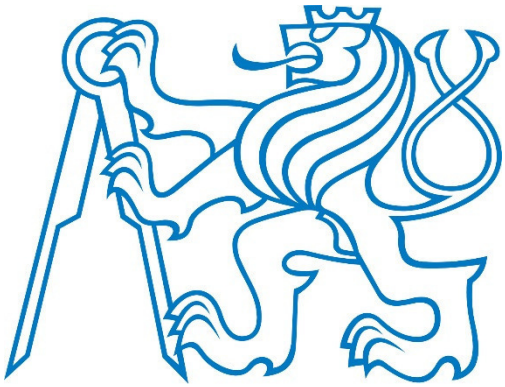
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta strojní

Ústav přístrojové a řídicí techniky



BAKALÁŘSKÁ PRÁCE

Návrh servopohonu se stěračovým motorem

Autor: Jan Krofta

Vedoucí práce: doc. Ing. Martin Novák Ph.D.

Praha 2017

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Krofta** Jméno: **Jan** Osobní číslo: **423361**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídící techniky**

Studijní program: **Teoretický základ strojního inženýrství**

Studijní obor: **bez oboru**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Návrh servopohonu se stěračovým motorem

Název bakalářské práce anglicky:

Design of a servo with a windshield wiper motor

Pokyny pro vypracování:

-Navrhněte koncept servopohon se stěračovým motorem
-Navrhněte a zapojte elektroniku (výkonová elektronika + řízení) s deskou Arduino
-Servopohon ověřte

Seznam doporučené literatury:

HOFREITER, Milan. Základy automatického řízení: příklady. 4. přepracované vydání. V Praze: České vysoké učení technické v Praze, 2016. 123 stran. ISBN 978-80-01-05899-2.
UHLÍŘ, Ivan a kol. Elektrické stroje a pohony. Vyd. 2. přeprac. Praha: ČVUT, 2007. 137 s. ISBN 978-80-01-03730-0.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

doc. Ing. Martin Novák Ph.D., ústav přístrojové a řídící techniky FS

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **19.04.2017** Termín odevzdání bakalářské práce: **16.06.2017**

Platnost zadání bakalářské práce: _____

Podpis vedoucí(ho) práce	Podpis vedoucí(ho) ústavu/katedry	Podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. _____ Datum převzetí zadání

_____ Podpis studenta

PROHLÁŠENÍ

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně s tím, že její výsledky mohou být dále použity podle uvážení vedoucího bakalářské práce jako jejího spoluautora. Souhlasím také s případnou publikací výsledků bakalářské práce nebo její podstatné části, pokud budu uveden jako její spoluautor. Dále prohlašuji, že všechny použité prameny jsou citovány.

V Praze dne…… Podpis ……

ANOTACE

Tato bakalářská práce se zabývá návrhem funkčního servopohonu za použití automobilového stěračového motoru a vývojové desky Arduino. V teoretické části jsou rozebrány jednotlivé prvky servopohonu a je zde stručně popsán princip PID regulátoru a PWM řízení rychlosti otáčení. Rovněž se zaobírám možnými způsoby uspořádání více servomotorů v rámci jednoho celku. V praktické části popisuji, jak jsem sám postupoval při návrhu zadaného servomotoru. Zabývám se výběrem vhodného enkodéru, potřebnými konstrukčními úpravami stěračového motoru, zapojením všech součástí a popisem vlastního obslužného programu. Na závěr předkládám průběhy odezvy mého servomotoru při různých nastaveních PID regulátoru a rovněž navrhuji možné způsoby zdokonalení tohoto konceptu.

KLÍČOVÁ SLOVA

servomotor, servo, stěračový motor, PID regulátor, PWM, Arduino, rotační enkodér

ABSTRACT

This bachelor thesis deals with construction of servomotor using windshield wiper motor and Arduino development board. In theoretical section of this paper I describe components of servomotor and the principle of PID regulator and PWM control of rotation speed. I also address the issue of topology of multiple servomotors included in one system. In practical section of this thesis I describe how I proceeded with construction of the servomotor. Topics such as choosing the right rotary encoder, modifying the wiper motor, connecting all parts together and writing code for Arduino board are covered. Response curves of the final design are shown and discussed. Lastly, I include thoughts on future possible improvements of my design.

KEYWORDS

servomotor, servo, windshield wiper motor, PID regulator, PWM, Arduino, rotary encoder

OBSAH

PŘEHLED POUŽITÝCH ZKRATEK

DC	Direct Current – stejnosměrný proud
PWM	Pulse Width Modulation – pulzně šířková modulace
CW	Clockwise – po směru hodinových ručiček
CCW	Counter clockwise – proti směru hodinových ručiček
USB	Universal Serial Bus – univerzální sériová sběrnice
TTL	Transistor–transistor logic – tranzsitorově-tranzistorová logika
UART	Universal Asynchronous Reciever/Transmitter – Asynchronní sériové rozhraní
IDE	Integrated Development Environment – integrované vývojové prostředí
ICSP	In-Circuit Serial Programming – Programování uvnitř systému
A/D	Analogově-digitální převodník
PID	Proporcionálně-integračně-derivační regulátor
PSD	Proporcionálně-sumačně-diferenční regulátor

1 ÚVOD

1.1 CÍL PRÁCE

Tématem mojí bakalářské práce je konstrukce funkčního servomotoru s použitím běžného automobilového stěračového motoru. Nejprve se budu stručně zabývat teorií servomotoru, popíši jeho jednotlivé části a princip regulace. V praktické části se budu věnovat praktickému návrhu a konstrukci vlastního zadaného servomotoru včetně obslužného programu pro vývojovou desku Arduino. Pokud se koncept servomotoru se stěračovým motorem osvědčí, mohl by být použit pro realizaci některých pohybových stupňů volnosti humanoidního robota. Tento robot je stavěn na ústavu přístrojové a řídící techniky pod vedením vedoucího mé práce, doc. Ing. Martinem Novákem Ph.D.

1.2 SERVOMECHANISMUS

Slovo „servo“ pochází z latiny, v překladu znamená „sloužit“. Servomechanismus je obecně systém využívající zpětné vazby k vykonání zásahu, díky kterému na výstupu nastane žádaný stav [1]. Výstupem tedy nemusí být pouze změna polohy (ať už úhlové nebo lineární), ale například i změna teploty či vlhkosti, průtoku, tlaku apod. [1]. V této práci se budu však zabývat vytvořením servopohonu ze stěračového motoru. Výstupem servomechanismu tedy bude hřídel natáčející se do poloh zadaných uživatelem.

1.3 SERVOMOTOR

Servomotor je rotační nebo lineární aktuátor, tedy mechanický prvek zajišťující kontrolovaný pohyb mechanismu. Skládá se z motoru (elektrického, pneumatického, hydraulického či piezoelektrického), čidla polohy a ovladače zajišťujícího požadovaný pohyb motoru. Servomotor na rozdíl od obyčejného motoru umožňuje přesné řízení polohy (úhlové nebo lineární). Od krokového motoru se liší principem řízení. Klasické krokové motory jsou řízeny elektrickými impulsy, kdy jeden impuls znamená změnu natočení hřídele o konstantní úhel, tzv. krok. Krokové motory tedy nepotřebují čidlo polohy, neboť jejich poloha je určena právě počtem pulzů a jím odpovídajících kroků. Jejich nevýhodou je, že když se překoná moment, kterým působí proti změně své polohy vnějším působením, dojde k tzv. přeskočení kroku a absolutní údaj o poloze se ztratí. Proto jsou na trhu i krokové motory se zabudovaným čidlem polohy, tedy enkodérem, které riziko ztráty údaje o poloze eliminují.

2 ČÁSTI SERVOMOTORU

2.1 MOTOR

Pro tuto bakalářskou práci je nejdůležitější elektrický motor, konkrétně stejnosměrný komutátorový motor s permanentními magnety ve statoru, avšak v průmyslu se hojně využívají i synchronní motory s permanentními magnety v rotoru anebo asynchronní motory s kotvou na krátko.

2.2 STĚRAČOVÝ MOTOR

Za stěračový motor se považuje jeden celek většinou komutátorového DC motoru, šnekové převodovky a doběhového vypínače. V souladu s předpisy musí být stěračový motor minimálně dvourychlostní. Pro komutátorový DC motor připadá v úvahu řízení rychlosti pomocí třetího pomocného kartáče, odporově (nevyužívá se pro vysoké ztráty) či elektronicky metodou PWM (nejdokonalejší, elektronická reverzace snižuje nároky na prostor, umožňuje plynulou změnu rychlosti). U klasického dvourychlostního stěračového motoru je samočinné doběhnutí do parkovací polohy řešeno spínačem mechanicky spojeným s motorem, který ve vhodnou chvíli nejenže odpojí motor od napájení, ale i zkratuje jeho vývody, čímž dojde k účinnému brzdění [2] [3].

2.3 SNÍMAČ POLOHY

Snímače polohy se dělí na absolutní a inkrementální. Signál vycházející z absolutního snímače polohy udává přesnou polohu motoru (např. 278°) ihned po připojení zdroje energie ke snímači. Inkrementální snímač polohy měří pouze přírůstek polohy (např. pootočení o 10°), pro získání absolutní polohy je nutné natočit hřídel motoru do referenčního bodu a potom přičítat změřené přírůstky [4].

Jednoduchým příkladem absolutního snímače polohy je potenciometr, odpor mezi jezdcem a vývodem odporové vrstvy absolutně určuje polohu jezdce. V průmyslu se často využívá optický snímač polohy, který využívá Grayův kód. Grayův kód je binární číselná soustava, kde se každé dvě po sobě jdoucí hodnoty liší v bitovém vyjádření změnou pouze jedné pozice. Pro n-bitový rotační absolutní optický snímač polohy je nejmenší krok $360°/2^n$ [4].

Výstupem inkrementálního snímače polohy jsou dva signály a velikost a směr přírůstku se určuje z jejich vzájemného fázového posuvu. Pracují na mechanickém, optickém či magnetickém principu [4].

2.4 ŘÍDÍCÍ ELEKTRONIKA

Ovládacím algoritmem je zpětnovazební smyčka, která porovnává aktuální polohu hřídele motoru s polohou požadovanou. Na základě rozdílu těchto hodnot (regulační odchylky) vyvozuje akční zásah, kterým chybu minimalizuje. V nejjednodušším případě je řízení pouze třípolohové (motor vypnutý, otáčení CW a CCW). Nevýhodou tohoto řešení je malá přesnost a zvýšené mechanické i elektrické namáhání servopohonu. Lepším řešením je použití PID regulátoru, jehož detailnější popis je níže. Rychlost motoru je v případě stejnosměrného komutátorového motoru řízena metodou PWM, v případě asynchronního motoru např. pomocí frekvenčního měniče. Směr otáčení se u stejnosměrného motoru řídí H-můstkem.

2.4.1 PID regulátor

Proporcionálně integračně derivační, zkráceně PID regulátor, je velmi široce používaný řídící mechanismus s negativní zpětnou vazbou. Do regulátoru vstupuje žádaná hodnota řízené proměnné, anglicky označovaná jako setpoint a její aktuální hodnota. Regulátor nejprve vypočítá regulační odchylku jako rozdíl těchto dvou hodnot.

$$e(t) = s(t) - y(t) \tag{1}$$

Kde $e(t)$ je regulační odchylka, $s(t)$ je žádaná hodnota (setpoint) a $y(t)$ je aktuální hodnota řízené veličiny. Výstupem regulátoru je akční zásah u(t) který se ve spojitém čase vypočítává podle rovnice (2).

$$u(t) = K_p e(t) + K_i \int_0^t e(t)\,dt + K_d \frac{de(t)}{dt} \tag{2}$$

Kde K_p je zesílení, K_i je integrační časová konstanta a K_d je derivační časová konstanta. Výstup PID regulátoru je tedy součtem výstupů třech dílčích částí.

2.4.1.1 Proporcionální regulátor

První část PID regulátoru se nazývá proporcionální. Aktuální regulační odchylku pouze násobí konstantou K_p, která se označuje jako zesílení. Výstup je tedy přímo úměrný vstupu.

Jeho samostatné použití vede k regulaci s takzvanou trvalou regulační odchylkou (TRO). Takový případ nastává, pokud velikost výstupu nedostačuje k další úpravě regulované veličiny. Příliš vysoká hodnota zesílení vede k rozkmitání regulované veličiny [5].

2.4.1.2 Integrační regulátor

Druhá část se nazývá integrační a jak vyplývá z jeho předpisu, výstup je přímo úměrný „součtu“ předchozích regulačních odchylek. Smyslem této složky je tedy právě eliminace TRO. Konstantou úměrnosti je integrační konstanta. Změnou její hodnoty se mění příspěvek této složky do celého regulátoru. Příliš vysoká hodnota integrační konstanty opět způsobuje rozkmitání regulované veličiny [5].

2.4.1.3 Derivační složka

Třetí část není sama o sobě použitelným regulátorem, reaguje totiž na časovou změnu (derivaci) regulační odchylky. Při ustálení odchylky je její výstup nulový a regulace neprobíhá. Pokud se regulační odchylka snižuje, resp. zvyšuje, je výstupem D složky záporná, resp. kladná hodnota, přímo úměrná rychlosti změny odchylky. Pokud se tedy aktuální hodnota řízené veličiny blíží žádané, záporná derivační složka tlumí výstup regulátoru, čímž snižuje riziko překmitu. Zvyšování derivační konstanty tedy může zlepšit rychlost odezvy regulátoru a zlepšit stabilitu. Ze své podstaty však D složka zesiluje šum, a i proto není její použití příliš časté [5].

Pokud je nějaká část v regulátoru nepřítomná, pak se takový regulátor nazývá P regulátor (není přítomná I a D složka), PI regulátor (není přítomná D složka) anebo méně častý PD regulátor (není přítomná I složka).

2.4.1.4 PSD regulátor

Klasický PID regulátor podle dříve uvedené rovnice pracuje ve spojitém čase. Realizace takového regulátoru proto není v číslicových systémech (jakým je například mikrokontrolér) možná. Když spojitý čas nahradíme konečnou časovou diferencí Δt, přejde rovnice PID regulátoru v rovnici PSD, tedy proporcionálně-sumačně-diferenčního regulátoru:

$$u(k) = u(0) + r_0 \left[e(k) + \frac{1}{T_I} \sum_{j=1}^{k} e(j)\Delta t + \frac{T_D}{\Delta t} (e(k) - e(k-1)) \right] \qquad (3)$$

Z této rovnice je mimo jiné patrné, že parametry sumační a diferenční části závisí i na periodě Δt. Při její změně je tedy potřeba příslušně upravit i koeficienty T_I a T_D [6].

2.4.1.5 Seřízení PID (PSD) regulátoru

K určení zesílení K_p a časových konstant K_i a K_d se využívá mnoho metod. Hodnoty lze vypočítat, k tomu je ale potřeba znát odezvu řízené veličiny na řídící veličinu. Ta je ale často obtížně určitelná, takže se využívá i jiných způsobů. Mezi ně patří metoda heuristická (poněkud méně vznešeně metoda pokusu a omylu), tedy odhadnutí hodnot a jejich postupná úprava, dokud se nedosáhne požadované odezvy regulátoru. Jako první se většinou nastavuje hodnota K_p, potom K_i a nakonec K_d. Lze se řídit následující orientační tabulkou vlivu změny jednotlivých složek na odezvu regulátoru.

