Jihočeská univerzita v Českých Budějovicích

Přírodovědecká fakulta

Řízení krokového motoru v částečně uzavřené smyčce

Bakalářská práce

Jan Veber

Školitel: Ing. Ladislav Ptáček, Ph.D.

Zbytiny 2021

Bibliografické údaje:

J. Veber, 2021: Řízení krokového motoru v částečně uzavřené smyčce [Control of stepper motor in partly closed loop, Bc. Thesis, in Czech] – 48 p., Faculty of Science, University of South Bohemia, České Budějovice, Czech Republic.

Abstrakt:

Cílem této práce je přiblížit studentům se zájmem o pohony metody jak základního, tak pokročilého řízení krokových motorů pomocí mikrokontroléru. Práce by měla vysvětlit principy samotných krokových motorů, druhy pohybových profilů a jejich vlastnosti, výhody, nevýhody a také metody jejich výpočtů. Dále chce tato práce vysvětlit komunikační protokol MODBUS a jeho implementaci do mikrokontroléru PIC32. Součástí této práce je také demonstrační sestava určená pro výuku, která umožní studentům vidět a lépe pochopit chování různých druhů řízení krokového motoru a také jeho integraci do nadřazeného řídicího systému přes MODBUS.

Přínosem práce by mělo být vysvětlení současných způsobů řízení krokových motorů v průmyslu a implementace jejich řídicích systémů pomocí datových sběrnic/sítí. K tomu má dopomoci demonstrační sestava, která díky své předpokládané robustnosti umožní podstoupit různé testy. Byla navržena tak, aby byla bezpečná a odolná proti zničení, což je zvlášť významné při obsluze neznalými osobami, tedy studenty. Díky modulárnosti sestavy bude možno ovládat motor také zařízením třetí strany. Bude tedy možné na sestavě otestovat například program v PC pro komunikaci na datové sběrnici MODBUS.

Abstract:

The goal of this bachelor thesis is to clarify drivers basic of electric driver to students interrested in motion and some extended methods of control stepper motors with microcontroller. This thesis should explain principles of stepper motor, kinds of motion profiles, their properties, advantages, disadvantages and also methods their computations. Second goal is to explain the fieldbus protocol MODBUS and its implementation into PIC32 microcontroller. Practical part of this thesis is design and construction of a demonstration kit which allows to students see and better understand basic of stepper motor control and its integration to master system via MODBUS.

Benefit of this thesis is that it introduce nowadays methods of control stepper motors to student to get view of industry implementation as well as control systems via fieldbuses/networks. The demonstration kit helps to achieve this goal. It was designed to be crash resistant, easy control and safe for users. The demonstration kit could be controlled by a third party device because of its modularity. Moreover the kit should be controlled by a PC software via MODBUS.

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47 b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

Ve Zbytinách dne,

dne 5. 4. 2021 Podpis studenta……...

Poděkování

Děkuji panu Ing. Ladislavu Ptáčkovi, Ph.D., za odborné vedení práce, věcné připomínky, dobré rady a vstřícnost při konzultacích a vypracovávání bakalářské práce. Dále děkuji mé manželce za pochopení. Poděkování patří také kolegům v mém zaměstnání, zejména pak panu Michalu Borovkovi za úžasný 3D tisk.

Obsah

1 Úvod 1
2 Teoretická část 2
2.1 Krokový motor 2
2.1.1 Dělení krokových motorů podle konstrukce 3
2.1.2 Princip krokového motoru 4
2.1.3 Výhody a nevýhody krokového motoru 7
2.2 Řízení krokového motoru 7
2.2.1 Základní princip řízení krokového motoru v otevřené smyčce 8
2.2.2 Princip řízení krokového motoru v uzavřené smyčce 15
2.2.3 Jednoduché řízení dynamiky pohybu 16
2.2.4 Trapezoidní řízení dynamiky pohybu 17
2.2.5 Trapezoidní řízení dynamiky pohybu s „S“ křivkou 19
2.3 Teoretická implementace řízení krokového motoru do mikrokontroléru 20
2.4 Průmyslové datové sběrnice 22
2.4.1 MODBUS 23
2.5 Mikrokontrolér Microchip PIC32MX 25
3 Praktická část 26
3.1 Použité komponenty 26
3.1.1 Krokový motor 26
3.1.2 Vývojová deska OLIMEX PINGUINO-MICRO 26
3.1.3 Dvojitý H můstek L298 27
3.1.4 Převodník napěťových úrovní CD40109 27
3.1.5 Optozávora 28
3.1.6 Převodník RS485 <-> TTL ST1480AC 28
3.2 Mechanická konstrukce výukové sestavy 29
3.3 Elektrické zapojení 31
3.4 Software 32
4 Dosažené výsledky 37
4.1 Testování 37
4.2 MODBUS 40
4.3 Funkce pohonu 42

4.4 Hardware 45
5 Závěr 47
Seznam literatury **49**
Přílohy **51**

1 Úvod

V druhé polovině 20. století se v průmyslu odehrálo mnoho překotných změn, kdy se během několika málo let přešlo od rukodělné práce k automatizovaným výrobním procesům. Tyto procesy pod tlakem trhu prošly dramatickými změnami od různých pístových, vačkových a podobných mechanických strojů ke strojům s elektrickými pohony.

Ze začátku stačily na elektrické pohony systémy brzda/spojka a koncové spínače. Ale rozvoj robotiky si vyžádal konstrukci pohonu s dostatečným výkonem, jehož polohu by navíc bylo možné přesně definovat. Pohon měl splňovat parametry co nejpřesnějšího polohování a současně mít rozumné výrobní náklady. Tomuto dobovému požadavku vyhovovaly krokové motory, jež měly bezkartáčovou konstrukci, měly vysoký statický, přídržný moment a také příznivou cenu. I když byly tyto motory z robotických aplikací postupně vytlačeny stejnosměrnými a poté střídavými servopohony s lepšími dynamickými vlastnostmi, jejich aplikace v moderních průmyslových zařízeních stále dávají smysl, a to zejména pokud jde o náročnost na jejich řídicí elektroniku a tím pádem na jejich cenu. Díky ceně jsou tyto motory velice oblíbené u amatérských stavitelů a vynálezců, bez nichž by tento svět možná ztratil inspiraci.

Výše uvedené neznamená, že vývoj v oblasti řízení krokových motorů ustal na nějakém mrtvém bodě. Naopak. Díky pokročilé a cenově dostupné elektronice je možné aplikovat mikrokrokové řízení, řízení v uzavřené smyčce a také ovládání a sběr dat po datových sítích. Tyto způsoby řízení a komunikace zaručují těmto pohonům v mých očích ještě dlouhou budoucnost .

Tato práce se na konkrétních příkladech věnuje technikám, jak řešit výše uvedené pokročilé funkce řízení krokových motorů. Nemá ambice sloužit jako *atlas* krokových motorů. Budu se zabývat jen jedním, avšak nejpoužívanějším druhem krokového motoru, a to hybridním v bipolárním zapojení.

2 Teoretická část

Tato část popisuje obecné principy krokových motorů a teorii jejich řízení. Dále zde bude popsána datová sběrnice MODBUS a použitý mikrokontrolér PIC32MX.

2.1 Krokový motor

Krokový motor je točivý bezkartáčový synchronní stroj, jehož rotace je způsobena postupným spínáním napájení jednotlivých cívek statorového vinutí podle předem daného vzorce. Synchronní je proto, že rychlost jeho otáčení je přímo úměrná rychlosti spínání jeho cívek až do limitní frekvence. Motor se skládá ze statoru a rotoru, jež mají vyniklé póly. Ve statoru se nachází statorové vinutí, rotor bývá sestaven z permanentních magnetů. Jedna otáčka krokového motoru je rozdělena na množství stabilních poloh – kroků. Standardně má motor 200 kroků čili jeden krok je 1,8°. Tento základní krok se také nazývá celokrok. Velikost tohoto kroku lze vyjádřit vzorcem:

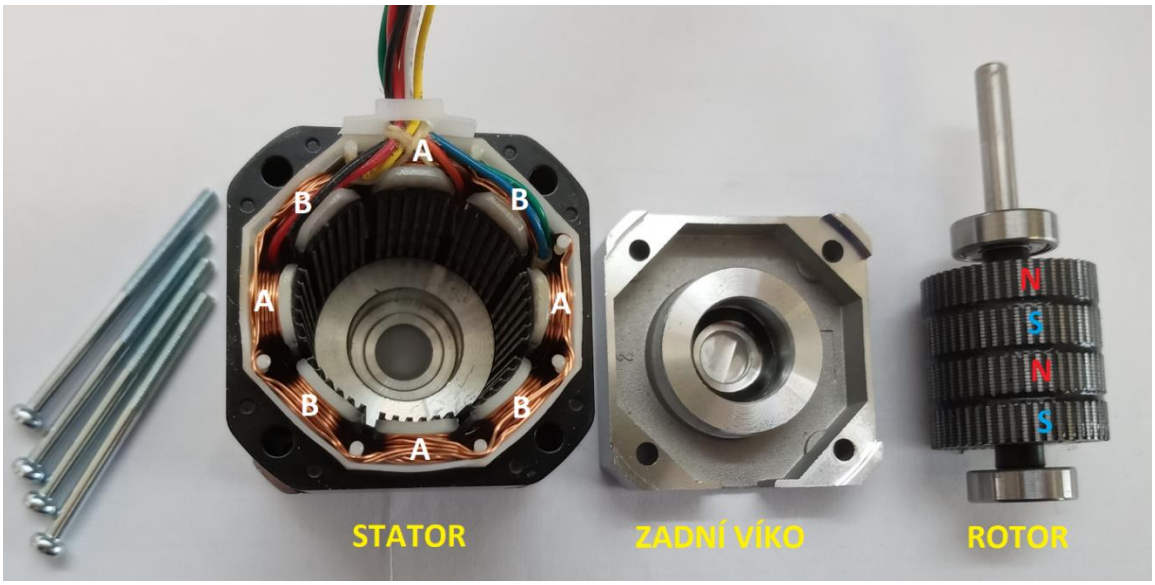
$$\Delta\phi = \frac{360^\circ}{m \cdot N}\ [^\circ]\ , \tag{1}$$

kde $\Delta\phi$ je velikost jednoho kroku ve stupních, m je počet fází statoru a N je počet vyniklých pólů (zubů) rotoru.

Samotný počet celokroků n je potom vyjádřen vzorcem:

$$n = m \cdot N\ [-] \tag{2}$$

Tyto kroky lze ještě dále elektronicky dělit, a to poměrem proudů mezi napájenými cívkami. Z výše uvedeného vyplývá, že krokový motor lze řídit téměř výhradně elektronicky.



