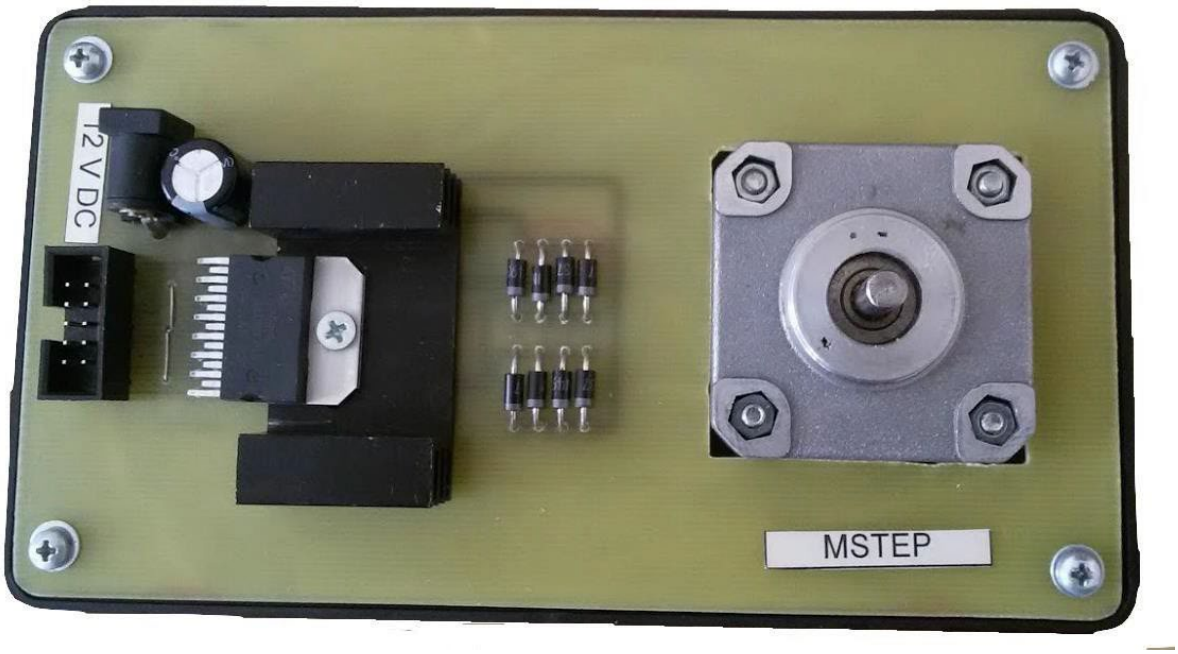
1 Krokový a stejnosměrný motor

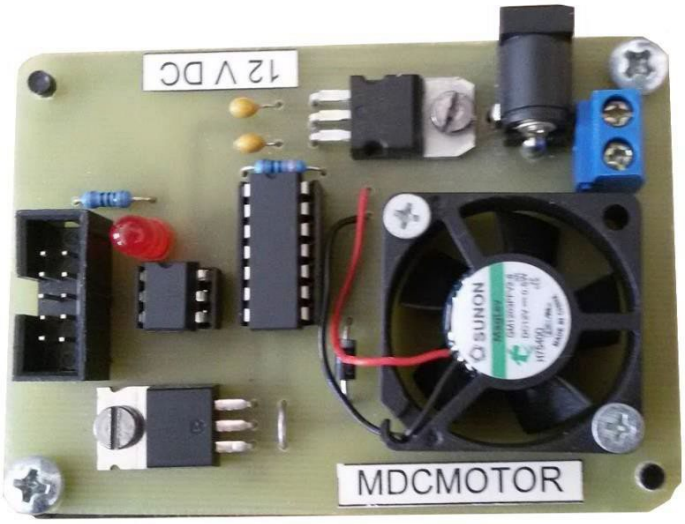
Na tomto cvičení se seznámíte s periférií krokového motoru a s periférií stejnosměrného motoru. Úkolem je pochopit princip činnosti a možnosti, jak uvedené periférie ovládat.