Tabulka 1 - Vliv změny konstant na činnost PID regulátoru [7]

	Reakční čas	**Překmity**	**Doba ustálení**	**TRO**	**Stabilita**
Zvyšování K_p	Zkrácení	Nárůst	Drobný nárůst	Zmenšení	Zhoršení
Zvyšování K_i	Drobné zkrácení	Nárůst	Nárůst	Podstatné zmenšení	Zhoršení
Zvyšování K_d	Drobné zkrácení	Zmenšení	Zkrácení	Malý vliv	Zlepšení

Poněkud sofistikovanější je Ziegler-Nicholsova metoda, rovněž nazývaná metoda kritického zesílení. Spočívá ve zjištění zesílení K_u, při kterém dojde k nekonečnému kmitání řízené veličiny a periody T_u těchto kmitů. Z těchto údajů se následně vypočítají parametry K_p, K_i a K_d podle následující tabulky.

Tabulka 2 - Výpočet konstant regulátoru Z-N metodou [7]

	K_p	**K_i**	**K_d**
P regulátor	$K_u/2$	-	-
PI regulátor	K_u /2.2	$T_u/1.2$	-
PID regulátor	K_u /1.7	T_u /2	T_u /8

Metoda však není příliš přesná a nelze ji použít například u systémů, kde rozkmitání řízené veličiny nepřichází v úvahu ani jednorázově pro účely seřízení [7].

2.4.2 PWM regulace

Pulzně šířková modulace (anglicky pulse width modulation, zkr. PWM) je v technické praxi velmi často využívaný způsob převodu analogového (spojitého) signálu na digitální (diskrétní).

V případě DC motoru je úhlová rychlost rotoru přímo úměrná napětí na jeho svorkách, změnou tohoto napětí lze tedy řídit otáčky, viz následující vztah.

$$\omega = k_n \frac{V_m - R_m I_a}{\phi} \; [s^{-1}] \tag{3}$$

Kde k_n [1] je konstanta rychlosti, V_m $[V]$ je napětí na svorkách motoru, $R_m[\Omega]$ je elektrický odpor kotvy, $I_a[A]$ je proud protékající motorem a ϕ $[Wb]$ je magnetický tok v motoru [8]. Změnu napětí lze realizovat použitím odporového děliče napětí s variabilním odporem, např. potenciometrem. Toto řešení je však nevýhodné, protože omezuje proud procházející motorem, a tedy i jeho moment. Ztrátový výkon je navíc zmařen v rezistoru, tento způsob je tedy i značně nehospodárný. K řešení tohoto problému se využívá právě metody PWM. Na svorky motoru se místo konstantního napětí přivádí napětí, které je střídavě nulové a odpovídající napětí zdroje. Čas se rozdělí na periody dostatečně krátké délky. Zdroj se poté připojuje vždy na určitou dobu v rámci trvání každé jednotlivé periody. Poměr doby v rámci periody, kdy je zdroj připojen a odpojen pak odpovídá poměru výstupního efektivního napětí a napětí na zdroji. Pokud je například doba trvání jedné periody 10 ms a zdroj o jmenovitém napětí 12 V je připojen po dobu 5 ms (a ve zbytku trvání periody je vypnut), pak bude výstupní efektivní hodnota napětí 50 % napětí zdroje, tedy 6 V. Úvaha vychází ze známého vztahu pro výpočet efektivní hodnoty střídavého napětí.

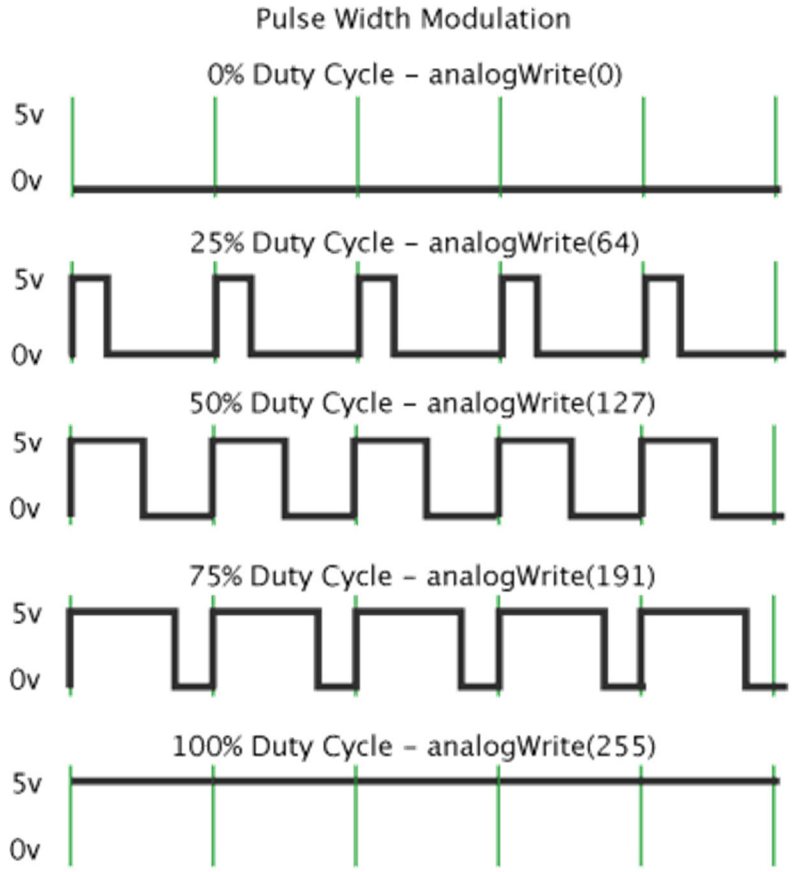
$$U_{ef} = \sqrt{\frac{1}{T}\int_0^T u^2(t)dt} \; [V] \tag{4}$$

Kde U_{ef} je efektivní hodnota napětí, T je doba trvání periody, u(t) je funkce popisující průběh napětí v závislosti na čase t. Za teoretického předpokladu použití bezztrátového spínacího prvku lze pro průběh napětí daného dvěma diskrétními hodnotami, z nichž je jedna nulová a druhá odpovídající napětí zdroje, vztah zjednodušit na:

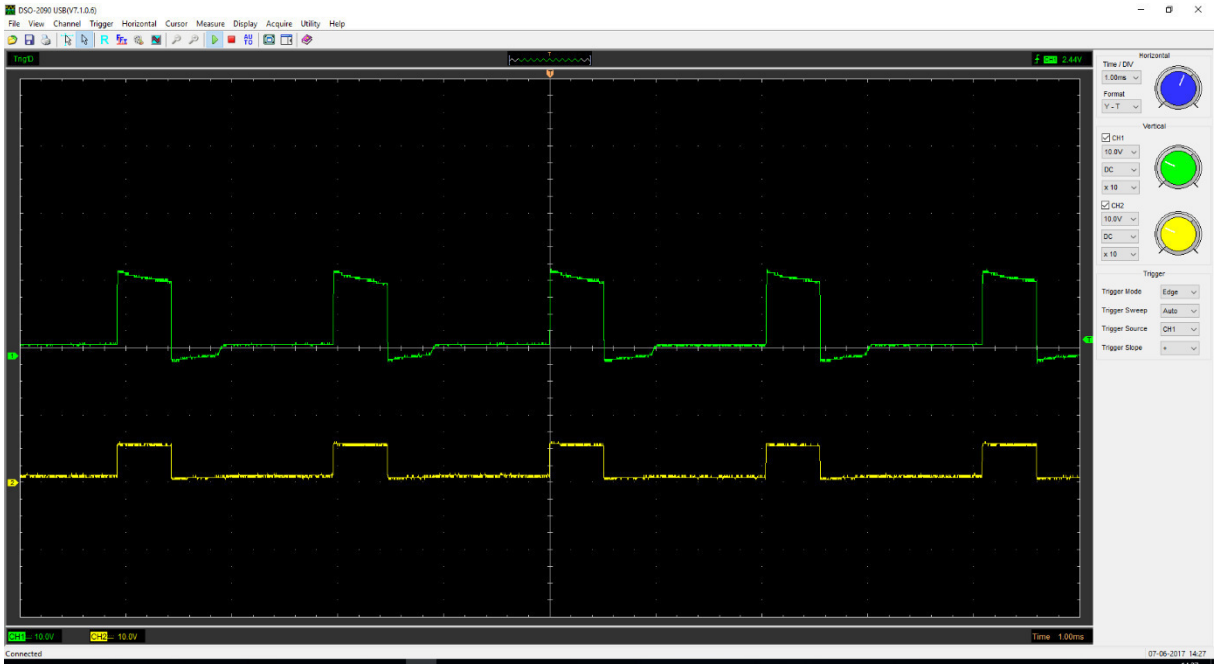
$$U_{ef} = U_M \frac{t_{ON}}{T} \; [V] \tag{5}$$

Kde U_{ef} je efektivní hodnota napětí, U_M je napětí zdroje, T je doba trvání periody a t_{ON} je doba, po kterou je zdroj připojen v rámci jedné periody. Zcela zřejmý je princip pulzně šířkové modulace z obrázku č.1. Řízení je teoreticky bezztrátové, prakticky ztráty nastávají právě

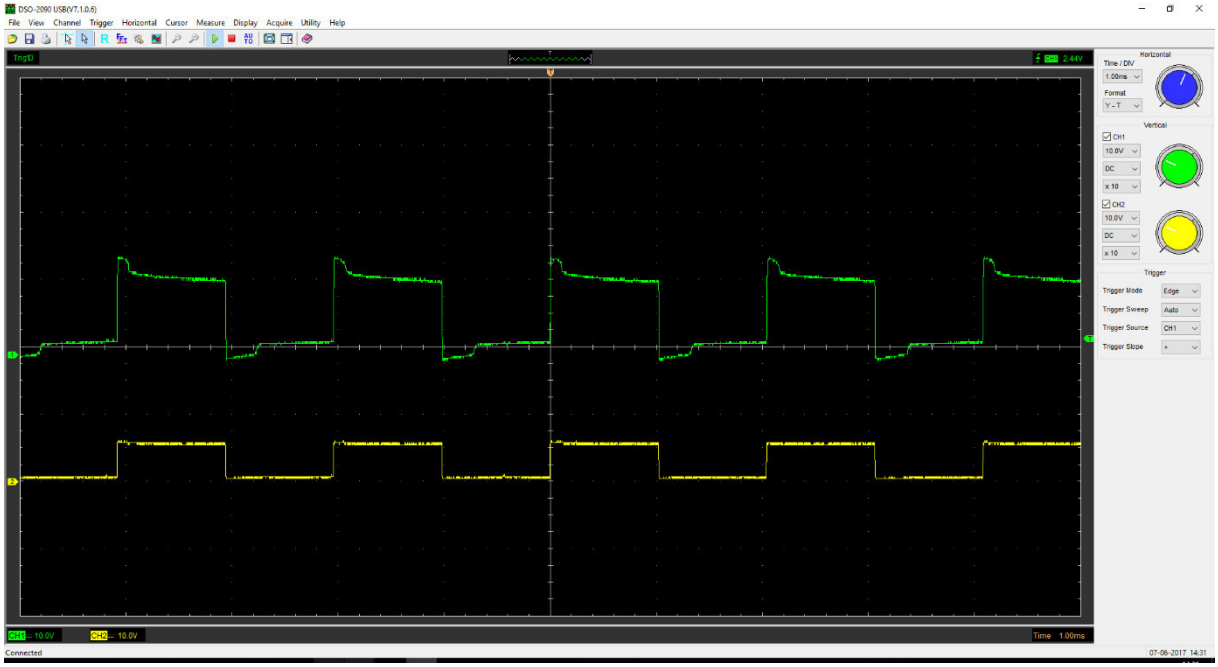
vlivem nenulového vnitřního odporu samotného spínacího členu. Délku jedné periody je nutno volit s ohledem na dynamiku napájeného zařízení. Pro moji aplikaci použitý stěračový DC motor má relativně velký moment setrvačnosti, a proto dostatečně rychlé střídání vypnutého a zapnutého stavu nezpůsobí kolísání otáček. Pro zajímavost jsem na osciloskopu odečetl průběhy napětí jak budícího PWM signálu, tak i výsledného napětí na svorkách motoru pro různé PWM hodnoty. Oscilogramy jsou na obrázcích 2 až 4. Je z nich dobře patrná odezva induktivní zátěže na skokové spínání a vypínání, které vede ke vzniku napěťových špiček, které jsou však vnitřní elektronikou motoru (cívkou, kondenzátorem a varistorem) částečně odfiltrovány. Na vývojové desce Arduino (viz dále), kterou při konstrukci servomotoru využívám, se PWM výstup realizuje funkcí analogWrite. Je k dispozici 256 úrovní výstupního efektivního napětí v rozsahu 0-5 V. Na obrázku 2 je výstup nastaven na úroveň 64 (odpovídá U_{ef} = 1,25 V), na obrázku 3 na úroveň 128 (odpovídá U_{ef} = 2,5 V) a na obrázku 4 na úroveň 254 (U_{ef} = 4,98 V). Žlutý je budící signál, tedy napětí mezi zemí a výstupem Arduina, zelené je napětí na svorkách motoru. 1 vertikální dílek představuje 10 V, 1 horizontální dílek představuje 1 ms.