Obrázek 1: Rozebraný krokový motor MICROCON SX17. Na vyniklých pólech statoru jsou patrné zuby stejně jako na rotoru. Zuby pólů rotoru jsou vůči sobě navzájem posunuty

2.1.1 Dělení krokových motorů podle konstrukce

Za dobu vývoje krokových motorů vznikly jejich tři hlavní typy – krokový motor s permanentními magnety, s proměnnou reluktancí a hybridní krokové motory.

a) Krokový motor s permanentním magnetem

Tento motor se též nazývá motor s aktivním rotorem. Rotor je tvořen permanentním magnetem a nemá vyniklé póly. Ve statoru jsou potom 4 cívky, jejichž postupným zapínáním a vypínáním dochází ke změně jejich pólů. Rotor se pak naklání do polohy, kde je nejvíce přitahován tou či onou cívkou. Díky tomu, že je rotor tvořen jedním magnetem, má tento motor pouze 4 polohy. Zároveň má díky permanentnímu magnetu vyšší intenzitu magnetického pole a tím pádem příznivější momentové charakteristiky než motory s proměnnou reluktancí. Motory s permanentním magnetem se obvykle zapojují buď jako unipolární, nebo jako bipolární. Unipolární motor lze zapojit jako bipolární, avšak bipolární zapojit jako unipolární nelze.

b) Krokový motor s proměnnou reluktancí

Motor této konstrukce se také nazývá drápkový motor. Pracuje na principu změny magnetického odporu (reluktance). Rotor je z magneticky měkkého materiálu a má vyniklé póly. Stator se skládá z magnetického materiálu s vyniklými póly a obsahuje vinutí jednotlivých fází. Počet pólů rotoru a statoru se musí lišit, aby se motor točil. Při přepínání jednotlivých fází statoru se rotor natočí tak, aby byl odpor magnetického

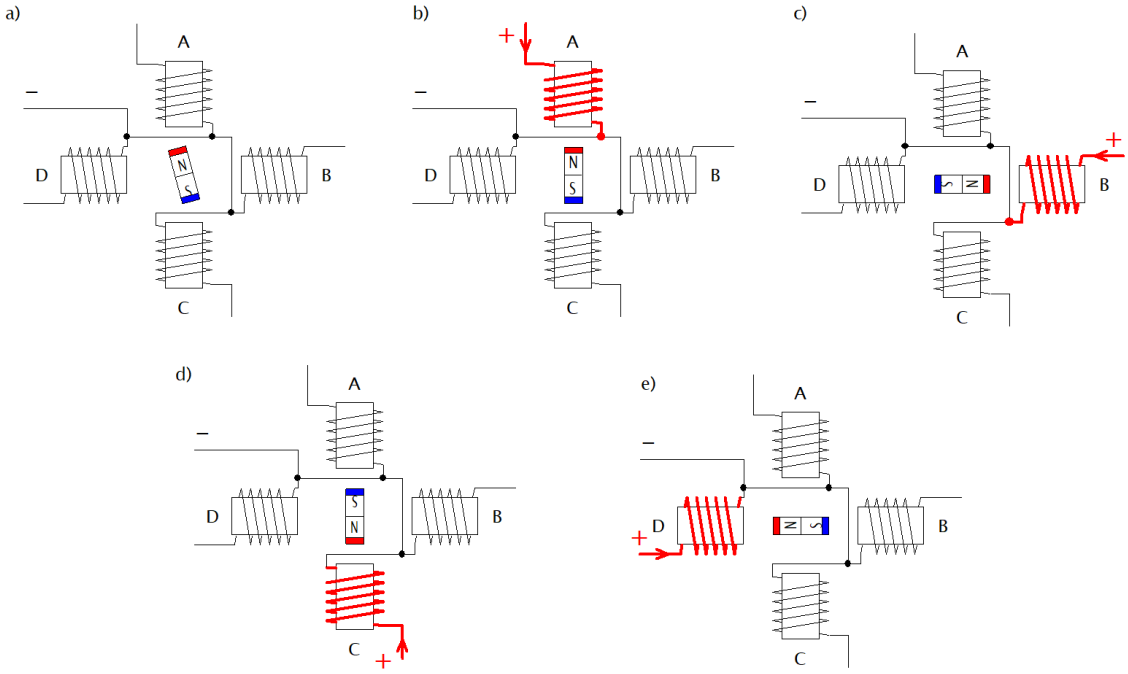
obvodu co nejmenší neboli nejbližší zub rotoru se natočí naproti nejbližšímu zubu statoru. Počet vyniklých pólů ovlivňuje výsledný počet kroků na otáčku. Proto mají tyto motory vysoké rozlišení. Díky rotoru tvořenému pouze magneticky měkkým materiálem má tento typ motoru malý záběrný moment, a proto ho nelze použít ve výkonových aplikacích.

c) Hybridní krokový motor

Hybridní krokový motor kombinuje vlastnosti předchozích dvou konstrukcí a zejména tomuto typu motoru je věnována tato práce. Rotor je tvořen permanentním magnetem, avšak s pólovými nástavci – zuby. Ty díky konstrukci střídají svou magnetickou polaritu. Tento motor tedy skloubil výhody obou výše uvedených typů motorů, a to momentové charakteristiky z motoru s permanentním magnetem a výhodu vysokého rozlišení z motoru s proměnlivou reluktancí. Stejně jako motory s permanentním magnetem lze zapojit vinutí statoru jako unipolární, nebo bipolární. Blíže o tom pojednává kapitola 2.3.

2.1.2 Princip krokového motoru

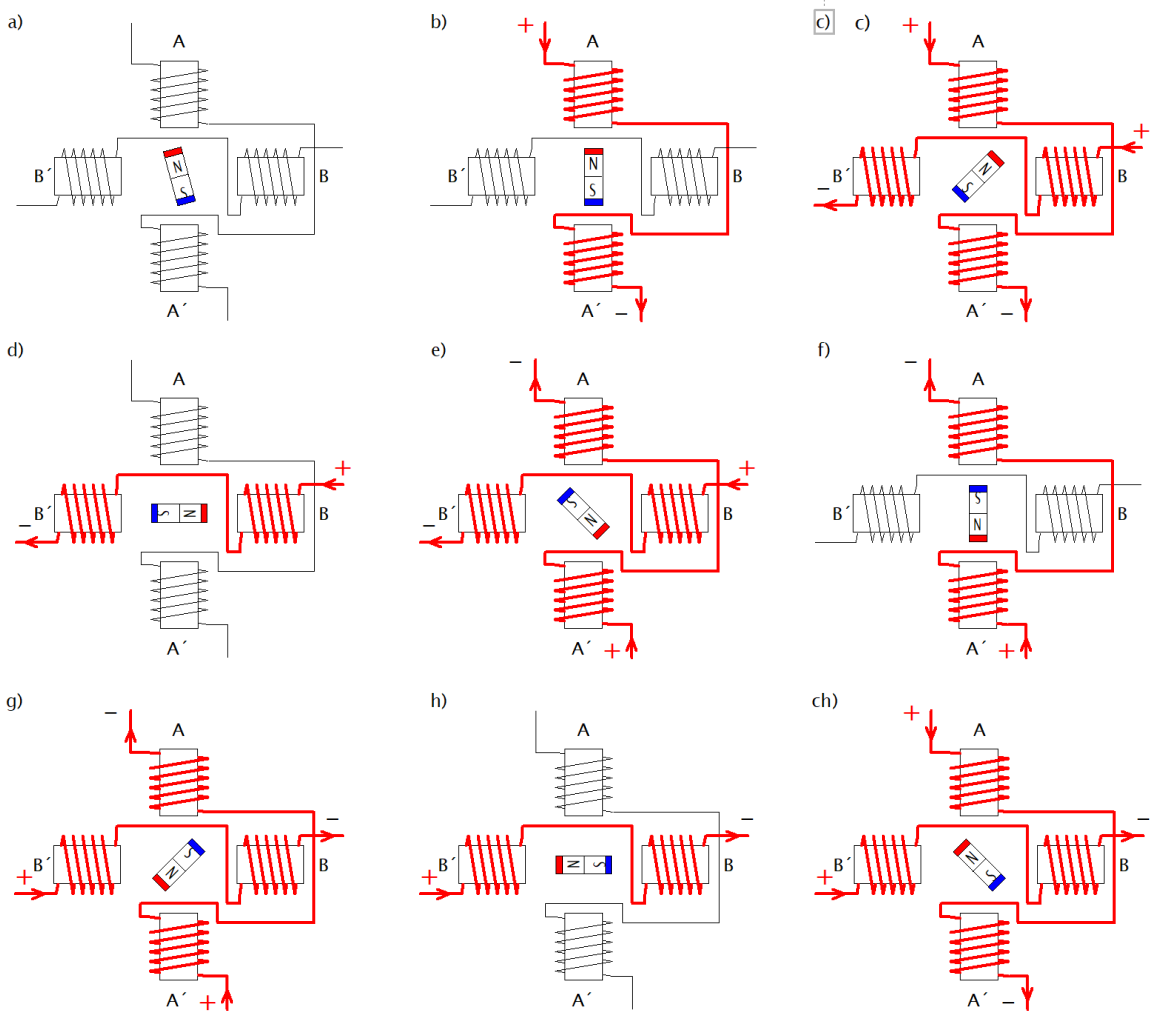
Na zjednodušeném schématu popíšu princip unipolárního krokového motoru, který má 4 celokroky na otáčku a buzenou pouze jednu fázi. Rotor je tvořen permanentním magnetem se dvěma póly a stator je tvořen čtyřmi cívkami – fázemi s jedním půlovým nástavcem. Tyto fáze si označme A, B, C a D. Motor v unipolárním zapojení se snadněji ovládá, protože se nemusí otáčet polarita napájecího napětí. Zároveň má však nevýhodu nižšího záběrného momentu právě proto, že je napájena vždy jen jedna, nebo maximálně dvě cívky. Postupně budeme napájet všechny čtyři cívky, aby byl princip motoru zřejmý.



Obrázek 2: Sekvence spínání cívek, rotor se točí doprava: a) rotor je v náhodné pozici, b) je napájena cívka A, c) je napájena cívka B, d) je napájena cívka C, e) je napájena cívka D

Z výše uvedených obrázků je vidět, že rotor (kotva) se otáčí za právě napájenou cívkou. Jinými slovy, při napájení jedné cívky statoru se z ní stává elektromagnet a ten přitáhne opačný pól rotoru. Postupným přepínáním cívek statoru vznikne točivé magnetické pole a díky tomu se rotor otáčí. Pro vytěžení maximálního momentu motoru se můžou budit dvě fáze najednou. Tím se sice posune pozice celokroku o 45°, ale počet kroků na otáčku zůstane stejný. Detailněji se tímto zabývá kapitola 2.2.1. Rychlost otáčení rotoru závisí přímo na rychlosti přepínání jednotlivých cívek, protože se jedná o synchronní točivý stroj. Pokud se přepínání napájení cívek zastaví a zůstane napájena jen jedna (nebo dvě), rotor setrvá v této pozici.

V dnešní době se díky pokročilé elektronice využívají nejvíce bipolární krokové motory. Ty fungují v principu naprosto stejně jako unipolární. Rozdílné je napájení a zapojení cívek jednotlivých fází. Každá fáze se skládá ze dvou cívek zapojených do série, nebo paralelně podle požadavků na rychlost, nebo moment. Cívky fáze A jsou rozděleny na cívku A a A´. Stejně tak jsou cívky fáze B rozděleny na cívku B a B´. Na obr. 3 si tedy znovu ukažme zjednodušený krokový motor z výše uvedeného příkladu, avšak v bipolárním sériovém zapojení. Pro úplnost ještě přidáme poloviční kroky, kterými celá sekvence také začne.