1.1 Přípravky pro výuku

Pro oba přípravky je ještě nutný stejnosměrný zdroj 12 V pro napájení motorků.



Obr. 1 modul krokového motoru

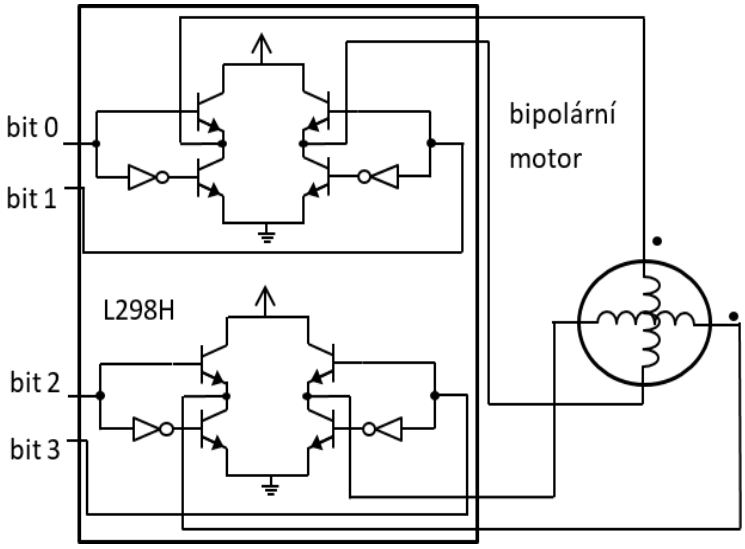


Obr. 2 Modul stejnosměrného motoru

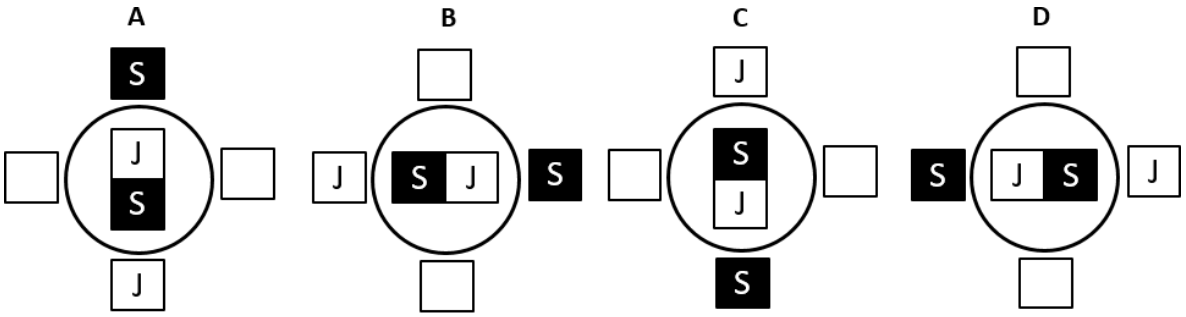
1.2 Bipolární krokový motor

Přípravek MSTEP obsahuje bipolární krokový motor ovládaný H-můstkem L298HN. Zjednodušené schéma zapojení je vidět na obrázku. H-můstek umožňuje obracet polaritu proudu v připojené cívce. Pokud bude bit 0 nastaven na logickou 1, bude sepnutý levý horní tranzistor a na začátek vertikální

cívky (označen tečkou) přivede napájecí napětí. Pokud bude současně bit 1 nastaven na logickou 0, bude sepnutý pravý dolní tranzistor v horním H-můstku a konec vertikální cívky se uzemní. Tím se nastaví polarita elektromagnetu jako na ukázce „A“ a dle toho se natočí i rotor. V tomto stavu jsou bity 2 a 3 nastaveny na logickou nulu a oba konce horizontální cívky jsou tedy uzemněny, cívkou neprochází proud a elektromagnet nemá určenu polaritu.