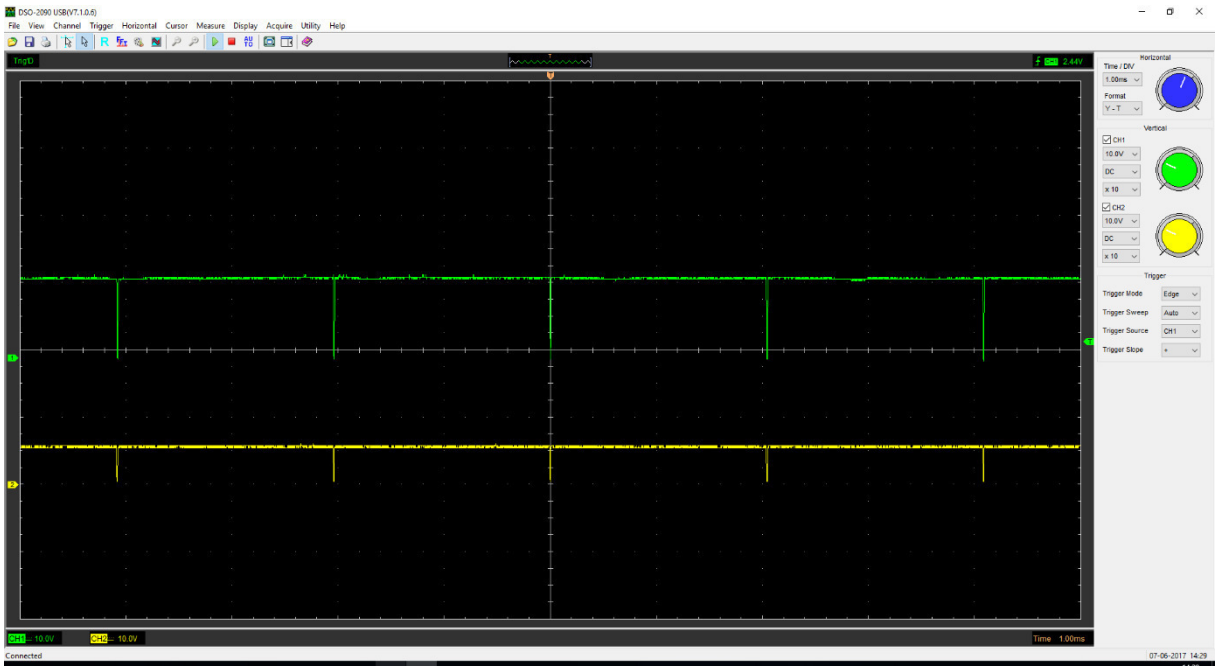
Obrázek 1) Pulzně šířková modulace [9]



Obrázek 2) PWM oscilogram, U_{ef} = 1,25 V



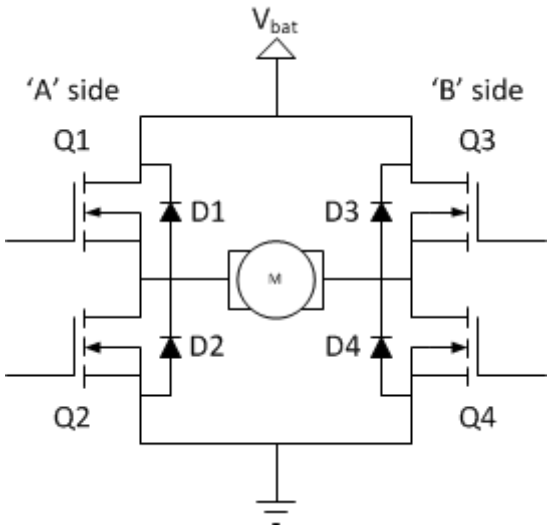
Obrázek 3) PWM oscilogram, U_{ef} = 2,5 V



Obrázek 4) PWM oscilogram, U_{ef} = 4,98 V

2.4.3 H-můstek

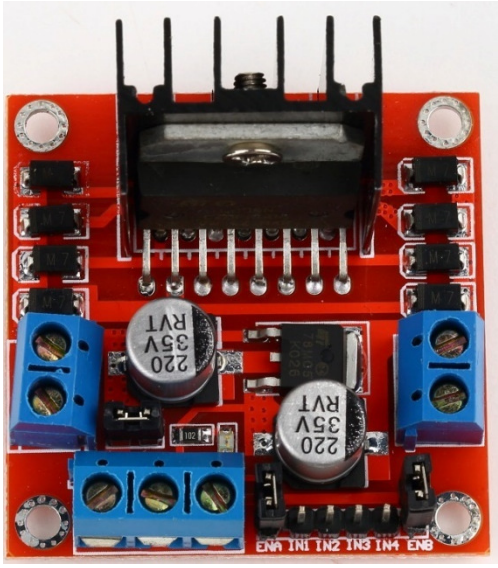
H-můstek je elektronický obvod umožňující připojení stejnosměrného napětí na svorky zátěže v obou možných směrech. V případě stejnosměrného elektrického motoru nám změna polarity umožní ovládat smysl otáčení hřídele. Elektrické schéma h-můstku je na obrázku číslo 5.



Obrázek 5) H-můstek [10]

Obvod sestává ze čtyř spínacích prvků, v případě obrázku č.2 unipolárních tranzistorů, čtyř diod, připojeného zdroje stejnosměrného napětí a připojené zátěže (motoru). Ve výchozím stavu, když není na gate žádného z tranzistorů přivedeno napětí motorem, neprochází proud a je v klidu. Pokud je přivedeno napětí na gate tranzistorů Q1 a Q4, motor se otáčí jedním směrem, pokud je napětí přivedeno na gate tranzistorů Q3 a Q4, motor se otáčí směrem druhým.

Pokud je napětí přivedeno na gate tranzistorů Q1 + Q3 nebo Q2+Q4, motorem neprochází proud. Pokud je napětí přivedeno na gate tranzistorů Q1 + Q2 nebo Q3 + Q4, dochází ke zkratu. Tyto kombinace jsou tedy nežádoucí. Diody mají význam pro ochranu tranzistorů, neboť při odpojení induktivní zátěže, jakou je právě například elektromotor, vznikne napěťová špička, která by tranzistory mohla prorazit. Proud vyvolaný těmito špičkami je přes diody zkratován a energie napěťové špičky se disipuje [10]. Na gate tranzistorů lze přivádět přímo PWM signál. H-můstky jsou k dispozici jako integrované obvody, které v sobě mají zpravidla více jednotlivých H-můstků. Tyto integrované obvody bývají uloženy v pouzdře s chladičem, neboť při spínání dochází vlivem nenulového odporu tranzistorů ke vzniku tepla. Rovněž jsou dostupné konstrukční moduly s H-můstkem (obr. 6). Tyto moduly mají připravené piny a svorkovnice pro pohodlné připojení do funkčních celků a dále například vlastní regulátor napětí.

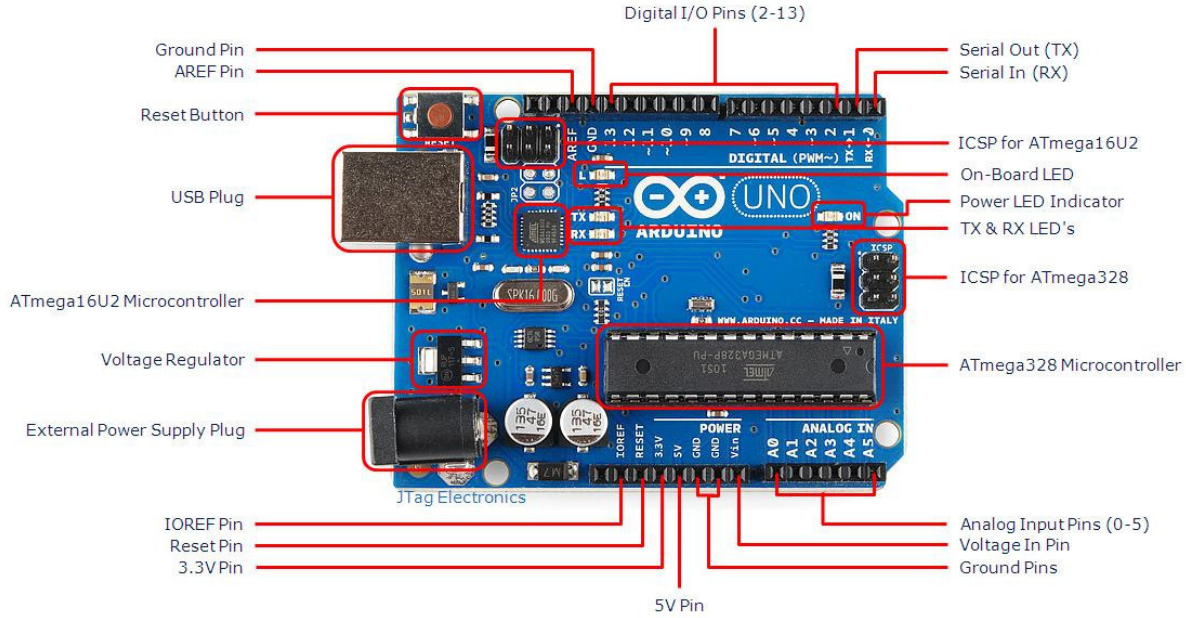


Obrázek 6) Modul s H-můstkem

3 MIKROKONTROLÉR ARDUINO UNO

Mikrokontrolér, který použiji při realizaci servopohonu, je open-source řešení Arduino Uno. Jedná se o vývojovou desku osazenou mikrokontrolérem Atmel ATmega328P, 16 MHz krystalem, regulátorem napětí a převodníkem úrovní USB na TTL (UART) pro snadné připojení k PC. Z mikrokontroléru je na desku vyvedeno 14 digitálních vstupně-výstupních pinů, z nichž 6 podporuje PWM (s rozlišením 10 bitů), a dále 6 vstupů vybavených 10 bitovými analogově-digitálními převodníky. Na desce dále najdeme rozhraní ICSP pro sériové programování vlastního mikrokontroléru, tlačítko RESET pro opětovné spuštění programu a

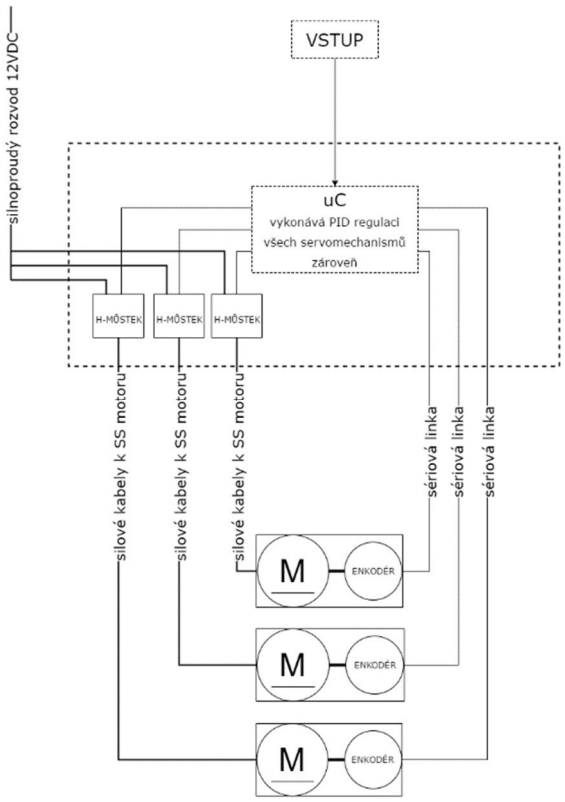
vestavěnou LED diodu. Napájení je možné buďto skrz USB (max. odběr 500 mA), anebo přes samostatný napájecí konektor. Regulátor napětí podporuje vstupní napětí 6–20 V (doporučeno 7-12 V) [11]. Deska pracuje s napětím logické jedničky 5 V. Vývojové desky Arduino se programují v jazyku Arduino programming language (programovací jazyk Arduino). Tento jazyk je založen na open-source frameworku Wiring, který je určen pro programování širšího spektra mikrokontrolérů. Je založen na programovacím jazyce C++, kterému se až na detaily podobá [12]. Kompilaci zdrojového kódu a jeho nahrání do desky Arduino obstarává software Arduino IDE. Kód lze psát přímo v tomto programu, postrádá však některé funkce zjednodušující orientaci v kódu, např. zobrazení čísel řádků, proto jsem při programování dával přednost Notepadu++. Užitečnou funkcí Arduino IDE je vestavěný sériový monitor pro komunikaci s deskou.



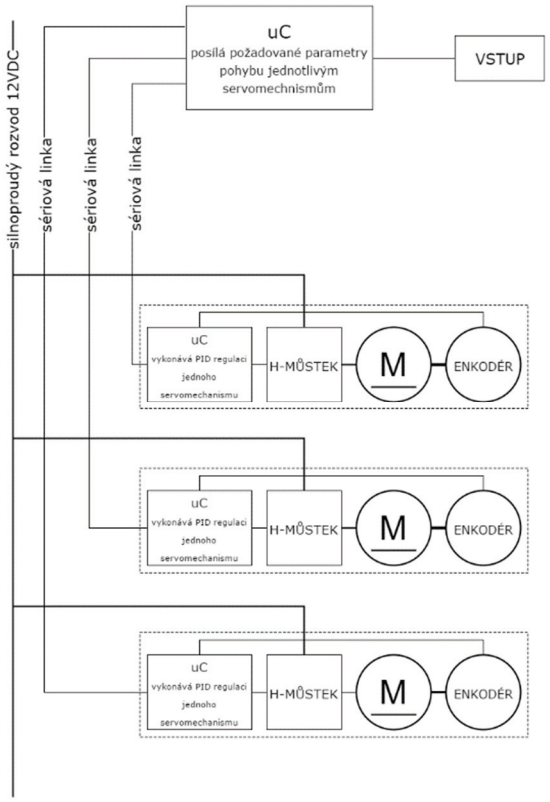
Obrázek 7) Mikrokontrolér Arduino UNO [13]

4 TOPOLOGIE SOUSTAVY SERVOMECHANISMŮ

Předpokládanou aplikací pro vyvíjený servopohon je humanoidní robot. Jako příklad takového robota v kapitole 5 uvádím stroj Honda Asimo s 57 stupni volnosti. Při návrhu servopohonu je potřeba vycházet z topologie této soustavy pohonů, způsobu jejich napojení jak na centrální řídící počítač, tak na zdroj napájení. Při návrhu samotné topologie vycházím z požadavků na odolnost proti rušení (DC motory a silnoproudá vedení jsou zdroji rušivého elektromagnetického pole) a zohledňuji také ekonomičnost možných variant.



Obrázek 8) Uspořádání s centrální regulací



Obrázek 9) Uspořádání s regulátory v kompaktním celku s motorem

Každá z uvedených struktur má své výhody i nevýhody. Výhodou centralizovaného uspořádání na obr. 8 je potřeba pouze jednoho řídícího počítače, což může být příznivé z ekonomického hlediska. Na druhou stranu sériové linky od snímačů polohy k centrálnímu počítači mohou být náchylné k rušení a spolehlivá zpětná vazba je pro kvalitní regulaci patrně nezbytná. Další nevýhodou je vyšší spotřeba poměrně drahých silových kabelů, neboť každý pohon musí mít zvláštní vedení. Naopak výhodou uspořádání servopohonu jako kompaktního celku (obr. 9) je snadná implementace do celé řady aplikací. Celek se připojí k silnoproudému rozvodu a po sériové lince se mu pouze posílají povely ve formě požadované polohy.

5 VÝBĚR SNÍMAČE POLOHY

Na začátku praktické části své bakalářské práce budu řešit výběr vhodného snímače polohy. Jelikož inkrementální senzor k absolutnímu určení polohy musí nejdřív vyhledat známou výchozí pozici, toto řešení se pro servopohon pro humanoidního robota nejeví jako ideální. Pro příklad si představme humanoidního robota Honda Asimo, který má 57 stupňů volnosti, tzn. 57 servomechanismů [14]. Pro správnou funkci by tedy všech těchto 57 servomechanismů muselo projet nejméně část své dráhy. Servopohon se stěračovým motorem samozřejmě není vhodným akčním členem pro realizaci každého z těchto stupňů volnosti, nešikovnost uvedeného řešení je ale i tak zjevná.

5.1 POROVNÁNÍ SNÍMAČŮ

Výběr snímače polohy se tedy omezuje na snímače absolutní. V následujících tabulkách je srovnání vybraných dostupných řešení.