Obrázek 3: Sekvence spínání cívek, rotor se točí doprava: a) rotor je v náhodné pozici, b) je napájena cívka A+A´-, c) jsou napájeny cívky A+A´- a B+B´-, d) je napájena cívka B+B´-, e) jsou napájeny cívky B+B´- a A´+A-, f) je napájena cívka A´+A-, g) jsou napájené cívky A´+A- a B´+B-, h) je napájena cívka B´+B-, ch) jsou napájené cívky B´+B- a A+A´-

Z obrázků je vidět, že napájení cívek jednotlivých fází je složitější vzhledem k unipolárnímu řízení. Potřebujeme 2× tolik spínacích prvků než v předchozím zapojení. Na jedno vinutí tedy potřebujeme 4 spínací prvky zapojené do tzv. H můstku. Další fakt je ten, že při polovičním kroku, kdy je rotor otočen o 45° vůči předchozímu celokroku, je moment motoru o cca. 30 % menší než při celokroku. Tento fenomén je blíže popsán v kapitole 2.2.1.

2.1.3 Výhody a nevýhody krokového motoru

Mezi hlavní výhody moderních krokových motorů patří zejména přesnost natočení, prakticky 100% opakovatelnost polohování, mechanická odolnost a také cenová dostupnost. Tyto motory jsou oblíbené zejména proto, že při správném dimenzování zátěže pracují spolehlivě a bez zpětné kontroly jejich polohy. Díky tomu nalezneme krokové motory v mnoha aplikacích od tiskáren přes CNC stroje po 3D tiskárny. Krokové motory jsou dokonce i ručičky ukazatelů v přístrojových deskách automobilů a jiných strojů.

Jako všechno na tomto světě je každá výhoda vyvážena nějakou nevýhodou. Krokové motory mají problém s maximálními otáčkami a to ten, že při rychlém přepínání cívek jednotlivých vinutí statoru se magnetický obvod motoru nestíhá saturovat/odsaturovat. Důsledkem toho klesne točivý moment ke kritickému minimu a motor se zastaví. Proto je rozumné používat krokové motory do cca. 2000 ot/min. Další nevýhodou je vznik rezonancí při otáčení rotoru, způsobených přepínáním cívek vinutí a díky tomu skokovému pohybu rotoru. Toto lze značně omezit tzv. mikrokrokováním. Nicméně z toho vyplývá další nevýhoda složitějšího řízení. Krokový motor má ještě jednu zásadní nevýhodu, a to absolutní neodolnost proti mechanickému přetížení, při němž totiž motor vypadne ze synchronního režimu a ztratí krok, nebo kroky. Pokud je mechanická zátěž v kritické oblasti, můžou se kroky ztrácet náhodně a tím pádem můžou komplikovat případnou diagnostiku problému v té dané aplikaci. Pokud je motor již kompletně přetížen, dojde k jeho zastavení. Znovu se rozběhne pouze po odlehčení a také je nutno ho spustit z nulové rychlosti. Toto lze řešit přídavným snímačem polohy hřídele, avšak to dále komplikuje řízení.

2.2 Řízení krokového motoru

Jak již bylo zmíněno výše, krokový motor lze řídit výhradně elektronicky, a to přepínáním statorových cívek ve vhodném pořadí. Pokud máme zdroj, který je schopen dodat proud o nominální hodnotě použitého krokového motoru, postačí nám pouze spínat jednotlivé cívky výkonovým tranzistorem. Tímto způsobem se však dostaneme k rozlišení maximálně poloviční kroku a motor bude mít velké rezonance. Pro zjemnění je výhodné v každém kroku měřit spínaný proud a regulovat ho pomocí PWM regulace. Detailně o tom pojednávají následující kapitoly.

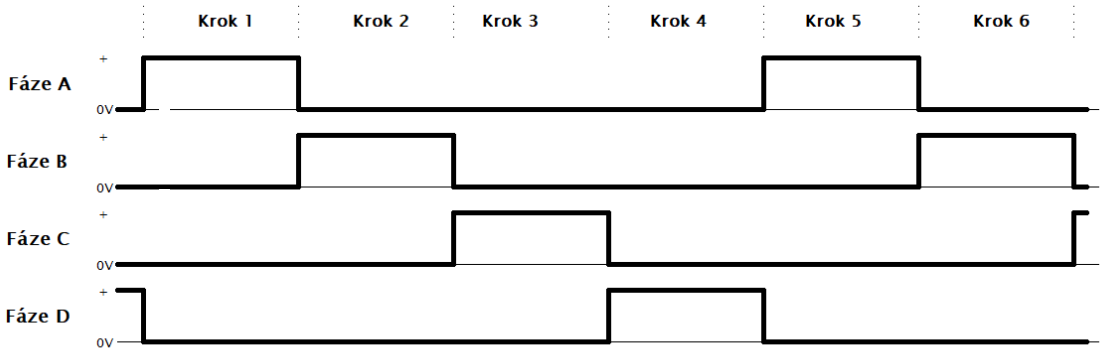
2.2.1 Základní princip řízení krokového motoru v otevřené smyčce

Řízením v otevřené smyčce rozumíme to, že pokud budeme budit vinutí statoru určitou sekvencí, budeme předpokládat, že se rotor skutečně natočil tak, jak jsme chtěli, čili využijeme jednu ze základních výhod krokového motoru, a to přesnost natočení. Tento způsob řízení by se dal rozdělit do následujících skupin.

2.2.1.1 Unipolární krokový motor

a) Celokrokové řízení s buzením jedné fáze v unipolárním zapojení

Za nejjednodušší způsob řízení krokového motoru lze považovat celokrokové s buzením pouze jedné fáze. Rotor se vždy natočí tak, aby byla vzdálenost mezi zubem rotoru a pólovým nástavcem statoru co nejmenší. Při použití této metody řízení bude mít motor malý záběrný moment a počet kroků bude odpovídat katalogovým údajům celokroků daného motoru. To je důvod, proč se v praxi téměř nepoužívá. Jak bude vypadat průběh signálů na jednotlivých cívkách, ukazuje obr. 4. Při otáčení v jednom směru bude sekvence spínání fází A, B, C, D, A, B… a při opačném směru bude sekvence D, C, B, A, D…

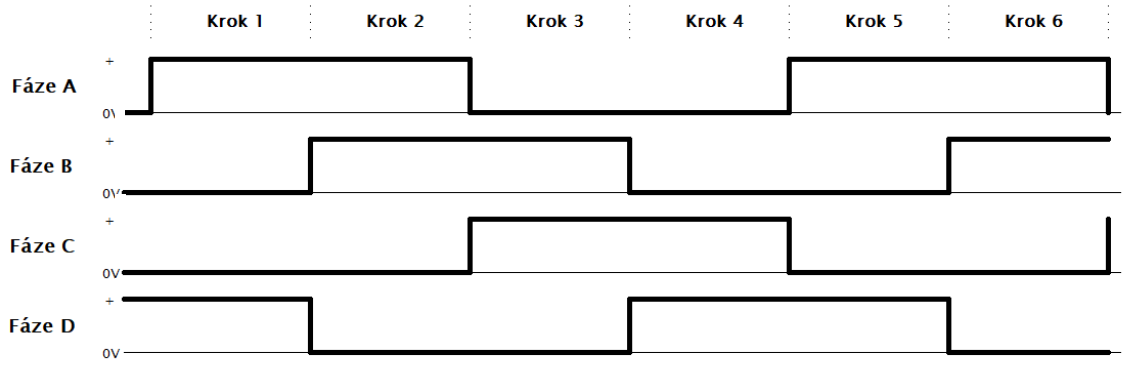


Obrázek 4: Celokrokové řízení s buzením jedné fáze v unipolárním zapojení

b) Celokrokové s buzením dvou fází v unipolárním zapojení

Pokud budeme chtít situaci ohledně malého momentu dále zlepšit, je vhodné budit vždy dvě cívky najednou. To je ostatně nejběžnější způsob řízení krokového motoru. Při tomto typu řízení je záběrný moment nejvyšší dosažitelný. Počet kroků je stejný jako u předchozího řízení, ale je o zub posunutý. Je to dáno tím, že při buzení dvou fází se rotor při celokroku natočí do polohy mezi pólovými nástavci. Je to zřejmé z obr. 3. Na obr. 5 je průběh signálů na jednotlivých cívkách. Sekvence spínání fází je A+D, A+B,

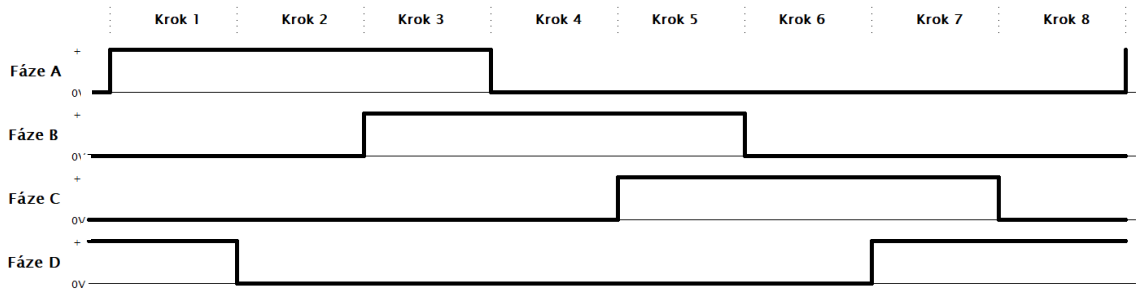
B+C, C+D, A+D... V opačném směru bude sekvence spínání fází A+D, C+D, B+C, A+B, A+D...



Obrázek 5: Celokrokové řízení s buzením dvou fází v unipolárním zapojení

c) Polokrokové řízení v unipolárním zapojení

Pokud chceme mít 2× lepší rozlišení polohovatelnosti motoru, můžeme řídit motor polokrokově. Princip spočívá v tom, že při buzení motoru dvěma fázemi vložíme do sekvence spínání fází ještě buzení jedné fáze. Jak již bylo zmíněno v kapitole 2.1.2, má tento způsob řízení jednu velkou nevýhodu, a sice tu, že v momentě, kdy se budí pouze jedna fáze, je moment motoru nižší o cca. 30 %. Vyplývá to z toho, že při buzení dvou fází se najednou vektory magnetických indukcí sčítají, a jelikož úhel, který tyto vektory svírají, je 45°, je jejich výsledná hodnota √2krát větší než hodnota každého z nich. Z toho vyplývá, že při buzení jednou fází bude magnetická indukce √2krát menší než při buzení dvěma fázemi. Toto lze řešit zvýšením budícího proudu v polovičním kroku. Na obr. 6 je znázorněn průběh signálů při polokrokovém řízení. Sekvence spínání jednotlivých fází bude A+D, A, A+B, B, B+C, C, C+D, D, A+D… Analogicky při změně smyslu otáčení bude sekvence A+D, D, C+D, C, B+C, B, A+B, A, A+D…