Obr. 3 principiální elektrické schéma



bit	0	1	2	3
stav	1	0	0	0

bit	0	1	2	3
stav	0	0	1	0

bit	0	1	2	3
stav	0	1	0	0

bit	0	1	2	3
stav	0	0	0	1

Obr. 4 princip činnosti krokového motoru

Pro otočení o jeden krok doprava se nastaví bit 2 do logické jedničky a bit 3 do logické nuly. Tím se pustí proud do horizontální cívky, a když se bity 0 a 1 nastaví do logické nuly, změní se rozložení dle stavu „B“. Střídáním stavů „A“, „B“, „C“ a „D“ dojde k postupnému otáčení motoru. Pro změnu směru otáčení stačí obrátit pořadí procházení stavů na „D“, „C“, „B“, „A“.

Podle obrázku by měl motor pouze 4 kroky, ve skutečnosti má motor v přípravku MSTEP 100 kroků na otáčku. Tento počet lze ještě zdvojnásobit vložením mezikroku. Mezikrok vznikne aktivováním dvou sousedních stavů a rotor tak ustálí mezi těmito stavy.

Arduino obsahuje třídu Stepper pro zjednodušení práce s krokovým motorem. Při inicializaci se zadává počet kroků motoru na otáčku a čísla pinů, na nichž jsou zapojeny ovládací signály. V případě zapojení přípravku MSTEP na port D by vypadala inicializace instance třídy Stepper následovně:

```
Stepper myStepper(100, 16,17,2,3);
```

Pro otočení motoru o „n“ kroků stačí zavolat metodu `void step(int n)` jejímž parametrem je počet kroků. Zadání záporného počtu kroků se bude motor otáčet opačným směrem.

Rychlost, s jakou se budou kroky provádět, se nastavuje metodou `void setSpeed(long rpm)`, která jako parametr přebírá počet otáček za minutu.

Třída Stepper udržuje rotor ve zmíněných mezistavech, vždy jsou aktivní dva sousední stavy. Sekvence pro otáčení je:

bit 0	bit 1	bit 2	bit 3
1	0	1	0
0	1	1	0
0	1	0	1
1	0	0	1

1.3 Stejnosměrný motor

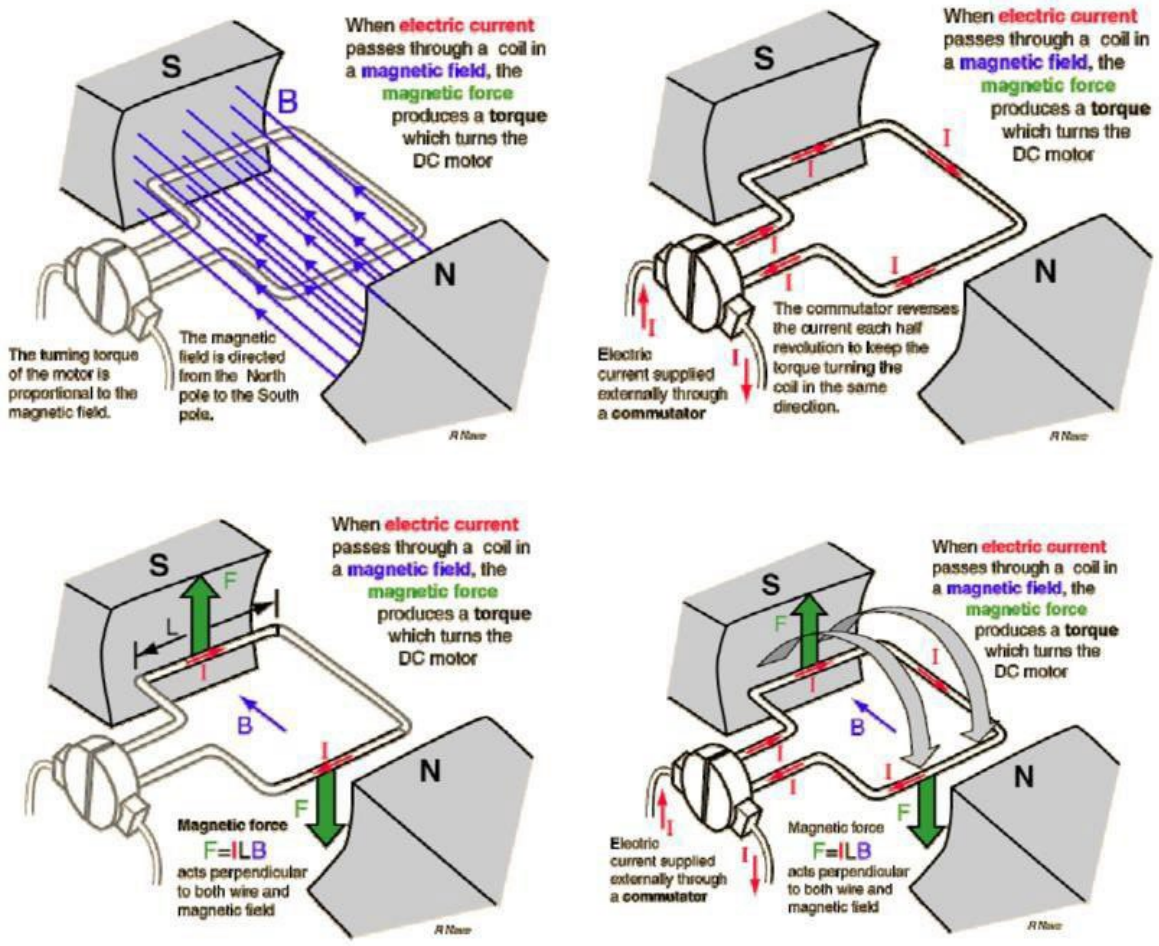
Ke svorkám motoru přivádíme stejnosměrný proud, který prochází vodiči kotvy. Protože se tyto vodiče nacházejí v magnetickém poli, působí na ně jistá síla a motor se otáčí. Směr otáčení lze určit např. pravidlem levé ruky. Kdyby však vodiči procházel trvale stejnosměrný proud, přestaly by se vodiče pohybovat po dosažení neutrální polohy - v ose mezi dvěma sousedními póly. Aby se kotva mohla otáčet dále původním směrem, musí se smysl proudu v nich po přechodu od jednoho pólu ke druhému změnit. Tuto změnu smyslu proudu ve vodičích rotorového vinutí obstarává komutátor.

V homogenním magnetickém poli mezi dvěma póly se otáčí závit, jehož začátek a konec jsou připojeny na dva kroužky, které se spolu s ním otáčejí. Magnetický tok, spřažený s otáčejícím se závitem, se mění s časem podle sinusovky a v závitu se indukuje střídavé napětí. Polarita obou kroužků se periodicky mění podle toho, zda vodič spojený s kroužkem je pod severním nebo jižním pólem. I proud, který prochází uzavřeným vnějším obvodem, je střídavý.

Komutátor u stejnosměrného motoru mění smysl proudu ve vodičích rotoru tak, že se rotor otáčí trvale jedním směrem.

Tažná síla motoru pulsuje, podobně jako pulsuje indukované napětí dynama. Čím více cívek má motor, tím je tah plynulejší. Zmenšuje se tím, že se některé cívky během komutace spojují kartáči nakrátko a nepřispívají k vytváření točivého momentu.

Při stejné polaritě pólů a při stejném smyslu proudu v kotvě je směr otáčení motoru a generátoru opačný, jak to plyne z obr 5., neboť při stejné polaritě pólů a při stejném směru otáčení je směr proudu v kotvě motoru a dynama opačný.