Tabulka 3 – Porovnání diskrétních snímačů úhlové polohy

Druh snímače	Absolutní magnetický snímač úhlu	Absolutní optický snímač úhlu
Značka a typ	AVAGO 6010 [15]	AVAGO AS38-H39E-B12S [16]
Rozsah	neomezený	neomezený
Rozlišení	10 bit (0,35°)	23 bit v rozsahu jedné otáčky a 16 bit v rozsahu více otáček
Rozměry	Ø23x31,5 mm	Ø38x40mm
Napájecí napětí	5 VDC	5 VDC, energy harvesting
Životnost	Prakticky neomezená	Prakticky neomezená
Orientační cena za 1ks	608,- Kč (Farnell)	3007,- Kč (Mouser)

Tabulka 4 – Porovnání analogových odporových snímačů úhlové polohy

Druh snímače	Přesný potenciometr na bázi vodivého plastu	Přesný potenciometr s dutým hřídelem na bázi vodivého plastu
Značka a typ	CTS Series 284 [17]	StrainSense PGL60 [18]
Odpor	50 kOhm	10 kOhm
Elektrický rozsah	220°	320°
Linearita	+- 2 %	+- 1 %
Rozlišení	Prakticky omezené A/D převodníkem	Prakticky omezené A/D převodníkem
Jmenovitý výkon	0,25 W	0,5W
Životnost	2 miliony cyklů	10 milionů cyklů
Rozměry	Ø16x21,3 mm	Ø16x17,55 mm
Orientační cena za 1ks	312,- Kč (Mouser)	Nezjištěna

Na základě srovnání parametrů jsem se nakonec rozhodl pro absolutní magnetický snímač úhlu AVAGO AEAT-6010. Za hlavní výhody považuji neomezenou životnost, dobrou odolnost vůči vlivům prostředí, snadnou implementaci skrz sériovou linku a neomezený rozsah. Výstupem snímače je přímo binárně vyjádřený úhel natočení, není tedy potřeba A/D převodník. Cena je sice vyšší než v případě přesných potenciometrů, ale je vykoupena právě lepší životností, spolehlivostí a snadnější implementací. Snímač se umisťuje na válcový hřídel o průměru 6 mm, na servomechanismu bude umístěn z opačné strany ozubeného kola, než je výstupní hřídel. Toto řešení sice bude vyžadovat konstrukční úpravu stěračového motoru, ale i tak bude výrazně levnější než použití snímače polohy s dutým hřídelem. Snímač se připojuje pětipinovým konektorem, v dokumentaci není uveden přesný typ protikusu, pouze jeho okótovaný výkres. Protikus s přesně odpovídajícími rozměry (celková šířka a výška) je konektor Molex 15134-0500, který je k dispozici u dodavatele Farnell (stejně jako samotný enkodér). Dodává se i jako hotový celek konektoru s připojenými vodiči o délce 150 mm.

5.2 ZÁVĚR SROVNÁNÍ

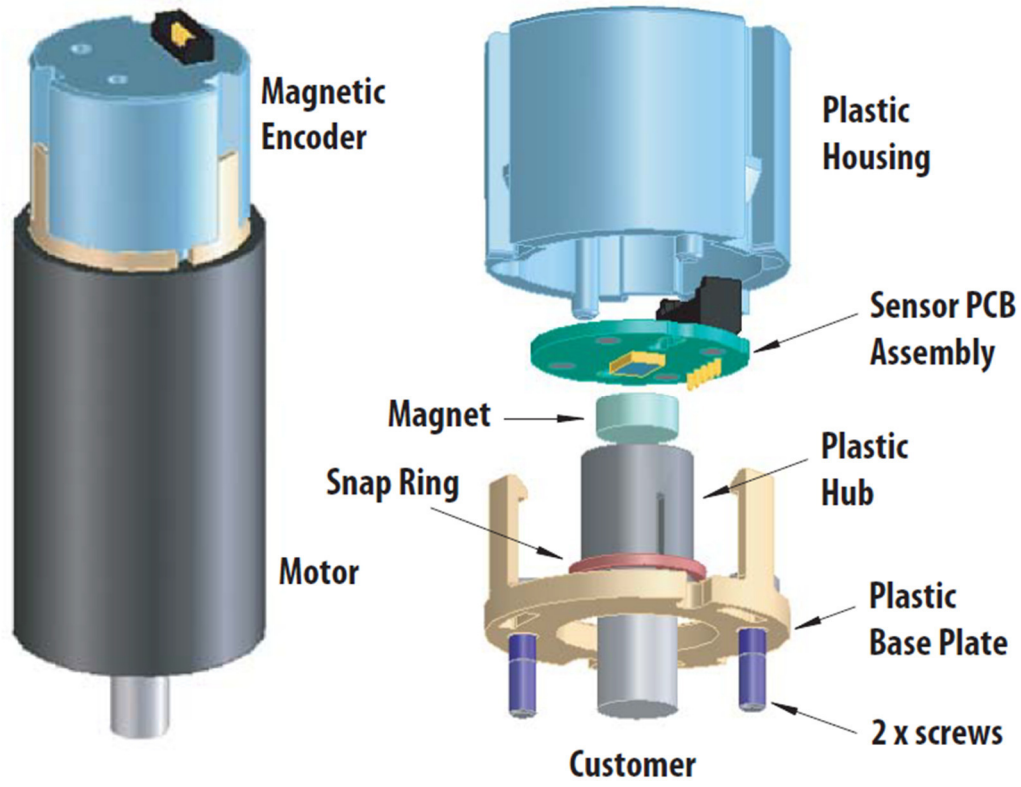
Tabulka 5 - Závěr srovnání enkodérů

	Přesný potenciometr na bázi vodivého plastu	**Přesný potenciometr s dutým hřídelem na bázi vodivého plastu**	**Absolutní magnetický snímač úhlu**	**Absolutní optický snímač úhlu**
Výhody	Nízká cena, snadná implementace, dostatečná přesnost, kompaktní rozměry	Možnost instalace na stranu výstupního hřídele (zmenšení celkových rozměrů), jednoduchá implementace, kompaktní rozměry	Vysoká přesnost, neomezený rozsah, prakticky neomezená životnost, digitální výstup	Velmi vysoká (pro tuto aplikaci až zbytečná) přesnost
Nevýhody	Instalace z druhé strany stěračového motoru zvyšuje rozměry celku, omezená (ale dostatečná) životnost, pouze analogový výstup, omezený rozsah	Vysoká cena, jen analogový výstup, omezený rozsah	Vyšší cena, citlivost na rušivé magnetické pole, větší rozměry	Vysoká cena, citlivost na čistotu prostředí, velké rozměry, omezený rozsah

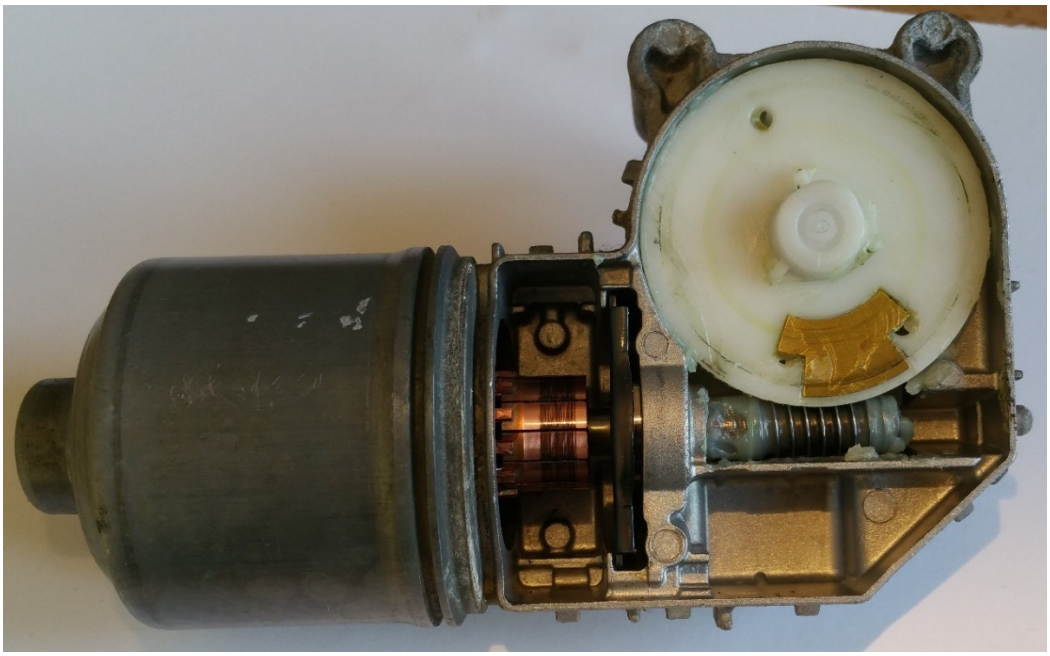
6 UPEVNĚNÍ SNÍMAČE POLOHY KE STĚRAČOVÉMU MOTORU

Výrobce enkodéru poskytuje návod na správné připevnění snímače k hřídeli, jehož natočení se měří [19]. Způsob připevnění k běžnému motoru je vidět na obrázku č.10. Výrobce počítá s existencí volného konce hřídele o průměru 6 mm, na který se nasazuje plastové pouzdro

opatřené magnetem, jehož magnetické pole je snímáno Hallovými sondami v enkodéru. Takovým hřídelem však stěračový motor nedisponuje. Výstupní hřídel použít nelze, neboť pak by bylo servo nepoužitelné. Rozhodl jsem se tedy pro instalaci hřídele na opačnou stranu šnekového kola (viz obrázek č.11).



Obrázek 10) Schéma připevnění enkodéru k motoru [15]

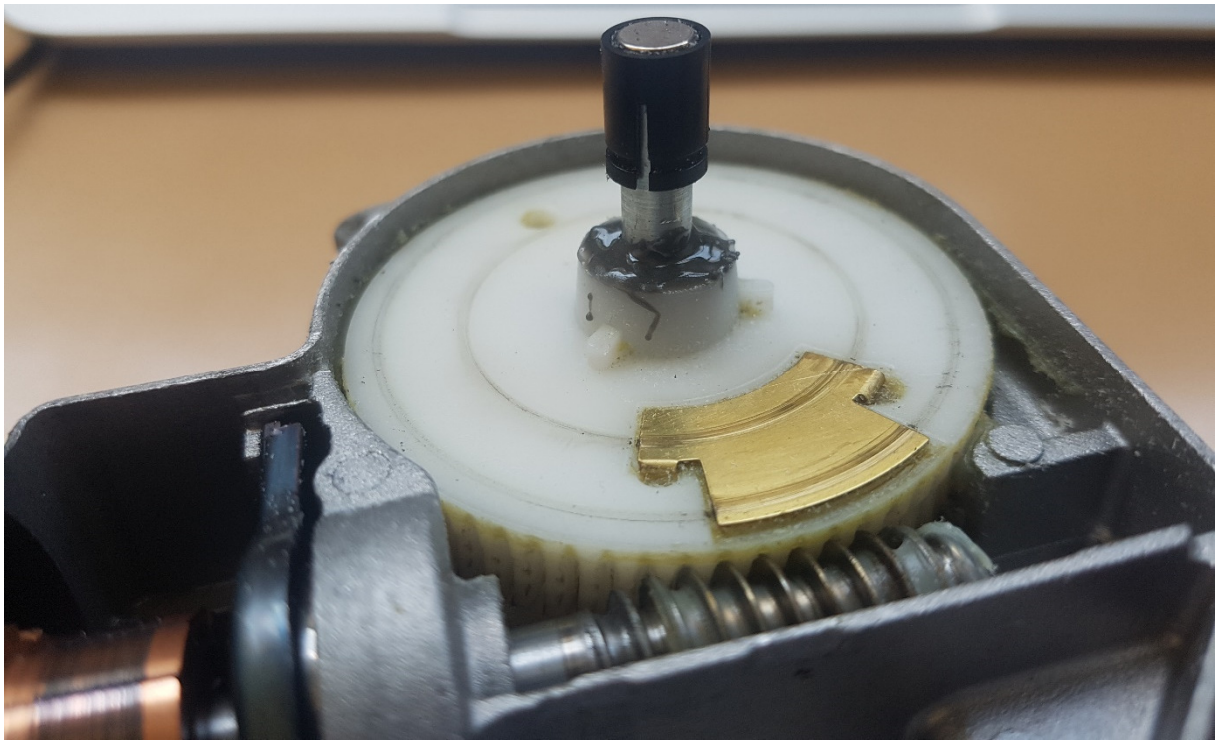


Obrázek 11) Otevřený stěračový motor

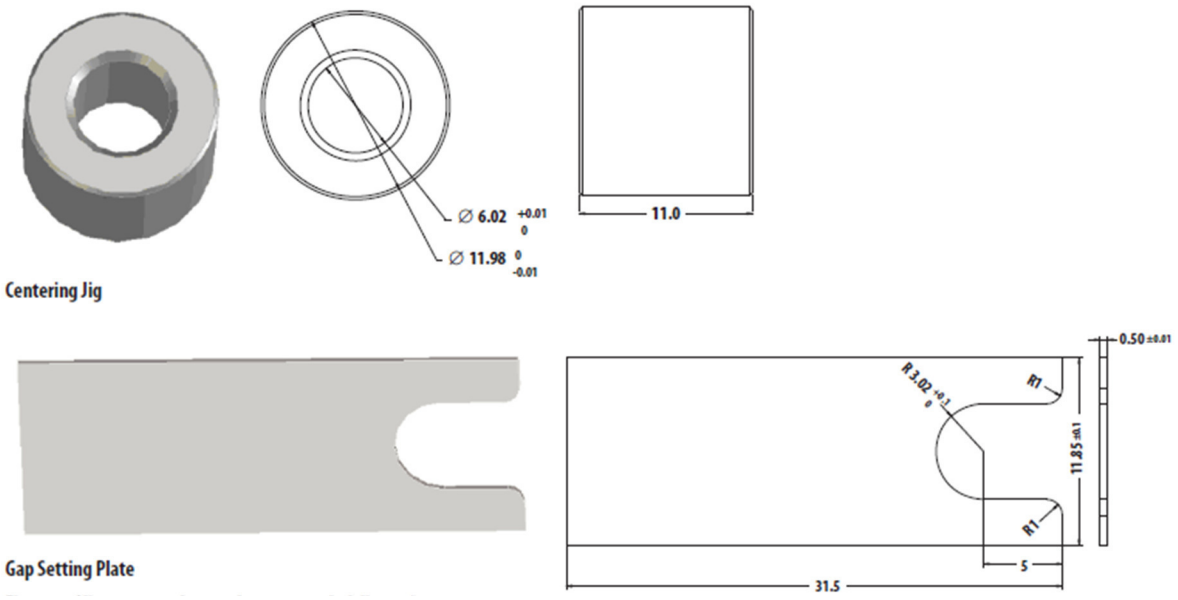
Do plastového pouzdra jsem nechal vyvrtat otvor (výkres viz příloha I), kterým nový hřídel vystoupí z pouzdra motoru na povrch. Využil jsem malé válcové díry ve šnekovém kole o průměru i hloubce 2 mm, do které se hřídel zasadí. Předpokládám, že otvor má nějaký technologický význam při výrobě kola. Díra na první pohled není dokonale vystředěná, k žádné lepší variantě uchycení hřídele jsem však nedospěl.

Výstupek, ve kterém je díra umístěna, má tvar komolého kužele a k přesnému zjištění jeho rozměrů by bylo nutné využít nějaké sofistikované měřící zařízení. Pokud by se rozměry podařilo zjistit, bylo by možné například přesným 3D tiskem vyrobit hřídel s koncem tvarově přizpůsobeným k dosednutí na tento výstupek.