Obrázek 6: Polokrokové řízení unipolárním zapojením

d) Mikrokrokování unipolárního motoru

Tato metoda již vyžaduje složitější způsob řízení, jelikož kromě pozměněné sekvence signálů jednotlivých fází je třeba ještě regulovat velikost proudů v jednotlivých krocích. Nejlépe je to PWM modulací, o které se pojednává níže v této kapitole. Princip je v tom, že jednotlivé celokroky rozdělíme do několika menších kroků, obvykle do 4, 8 nebo 16. Samozřejmě je můžeme dělit ještě dále, ale v praxi to už nemá díky mechanickým odporům žádný smysl. Takže ze standardního motoru s rozlišením 200 kroků/1,8° můžeme vytvořit mikrokrokováním 800 kroků/0,45°, 1600 kroků/0,225° nebo 3200 kroků/0,1125°. Jako bonus se sníží hlučnost a celkově bude mít motor jemnější chod. Pokud tedy kroky takto rozdělíme, můžeme lépe regulovat poměry magnetických sil v motoru mezi buzenými vinutími a tím dosáhneme teoreticky libovolné polohovatelnosti mezi sousedními celokroky. Avšak díky výše zmíněným mechanickým odporům nelze v praxi zaručit dobrou opakovatelnost. Aby se motor otáčel přesně v úhlu děleném požadovaným mikrokrokem, použijeme vzorec:

$$I_A = I_{max} \cdot sin\left(\frac{180°}{n}\right)[A], \tag{3}$$

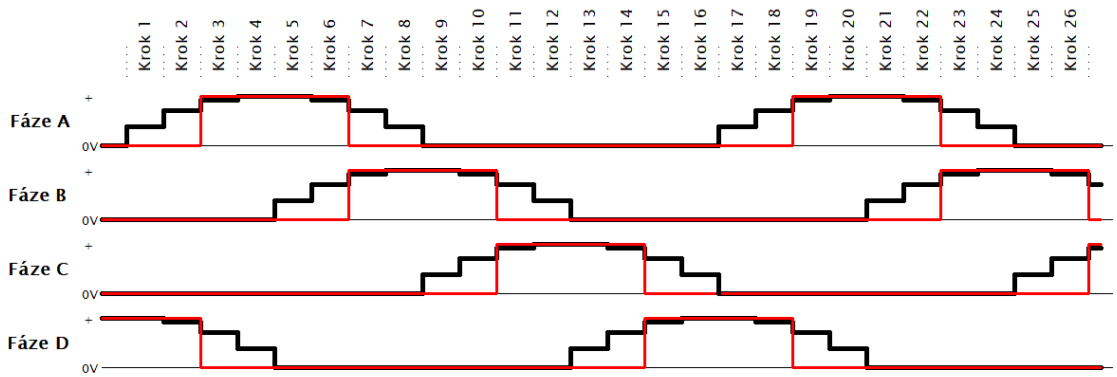
kde I_A je počítaný proud fáze A, I_{max} je maximální předpokládaný proud fází a n je číslo kroku. Výsledný průběh proudů ve fázi se bude podobat sinusoidě tím více, čím více mikrokroků se použije. Pro fáze B až D použije stejný vzorec, jen se proti předchozí fázi posune o $\frac{\pi}{2}$ neboli o 90°. Příklad výpočtu pro jednu fázi zobrazuje tab. 1.

Tabulka 1: Příklad výpočtů proudů pro jednu fázi v unipolárním zapojení

Max. proud [A]	Počet mikrokroků
2	8

Číslo mikrokroku	Proud fáze A [A]
0	0,00
1	0,77
2	1,41
3	1,85
4	2,00
5	1,85
6	1,41
7	0,77

Na obr. 7 je znázorněn průběh signálů na čtyřech fázích unipolárního motoru při mikrokrokování v rozlišení 1 krok na 4 mikrokroky čili celkem 8 mikrokroků na 180°. Pro názornost je obrázek proložen červeným signálem při celokrokovém řízení.



Obrázek 7: Mikrokrokování při unipolárním zapojení

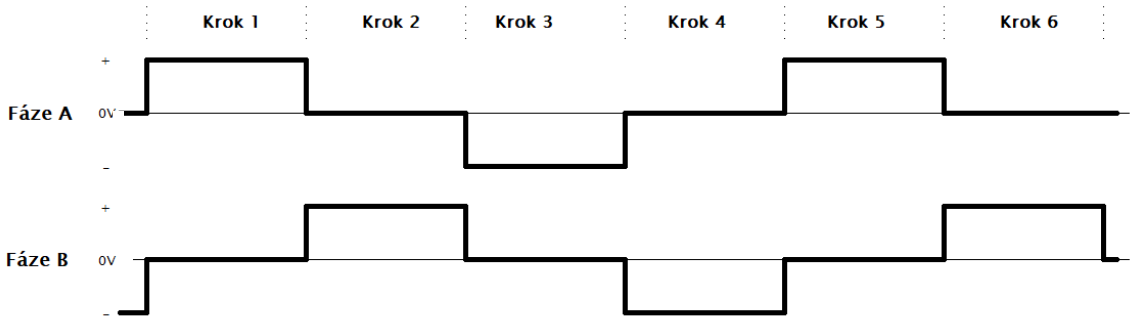
2.2.1.2 Bipolární krokový motor

Oproti unipolárnímu motoru je řízení bipolárního motoru, kde se ovládají čtyři fáze, pouze dvoufázové, ale s proměnlivou polaritou. U bipolárního motoru už si nevystačí ne s jedním spínacím tranzistorem na fázi. Musíme použít čtyři tranzistory zapojené do H-můstku kvůli již zmíněným změnám polarity. Začněme opět od nejjednoduššího způsobu řízení.

a) Celokrokové řízení s buzením jedné fáze v bipolárním zapojení

Stejně jako u unipolárního zapojení je tento způsob řízení téměř nepoužívaný, protože motor značně rezonuje a dosáhne jenom katalogového rozlišení. Vzhledem k buzení pouze jedné fáze není ani dosaženo maximálního možného momentu motoru.

Na obr. 8 je průběh proudů při tomto způsobu řízení. Sekvence spínání fází bude A+, B+, A-, B-, A+… Při otáčení na druhou stranu to bude A+, B-, A-, B+, A+…

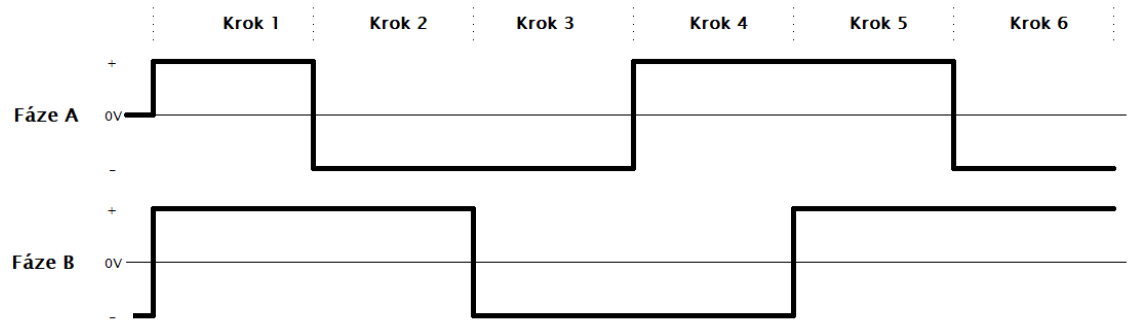


Obrázek 8: Celokrokové řízení s buzením jedné fáze v bipolárním zapojení

b) Celokrokové řízení s buzením dvou fází v bipolárním zapojení

Pokud budeme budit dvě fáze najednou, tak se stejně jako u unipolárního zapojení posune celokrok o jeden zub a záběrný moment se dostane na maximum. Počet katalogových kroků zůstane stejný jako v předchozím případě. Ovšem rezonance díky skokovému pohybu rotoru zůstanou.

Na obr. 9 je průběh signálů při tomto způsobu řízení. Sekvence spínání fází je A+B+, A-B+, A-B-, A+B-, A+B+… Opačně se bude motor točit při sekvenci A+B+, A+B-, A-B-, A-B+, A+B+…

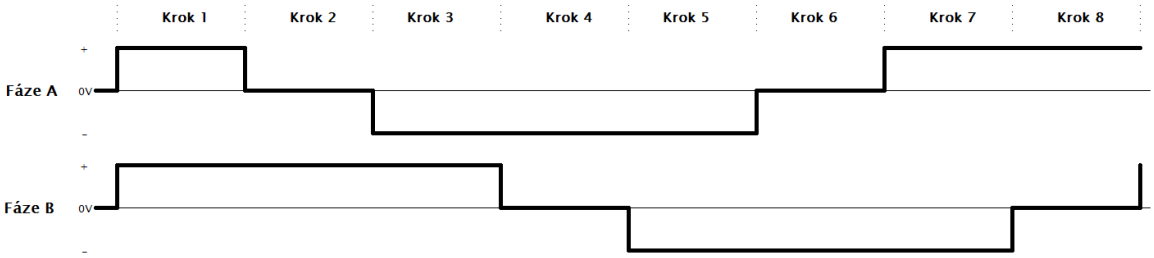


Obrázek 9: Celokrokové řízení s buzením dvou fází v bipolárním zapojení

c) Polokrokové řízení v bipolárním zapojení

Posledním „jednoduchým“ způsobem řízení bipolárního motoru je polokrokové řízení. Jak již bylo uvedeno u unipolárního motoru, z tohoto způsobu řízení vyplývají dvě zásadní fakta. První je dvojnásobné rozlišení motoru oproti katalogovým údajům a druhé je o cca 30 % zmenšený moment motoru v okamžiku buzení jedné fáze, a to ze stejných důvodů jako u unipolárního motoru.

Na obr. 10 je znázorněn průběh signálů při polokrokovém řízení. Sekvence spínání fází bude A+B+, B+, A-B+, A-, A-B-, B-, A+B-, A+, A+B+… v jednom směru otáčení a A+B+, A+, A+B-, B-, A-B-, A-, A-B+, B+, A+B+…



Obrázek 10: Polokrokové řízení v bipolárním zapojení

d) Mikrokrokování bipolárního motoru

Jestliže chceme mít co největší volnost v polohování a maximálně potlačit rezonance a tím pádem hluk motoru, musíme přistoupit k mikrokrokování. Stejně jako u unipolárního motoru spočívá princip v rozdělení jednoho kroku do několika mikrokroků s různými proudy pomocí PWM modulace. Tím se dosáhne téměř sinusového průběhu budících proudů v závislosti na počtu mikrokroků. I zde platí, že rozdělovat kroky na více než 16 mikrokoků nemá valný význam, ať už z důvodu nízké polohové opakovatelnosti, nebo z důvodu vysoké spotřeby výpočetního výkonu řídicího systému.