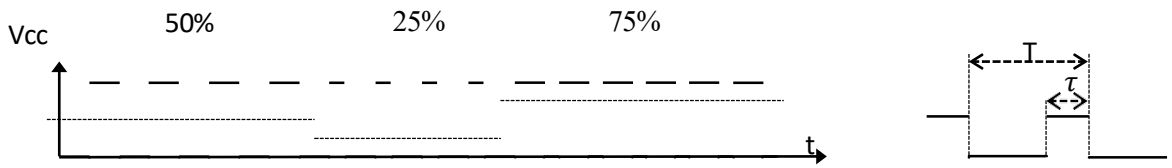


Obr. 5 princip stejnosměrného motoru

1.3.1 Řízení otáček stejnosměrného motoru – PWM

Otáčky stejnosměrného motoru jsou závislé na velikosti napájecího napětí. Snížení otáček motoru lze tedy dosáhnout snížením napájecího napětí. Tím však také klesne jeho tažná síla. Toto lze částečně kompenzovat tzv. pulzně šířkovou modulací (PWM: Pulse-Width Modulation). Do motoru budou zasílány krátké impulsy s amplitudou napájecího napětí. Čím větší bude poměr délky impulsu a délky periody signálu (délka pracovního cyklu) tím rychleji se bude motor otáčet. Pokud bude frekvence signálu rychlejší než časová konstanta motoru, bude se motor otáčet rychlostí, jakou by se otáčel při připojení napájení odpovídající střední hodnotě vstupního signálu. Výhodou je to, že v době kladné hodnoty impulsu táhne motor maximální silou.

Na obrázku jsou ukázány tři délky pracovního cyklu (duty cycle) 50%, 25% a 75%. Čárkovaně je zvýrazněna střední hodnota signálu, která udává i otáčky motoru.



Obr. 6 princip pulzně šířkové modulace

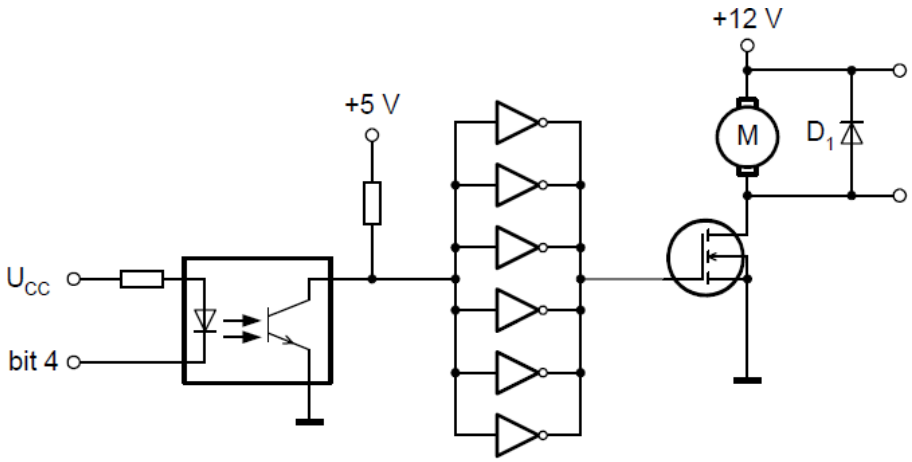
Pracovní cyklus (D) se vyjadřuje jako poměr délky periody signálu (T) vůči délce trvání logické jedničky (τ) vynásobený 100%.

$$D = \frac{\tau}{T} * 100\%$$

Stejným způsobem lze regulovat například i jas LED diody. Zde se využívá setrvačnosti lidského zraku, který při dostatečné frekvenci přestane vnímat blikání a změnu pracovního cyklu vnímá jako změnu jasu LED diody. PWM regulace využívají i spínané napájecí zdroje, které mají na výstupu kondenzátor vyhlazující a průměrující výstupní napětí.

Arduino umožňuje snadné generování PWM signálu prostřednictvím funkce `void analogWrite(char pin, int val)`. Parametr `pin` může být jeden z pinů 3, 5, 6, 9, 10, 11. To jsou piny, na kterých jsou výstupy některého z čítačů/časovačů mikrokontroléru přímo hardwarově podporující PWM výstup. Parametr `val` představuje délku pracovního cyklu. Může nabývat hodnoty 0 až 255, přičemž 0 znamená stabilní nulový výstup a 255 stabilní jedničkový výstup. Hodnota 128 představuje střídu 1:1 (pracovní cyklus 50%).

