0

void setup()
{  }

void loop()
{
  // přečteme analogovou hodnotu z potenciometru
  int val = analogRead(pot);

  // pokud je potenciometr ve střední poloze, motor se neotáčí
  if( (val > 500) | (val < 523) )
  {
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
  }

  else
  {
    // chceme otáčet motorem jedním směrem
    while (val >= 523)
    {
      // mapujeme proměnnou speed_ na rozsah 5 až 500 otáček / min.
      int speed_ = map(val, 523, 1023, 5, 500);
```

```
      // nastavíme motoru rychlost
      stepper.setSpeed(speed_);

      // pootoč motorem o jeden krok zvoleným směrem
      stepper.step(1);

      val = analogRead(pot);
    }

    // chceme otáčet motorem opačným směrem
    while (val <= 500)
    {
      // mapujeme proměnnou speed_ na rozsah 5 až 500 otáček / min.
      int speed_ = map(val, 500, 0, 5, 500);
      // nastavíme motoru rychlost
      stepper.setSpeed(speed_);

      // pootoč motorem o jeden krok zvoleným směrem
      stepper.step(-1);

      val = analogRead(pot);
    }
  }
}
```

Knihovna AccelStepper

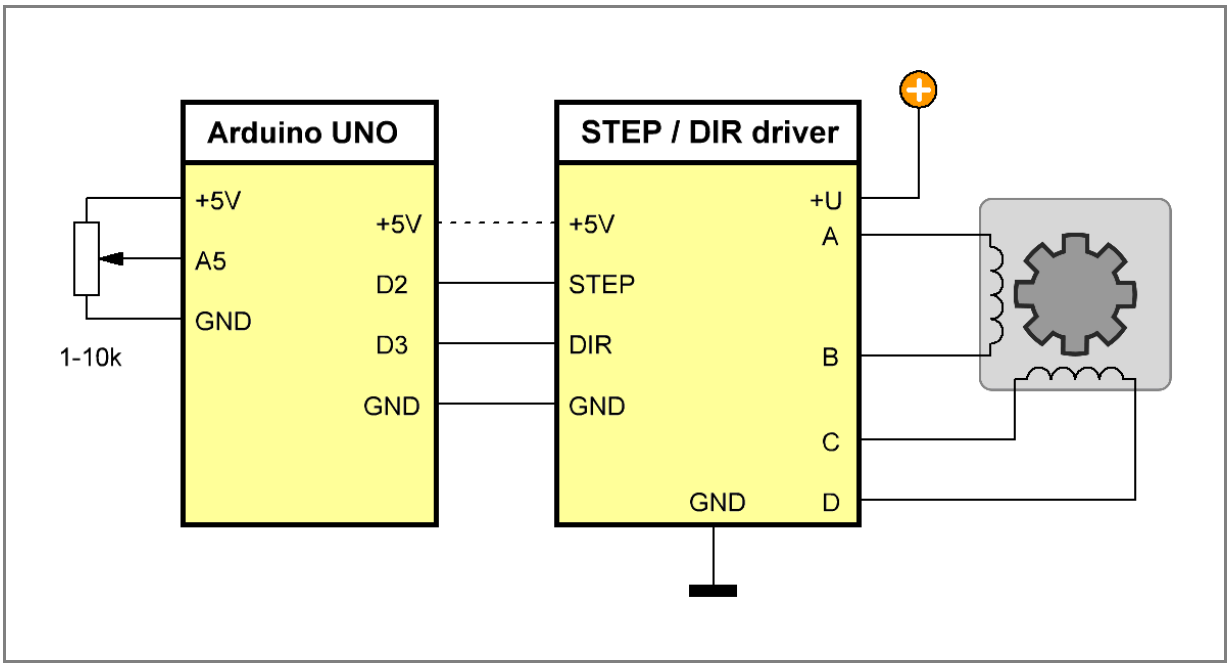
Nyní se podívejme na poněkud složitější řešení, které používá knihovnu AccelStepper.