Protože se ovládají jenom dvě fáze a musí se měnit polarita, použijeme pro výpočet budicích proudů vzorce:

$$I_A = I_{max} \cdot sin\left(\frac{360°}{n}\right)[A] \tag{4}$$

a

$$I_B = I_{max} \cdot cos\left(\frac{360°}{n}\right)[A], \tag{5}$$

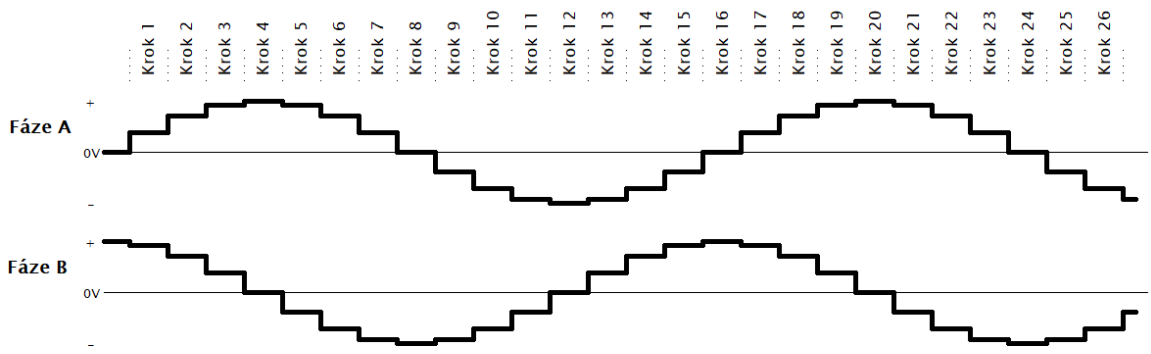
kde I_A proud fáze A, I_B je proud fáze B, I_{max} je maximální předpokládaný proud fází a n je číslo kroku. Rozdíl je oproti unipolárnímu motoru v tom, že počítáme v celém rozsahu jednotkové kružnice a fázový posun fáze B oproti fázi A nám zajistí funkce cosinus. Tab. 1 nám ukazuje vzorový výpočet dvou fází s maximálním proudem 2 A a s počtem 8 mikrokroků na celokrok, takže 16 kroků na 360° [1].

Tabulka 2: Příklad výpočtů proudů pro fáze A a B v bipolárním zapojení

Max. proud [A]	**Počet mikrokroků**
2	16

Číslo mikrokroku	**Proud fáze A [A]**	**Proud fáze B [A]**
0	0,00	2,00
1	0,77	1,85
2	1,41	1,41
3	1,85	0,77
4	2,00	0,00
5	1,85	−0,77
6	1,41	−1,41
7	0,77	−1,85
8	0,00	−2,00
9	−0,77	−1,85
10	−1,41	−1,41
11	−1,85	−0,77
12	−2,00	0,00
13	−1,85	0,77
14	−1,41	1,41
15	−0,77	1,85

Na obr. 11 je potom znázorněn průběh proudů na fázích A a B vypočítaných v tab. 2.



Obrázek 11: Mikrokrokování při bipolárním zapojení

2.2.2 Princip řízení krokového motoru v uzavřené smyčce

Řízením v uzavřené smyčce se označuje, že dosažení každého předpokládaného kroku/mikrokroku se kontroluje pomocí snímače polohy. Pokud je kontrolovaný krok dosažený, může se pokračovat dalším krokem. Pokud se požadovaného kroku nedosáhne, může se řídicí systém zachovat několika způsoby:

a) Čeká, dokud se požadované polohy nedosáhne, a poté pokračuje dále. Pokud se požadované polohy nedosáhne v určitém čase, řídicí systém zastaví sekvenci pohybu a vyhlásí chybu.
b) Čeká, dokud se požadované polohy nedosáhne a poté pokračuje dále. Pokud se požadované polohy nedosáhne v požadovaném čase, řídicí systém zvýší proud do buzené fáze (do fází). Zvyšování proudu může proběhnout v několika iteracích po vypršení časových limitů. Pokud se ani po maximálním povoleném zvýšení proudu motor neotočí do požadované polohy, řídicí systém zastaví sekvenci pohybu a vyhlásí chybu.
c) Řídicí systém pracuje v režimu, kdy se permanentně snaží dosáhnout požadované polohy, ale nezvyšuje proud do buzených fází a ani nevyhlašuje chybu nedosažení požadované polohy. Jen vyhlásí status dosažené polohy, pokud se mu podaří této polohy dosáhnout. Pracovně ho nazvu režim „pružinka“, protože se motor bude chovat jako napnutá pružina s konstantní silou.

Společným a logickým rysem všech těchto režimů je, že řídicí systém kontroluje polohu v obou směrech čili nestačí pulzy ze snímače jen inkrementovat, ale také dekrementovat. To závisí na směru otáčení motoru a také na směru neočekávaných

vnějších sil, působících na motor z vnějška. Díky tomu může systém vypočítat nový počet kroků k dosažení původně požadované polohy, například při dočasném přetížení motoru. Vždy je zapotřebí dvou signálů vzájemně posunutých o π, aby došlo k rozeznání směru otáčení. Více o tomto pojednává kapitola 2.3.

Vzhledem k amatérským podmínkám, při kterých vzniká tato práce, se nebudou kontrolovat všechny polohy mikrokroků, ale budou se kontrolovat v celých krocích. Je to logické, krokový motor má v těchto pozicích nejvyšší moment a tím pádem jsou tyto pozice lépe dosažitelné a méně ovlivněné mechanickými odpory v mechanické soustavě. Proto je v názvu této práce obsažena „částečně uzavřená smyčka“. V našem případě to znamená 200 kontrol na jednu otáčku motoru. Kdyby se použil například průmyslový snímač polohy Sick DFS60B-S4PA10000 [8], mohli bychom kontrolovat 10000× za jednu otáčku.

2.2.3 Jednoduché řízení dynamiky pohybu

Nejjednodušší forma polohování krokového motoru je sekvence požadovaného počtu kroků o určité frekvenci. Toto polohování má velkou výhodu v tom, že řídicí systém může velice jednoduše vypočítat čas, za který se motor dostane do požadované pozice, respektive určí frekvenci kroků, aby se motor dostal do požadované pozice v žádaném čase. Požadovanou frekvenci kroků lze vyjádřit vzorcem:

$$f = \frac{1}{T}\ [Hz], \tag{6}$$

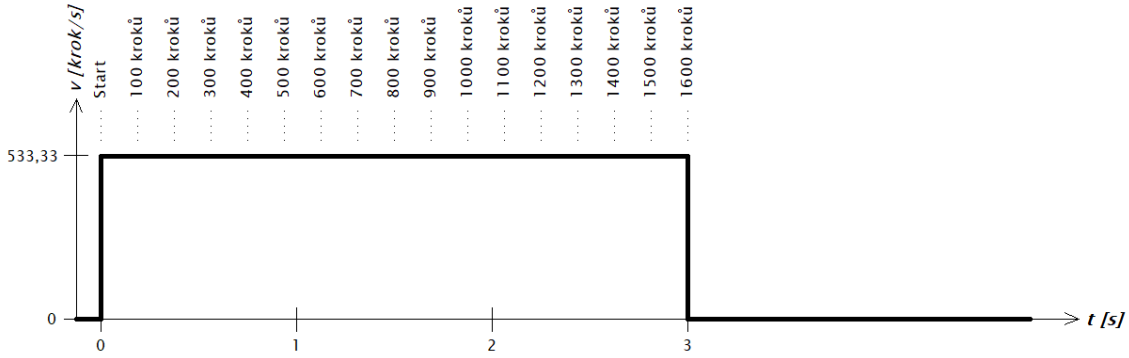
kde *T* je perioda, kterou spočítáme podle vzorce:

$$T = \frac{t}{n}\ [-], \tag{7}$$

kde t je požadovaný čas pohybu a *n* je požadovaný počet kroků (délka pohybu).

Pokud tedy budeme chtít s motorem popojet o 1600 kroků za 3 s, vypočítáme podle (7) periodu 0,001875. Dosadíme-li tuto hodnotu do (6), frekvence pro mikrokontrolér bude 533,33 Hz. Tuto frekvenci můžeme také chápat jako rychlost 533,33 kroků za sekundu. Tento způsob udávání rychlosti krokových motorů se také často používá.

Obr. 12 ukazuje průběh tohoto jednoduchého rychlostního profilu. Z obrázku je vidět, že akcelerace a decelerace motoru jsou skokové, průběh rychlostního profilu je tedy obdélníkový. Z toho plyne velká nevýhoda uvedeného způsobu řízení pohybu krokového motoru, a to omezení rozsahu rychlostí motoru. Díky setrvačným silám v rotoru a v mechanické soustavě nelze využít maximální rychlosti. Při vysoké cílové rychlosti by se díky velké akceleraci a deceleraci ztrácely kroky, nebo by se motor zcela zablokoval. Tím by zmizela hlavní vlastnost krokového motoru, a to přesnost polohování. Tento profil se proto využívá u zapisovacích zařízení, u pohonů nenáročných na dynamiku a všude, kde nehrozí náhlé změny zatížení atd.



Obrázek 12: Jednoduchý rychlostní profil

2.2.4 Trapezoidní řízení dynamiky pohybu

Pokud budeme chtít využívat maximální rozsah použitelných otáček motoru, je vhodné použít trapezoidní rychlostní profil. Tento profil kontroluje akceleraci a deceleraci. Díky tomu lze využít i vysoké konstrukční rychlosti. Během kontrolované akcelerace a decelerace je možno držet dynamické zatížení motoru v požadovaných mezích a tím zabránit ztrácení kroků při využití vysokých cílových rychlostí.

Při výpočtu trapezoidního profilu budeme vycházet z toho, že se tento profil skládá ze tří sekcí: akcelerace – d_1, konstantní rychlost – d_2 a decelerace – d_3, viz obrázek 13. Tyto sekce si můžeme představit jako trojúhelník, obdélník a trojúhelník. Parametry pro požadovaný pohyb lze zadat několika způsoby. Záleží na tom, co je pro daný pohyb prioritou. V následujícím příkladu budeme požadovat, aby motor urazil určitou vzdálenost za danou dobu s tím, že zadáme požadovanou dobu akcelerace a decelerace. Budeme uvažovat, že akcelerace a decelerace jsou shodné. Takže prioritou je pro nás čas a dráha.

Příklad: Motor má urazit dráhu d = 1500 kroků za dobu t = 5 s. Akcelerace i decelerace bude trvat 0,75 s.

Je tedy potřeba zjistit akceleraci/deceleraci a maximální cílovou rychlost.

Řešení:

1. **Délka akcelerace a decelerace**

Pokud je akcelerace konstantní bude obecně platit, že:

$$d = v_i t + \frac{1}{2} a t^2 \ [m \cdot s^{-2}], \tag{8}$$

kde d je dráha po dobu akcelerace, v_i je počáteční rychlost, a je akcelerace a t je čas akcelerace.

Dosazením do (8) a za předpokladu, že $v_i = 0$, dostaneme:

$$d_1 = \frac{1}{2} a (0{,}75)^2 \ [m \cdot s^{-2}] \tag{9}$$

2. **Maximální rychlost**

Určíme maximální rychlost podle vzorce:

$$v_{max} = v_i + at \ [m \cdot s^{-1}], \tag{10}$$

kde v_{max} je maximální rychlost, v_i počáteční rychlost, a akcelerace a t čas akcelerace.