Nakonec jsem se tedy rozhodl pro hřídel ocelový, který jsem si podle vlastního návrhu (příloha II) nechal vysoustružit v dílně. Hřídel jsem zasadil do zmíněné díry ve šnek. kole a zafixoval lepidlem. Šnekové kolo je plastové, nejspíše z obtížně lepitelného polypropylenu. Lepené části jsem pečlivě odmastil acetonem, neboť šnekový převod je mazán vazelínou, která je pod krytem stěračového motoru skutečně všudypřítomná. Nejprve jsem se pokusil hřídel přilepit běžným kyanoakrylátovým lepidlem, spoj však nevydržel dlouhou dobu a při testování servopohonu selhal. Úspěchu jsem dosáhl s dvousložkovým epoxidovým lepidlem, povrch jsem navíc před aplikací krátce ožehnul plamenem, neboť je to pro úspěšné lepení tohoto typu plastu nutné [20]. Výsledek, tedy hřídel přilepený ke šnekovému kolu, je vidět na obrázku č. 12.



Obrázek 12) Hřídel s pouzdrem s magnetem na šnekovém kole



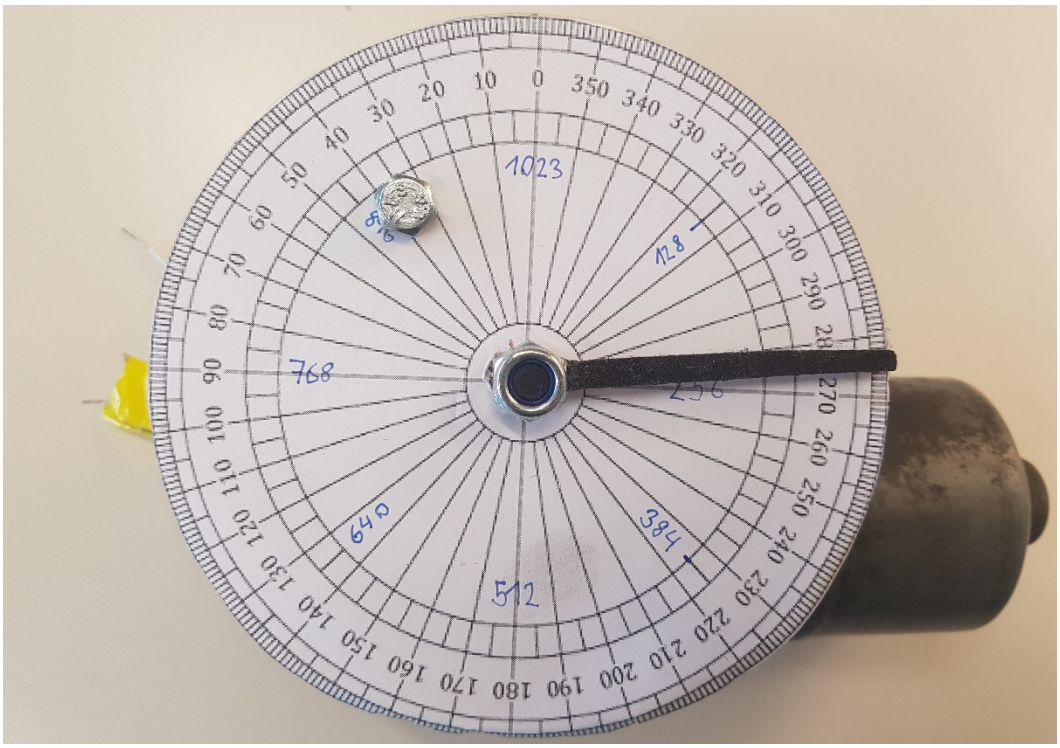
Obrázek 13) Zarovnávací přípravky [15]

Zásadním problémem je správné vycentrování hřídele s magnetem vůči pouzdru snímače, respektive vůči základně (na obr. č.10 označena jako „Plastic base plate“), do které se pouzdro zaklapne. Výrobce enkodéru dodává pro tento účel dvoudílný zarovnávací set (viz obrázek č.13), tento se mi však nepodařilo nikde sehnat. Vzhledem k tomu, že jsou k dispozici výrobní výkresy obou částí této sady, vytvořil jsem si jejich 3D modely a nechal je zhotovit na 3D tiskárně. Tisk se příliš nezdařil, odchylky od požadovaných rozměrů byly relativně velké, povrchy byly velmi hrubé a nerovnoměrné. Přesto byl vystřeďovací přípravek (na obr. 13

Obrázek 14) Detail připevnění enkodéru k motoru

označen jako „Centering Jig“) užitečný a s jeho pomocí jsem dosáhl poměrně uspokojivého zarovnání. Přípravek pro vymezení vzdálenosti od snímačů (na obr. 13 označen jako „Gap Setting Plate“) jsem nakonec nevyužil, tloušťka byla kolísavá a pouzdro s magnetem šlo po hřídeli axiálně posouvat jen velmi těžko. Pouzdro snímače jsem v poloze vymezené vystřeďovacím přípravkem zafixoval pomocí tavné pistole, jak je to vidět na obr. 14. Výrobce sice dodává přímo se snímačem i dva šrouby určené k montáži, pro moji aplikaci jsou však příliš dlouhé a výroba by se zbytečně zkomplikovala vyřezáváním závitů do tenkostěnného krytu motoru.

Použitý magnetický enkodér je citlivý zejména na házení hřídele (se zvětšujícím se házením klesá linearita výstupu na vstupu). I když je házení viditelné i pouhým okem, enkodér funguje přijatelně spolehlivě. Soudě podle improvizovaného úhloměru, který jsem na výstupní hřídel připevnil (obr. 15), je opakovatelnost překvapivě dobrá. Kvalita uložení hřídele je však pro přesnost servomechanismu v porovnání s rozlišením enkodéru v mém případě zcela rozhodujícím faktorem.



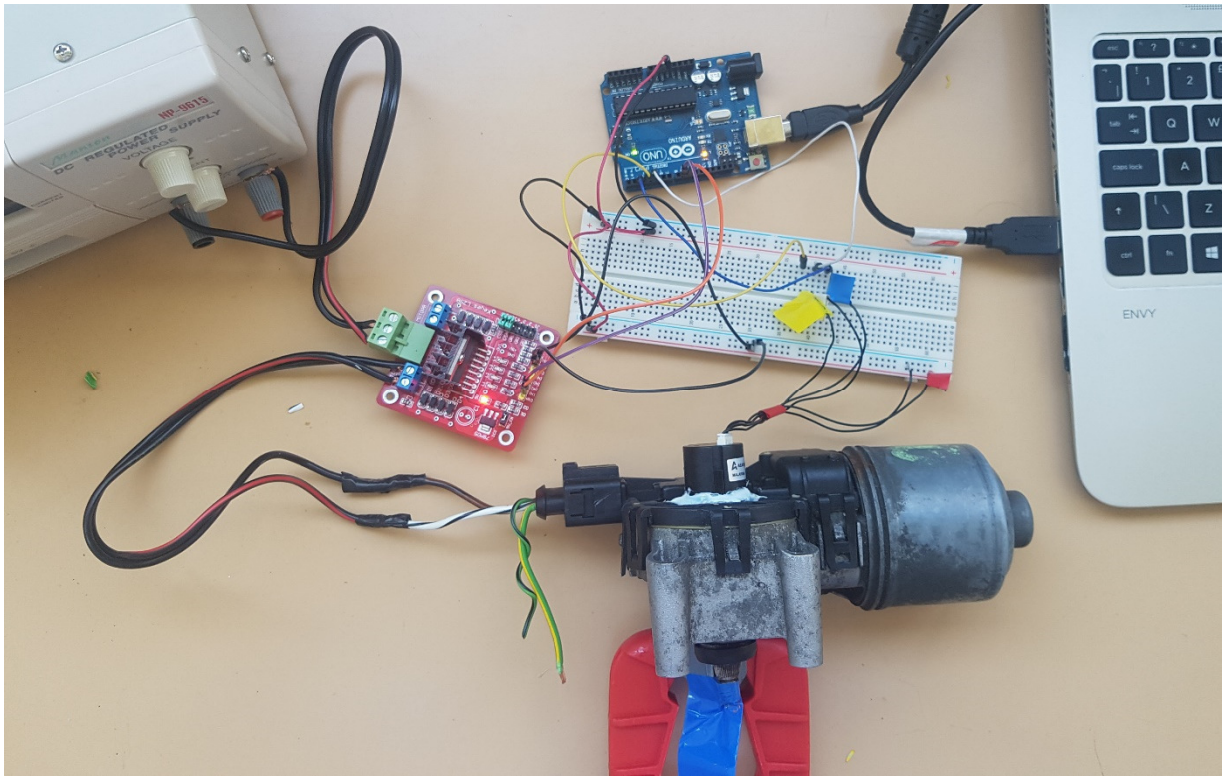
Obrázek 15) Úhloměr s ručičkou pro znázornění polohy hřídele

7 ZAPOJENÍ ŘÍDÍCÍ ELEKTRONIKY

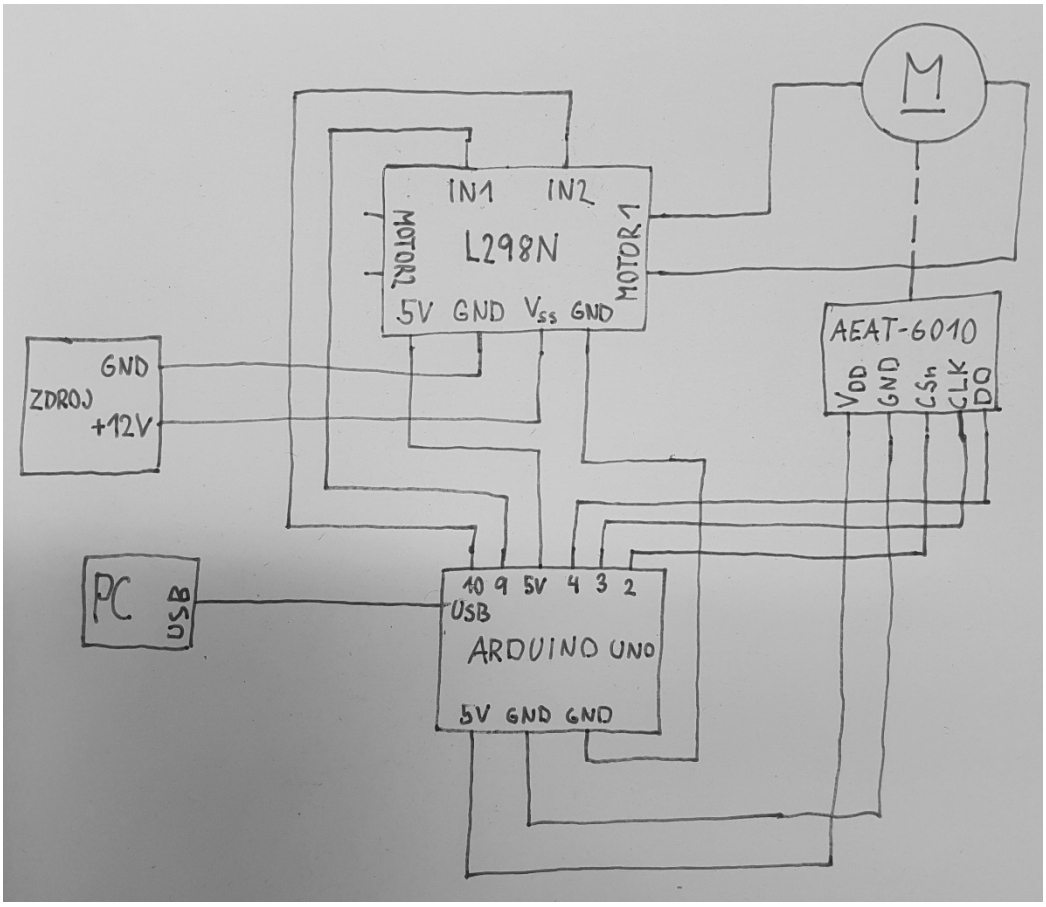
Můj servopohon se skládá z následujících části:

1. Stěračový motor s připevněným enkodérem
2. Modul s H-můstkem L298N
3. Deska Arduino UNO
4. Laboratorní zdroj stejnosměrného napětí
5. Osobní počítač se SW pro komunikaci po sériové lince

Pro zjednodušení zapojení jsem rovněž používal nepájivé kontaktní pole. Slaboproudé elektrické vedení je realizováno vodiči s kolíky, které přímo pasují do nepájivého pole i na piny Arduina a modulu s H-můstkem. Silnoproudé vedení je realizováno silnějšími lanky. Využil jsem původní konektor od stěračového motoru, pouze jsem ho zbroušením upravil, aby šel snáze sundávat (původně není k častému sundávání určen, takže je velmi pevný). Všechny součásti pohonu jsou vidět na obr. 16, blokové schéma zapojení je na obr. 17.



Obrázek 16) Sestavený servopohon



Obrázek 17) Blokové schéma servomotoru

8 OVLÁDACÍ PROGRAM

Ovládací program servopohonu se stěračovým motorem je spuštěn na desce Arduino UNO a má tyto funkce:

1. Zajišťuje komunikaci s řídícím počítačem, v mém případě PC. Toto zahrnuje načítání žádaných hodnot polohy od uživatele a odesílání aktuálních údajů o žádané poloze, PWM výstupu a skutečné poloze.
2. Zajišťuje komunikaci se snímačem polohy – enkodérem, tedy z něj načítá aktuální údaj o poloze.
3. Realizuje samotný PID (PSD) algoritmus na základě parametrů K_p, K_i a K_d, žádané hodnoty (Setpoint) načtené od uživatele a skutečné polohy načtené z enkodéru. Výstupem je akční zásah v rozsahu -255 až 255, kde znaménko určuje směr otáčení.
4. Upravený (viz dále) výstup regulátoru převádí na PWM signál, který je přiveden na vstup H-můstku odpovídající požadovanému směru otáčení.

Nyní bude následovat vysvětlení všech částí zdrojového kódu programu. Regulační PID algoritmus přechází vzhledem k realizaci na digitálním zařízení v PSD, vzhledem k názvu knihovny a zvyklostem se o něm však budu zmiňovat stále jako o PID.

8.1 DEKLARACE PROMĚNNÝCH, FUNKCE SETUP

```
01 #include <PID_v1.h>
02
03 const int CS=2;
04 const int CLK=3;
05 const int DI=4;
06 const int CW=9;
07 const int CCW=10;
08 double Setpoint, Input, Output, mapout, thres;
09 double Kp=1.55, Ki=0, Kd=0, thres=10 ;
10
11 PID myPID(&Input, &Output, &Setpoint, Kp, Ki, Kd, DIRECT);
12
13 void setup() {
14  Serial.begin(115200);
15  pinMode(CS, OUTPUT);
16  pinMode(CLK, OUTPUT);
17  pinMode(DI, INPUT);
18  pinMode(CW, OUTPUT);
19  pinMode(CCW, OUTPUT);
20
21  digitalWrite(CS, HIGH);
22  digitalWrite(CLK, HIGH);
23  myPID.SetOutputLimits(-255,255);
24  myPID.SetMode(AUTOMATIC);
25  Setpoint=200;
26
27
29 }
```

Zdrojový kód začíná připojením knihovny, která realizuje PID regulaci. Následuje deklarace proměnných. Prvních pět proměnných odkazuje na vstupní a výstupní piny pro příjem údaje o poloze z enkodéru a ovládání H-můstku. Pro přehlednost jejich význam uvádím v následující tabulce.