Doporučené vybavení:

- Arduino UNO, doplněné Connector shieldem
- driver SMCB10 s obvodem A3982
- krokový motor velikosti NEMA 17
- knihovna AccelStepper

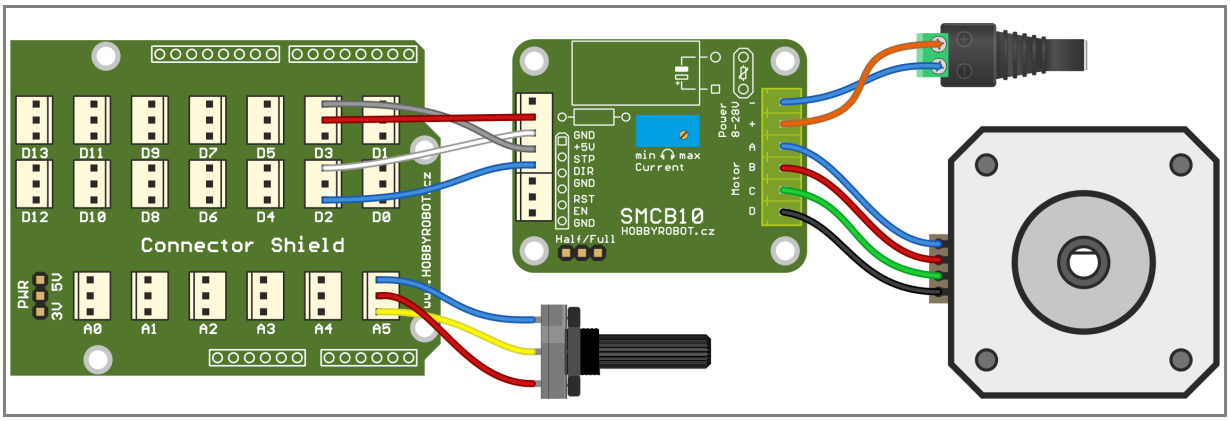
Program bude samozřejmě fungovat i s libovolným jiným driverem, řízeným signály STEP a DIR a jakýmkoli bipolárním krokovým motorem.

Blokové schéma:



Pokud nemá váš driver k dispozici výstup pro napájení externích zařízení s napětím 5 V, je nutno zjistit samostatné napájení Arduina napětím velikosti 7 až 12 V nebo ho trvale napájet z USB portu.

Názorný obrázek zapojení:



Program:

Program je určen pro spolupráci s drivery řízenými signály STEP a DIR.

Nepřehlédněte, že program používá knihovnu AccelStepper a pokud potřebujete motor připojit jiným způsobem, prostudujte nejprve příručku.

Potenciometr je možno připojit na kterýkoli z analogových vstupů Arduina, ale v programu je třeba patřičně upravit tuto definici:

SPEED_PIN 5

Maximální rychlost otáčení motoru určena konstantou:

MAX_SPEED 2000 (kroky za sekundu)

a minimální rychlost konstantou:

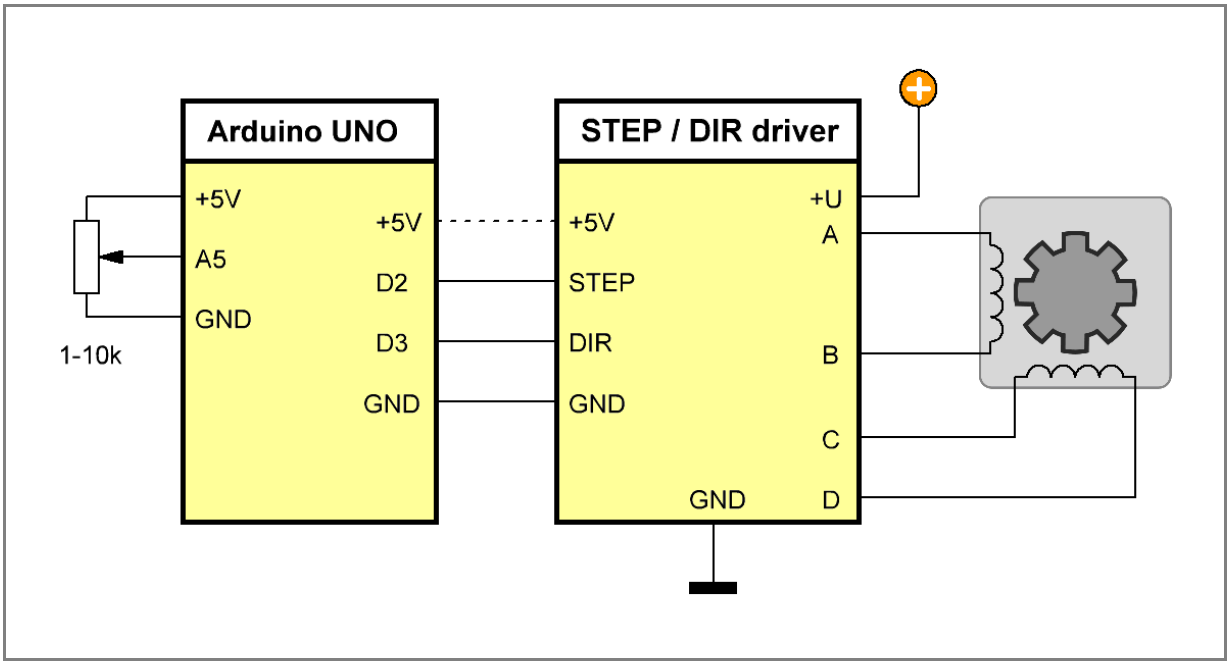
MIN_SPEED 0 (kroky za sekundu)