Pokud (10) doplníme a vyjádříme z něj a, dostaneme:

$$a = \frac{v_{max}}{0{,}75} \ [m \cdot s^{-2}] \tag{11}$$

Dosadíme-li (11) do (9), dostaneme, že:

$$d_1 = \frac{1}{2} \left(\frac{v_{max}}{0{,}75}\right) (0{,}75)^2$$

$$d_1 = \frac{1}{2} \left(\frac{v_{max}}{0{,}75}\right) (0{,}75)^2$$

$$d_1 = 0{,}375 v_{max} \ [m] \tag{12}$$

3. **Konstantní rychlost**

Podle (8) spočítáme konstantní dráhu:

$$d_2 = v_{max}t + \frac{1}{2}at^2$$

$$d_2 = v_{max}(5 - 0{,}75 - 0{,}75) + \frac{1}{2}(0)(0{,}75)^2$$

$$d_2 = 3{,}5v_{max}$$

Víme, že celková dráha $d = d_1 + d_2 + d_3$. Také víme, že $d_1 = d_3 = 0{,}375v_{max}$, čili:

$$1500 = 0{,}375v_{max} + 3{,}5v_{max} + 0{,}375v_{max}$$

$$v_{max} = 352{,}941\ [krok/s]$$

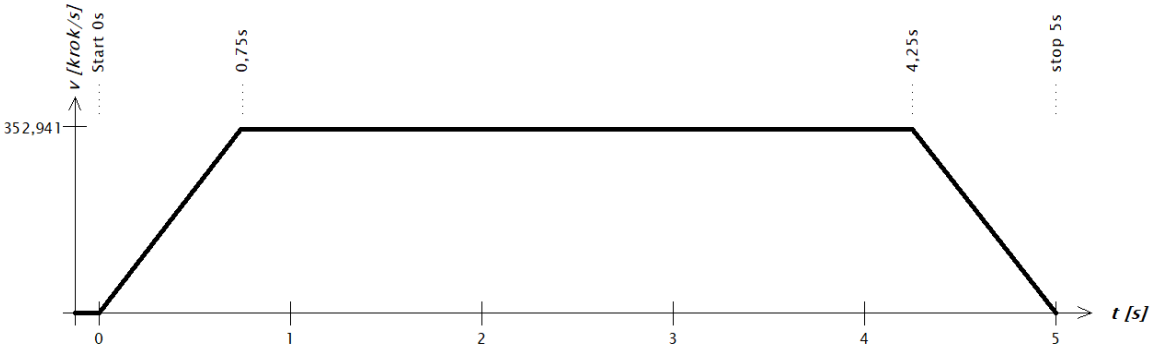
4. **Akcelerace a decelerace**

Dosadíme-li do (10), tak dostaneme:

$$352{,}941v_{max} = 0 + a \cdot 0{,}75$$

$$a = 470{,}588\ [krok/s]$$

Jelikož prakticky nikdy nevyjde vypočítaný krok na celé číslo, je nutno toto nějakým způsobem vykompenzovat. Způsob řešení tohoto problému je popsán v kapitole 2.3.



Obrázek 13: Trapezoidní profil z předchozího příkladu

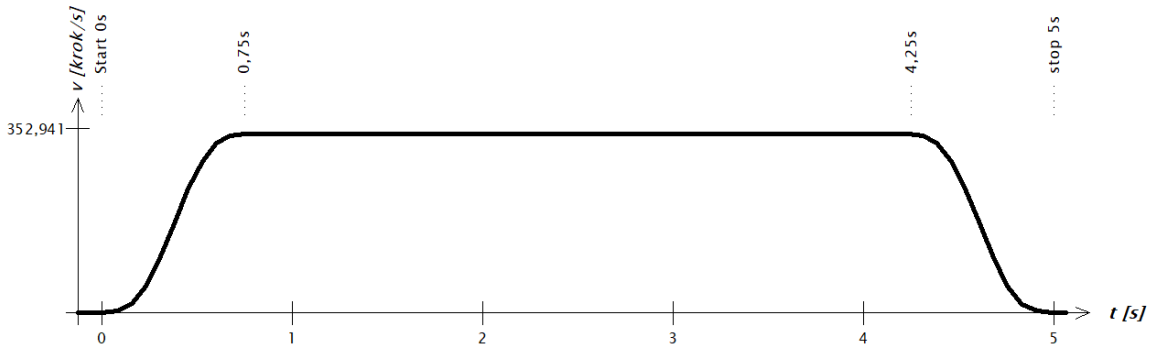
2.2.5 Trapezoidní řízení dynamiky pohybu s „S“ křivkou

I když to není z obr. 13 přímo patrné, trapezoidní profil má nevýhodu v tom, že na začátku a konci akcelerace i decelerace vznikají záškuby motoru způsobené náhlými změnami akcelerace a decelerace. Pro jejich odstranění je třeba „otupit“ přechody pro každou změnu rychlosti, tedy ve čtyřech místech našeho grafu. Toto „otupení“ lze zajistit tzv. S-křivkou. Na dostupných veřejných zdrojích je popsána řada matematických metod, jak toho dosáhnout.

Pro náš účel jsem jako nejjednodušší postup zvolil přepočítat námi vypočítanou lineární akceleraci funkcí sinus podle vzorce, jenž vznikl jako výtažek ze všech získaných informací:

$$v_t = \left(\left(\frac{t}{t_{acc}} \cdot 2\pi\right) - \sin\left(\frac{t}{t_{acc}} \cdot 2\pi\right)\right) \cdot \frac{v_{max}}{2\pi}, \tag{13}$$

kde v_t je rychlost v čase t, t_{acc} je celkový čas akcelerace a v_{max} je cílová rychlost po skončení akcelerace, vypočítaná podle vzorců pro trapezoidní profil. Nový tvar křivky je znázorněn na obrázku 14.



Obrázek 14: Profil z obr. 13 přepočítaný vzorcem (13) – S křivka

2.3 Teoretická implementace řízení krokového motoru do mikrokontroléru

Pokud chceme implementovat výše uvedené znalosti do mikrokontroléru, a to zejména profil s S-křivkou, musíme vzít na vědomí určitá specifika, např. omezenou velikost RAM paměti, rychlost vykonání instrukcí a nutnost maximálního využití vestavěných periferií a přerušení. Většina komerčních aplikací využívá paměť RAM k uložení předem kompletně vypočítaného profilu. Poté už jen mikrokontrolér vykoná požadovaný profil. Problém je v tom, že při velmi dlouhých pohybech nemusí kapacita RAM u levného mikrokontroléru stačit. Toto se dá řešit počítáním parametrů pro každý krok v reálném čase. Zde však narazíme na matematické operace, které zaberou mnoho strojových cyklů mikrokontroléru. To není rozhodně ideální pro vykonávání v přerušení. Na internetu je k nalezení několik metod, jak toto eliminovat různými aproximacemi a podobně, a to s cílem zbavit se náročných výpočtů.

Aby nebyla tato práce pouhým přepsáním známých faktů, pokusil jsem se o jiný přístup. Celý průběh rozdělím na pevně určený počet bodů. Mezi každými dvěma body uvažuji konstantní rychlost a změnu rychlosti počítám pouze pro určené body.

Princip řízení bude tedy následující:

a) Po zadání parametrů se vypočítá trapezoidní profil viz kapitola 2.2.4, respektive jeho stěžejní parametry, jimiž budou maximální rychlost a doba konstantní rychlosti. Časy akcelerace a decelerace, dráha a celková doba pohybu již budou známy při zadání parametrů.
b) Z časů a rychlostí spočítáme, kolik bude která fáze profilu potřebovat vykonat kroků.
c) Počet vypočítaných kroků jednotlivých fází profilu zaokrouhlíme na celá čísla.
d) Sečteme vypočítané kroky a porovnáme je s požadovanou hodnotou. Pokud nějaké kroky chybí, přičteme je do fáze pohybu konstantní (nejvyšší) rychlostí. Zde totiž bude vliv na celkový čas pohybu minimální. Pokud budou vypočítané kroky přebývat, kroky podle předchozího postupu odečteme.
e) V první fázi si čas akcelerace rozdělíme na 10 stejných částí. Pro každou část si spočítáme cílovou periodu rychlosti podle vzorce:

$$T_{act} = t_{act} \cdot \frac{v_{max}}{t_{acc/dec}}, \tag{14}$$

kde T_{act} je aktuální perioda v bodě, t_{act} je čas v počítaném bodě, v_{max} je maximální rychlost a $t_{acc/dec}$ je spočítaný čas akcelerační/decelerační rampy.

f) Spočítáme si rozdíl mezi T_{act} a předchozí periodou T_{act-1}. Tím nám vznikne časový úsek s počáteční periodou T_{act-1} a koncovou periodou T_{act}. Z těchto dvou period si spočítáme průměrnou periodu v úseku T_{avg} pomocí vzorce:

$$T_{avg} = \frac{T_{act} + Tv_{act-1}}{2} \tag{15}$$

g) Z této průměrné periody si spočítáme počet kroků, jenž se vejde do aktuálního úseku podle vzorce pro výpočet periody v obdélníkovém profilu (7). Pokud je výsledný počet kroků menší než 1, spočítáme si následující úsek a výsledný počet kroků přičteme k předchozímu. Takto to opakujeme, dokud není počet kroků alespoň 1.
h) Poté zase spočítáme následující časové úseky, dokud nedosáhneme bodu T_{max}.

i) Tyto body si uložíme do RAM. Vypočtené periody a počty kroků si ihned uložíme do decelerační tabulky, ale v opačném pořadí.
j) Vznikne nám 20 bodů, se kterými se bude dále pracovat. Zároveň budeme vědět, kolik bude třeba vykonat kroků mezi body 0–10 (akcelerace), 10–11 (konstantní rychlost) a 11–20 (decelerace). Trik bude spočívat v tom, že budeme počítat mezi každými dvěma body konstantní rychlost, což lze už provést rychle v reálném čase. Křivka sice nebude úplně plynulá, ale při rozdělení na 10 částí by už mohly zmizet záškuby motoru. V případě neuspokojivé plynulosti pohybu při akceleraci bude možno rozdělit pohybový profil na více částí.

Výhodou tohoto řízení by mělo být to, že ať bude pohyb jakkoli dlouhý, tak se bude vždy skládat jen z dvaceti parametrů (případně více při jemnějším rozdělení). Takže nebude problém s kapacitou RAM. Podrobněji bude toto popsáno v kapitole 3.5. Pokud se bude operovat v uzavřené smyčce, bude se navíc kontrolovat každý x-tý krok, zda se hřídel motoru nachází ve správné poloze. Kontrolní poloha by se měla nacházet v oblasti celokroku, kvůli větší jistotě předpokládané polohy.

2.4 Průmyslové datové sběrnice

S rozvojem technologií a potřebou přenášet čím dál více dat vznikl problém s ohromným počtem vodičů ve strojních zařízeních. Toto bylo třeba nějak redukovat, a proto vznikly datové sběrnice. Pokud přeskočíme paralelní a nerozšířené sběrnice, tak to bylo sériové rozhraní RS485, které dokázalo obsluhovat až 32 zařízení na dvou vodičích a to vše až na vzdálenost 1200 m při rychlosti až 10 Mb/s. Poté přišla na řadu různá moderní síťová řešení využívající protokol Ethernet. Jako příklad lze uvést ETHERNET/IP, PROFINET, ETHERCAT a další. Těm se ale tato práce nebude věnovat.

RS485 považujeme za fyzickou vrstvu, která je dána napěťovými úrovněmi vodičů, standardizovanými přenosovými rychlostmi a umožňuje pracovat s různými protokoly. Někdy se jí také říká proudová sběrnice, protože na delší vzdálenosti se musí ukončit tzv. terminátory o hodnotě cca. 120 Ω. Na základech RS485 se dá provozovat protokol MODBUS RTU, MODBUS ASCII, DEVICE NET, PROFIBUS, CANOPEN a další. I když už jsou tyto sběrnice vytlačovány moderními ethernetovými řešeními, své pevné místo v průmyslu ještě nějakou dobu mít budou kvůli své snadné

implementovatelnosti, robustnosti a odolnosti v silně elektromagneticky zarušeném prostředí [4].