Tabulka 6 – Vstupně-výstupní piny

CS	CLK	DI	CW	CCW
Připojeno na chip select pin enkodéru	Připojeno na clock pin enkodéru	Data input, pin je připojen na data output pin enkodéru	Připojeno k vstupu IN1 H-můstku, realizuje otáčení po směru hod. ruč.	Připojeno k vstupu IN2 H-můstku, realizuje otáčení proti směru hod. ruč.

Další proměnné (Input, Output, Setpoint) jsou zavedeny pro účely knihovny PID regulátoru. Input je aktuální stav řízené proměnné, Output je regulační proměnná a Setpoint je požadovaná hodnota řízené proměnné. Proměnné mapout a thres jsou zavedeny za účelem přepočtu výstupní proměnné regulátoru na konečnou PWM hodnotu.

Na dalším řádku se definují konstanty regulátoru, tedy proporcionální Kp, integrační (sumační) Kp a derivační (diferenční) Kd. Poté je na řádku 11 inicializován PID algoritmus. Konkrétně jsem využil oficiální PID knihovnu Arduino PID Library v nejnovější verzi 1.0.1. V této verzi se knihovna dle autora robustností blíží průmyslově používaným regulátorům. V argumentu funkce PID se nastavuje název (zde „myPID“), pojmenování potřebných proměnných a „smysl“ regulace (DIRECT, nebo REVERSE). První možnost znamená, že kladná regulační odchylka vyvolává kladný akční zásah, druhá znamená opak.

Ve funkci setup se inicializuje sériová komunikace s přenosovou rychlostí 115200 bps. Toto je potřeba pro sériovou komunikaci s PC. Dále je definováno, které piny jsou vstupní a které výstupní. Nastavením logické jedničky na pinech CS a CLK enkodéru dojde k jeho deaktivaci, čímž je připraven pro další použití. Na řádku 23 je nastaven rozsah řídící veličiny regulátoru. Použil jsem 0–255 v obou směrech, což odpovídá rozsahu 8bitového PWM, které Arduino podporuje. Později je však výstup ještě upraven. Dále je nastaven výchozí Setpoint.

8.2 KOMUNIKACE SE SNÍMAČEM POLOHY

```
30 double readSensor(){
31  unsigned int dataOut = 0;
32 
33  digitalWrite(CS, LOW);
34  delayMicroseconds(1);
35 
36  for(int x=0; x<10; x++){
37      digitalWrite(CLK, LOW);
38      delayMicroseconds(1);
39      digitalWrite(CLK, HIGH);
40      delayMicroseconds(1);
41      dataOut = (dataOut << 1) | digitalRead(DI);
42  }
43  digitalWrite(CS, HIGH);
44  return (double)dataOut;
45 
46 }
```

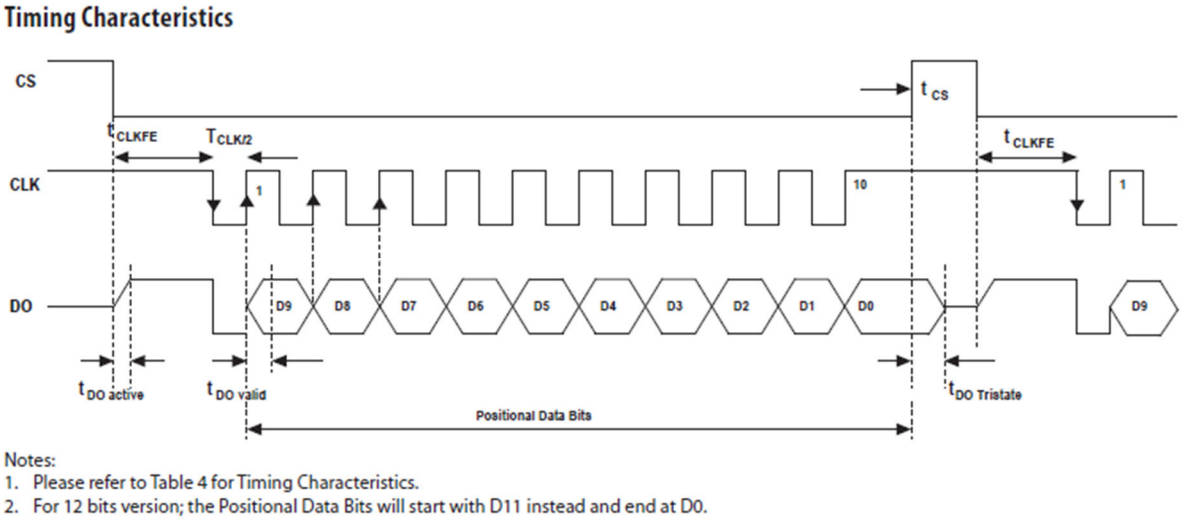
Řádkem 30 začíná definice funkce readSensor, která slouží ke čtení hodnot polohy z magnetického absolutního enkodéru. Komunikace probíhá přes SSI, tedy sériové synchronní rozhraní. Průběh komunikace se vyčte z časovací charakteristiky, která je součástí datasheetu enkodéru. Komunikace se skládá z následujících po sobě jdoucích částí:

1. Pro účely této funkce se deklaruje proměnná dataOut a přidělí se jí výchozí, nulová hodnota.
2. Na pinu CS enkodéru se nastaví logická nula, čímž se enkodér aktivuje. Následuje prodleva t_CLKFE. (řádky 33,34)
3. Následně se otevírá for cyklus, který probíhá od 0 do 9. Každý jednotlivý cyklus načte jeden bit z celkových deseti bitů, které tvoří jeden údaj o poloze.
    a. Na pin CLK enkodéru se přivede logická nula po dobu T_CLK/2 (řádky 37, 38)
    b. Na pin CLK enkodéru se přivede logická jednička opět po dobu T_CLK/2 (řádky 39, 40)

c. Do proměnné dataOut se načte hodnota, která vznikne provedením operátoru AND nad předchozí hodnotou proměnné posunutou o 1 bit doleva a aktuální logické hodnoty na pinu DI. (řádek 41)

4. Cyklus se desetkrát opakuje, poté se na pin CS opět přivede logická jednička, čímž se enkodér deaktivuje až do opětovného zavolání funkce. (řádek 43)
5. Funkce readSensor vrátí hodnotu dataOut (řádek 44).

Délka prodlev se vyčte z datasheetu enkodéru.



Obrázek 18) Časovací charakteristika enkodéru [15]

8.3 NAČÍTÁNÍ HODNOT SETPOINT ZE SÉRIOVÉ LINKY

```
48 void loop() {
49
50
51  while(Serial.available() > 0) {
52      int SerInp = Serial.parseInt();
53      if(Serial.read() == '\n') {
54          Setpoint = constrain(SerInp,128,840);
55      }
56  }
```

Na řádku 48 začíná hlavní smyčka programu. První se odehrává načtení požadované hodnoty natočení hřídele (setpoint) ze sériové linky. Na řádku 51 začíná while cyklus, který v případě, že je na sériové lince k dispozici hodnota tuto hodnotu načte do proměnné SerInp (řádek 52). Ve chvíli, kdy je po sériové lince přijat znak „newline“, což je typicky po stisknutí klávesy enter po zadání požadované hodnoty do sériového monitoru, se hodnota SerInp nastaví jako nový Setpoint, se kterým dále pracuje PID knihovna (řádek 54).

8.4 VÝPOČET VÝSTUPU A JEHO ZPRACOVÁNÍ

Můj servomechanismus nemá implementován výpočet a zvolení nejkratší úhlové vzdálenosti z aktuální do žádané polohy při průchodu nulou. Proto je funkcí constrain omezen pohyb hřídele pouze od polohy 128 do polohy 840. V případě překmitnutí za nulu (v obou směrech) by totiž došlo k další plné otáčce hřídele, čímž by došlo k nekontrolovanému roztočení hřídele. Možnostmi implementace „průchodu nulou“ se budu zabývat později.

```
57  Input = readSensor();
58  myPID.Compute();
59  mapout = map(abs(Output),0,255,50,255);
60
61  if(Output>thres) {
62      analogWrite(CW,mapout);
63      digitalWrite(CCW,LOW);
64  }
65  else if(Output<(-thres)){
66      analogWrite(CCW,mapout);
67      digitalWrite(CW,LOW);
68  }
69  else{
70      digitalWrite(CW,LOW);
71      digitalWrite(CCW,LOW);
72  }
```

Na řádku 57 se zavolá funkce readSensor, kterou jsem popsal výše a vrací aktuální polohu hřídele. Tato poloha se pro účely PID knihovny uloží do proměnné Input. Následně se zavolá knihovna PID, která spočítá aktuální hodnotu řídící proměnné. Vzhledem k tomu, že minimální napětí potřebné k otáčení nezatíženého motoru je cca 2,4 V, což odpovídá PWM hodnotě 50, výstup PID regulátoru se převádí pomocí funkce map na rozsah odpovídající napětím 2,4 – 12 V (proměnná mapout tedy nabývá hodnot 50-255) (řádek 59).

Další část programu vyhodnocuje, zdali je výstup kladný nebo záporný. Podle toho se následně určuje, na který výstupní pin se přivede PWM signál. O to se stará funkce analogWrite, která jako argumenty přijímá hodnotu od 0 do 255 a příslušný výstupní pin. Výstupní signál se navíc na H-můstek přivádí jen tehdy, pokud je větší než nastavená hodnota proměnné thres. Důvodem je kolísání údaje o poloze z enkodéru, i když je hřídel v klidu. Důležité je vždy nevyužívaný výstupní pin uzemnit pomocí funkce digitalWrite. Pokud se to neudělá, tak po první změně směru otáčení se začne přivádět PWM signál na oba dva vstupy H-můstku zároveň, což způsobí nefunkčnost servomechanismu.

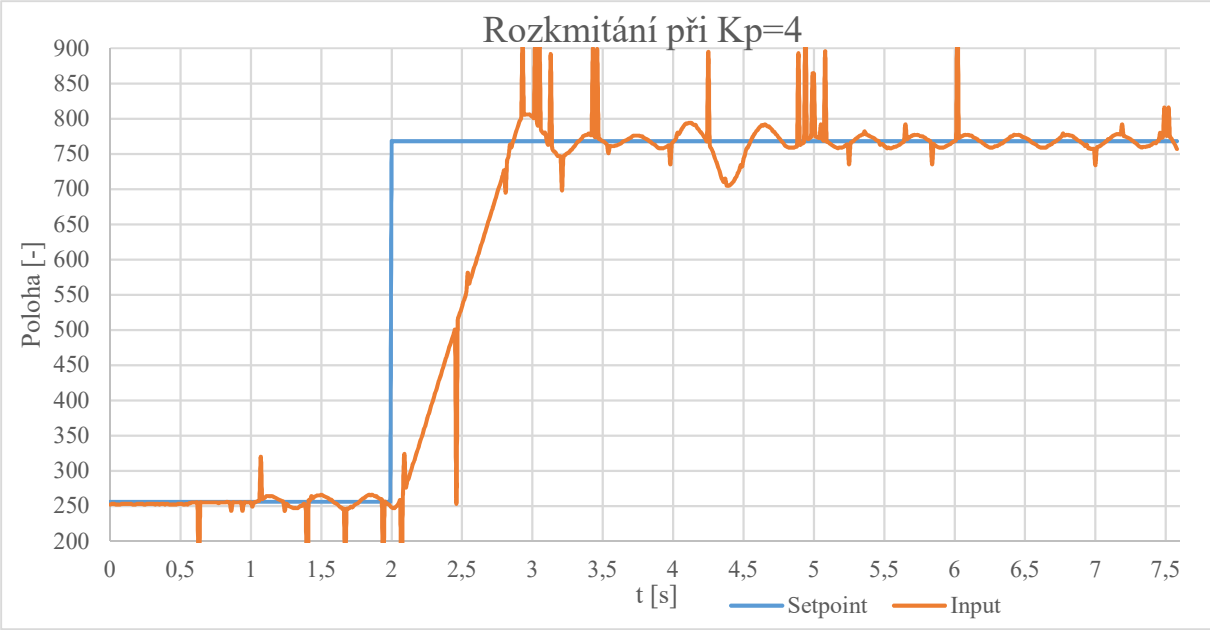
8.5 TISK PRACOVNÍCH PROMĚNNÝCH DO SÉRIOVÉ LINKY.

```
74   Serial.print(Setpoint);
75   Serial.print("\t");
76   Serial.print(Output);
77   Serial.print("\t");
78   Serial.println(Input);
79
80   delay(1);
81 }
```

Poslední část programu „vytiskne“ aktuální hodnoty proměnných Setpoint, Output a Input do sériové linky. Toto umožňuje například jejich vykreslení do grafu. Hlavní smyčka končí prodlevou o délce 1 ms, která chod programu stabilizuje (řádek 80). Poté se celý cyklus opakuje.

9 SEŘÍZENÍ REGULÁTORU

Při seřizování regulátoru jsem vycházel z Ziegler-Nicholsovy metody, kterou jsem popsal v teoretické části práce. Sledoval jsem průběh polohy při změně setpointu z hodnoty 256 na hodnotu 768. Nejprve jsem tedy postupně zvyšoval zesílení K_p, dokud se servomotor nerozkmital ustálenými kmity (obr. 19). Tento stav nastal při $K_p = 4 = K_u$. Průběh odezvy servomotoru jsem si v MS Excel vykreslil do grafu a odečetl jsem periodu kmitů $T_u = 0{,}4$ s. Z těchto dvou údajů jsem si podle vztahů uvedených v tab. 2 vypočítal hodnoty jednotlivých parametrů (tab. 7). Ve všech průbězích je patrný výrazný šum, jehož příčinu se mi již nepodařilo zjistit a odstranit. Hlavní podezření padá na vodiče vedoucí od enkodéru do Arduina, které se mohly porušit, na vině může však být i nepřesnost vzniklá při opětovném lepení hřídele ke šnekovému kolu poté, co nevydržel první, kyanoakrylátový spoj.