```
/*
StepperPotControl.ino
Jednosměrné řízení rychlosti otáčení krokového motoru potenciometrem
Driver STEP / DIR (SMCB10), motor MEMA 17 generic
JeDe robot s.r.o., říjen 2019
www.edurobot.cz
www.robodoupe.cz
*/

#include <AccelStepper.h>
// Knihovna AccelStepper

AccelStepper stepper1(AccelStepper::DRIVER, 2, 3);
// STEP (2), DIR (3)
// Nastavení druhu řídícího rozhraní a přiřazení pracovních pinů
// Viz příručka knihovny AccelStepper

// přiřazení pinů
#define  SPEED_PIN 5
// Potenciometr (1-10k) je připojen na vstup A5

// konstanty
#define  MAX_SPEED 2000
#define  MIN_SPEED 0
// Nastavení maxima a minima rychlosti otáčení motoru v krocích za sekundu

// proměnné
float current_speed = 0.0;        // Okamžitá rychlost otáčení motoru v krocích za sekundu
int analog_value = 0;             // Okamžitá hodnota napětí na analogovém vstup A5
                                  // určuje rychlost otáčení motoru.

// ZAČÁTEK PROGRAMU
void setup()
{
  // Knihovně AccelStepper předáme maximální rychlost otáčení motoru
```

```
  stepper1.setMaxSpeed(MAX_SPEED);
}

void loop()
{
  analog_value = analogRead(SPEED_PIN);
   // Přečteme hodnotu z pinu A5 (0 až 1023)…

  current_speed = (((analog_value/1023.0) * (MAX_SPEED - MIN_SPEED)) + MIN_SPEED);
  // … a přepočteme ji na rychlost otáčení motoru

  stepper1.setSpeed(current_speed);
  // Nastavíme novou rychlost otáčení…

  stepper1.runSpeed();
  // … a předáme driveru
}
```

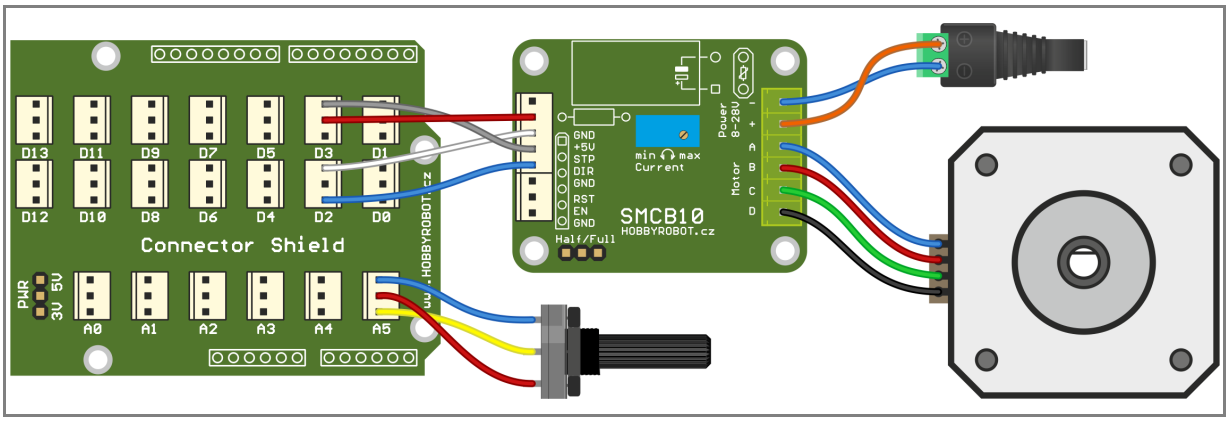
Program vylepšíme o obousměrné řízení s nulou (tedy zastaveným motorem) uprostřed dráhy potenciometru.

Blokové schéma:



Pokud nemá váš driver k dispozici výstup pro napájení externích zařízení s napětím 5 V, je nutno zjistit samostatné napájení Arduina napětím velikosti 7 až 12 V nebo ho trvale napájet z USB portu.

Názorný obrázek zapojení:



Program:

```
/*
StepperPotControlBidir.ino
Obousměrné řízení rychlosti otáčení krokového motoru potenciometrem
Driver STEP / DIR (SMCB10), motor MEMA 17 generic
JeDe robot s.r.o., říjen 2019
www.edurobot.cz
www.robodoupe.cz
*/

#include <AccelStepper.h>
// Knihovna AccelStepper

AccelStepper stepper1(AccelStepper::DRIVER, 2, 3);
// STEP (2), DIR (3)
// Nastavení druhu řídícího rozhraní a přiřazení pracovních pinů
// Viz příručka knihovny AccelStepper