Přípravek MDCMOTOR obsahuje jednoduchý budič pro řízení stejnosměrného motoru. Signál z bitu 4 (pátý bit) ovládá LED v optočlenu, optočlen zajišťuje galvanické oddělení motoru od procesoru. Výstupní signál z tranzistoru v optočlenu je posílen a invertován a ovládá spínací MOSFET. Motor se točí v případě, že na bitu 4 je log. 0. Dioda chrání tranzistor před přepěťovými špičkami, které produkuje motor jako zátěž induktivního charakteru. Protože bit 4 není jedním z bitů, na kterých je možné generovat PMW signál, je na přípravku bit 4 propojen s bitem 3, který PWM podporuje.



Obr. 7 elektrické schéma stejnosměrného motoru

2 Vzorové příklady

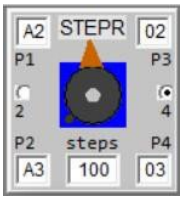
Uvedené vzorové příklady jsou jen základem pro ovládání příslušných periférií. Jde jen o základní principy, které můžete použít při řešení dalších úloh.

2.1 Krokový motor

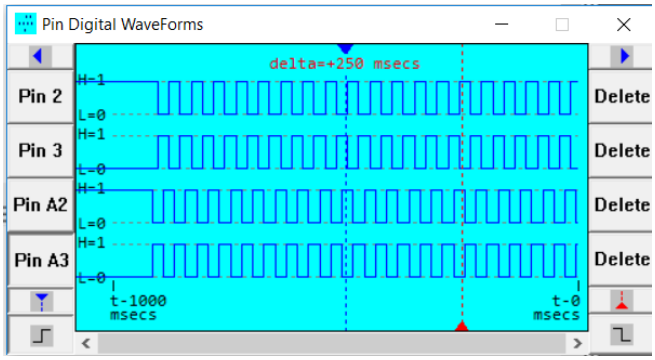
Uvedený program je připravený pro použití s periférií MSTEP. Tento přípravek je třeba ještě napájet externím zdrojem napětí 12V. Periférii propojte přes PortD. Zkontrolujte, zda jsou, na konektorové desce propojky pinů 0 a 1 nastaveny na piny 16 a 17 mikrokontroléru. Do mikrokontroléru nahrajte níže uvedený kód. Lze jej načíst z LMS MOODLE soubor MSTEP.ino

```
#include <Stepper.h>
// pocet kroku na otacku dle vaseho krokoveho motoru
const int steps = 100;

// inicializace krokoveho motoru myStepper
Stepper myStepper(steps, 16,17,2,3);

int stepCount = 100;         //pocet kroku

void setup() {
      myStepper.setSpeed(50);


}

void loop() {

      myStepper.step(stepCount);
      delay(1000);
      myStepper.step(-stepCount);
      delay(1000);

}
```