2.4.1 MODBUS

MODBUS je komunikační protokol, který je založen na architektuře master-slave. K přenosu dat využívá různých rozhraní jako RS485, RS422, RS232 a také TCP/IP sítě Ethernet [5]. Byl vyvinut firmou Schneider už v roce 1979 a od té doby se rozšířil v mnoha odvětvích. Jedním z důvodů širokého rozšíření bylo to, že protokol je otevřený a za jeho používání se nemusí platit [6]. Pro tuto práci bylo zvoleno rozhraní RS485, takže se zaměříme na protokol MODBUS RTU.

Zpráva protokolu MODBUS RTU se podle tab. 3 skládá z adresy zařízení Slave ID, kódu funkce, speciálních dat v závislosti na kódu funkce a kontrolního součtu CRC.

Tabulka 3: Struktura zprávy MODBUS RTU

Slave ID	**Kód funkce**	**Speciální data**	**CRC**
1 byte	**1 byte**	**2 − x byte**	**2 byte**

Kód funkce a speciální data tvoří PDU (Protocol Data Unit) neboli datovou jednotku. Data v modulu jsou uložena ve 4 tabulkách, z nichž dvě jsou pouze pro čtení (r) a dvě umožňují čtení i zápis (r/w). V každé tabulce je umístěno 9999 registrů.

Tabulka 4: Popis registrů MODBUS [5]

Číslo registru	**HEX adresa registru**	**Typ**	**Název**	**Délka dat**
1–9999	0000-270E	r/w	Cívky	1 bit
10001–19999	0000-270E	r	Diskrétní vstupy	1 bit
30001–39999	0000-270E	r	Vstupní registry	16 bit
40001–49999	0000-270E	r/w	Zádržné registry	16 bit

Zpráva protokolu MODBUS používá adresu registru. Například první zádržný registr má číslo 40001, ale jeho HEX adresa 0000. Rozdílu mezi těmito hodnotami se říká offset. Z tab. 4 vyplývá, že offset pro jednotlivé tabulky je 1, 10001, 30001 a 40001. Pro lepší pochopení uvedeme příklad:

Ze zařízení s adresou 17 chceme vyčíst hodnoty zádržných registrů 40108 až 40110. HEX zpráva na datové lince bude 11 03 006B 0003 7687, kde:

11 je adresa zařízení (11 HEX = 17 DEC)

03 je kód funkce (viz tab. 5)

006B je adresa prvního čteného registru (40108 – 40001 = 107 DEC = 6B HEX)

0003 je počet požadovaných registrů (3 HEX = 3 DEC) **7678** je CRC

Adresované zařízení nám odpoví například **11 03 06 AE41 5652 4340 49AD**:

11 je adresa zařízení (11 HEX = 17 DEC)

03 je kód funkce (viz tab. 5)

06 je počet následujících bytů (06 HEX = 6 DEC)

AE41 je hodnota registru 40108 (AE41 HEX = 44609 DEC)

5652 je hodnota registru 40109 (5652 HEX = 22098 DEC)

4340 je hodnota registru 40110 (4340 HEX = 17216 DEC)

49AD je CRC

Z výše uvedeného příkladu je vidět, že adresované zařízení vždy odpoví svou adresou a kódem funkce, na kterou odpovídá. Tuto sekvenci následuje údaj o počtu následujících bytů, jenž slouží k tomu, aby se mohl detekovat konec zprávy a tím pádem i kontrolní součet CRC (Cyclical Redundancy Check) [5]. Výpočtu CRC se bude věnovat kapitola 3.4.

Tabulka 5: Funkce protokolu MODBUS [5]

Kód funkce	**Co funkce dělá**	**Typ dat**	**Přístup**
01 (01 HEX)	Čte stav cívky	1 bit	r
02 (02 HEX)	Čte stav vstupu	1 bit	r
03 (03 HEX)	Čte zádržné registry	16 bit	r
04 (04 HEX)	Čte vstupní registry	16 bit	r
05 (05 HEX)	Nastavuje jednu cívku	1 bit	w
06 (06 HEX)	Nastavuje jeden registr	16 bit	w
15 (0F HEX)	Nastavuje více cívek	1 bit	w
16 (10 HEX)	Nastavuje více registrů	16 bit	w

2.5 Mikrokontrolér Microchip PIC32MX

Pro tuto práci byl vybrán 32bit mikrokontrolér rodiny PIC32MX od firmy Microchip. Vzhledem k jeho nízké ceně nemělo smysl přemýšlet o 16bit nebo 8bit alternativách. Tento mikrokontrolér lze taktovat až na 80 MHz a nabízí kromě 256 kB flash paměti a 32 kB RAM také mnoho HW periferií. To je v dnešní době komunikace velmi důležité. Konkrétní použitý typ 32MX440F256H má například USB 2.0 port, dva I^2C moduly, dva UART moduly podporující RS232, RS485, LIN a IrDA HW dekódování a nakonec dva SPI moduly. Z ostatních periferií lze vyzdvihnout až pět PWM komparátorů, pět externích zdrojů přerušení, až šestnáct analogových vstupů s vzorkovací frekvencí až 1 kHz. Všechny výše zmíněné periferie v praxi znamenají, že díky jimi generovaným přerušením se o ně programátor nemusí víceméně starat a zbytečně rozšiřovat kód programu. To je velká výhoda, která se projeví v rychlejším vývoji cíleného zařízení [7].

K programování mikrokontrolérů PIC se používá volně dostupné prostředí MPLAB X IDE od firmy Microchip. K tomuto programu je nutno mít ještě volně dostupný compiler XC32 a také vhodný programátor pro mikrokontroléry PIC. V našem případě byl použit programátor PICkit3. Prostředí MPLAB X podporuje jak jazyk C, tak jazyk C++ [7].

Mikrokontroléry rodiny PIC32 lze programovat buď klasickým programováním s nastavováním registrů mikrokontroléru a s uživatelským strukturováním programu, nebo lze využít plugin do MPLAB X, který se jmenuje MPLAB Harmony. Tento plugin obsahuje spousty již vytvořených funkcí, které programátorovi usnadní například práci s komunikačními periferiemi, SD kartami, displeji atd. Zároveň tento plugin přináší novou filozofii programování, která je založená na takzvaných úkolech (anglicky task). Při tomto přístupu k programování lze kontrolovat několik úkolů zdánlivě najednou (multitasking) tak, že jsou na sobě časově nezávislé. Na tyto úkoly lze pohlížet jako na nezávislá programová vlákna. Například se nemůže stát, že by program uvízl v nějaké nekonečné smyčce. Může se stát, že se zacyklí jedno vlákno, ale zbytek programu může pokračovat dál a tím pádem umožnit programátorovi kontrolovat nečekané situace [7]. Výhodou je, že plugin MPLAB Harmony lze bez problémů kombinovat s klasickým způsobem programování. Ne vždy totiž nabízená již hotová řešení odpovídají požadavkům programátora. Samotným programováním se bude zabývat kapitola 3.4.

3 Praktická část

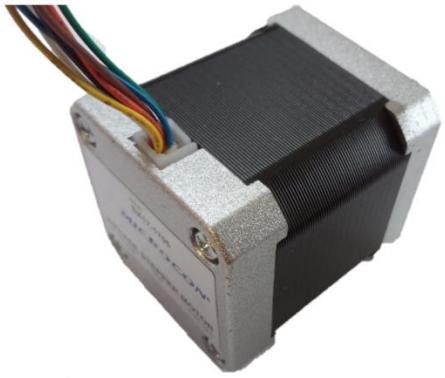
Tato část práce se zabývá reálným zařízením, vytvořeným na základě poznatků uvedených v teoretické části.

3.1 Použité komponenty

Vzhledem k samofinancování projektu a také k tomu, že tato práce má být jakýmsi motivátorem pro studenty, jsem se snažil o co nejrozumnější cenu jednotlivých komponent. Protože jsem vášnivý elektro-kutil, měl jsem k dispozici spoustu použitých součástek z různých vyhozených elektronických zařízení.

3.1.1 Krokový motor

Jako pohon jsem si vybral motor velikosti NEMA17 firmy MICROCON, typ SX17-1005LQCEF. Princip krokového motoru je již popsán v kapitole 2.1. Tento dvoufázový motor je použit v bipolárním zapojení. I když dle mých praktických zkušeností z průmyslu se nejedná o žádný high-tech výrobek, byla to z finančních důvodů dobrá volba.

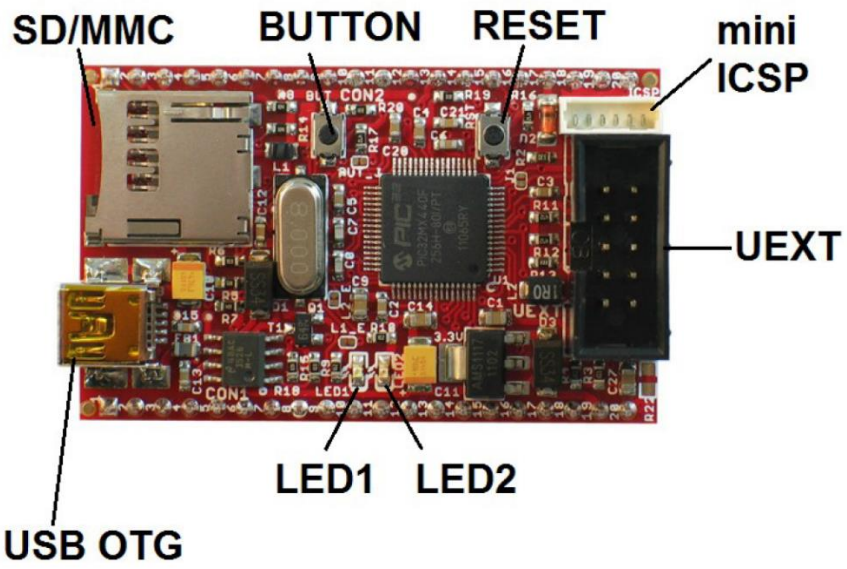


Obrázek 15: Krokový motor MICROCON SX17-1005LQCEF

3.1.2 Vývojová deska OLIMEX PINGUINO-MICRO

Tato vývojová deska osazená MCU PIC32MX440 je jednou z mnoha platforem, určených pro amatérský vývoj různých elektronických a IOT zařízení. V MCU na desce je již z výroby nahrán bootloader, který umožňuje download z vývojového prostředí Pinguino IDE. Toto prostředí je velice podobné vývojovému prostředí Arduino, což z něj činí dobrý nástroj pro začátečníky s tím, že zde je k dispozici výkonný 32bitový

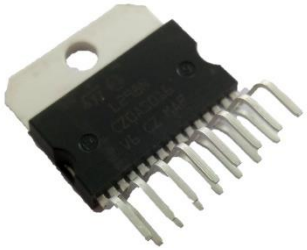
procesor oproti 8bitovému v deskách Arduina [9]. Nicméně v této aplikaci jsem bootloader smazal a desku jsem použil pouze kvůli hotovému osazení MCU PIC32. Tu jsem poté programoval z prostředí MPLAB-X přes rozhraní ICSP.