// přiřazení pinů
#define  SPEED_PIN 5
// Potenciometr (1-10k) je připojen na vstup A5

// konstanty
#define  MAX_SPEED 2000
#define  MIN_SPEED 0
// Nastavení maxima a minima rychlosti otáčení motoru v krocích za sekundu

// proměnné
float current_speed = 0.0;        // Okamžitá rychlost otáčení motoru v krocích za sekundu
int analog_value = 0;             // Okamžitá hodnota napětí na analogovém vstup A5
                                  // určuje rychlost otáčení motoru.
```

```
// ZAČÁTEK PROGRAMU
void setup()
{
// Knihovně AccelStepper se předá maximální rychlost otáčení motoru
  stepper1.setMaxSpeed(MAX_SPEED);
}

void loop()
{
  analog_value = analogRead(SPEED_PIN);
  // Přečteme hodnotu z pinu A5 (0 to 1023)

  if (analog_value < 460)
  // Je-li přečtená hodnota menší než 460...
  {
    current_speed = map(analog_value, 0, 460, MAX_SPEED, MIN_SPEED);
    // ... změníme ji z rozsahu 0 až 460 na rozsah
    // daný konstantami MAX_SPEED a MIN_SPEED
  }
  else if (analog_value > 563)
  // Je-li přečtená hodnota větší než 563...
  {
    current_speed = (map(analog_value, 563, 1023, MIN_SPEED, MAX_SPEED))* -1;
    // ... změníme ji z rozsahu 563 až 1023 na rozsah
    // daný konstantami MIN_SPEED a MAX_SPEED
    // Vynásobením konstantou -1 se hodnota current_speed stane zápornou
    // a směr otáčení motoru se změní
  }
  else
  // je-li přečtená hodnota mimo výše uvedené rozsahy...
  {
    current_speed=0;
    // ... motor zastavíme
  }

stepper1.setSpeed(current_speed);
// Nastavíme novou rychlost otáčení...

stepper1.runSpeed();
// ... a předáme driveru
}
```

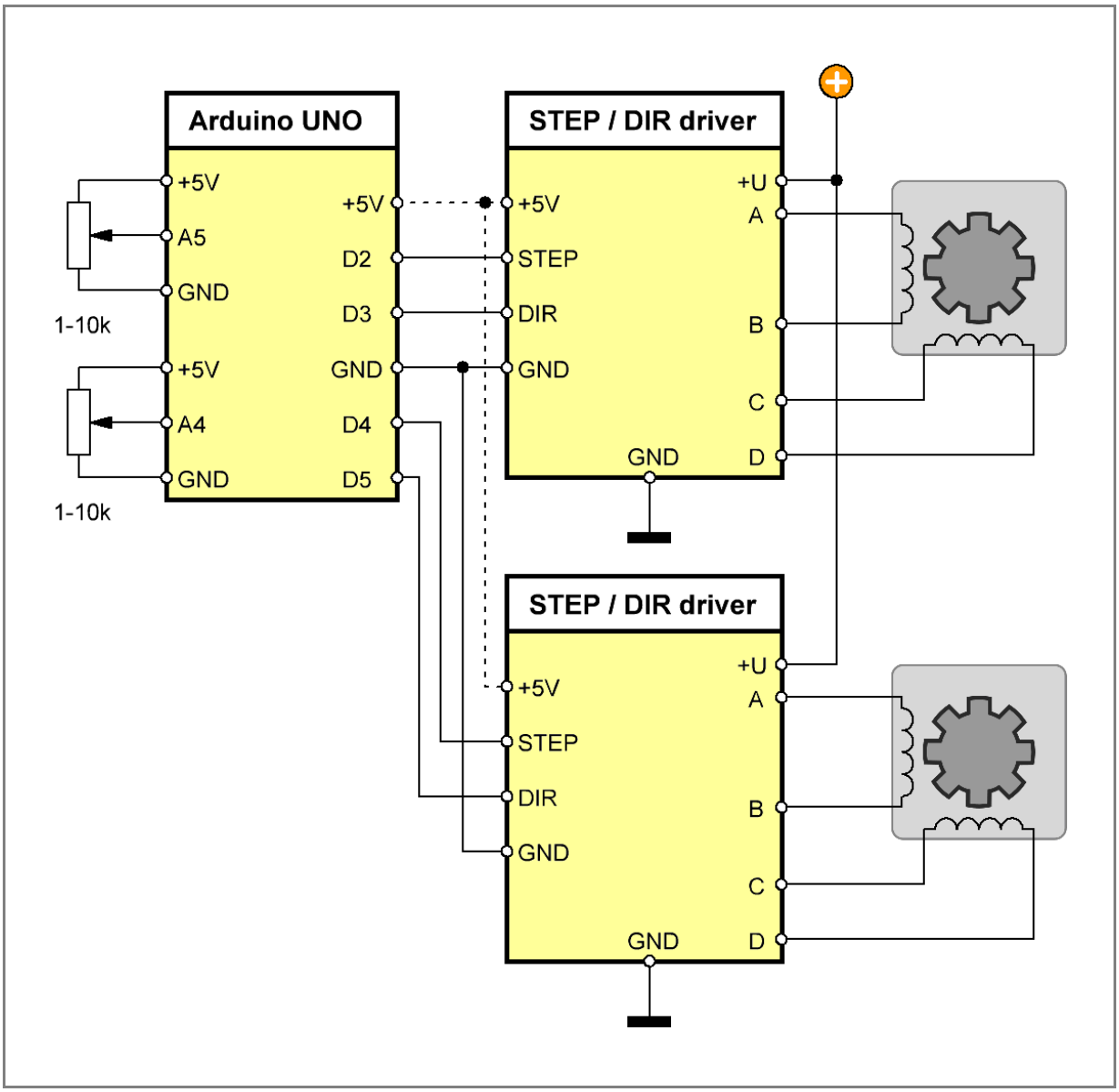
Nyní si ukážeme, že díky knihovně AccelStepper můžeme stejně řídit i více krokových motorů naráz a nezávisle na sobě.

Proti minulému pokusu do zapojení přibyl druhý driver krokového motoru a samozřejmě i motor.

Doporučené vybavení:

- Arduino UNO, doplněné Connector shieldem
- 2x driver SMCB10 s obvodem A3982
- 2x krokový motor velikosti NEMA 17
- knihovna AccelStepper

Blokové schéma:



Připomeňme si, že druhý driver krokového motoru je připojen na piny D4 (STEP) a D5 (DIR) Arduina a druhý potenciometr na analogový vstup A4.

S výhodou je možno pro řízení obou motorů použít miniaturní analogový joystick.

Program:

```
/*
TwoStepperPotControlBidir.ino
Nezávislé obousměrné řízení rychlosti otáčení dvou krokových motorů.
Rychlost otáčení je nastavována potenciometry
JeDe robot s.r.o., říjen 2019
www.edurobot.cz
www.robodoupe.cz
*/

#include <AccelStepper.h>
// Knihovna AccelStepper

AccelStepper stepper1(AccelStepper::DRIVER, 2, 3);
AccelStepper stepper2(AccelStepper::DRIVER, 4, 5);
// Nastavení druhu řídícího rozhraní
// a přiřazení pracovních pinů STEP (2, 4) a DIR (3, 5)
// Viz příručka knihovny AccelStepper

// Přiřazení pinů
#define  SPEED_PIN_1 5
// Potenciometr_1 je připojen na vstup A5
#define  SPEED_PIN_2 4
// Potenciometr_2 je připojen na vstup A4

//Definice konstant
#define  S1_MAX_SPEED 1500
#define  S1_MIN_SPEED 0
#define  S2_MAX_SPEED 1500
#define  S2_MIN_SPEED 0
// Nastavení maxima a minima rychlosti otáčení motoru v krocích za sekundu

// Definice proměnných
float current_speed_S1 = 0.0;
float current_speed_S2 = 0.0;
// Okamžité rychlosti otáčení motoru v krocích za sekundu
int analog_1 = 512;
// Okamžitá hodnota napětí na analogovém vstup A5
int analog_2 = 512;
// Okamžitá hodnota napětí na analogovém vstup A4
// určuje rychlost otáčení motoru.

// ZAČÁTEK PROGRAMU
void setup()
{
// Knihovně AccelStepper se předá maximální rychlost otáčení motorů
  stepper1.setMaxSpeed(S1_MAX_SPEED);
```

```
  stepper2.setMaxSpeed(S2_MAX_SPEED);
}

void loop()
{
  // první motor
  analog_1 = analogRead(SPEED_PIN_1);
  // Přečteme hodnotu z pinu A5 (0 až 1023)

  if (analog_1 < 460)
  // Je-li přečtená hodnota menší než 460...
  {
    current_speed_S1 = map(analog_1, 0, 460, S1_MAX_SPEED, S1_MIN_SPEED);
    // ... změníme ji z rozsahu 0 až 460 na rozsah
    // daný konstantami MAX_SPEED a MIN_SPEED
  }
  else if (analog_1 > 563)
  // Je-li přečtená hodnota větší než 563...
  {
    current_speed_S1 = (map(analog_1, 563, 1023, S1_MIN_SPEED, S1_MAX_SPEED))* -1;
    // ... změníme ji z rozsahu 563 až 1023 na rozsah
    // daný konstantami MIN_SPEED a MAX_SPEED
    // Vynásobením konstantou -1 se hodnota proměnné current_speed_S1
    // stane zápornou a směr otáčení motoru se změní
  }
  else
  // je-li přečtená hodnota mimo výše uvedené rozsahy...
  {
    current_speed_S1=0;
    // ... motor zastavíme
  }

stepper1.setSpeed(current_speed_S1);
// Nastavíme novou rychlost otáčení...