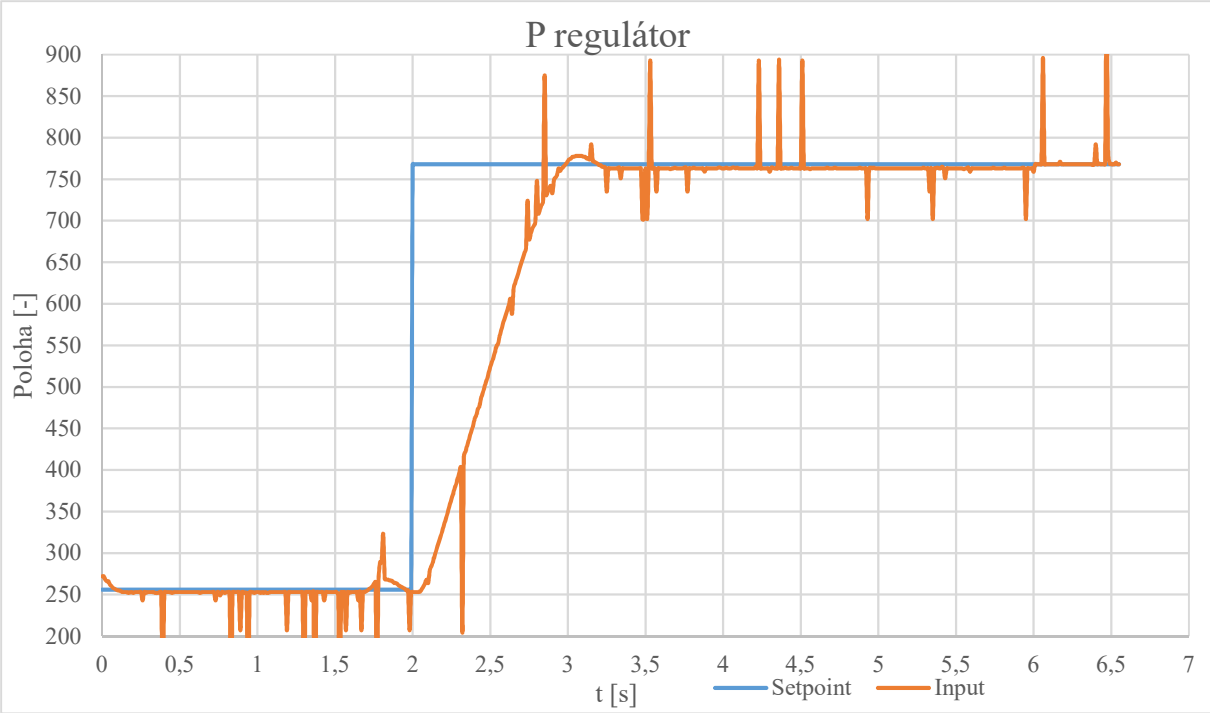


Obrázek 19) Rozkmitání regulátoru

Tabulka 7 – Vypočítané hodnoty parametrů regulátoru.

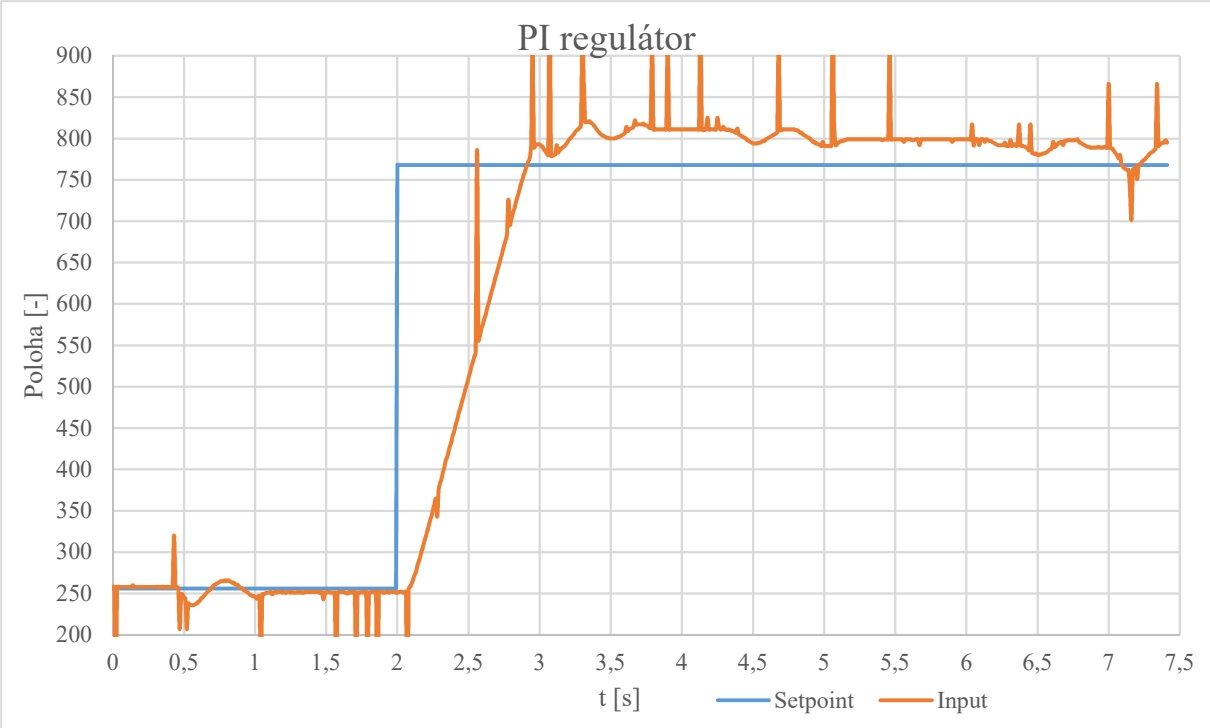
	K_p	K_i	K_d
P regulátor	2	-	-
PI regulátor	1,8	0,33	-
PID regulátor	2,4	0,2	0,05

Na následujících grafech je odezva na změnu Setpointu z hodnoty 256 na hodnotu 768 při nastavení parametrů dle tab. 7.



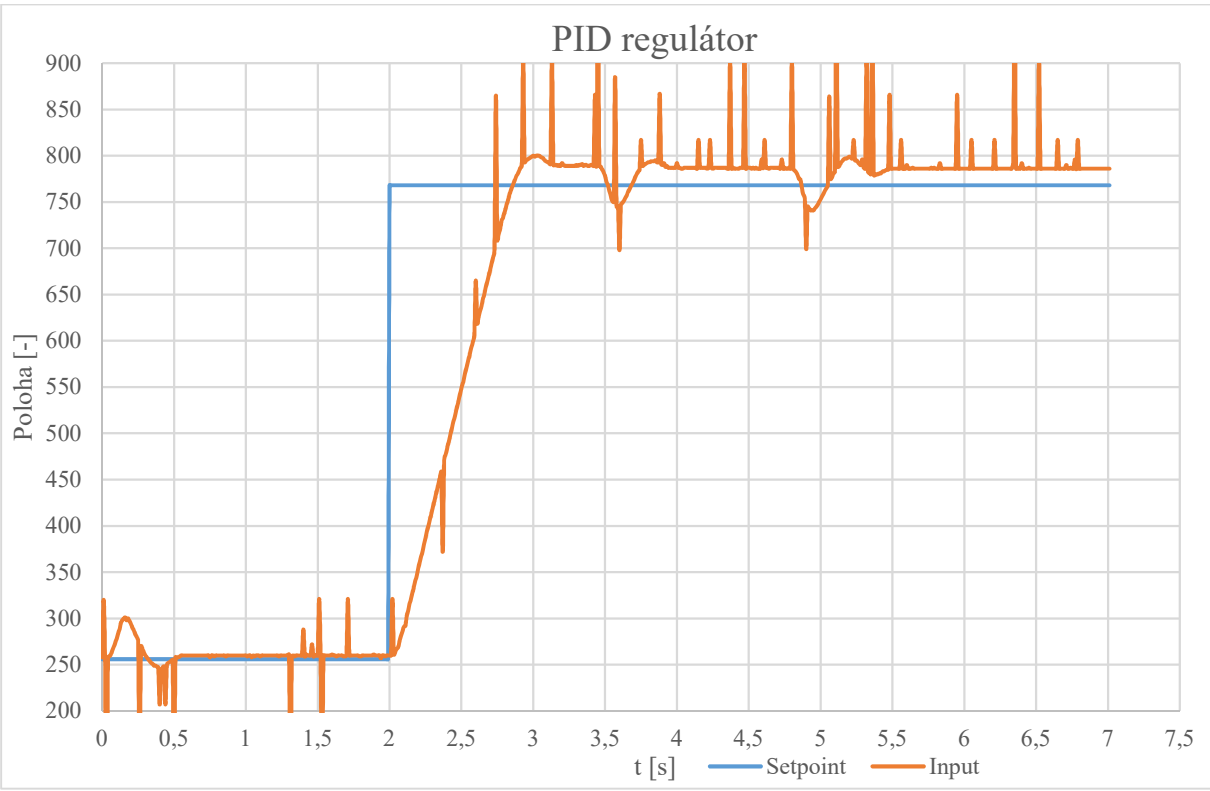
Obrázek 20) P-regulátor

P – regulátor s parametrem Kp = 2 reaguje na skokovou změnu žádané hodnoty velmi příznivě (obr. 20). Z grafu je patrné drobné překmitnutí, pak se průběh ale ustálí na hodnotě cca 765, u P – regulátoru nevyhnutelná trvalá regulační odchylka je tedy 3. Při přepočítání na úhlové stupně se jedná o cca 1°. Takovou odchylku považuji vzhledem ke zhoršené přesnosti enkodéru (vlivem nedokonale zarovnaného hřídele) za zcela přijatelnou. Pokud bychom servopohon využívali například pro realizaci předloktí humanoidního robota o délce 0,3 m, pak by odchylka konce předloktí od požadované polohy vlivem TRO dosáhla cca. 5 mm.



Obrázek 21) PI regulátor

I když by zvýšení konstanty Ki, a tedy příspěvku integrační části regulátoru mělo přispět k redukci TRO, v tomto případě se stal pravý opak (obr. 21). Dávám to za vinu výraznému šumu. Zvýšení parametru Ki tedy odezvu regulátoru jednoznačně zhoršilo.

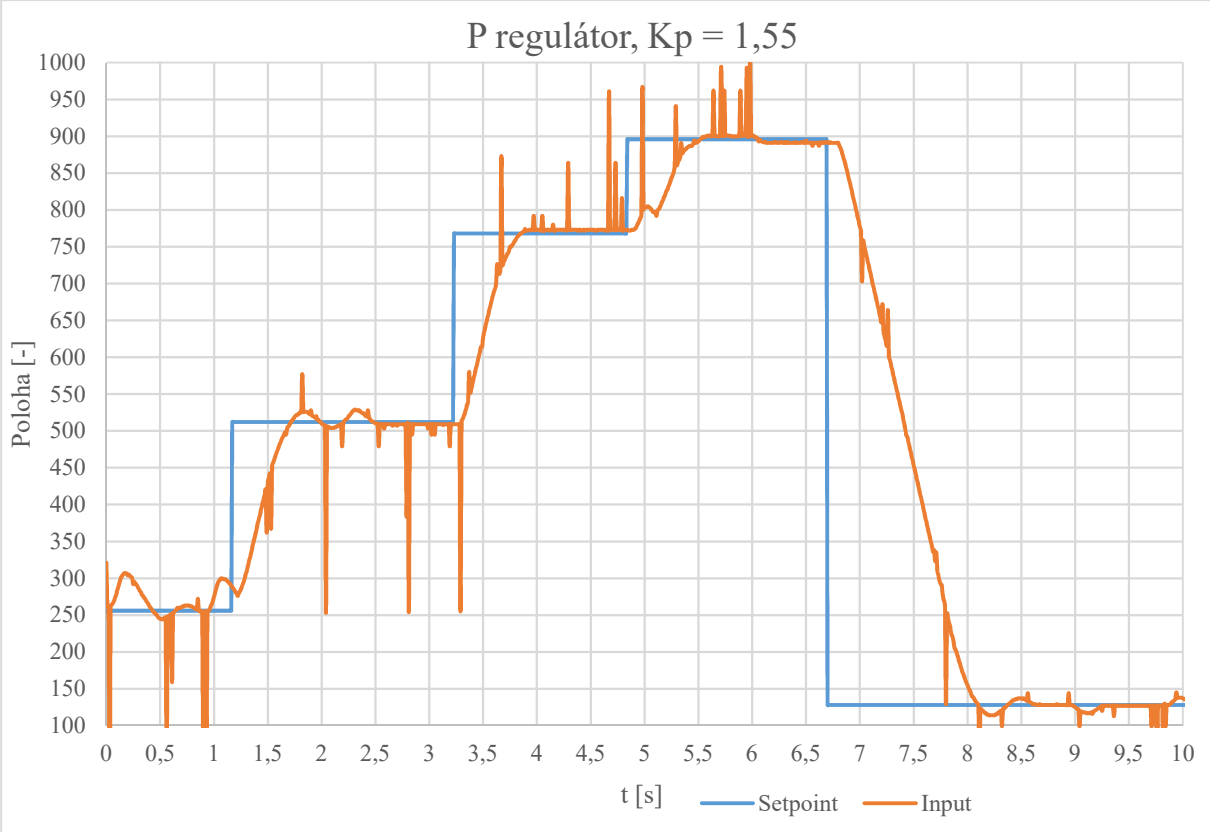


Obrázek 22) PID regulátor

Na obr. 22 je průběh při aktivaci všech tří složek PID regulátoru. Stále se dosahuje trvalé regulační odchylky kolem 25, tedy asi 9°. Šum při snímání řízené veličiny je jedním z případů, kdy nelze D složku použít, neboť tato má tendenci vliv šumu zesilovat. Přidáním D složky tedy ke zlepšení odezvy regulátoru nedochází.

9.1 VÝSLEDEK SEŘÍZENÍ

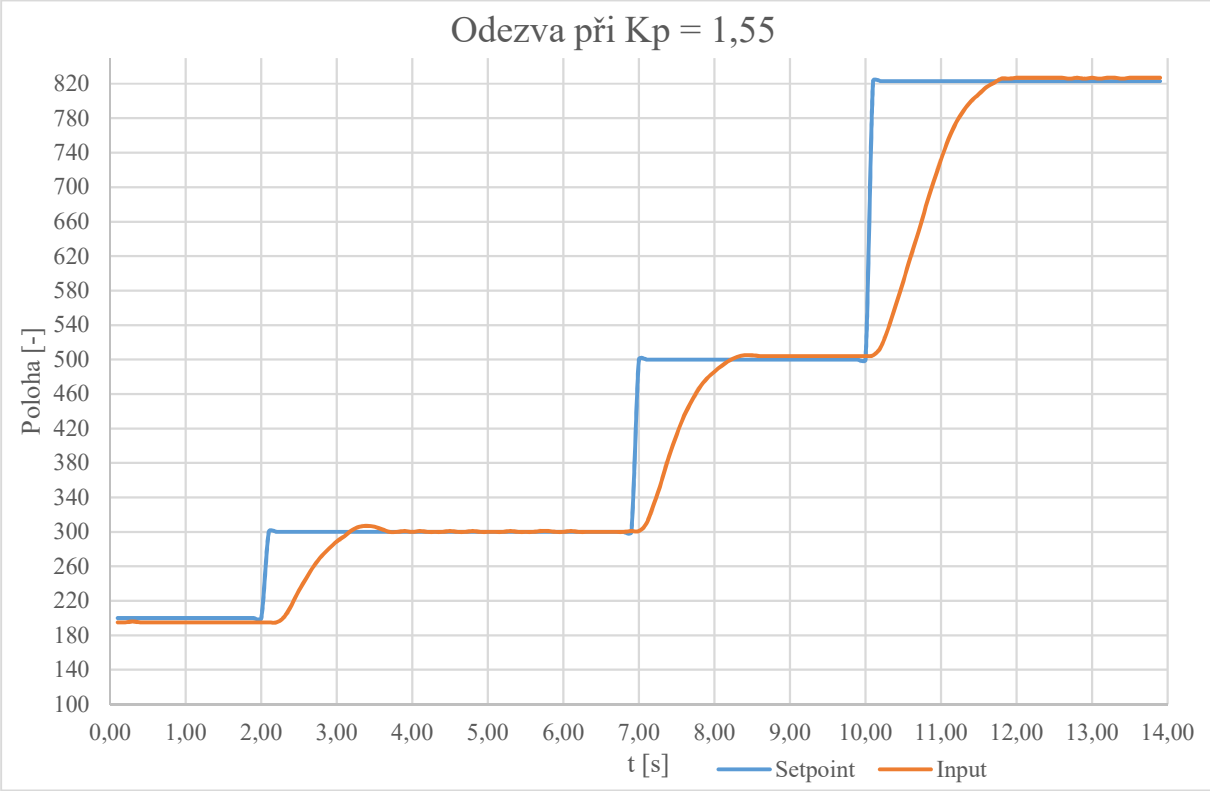
Nakonec jsem se rozhodl použít pouze P regulátor, neboť i při dalším zkoumání nad rámec výše uvedených průběhů jsem dospěl k závěru, že I a D složky nemají pro moji aplikaci význam. Metodou pokusu a omylu jsem dospěl k ideální hodnotě zesílení Kp = 1,55. P regulátor sice pracuje s určitou TRO, ta byla ale většinou do 1°, tedy pro celou řadu aplikací zanedbatelná. Odezva na skokovou změnu setpointu z hodnoty 256 postupně na hodnoty 512, 768, 896 a 128 je na obrázku č. 23.