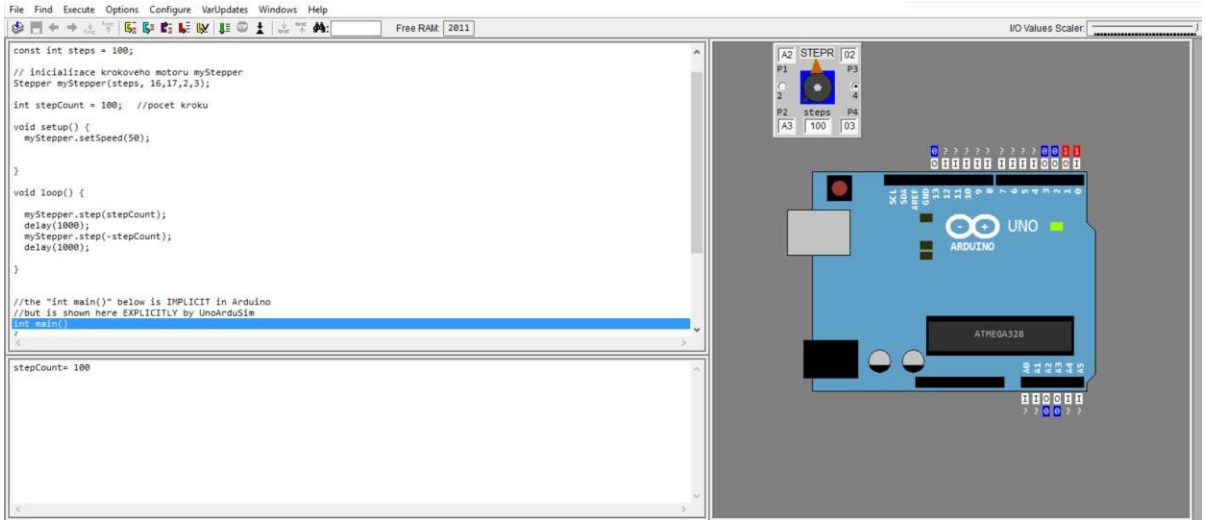
2.1.1 Krokový motor pomocí UnoArduSim



Toto V/V zařízení emuluje 6V bipolární nebo unipolární krokový motor s integrovaným řadičem poháněným buď pomocí dvou (na P1, P2) nebo čtyř (na P1, P2, P3, P4) řídicích signálů. Mužete také nastavit počet kroků na otáčku. Pro ovládání krokového motoru doporučuji použít knihovnu „Stepper.h“ a využít tak funkce „setSpeed ()“ a „step ()“ pro ovládání krokového motoru. Po spuštění aplikace UnoArduSim je třeba načíst zdrojový kód aplikace, to uděláte pomoci menu „File“ a volby „Load INO or PDE Prog..“ a načtete soubor MSTEP.ino. Po načtení programu je ještě třeba nahrát HW konfiguraci. To můžete udělat buď ručně pomocí menu „Configure“ a volby „I/O devices“ nebo následným načtením přes tlačítko „Load“ je uložena v souboru MSTEP_HW.txt.



Obr. 8 průběh signálů na jednotlivých pinech krokového motoru



Obr. 9 Krokový motor pomocí UnoArduSim

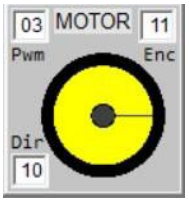
2.2 Stejnosměrný motor

Uvedený program je připravený pro použití s periférií MDCMOTOR. Tento přípravek je třeba ještě napájet externím zdrojem napětí 12V. Periférii propojte přes PortD. Zkontrolujte, zda jsou, na konektorové desce propojky pinů 0 a 1 nastaveny na piny 16 a 17 mikrokontroléru. Do mikrokontroléru nahrajte níže uvedený kód. Lze jej načíst z LMS MOODLE soubor MDCMOTOR.ino

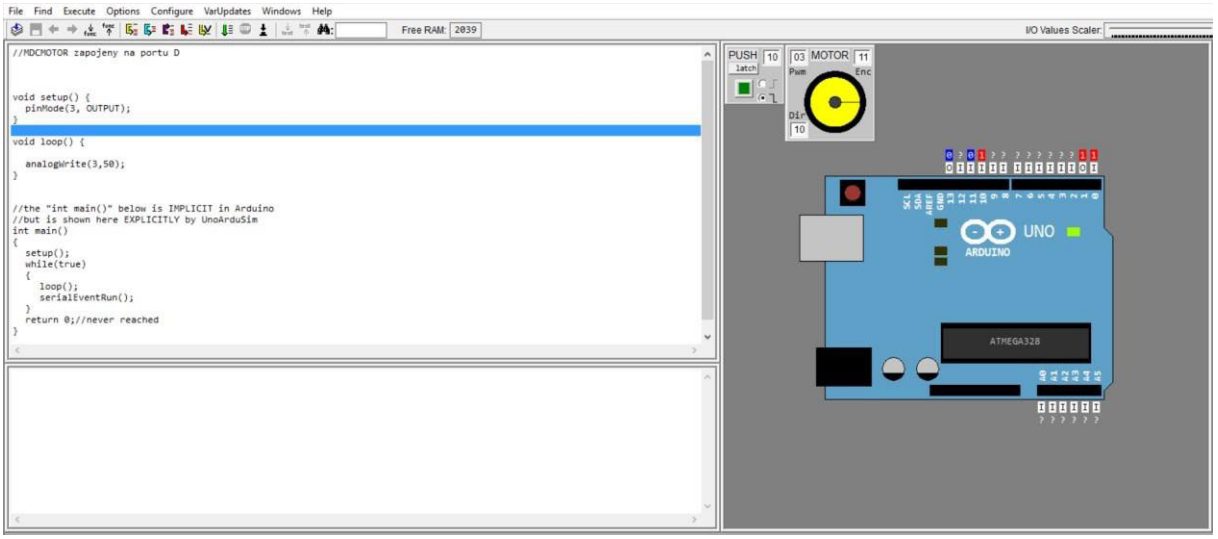
```
//MDCMOTOR zapojeny na portu D



void setup() {
     pinMode(3, OUTPUT);
}

void loop() {

     analogWrite(3,50);
}
```