stepper1.runSpeed();
// ... a předáme driveru


// Druhý motor
  analog_2 = analogRead(SPEED_PIN_2);
  // Přečteme hodnotu z pinu A4 (0 až 1023)

  if (analog_2 < 460)
  // Je-li přečtená hodnota menší než 460...
  {
    current_speed_S2 = map(analog_2, 0, 460, S2_MAX_SPEED, S2_MIN_SPEED);
    // ... změníme ji z rozsahu 0 až 460 na rozsah
    // daný konstantami MAX_SPEED a MIN_SPEED
```

```
  }
  else if (analog_2 > 563)
  // Je-li přečtená hodnota větší než 563...
  {
    current_speed_S2 = (map(analog_2, 563, 1023, S2_MIN_SPEED, S2_MAX_SPEED))* -1;
    // ... změníme ji z rozsahu 563 až 1023 na rozsah
    // daný konstantami MIN_SPEED a MAX_SPEED
    // Vynásobením konstantou -1 se hodnota proměnné current_speed_S2
    // stane zápornou a směr otáčení motoru se změní
  }
  else
  // je-li přečtená hodnota mimo výše uvedené rozsahy...
  {
    current_speed_S2=0;
    // ... motor zastavíme
  }

stepper2.setSpeed(current_speed_S2);
// Nastavíme novou rychlost otáčení...

stepper2.runSpeed();
// ... a předáme driveru
}
```

Doposud jsme rychlost a směr otáčení krokových motorků řídili potenciometrem, tedy stejnosměrným napětím, přiváděným na analogový vstup Arduina. Nyní se podíváme, jak je možno krokový motor řídit modelářskými servopulsy.

Řízení modelářského servomechanismu:

Poloha výstupního hřídele serva odpovídá proporcionálně šířce řídícího impulzu. Řídící impulz je pozitivní s amplitudou od 3 V do napájecího napětí, aktivní dobou trvání proměnnou od 1 do 2 ms a opakovací frekvencí obvykle 50 Hz (Evropa) nebo 60 Hz (USA a Japonsko). U takzvaných digitálních serv ale může být opakovací frekvence i mnohem vyšší. Době trvání řídícího pulzu odpovídá rozsah polohy výstupního hřídele serva ±45°. Většina serv dovoluje zvětšit mechanický rozsah pohybu výstupní osy na ±90° zvětšením rozsahu řídících impulzů od 0,5 do 2,5 ms.

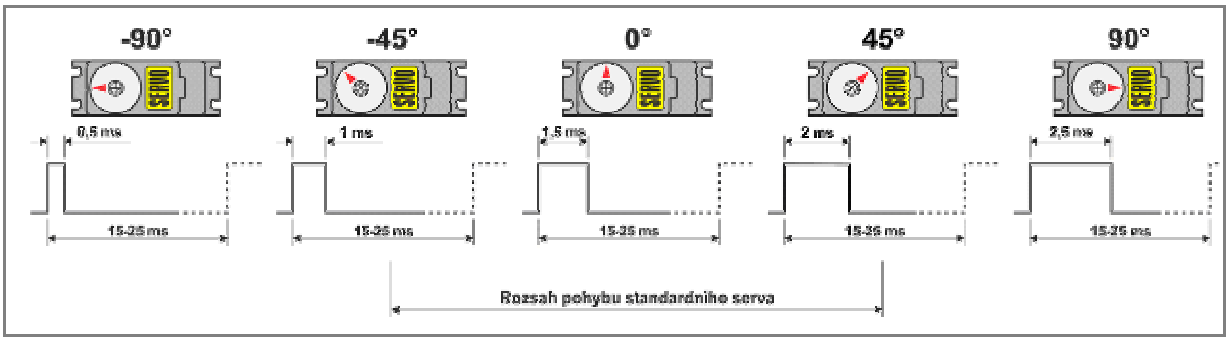
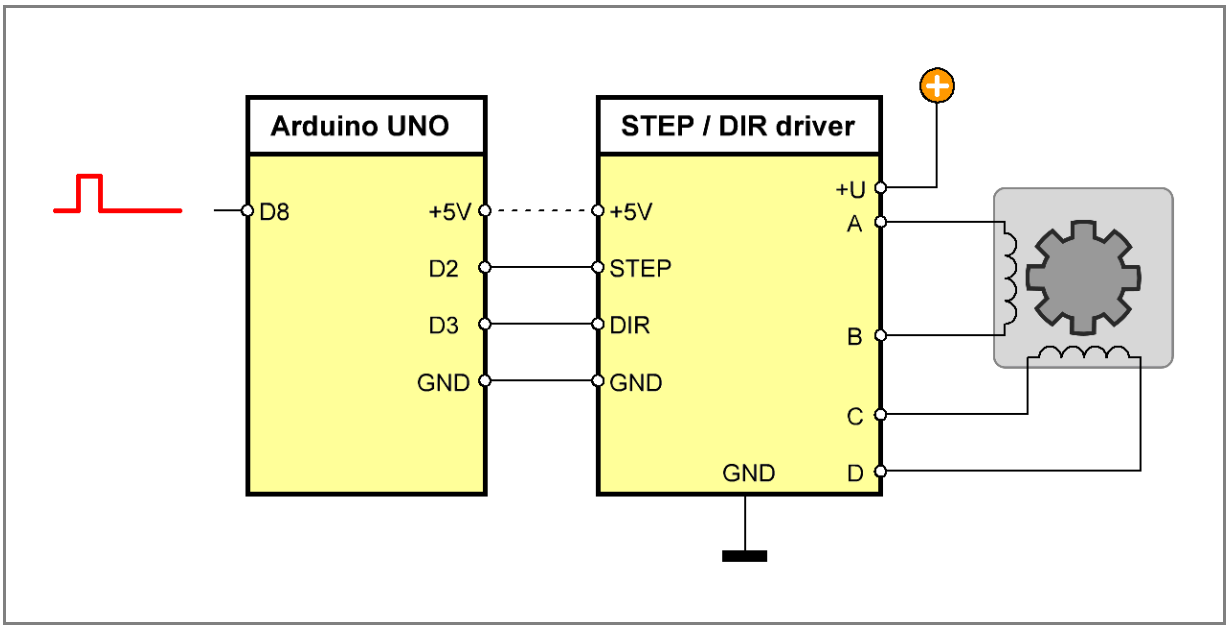


Schéma zapojení:



V tomto zapojení neřídíme rychlost otáčení krokového motoru potenciometrem (tedy analogovým signálem), ale digitálním pulsem s různou dobou trvání. Protože puls může přijít kdykoli v době běhu programu a mohli bychom ho zmeškat, případně naopak bychom si čekáním na jeho příchod blokovali činnost programu, je třeba použít přerušení. Ve vzorových programech je použita knihovna PinChangeInterrupt, která umožňuje sledovat události na všech pinech Arduina. Pro vstup servosignálu je v tomto případě použit pin D8, který tato knihovna testuje jako první a tím pádem má minimální časovou režii.

Program 1:

Jednosměrné řízení rychlosti otáčení krokového motoru servopulsem