Obrázek 23) P regulátor

Výrazný šum se bohužel objevil až poté, co jsem dokončil ovládací program. Už před jeho dokončením jsem si ale odečetl průběh odezvy P-regulátoru při parametru Kp = 1,55 (obr. 21). Při odečítání tohoto průběhu se ovládací program lišil ve způsobu přepočtu PWM výstupů nižších než 50. Nebyla použita funkce map, hodnoty nacházející se mezi hodnotami thres a 50

se skokově měnily právě na hodnotu 50. Dále byla na konci hlavní smyčky programu nastavena mnohem vyšší hodnota delay – 100 ms (ověřil jsem, že nižší hodnota delay příčinou šumu není). V následujícím grafu je znázorněna odezva na změnu setpointu z hodnoty 200 postupně na hodnoty 300, 500 a 800.



Obrázek 24) P regulátor bez šumu

10 PRŮCHOD NULOVOU POLOHOU

Původně jsem zamýšlel následující část zdrojového kódu jako součást řídícího programu, později jsem si však uvědomil, že by to vyžadovalo zásahy do zdrojového kódu samotné PID knihovny, neboť se mění způsob výpočtu regulační odchylky.

Vymýšlením způsobu, jakým pro libovolnou kombinaci poloh žádané a skutečné hodnoty (proměnných Setpoint a Input) určit nejkratší vzdálenost s přihlédnutím k možnosti průchodu hodnotou 0 = 1023 a jí odpovídající směr otáčení jsem strávil poměrně hodně času. Nakonec jsem skončil u definování 4 variant kombinací hodnot Setpoint a Input na základě jejich příslušnosti do intervalů 0 až 511 a 512 až 1023 a přiřazení vhodného vztahu pro výpočet vzdálenosti oběma směry každé z těchto kombinací. Požadovaný směr otáčení se poté určuje porovnáním obou možností, vybírá se pochopitelně kratší z nich.

```
void setup(){
02 }
03
04 void loop() {
05  boolean A;
06  boolean B;
07  boolean C;
08  int D;
09  double error;
10  boolean DIR;
11
12  if(Input>0&&Input<=512) {
13      A=1;
14  }
15  else {
16      A=0;
17  }
18
19  if(Setpoint>512&&Setpoint<=1023) {
20      B=1;
21  }
22  else {
23      B=0;
24  }
25
26  if(Input>Setpoint) {
27      C=1;
28  }
29  else {
30      C=0;
31  }
32
33  D=Input-Setpoint;
```

V hlavní smyčce programu se definují logické proměnné A, B, C a DIR a celočíselné proměnné D a error. Hodnoty proměnných A resp. B vyjadřují, zdali se proměnná Input, resp. Setpoint nachází v intervalu od 0 do 512 či v intervalu od 512 do 1023. Proměnná C vyjadřuje, která

poloha má větší hodnotu. Proměnná DIR určuje výstupní směr otáčení, hodnota 1 odpovídá otáčení po směru hodinových ručiček, hodnota 0 otáčení proti směru hod. ruč. Proměnná D se rovná rozdílu proměnných Input a Setpoint. Proměnná error odpovídá skutečné regulační odchylce.

```
35  if((A==0&&B==0&&C==0)||(A==1&&B==1&&C==0)) {
36      error = -D;
37      DIR = 1;
38  }
39  else if ((A==0&&B==0&&C==1)||(A==1&&B==1&&C==1)) {
40      error = D;
41      DIR = 0;
42  }
43  else if (A==0&&B==1) {
44      if((1023-D)>D) {
45          error = D;
46          DIR = 0;
47
48      }
49      else {
50          error = 1023-D;
51          DIR = 1;
52
53      }
54  }
55  else if (A==1&&B==0) {
56      if((1023+D)>-D) {
57          error = -D;
58          DIR = 1;
59
60      }
61      else {
62          error = 1023+D;
63          DIR = 0;
64
65      }
```

Část programu na řádcích 35 až 41 ošetřuje případ, kdy jsou obě proměnné Setpoint i Input v jednom intervalu (0 až 512 nebo 512 až 1023). V takovém případě je výpočet jednoduchý a určení odchylky i směru spočívá vychází ze znaménka proměnné D.

Část programu na řádcích 43 až 65 pak ošetřuje případy, kdy se každá proměnná nachází v jiném intervalu. Tyto možnosti jsou dvě a pro každou se vzdálenost po a proti směru hodinových ručiček počítá jinak. Program opět vybere nejkratší možnou vzdálenost a k ní příslušící směr otáčení.

11 ZÁVĚR

Cílem práce bylo navrhnout servopohon se stěračovým motorem, sestavit ho za pomoci desky Arduino a ověřit jeho funkci. Úkol jsem splnil, při práci jsem však narazil na několik problémů, které by se při případné implementaci tohoto typu servopohonu například do humanoidního robota musely řešit.

Je třeba si uvědomit, že stěračový motor je zařízení vyvinuté tak, aby co nejefektivněji při zachování nízkých výrobních nákladů plnilo jeden účel – pohon automobilových stěračů. Konkrétně se toto projevilo při montáži externího hřídele pro využití enkodérem. Šnekové kolo, vyrobené z obtížně lepitelného plastu pravděpodobně technologií vstřikování, neumožňuje zcela uspokojivě vyřešit připevnění hřídele – tak aby byla dodržena přesnost tolik potřebná pro přesnou funkci enkodéru. V práci jsem navrhnul způsob, jak problém uchycení za cenu větší složitosti řešit. Další možný způsob by mohl spočívat v obstarání detailní výrobní dokumentace k motoru, která však není vzhledem k utajení snadno dostupná.

Poněkud menší problém představuje zarovnání enkodéru vůči hřídeli s magnetem. K tomuto účelu výrobcem navržené zarovnávací přípravky jsem si musel nechat vyrobit, neboť nejsou dostupné. Díly byly vyrobeny metodou 3D tisku, přesnost však nebyla příliš dobrá. Pro účel ověření schůdnosti tohoto řešení servopohonu je dosažená přesnost zarovnání (a tím ovlivněná přesnost enkodéru) dostatečná, kdyby se však měla realizovat výroba více těchto servopohonů, jistě by se vyplatilo nechat si tyto tvarově poměrně jednoduché součástky vyrobit s dodržením předepsaných tolerancí.

Velmi přínosné pro mě bylo programování obslužného programu, neboť jsem před realizací této práce neměl s programováním skoro žádné zkušenosti. Mnou napsaný program sice jistě není nijak přelomový, avšak vcelku dobře plní svoji funkci. Obzvlášť zajímavé bylo vytváření části programu, která komunikuje s enkodérem. Rozhodl jsem se pro využití již napsané a ověřené oficiální PID knihovny pro Arduino. Knihovna je velmi spolehlivá a její využití je intuitivní.

Po dokončení ovládacího programu jsem se pokusil nalézt optimální hodnoty parametrů regulátoru. Bohužel se na výstupu z enkodéru objevil šum, který ladění značně komplikoval. Na důvod jeho vzniku jsem nepřišel. Průběhy odezvy jsem si však zaznamenával i v průběhu vytváření programu, kdy ještě šum přítomný nebyl. Součástí práce je tedy i průběh při naladění, se kterým jsem byl nejvíce spokojen, tedy P regulátor s parametrem Kp = 1,55 odečtený ještě

před vznikem šumu. Program se však poněkud lišil a průběh s finálním programem, ale bez šumu by pravděpodobně rovněž vypadal odlišně.

Pokud by to bylo pro danou aplikaci potřeba, knihovnu by bylo možné upravit s pomocí mnou navrženého algoritmu pro určení nejkratší úhlové vzdálenosti mezi dvěma polohami hřídele. Toto by řešilo dva problémy. Jeden z nich nastává, když servopohon např. z pozice 1000 do polohy 100 „cestuje“ proti směru hodinových ručiček, místo aby využil mnohem kratší cesty právě přes nulu. Druhý problém s prvním úzce souvisí a nastává při překmitnutí při najíždění do polohy blízké nule přes nulu a následném nekontrolovaném roztočení serva. Původně jsem chtěl kód, který tyto problémy řeší, implementovat do řídícího programu. Uvědomil jsem si však, že by to vyžadovalo změny v samotné PID knihovně, neboť kód mění způsob výpočtu regulační odchylky.

Na úplný konec se zamyslím nad samotným smyslem předělání stěračového motoru na servopohon. Na trhu je k dispozici nepřeberné množství komerčních servopohonů rozličných parametrů, které však většinou sdílejí jednu vlastnost a tou je vysoká cena. Naopak ceny použitých, ale plně funkčních stěračových motorů se v bazarech pohybují v řádu stokorun. Souhrnná cena enkodéru, H-můstku a desky Arduino je kolem tisíce korun. Za cenu pod dva tisíce korun lze tedy získat kompaktní servopohon s vysokým točivým momentem, vestavěnou samosvornou šnekovou převodovkou a při použití řídícího programu, který podporuje „průchod nulou“ i s neomezeným rozsahem. Tento servopohon pochopitelně není určen pro průmyslové aplikace, ale může přijít vhod domácím kutilům a při realizaci např. robotických projektů s omezeným rozpočtem, jakým je třeba stavba humanoidního robota na ústavu přístrojové a řídící techniky ČVUT. Pro tyto účely je přestavba stěračového motoru na servopohon dobrou možností jeho opětovného využití.

BIBLIOGRAFIE

[1] YOUNKIN, George. *Industrial servo control systems: fundamentals and applications*. 2nd ed., rev. and expanded. New York: Marcel Dekker, 2003. ISBN 08-247-0836-9.

[2] ŠŤASTNÝ, Jiří a Branko REMEK. *Autoelektrika a autoelektronika*. Vyd. 6. Praha: T. Malina, 2003. ISBN 80-862-9302-5.

[3] DENTON, Tom. *Automobile electrical and electronic systems: automotive technology : vehicle maintenance and repair*. 4th ed. New York: Routledge, 2012. ISBN 978-008-0969-435.

[4] ONWUBOLU, Godfrey. *Mechatronics: principles and applications*. 1. Burlington, MA: Elsevier Butterworth-Heinemann, 2005. ISBN 0750663790.

[5] PID Theory Explained. *National Instruments: White papers* [online]. Austin: National Instruments, 2011 [cit. 2017-06-06]. Dostupné z: http://www.ni.com/white-paper/3782/en/

[6] ZÍTEK, Pavel, Jaroslav HLAVA a Milan HOFREITER. *Automatické řízení*. Vyd. 2. přeprac. Praha: Vydavatelství ČVUT, 1999. ISBN 80-010-2044-4.

[7] Ziegler-Nichols Tuning Rules for PID. *MICROSTAR LABORATORIES* [online]. Bellevue: Microstar Laboratories, 2017 [cit. 2017-06-07]. Dostupné z: http://www.mstarlabs.com/control/znrule.html

[8] VOŽENÍLEK, Petr, Vladimír NOVOTNÝ a Pavel MINDL. *Elektromechanické měniče*. Vyd. 1. Praha: Vydavatelství ČVUT, 2005. ISBN 9788001031377.

[9] HIRZEL, Timothy. Pulse width modulation. In: *Arduino* [online]. Arduino, 2017 [cit. 2017-06-06]. Dostupné z: https://www.arduino.cc/en/uploads/Tutorial/pwm.gif

[10] TANTOS, Andras. H-Bridges – the Basics. In: *Modular Circuits* [online]. 2011 [cit. 2017-06-06]. Dostupné z: http://www.modularcircuits.com/blog/articles/h-bridge-secrets/h-bridges-the-basics/

[11] Arduino - ArduinoBoardUno. *Arduino* [online]. 2017 [cit. 2017-02-20]. Dostupné z: https://www.arduino.cc/en/Main/ArduinoBoardUno

[12] About \ Wiring. *Wiring* [online]. Ivrea: Wiring, 2011 [cit. 2017-06-06]. ISSN 2011-8376. Dostupné z: http://wiring.org.co/about.html

[13] Arduino Uno R3 Board. In: *JTAG Electronics* [online]. 2017 [cit. 2017-02-22]. Dostupné z: http://www.jtagelectronics.com/wp-content/uploads/2015/08/Arduino-Uno-R3-with-Part-Labels.jpg

[14] ASIMO Specifications. *ASIMO* [online]. b.r. [cit. 2016-11-16]. Dostupné z: http://asimo.honda.com/asimo-specs/

[15] AVAGO. *AEAT-6010/6012 Magnetic Encoder Data Sheet: 10 or 12 bit Angular Detection Device*. 2011. Dostupné také z: http://docs.avagotech.com/docs/AV02-0188EN

[16] AVAGO. *AS38-H39E-B13S Data Sheet: 39 Bits Multi-Turn Absolute House Encoder with BiSS-C Mode Output*. 2016. Dostupné také z: http://www.mouser.com/ds/2/678/ub-005503_DS_AS38-H39E-B13S_2016-02-170-934960.pdf

[17] CTS ELECTROCOMPONENTS. *Series 284 Precision Potentiometer Data Sheet*. 2013. Dostupné také z: http://www.ctscorp.com/wp-content/uploads/2015/11/Series-284.pdf

[18] STRAINSENSE. *Model PGL 60 Data Sheet: Conductive Plastic Hollow Shaft Sensor*. 2012. Dostupné také z: http://www.strainsense.co.uk/downloads/suppliers/potentiometers/potentiometer-pgl-e.pdf

[19] AVAGO TECHNOLOGIES. *AEAT-6010 Application Note 5317: Magnetic Encoder components kit assembly guideline*. AVAGO Technologies, 2007.

[20] DROBNÝ, Daniel. Lepení plastů. In: *Lepidla.cz* [online]. Broumov: Lepidla.cz, 2013 [cit. 2017-06-06]. Dostupné z: http://www.lepidla.cz/cs/a/lepeni-plastu.html

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

Na přiloženém CD je rovněž program pro Arduino ve formátu *.ino a *.txt.