Uvedený program roztočí stejnosměrný motor.

2.2.1 Stejnosměrný motor pomocí UnoArduSim



Toto V/V zařízení emuluje stejnosměrný motor napájený 6 V s integrovaným regulátorem řízení poháněné pulzně-šířkovou modulací (na vstupu PWM), a řízením směru otáčení (na vstupu Dir). DC Motor je připojený na rotační enkodér, který generuje výstupní signál (na výstu Enc). Pro řízení doporučuji použít funkci „analogWrite ()“. Pro řízení DC motoru lze použít pin s PWM 490 Hz (na pinech 3,9,10,11) nebo 980 Hz (na pinech 5,6). DC Motor je přesně modelován jak mechanicky tak i elektricky, proto na rozběh motoru je třeba dát větší rychlost než při postupném snižování rychlosti. Enkodér na DC motoru umožňuje rozlišit rotační změny s přesnosti 22.5°. Pro ovládání směru otáčení DC motoru je připojen přepínač ke vstupu Dir. Po spuštění aplikace UnoArduSim je třeba načíst zdrojový kód aplikace, to uděláte pomoci menu „File“ a volby „Load INO or PDE Prog..“ a načtete soubor MDCMOTOR.ino. Po načtení programu je ještě třeba nahrát HW konfiguraci. To můžete udělat buď ručně pomocí menu „Configure“ a volby „I/O devices“ nebo následným načtením přes tlačítko „Load“ je uložena v souboru MDCMOTOR_HW.txt.



Obr. 10 Stejnosměrný motor pomocí UnoArduSim

3 Cvičení 4 pro Arduino UNO

Odevzdávejte jednotlivé úkoly do samostatného adresáře. A do každého z adresářů nahrajte tři soubory. Zdrojový kód (*.ino) konfiguraci MCU (myArduPrefs.txt) a konfiguraci periférií (myIODevs.txt)

3.1 Úkol 7 – krokový motor s využitím třídy Stepper

Vytvořte program s využitím třídy Stepper, který bude po sériové lince přijímat číslo reprezentující počet kroků motoru. Kladné číslo pro otáčení v jednom směru, záporné pro opačný směr. Po přijetí čísla se motor v přípravku MSTEP otočí o zaslaný počet kroků rychlostí 30rpm a zůstane stát. Přípravek MSTEP připojte na port D, počet kroků na otáčku je 100, piny pro ovládání jsou: 16,17,2,3. Pro načtení textu ze sériové linky můžete využít kód z předchozích cvičení. (V případě, že používáte jen aplikaci UnoArduSim piny si zvolte dle potřeby.)

3.2 Úkol 8 – stejnosměrný motor řízený přes PWM z funkce analogWrite

Připojte přípravek MDCMOTOR k portu D. Ze sériové linky načtěte číslici -100 až 100, která bude představovat rychlost otáčení motoru. Hodnota 0 bude pro zastavený motor, hodnota -100 a 100 bude pro maximální rychlost v daném směru. Pomocí funkce `analogWrite` generujte na pinu 3 PWM signál regulující otáčky motoru. Motor se však bude roztáčet postupně na danou rychlost. Například bude-li stát (což na začátku programu bude) tak při zaslání hodnoty 100 bude po dobu 2 sekund motor postupně zvyšovat otáčky až na hodnotu 100 (tedy maximální rychlost).