```
/*
stepperServoPulseControl.ino
Jednosměrné řízení rychlosti otáčení krokového motoru servopulsem
Driver STEP / DIR (SMCB10), motor MEMA 17 generic
JeDe Robot s.r.o., říjen 2019
```

```
www.edurobot.cz
www.robodoupe.cz
*/

#include <AccelStepper.h>
// Knihovna AccelStepper (https://www.airspayce.com/mikem/arduino/AccelStepper/)
#include <PinChangeInterrupt.h>
// Knihovna PinChangeInterrupt (https://github.com/NicoHood/PinChangeInterrupt)

AccelStepper stepper1(AccelStepper::DRIVER, 2, 3);
// STEP (2), DIR (3)
// Nastavení druhu řídícího rozhraní a přiřazení pracovních pinů
// Viz příručka knihovny AccelStepper

// přiřazení pinů
#define pulse_pin 8
// Při testu událostí je tento pin první na řadě

// konstanty
#define  MAX_SPEED 2000
#define  MIN_SPEED 0
// Nastavení maxima a minima rychlosti otáčení motoru v krocích za sekundu

#define  MIN_PULSE_WIDTH 750
#define  MAX_PULSE_WIDTH 2265
// Minimální a maximální délka servopulsu v mikrosekundách

// proměnné
float current_speed = 0.0;        // Okamžitá rychlost otáčení motoru v krocích za sekundu
float analog_value = 0.0;         // Převod délky pulsu na rozsah 0 až 1023
                                   // určuje rychlost otáčení motoru.
int duration;                     // Doba trvání servopulsu

volatile unsigned long start = 0;
volatile float pulse_length = 0;

// ZAČÁTEK PROGRAMU
void setup()
{
  // Knihovně AccelStepper předáme maximální rychlost otáčení motoru
  stepper1.setMaxSpeed(MAX_SPEED);

  // nastavíme pin pro čtení servopulsu jako vstupní...
  pinMode(pulse_pin, INPUT);

  // ... a připojíme na něj obsluhu přerušení
  attachPinChangeInterrupt(digitalPinToPinChangeInterrupt(pulse_pin), onRising, CHANGE);
}
```

```
void loop()
{
  analog_value = map(pulse_length, MIN_PULSE_WIDTH, MAX_PULSE_WIDTH, 0, 1023);
  // změřenou délku servopulsu převedeme na rozsah 0 - 1023...

  current_speed = (((analog_value/1023) * (MAX_SPEED - MIN_SPEED)) + MIN_SPEED);
  // ... a přepočteme ji na okamžitou rychlost otáčení motoru

  stepper1.setSpeed(current_speed);
  // Nastavíme novou rychlost otáčení...

  stepper1.runSpeed();
  // ... a předáme driveru
}

// FUNKCE
void onRising()
{
  processPin();
}

void processPin()
{
  uint8_t trigger = getPinChangeInterruptTrigger(digitalPinToPCINT(pulse_pin));

if(trigger == RISING)
  {
    start = micros();
  }
else if(trigger == FALLING)
  {
    pulse_length = micros() - start;
  }
}
```

Program 2:

Obousměrné řízení rychlosti otáčení krokového motoru servopulsem

```
/*
StepperServoPulseControlBidir.ino
Obousměrné řízení rychlosti otáčení krokového motoru servopulsem
Driver STEP / DIR (SMCB10), motor MEMA 17 generic
JeDe Robot s.r.o., říjen 2019
www.edurobot.cz
www.robodoupe.cz
*/
```

```
#include <AccelStepper.h>
// Knihovna AccelStepper
#include <PinChangeInterrupt.h>
// Knihovna PinChangeInterrupt

AccelStepper stepper1(AccelStepper::DRIVER, 2, 3);
// STEP (2), DIR (3)
// Nastavení druhu řídícího rozhraní a přiřazení pracovních pinů
// Viz příručka knihovny AccelStepper

// přiřazení pinů
#define pulse_pin 8

// konstanty
#define  MAX_SPEED 2000
#define  MIN_SPEED 0
// Nastavení maxima a minima rychlosti otáčení motoru v krocích za sekundu

#define  MIN_PULSE_WIDTH 750
#define  MAX_PULSE_WIDTH 2265
// Minimální a maximální délka servopulsu v us

// proměnné
float current_speed = 0.0;        // Okamžitá rychlost otáčení motoru v krocích za sekundu
float analog_value = 0.0;         // Převod délky pulsu na rozsah 0 až 1023
                                  // určuje rychlost otáčení motoru.
int duration;                     // Doba trvání servopulsu

volatile unsigned long start = 0;
volatile float pulse_length = 0;

// ZAČÁTEK PROGRAMU
void setup()
{
  // Knihovně AccelStepper předáme maximální rychlost otáčení motoru
  stepper1.setMaxSpeed(MAX_SPEED);
  pinMode(pulse_pin, INPUT);
  attachPinChangeInterrupt(digitalPinToPinChangeInterrupt(pulse_pin), onRising, CHANGE);
}

void loop()
{
  analog_value = map(pulse_length, MIN_PULSE_WIDTH, MAX_PULSE_WIDTH, 0, 1023);

  if (analog_value < 460)
  // Je-li přečtená hodnota menší než 460...
  {
    current_speed = map(analog_value, 0, 460, MAX_SPEED, MIN_SPEED);
```

```
    // ... změníme ji z rozsahu 0 až 460 na rozsah
    // daný konstantami MAX_SPEED a MIN_SPEED
  }
  else if (analog_value > 563)
  // Je-li přečtená hodnota větší než 563...
  {
    current_speed = (map(analog_value, 563, 1023, MIN_SPEED, MAX_SPEED))* -1;
    // ... změníme ji z rozsahu 563 až 1023 na rozsah
    // daný konstantami MIN_SPEED a MAX_SPEED
    // Vynásobením konstantou -1 se hodnota current_speed stane zápornou
    // a směr otáčení motoru se změní
  }
  else
  // je-li přečtená hodnota mimo výše uvedené rozsahy...
  {
    current_speed = 0;
    // ... motor zastavíme
  }

  stepper1.setSpeed(current_speed);
  // Nastavíme novou rychlost otáčení…

  stepper1.runSpeed();
  // … a předáme driveru
}

// FUNKCE
void onRising()
{
  processPin();
}

void processPin()
{
  uint8_t trigger = getPinChangeInterruptTrigger(digitalPinToPCINT(pulse_pin));

if(trigger == RISING)
  {
    start = micros();
  }
else if(trigger == FALLING)
  {
    pulse_length = micros() - start;
  }
}
```