Obrázek 16: Vývojová deska OLIMEX PINGUINO-MICRO [9]

3.1.3 Dvojitý H můstek L298

Široce využívaný H-můstek L298 jsem zvolil jako osvědčenou cestu k výkonovému ovládání krokového bipolárního motoru. Svým širokým rozsahem napájecího napětí od 4,5 do 36 VDC a maximálním proudem 2 A je vyhovující pro použitý krokový motor [10].



Obrázek 17: H-můstek L298 v pouzdře Multiwatt 15

3.1.4 Převodník napěťových úrovní CD40109

I když to nebylo v původním plánu, musel jsem použít převodník napěťových úrovní ze 3,3 V na 5 V. I když je PIC32MX tolerantní k 5 V [7] na digitálních vstupech, tak L298 bohužel nedokáže pracovat se signálem 3,3 V [10].

Obrázek 18: Převodník CD40109

3.1.5 Optozávora

Optozávora ve formě, jaká byla použita v této práci, se skládá pouze z aktivního prvku (IR LED diody) a spínacího prvku (fototranzistor). Tyto dva prvky jsou nasměrovány proti sobě, kdy LED dioda vytváří jakýsi světelný most, jímž osvětluje fototranzistor. Ten je při osvětleném stavu vodivý čili může jím procházet proud signálového vedení. Pokud se vloží mezi LED diodu a fototranzistor překážka, tranzistor přestane vést elektrický proud.

Z výše uvedeného vyplývá, že jsme schopni detekovat překážku. Toho využijeme při detekování polohy. Nevýhodou tohoto typu snímače je, že nemá žádný tvarovací obvod, a proto může být ovlivněn rušivým denním světlem. Výhodou je naopak cena, která se rovná půlce kostky másla.

Obrázek 19: Optozávora TCST2103 [11]

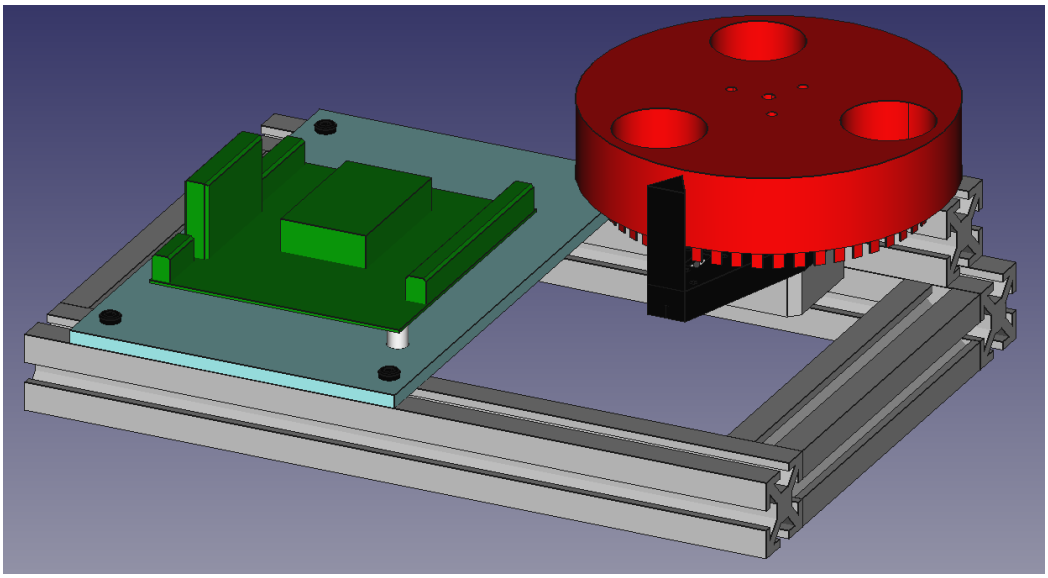
3.1.6 Převodník RS485 <-> TTL ST1480AC

Protože linka RS485 v half-duplexním zapojení pracuje na principu vyhodnocování napěťových rozdílů mezi dvěma vodiči a na MCU jsou pro sériovou komunikaci vyhrazeny dva sériové vývody Rx a Tx na úrovni TTL, musí se tyto dva nesourodé principy nějak přizpůsobit. K tomu slouží právě integrovaný obvod firmy ST 1480AC, který ke své funkci nepotřebuje žádné další součástky. Jediné, co je třeba hlídat z MCU, je směr toku dat. Díky tomu je kromě Rx a Tx vodičů nutno zapojit ještě třetí vodič, který určuje, jestli se data přijímají, nebo vysílají.

Obrázek 20: Převodník RS485<->TTL ST1480AC

3.2 Mechanická konstrukce výukové sestavy

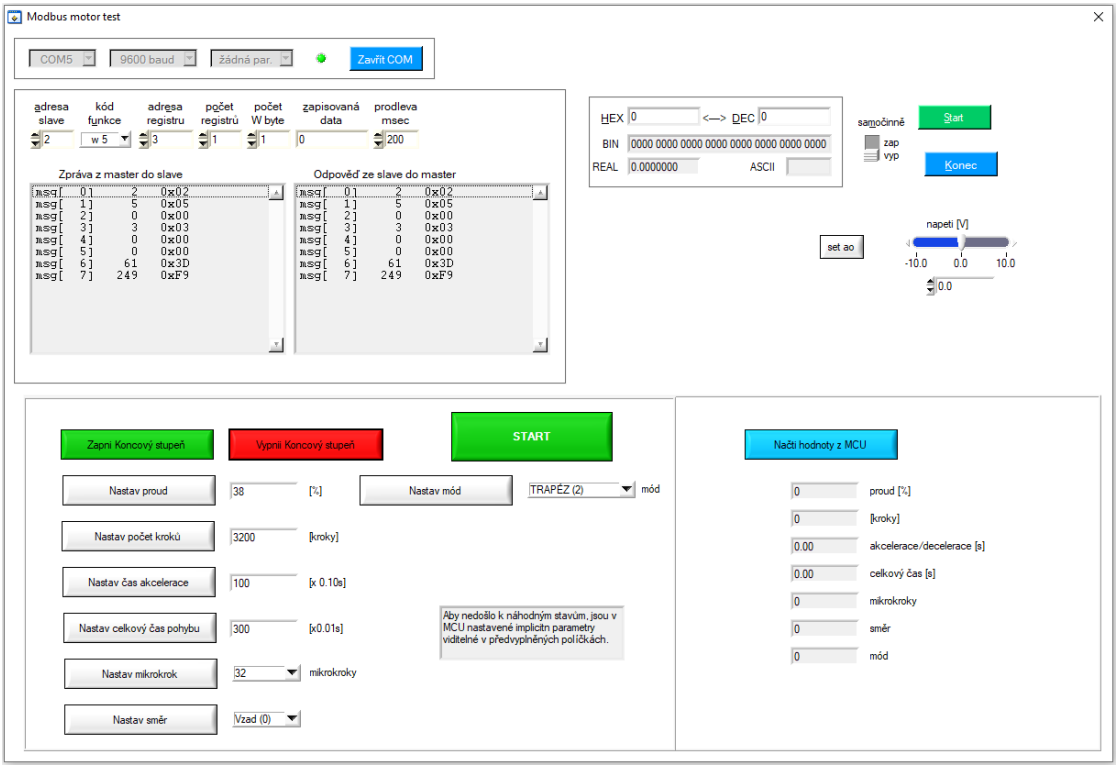
Dle původního záměru a požadavků vedoucího práce neměla být sestava větší než velikost papíru A4 a měla obsahovat lineární pohon s krokovým motorem, ovládací modul s Arduinem a display. Dalším z požadavků byla celková bezpečnost, zejména pohonu. To se mi podařilo s pomocí kolegů konstruktérů vyřešit tak, že by byla šroubovice lineárního pohonu na obou koncích bez závitu a vozík pohonu by měl na obou stranách odpružení. Čili při poruše programu nebo koncových čidel by na konci šroubovice vyjel vozík ze závitu a pouze by se opíral o pružinu. To by bylo aplikováno na obou koncích. Myslím, že by to bylo funkční z hlediska bezpečnosti, nikoli z hlediska schopnosti demonstrovat dynamické vlastnosti různých pohybových profilů. Po testech pohonu s vrtačkou jsem si uvědomil, že díky nízkému stoupání závitu šroubovice M12 by byl pohon pomalý a nebyl by znatelný rozdíl mezi jednotlivými pohybovými profily. Na jinou šroubovici s jiným stoupáním bych finančně nedosáhl, nehledě na to, že průmyslově používané šroubovice mají kuličková ložiska a nelze s nimi vyjet ze závitu. Proto jsem po diskusi s vedoucím práce navrhl celkovou změnu koncepce pohonu z lineárního na rotační. Pohon se skládá z motoru upevněného na nosníku, kde jsou čidla pro enkodér. Na hřídeli motoru je upevněn kotouč ozubením pro enkodér a také díry velikosti malé skleničky (tzv. malý panák, 0,02 l), kam se může umístit zmíněná nádoba s kapalinou, např. s vodou. Na hladině vody je totiž nejlépe vidět dynamické změny pohybu.



Obrázek 21: Návrh výukové sestavy po změně koncepce (kresleno v programu FreeCAD)

Jako druhý problém se ukázalo být samotné řídicí Arduino. Při přemýšlení o původním návrhu jsem si představil, jak se asi bude s takovýmto zařízením pracovat. Došlo mi, že student, který vymyslí novou funkci a naprogramuje ji do PIC32MX, bude muset následně programovat Arduino, které mu přes MODBUS umožní otestovat zmíněnou funkci. Tento způsob jsem považoval za poněkud složitý a demotivující.

Proto jsem se rozhodl pro změnu v podobě nahrazení Arduina vlastní nově vytvořenou PC aplikací. Ta by měla zajistit flexibilitu a rychlé testování příkazů na MODBUS pouhým zapsáním příkazu do políčka v PC aplikaci. Tuto aplikaci jsem napsal v prostředí LabWindows/CVI a funguje s USB/RS485 převodníkem. Jako základ jsem použil aplikaci na testování komunikace se zařízeními na MODBUS RTU. Tuto aplikaci jsme napsali s kolegy v zaměstnání a byla terčem neustálého upravování dle našich požadavků. Výsledkem je aplikace schopná komunikace na MODBUS RTU jako master. Jelikož byla původní aplikace uživatelsky velice nepříjemná, upravil jsem její grafické rozhraní GUI (Graphics User Interface). Přidal jsem tlačítka a pole určená pro snadné ovládání této konkrétní výukové sestavy. Na základní funkce pohonu tedy není potřeba složitě vyplňovat datová pole a jednotlivě odesílat. Pod novými tlačítky jsou skryté příkazy, nebo sekvence příkazů, aby se nemusel uživatel o nic starat a aby nedělal zbytečné chyby. Stačí pouze stisknout tlačítko požadované funkce. Uživateli je však v rámci testování ponechána možnost zaslat do zařízení jakoukoliv funkci.



Obrázek 22: Ukázka PC aplikace pro posílání příkazů do MCU

3.3 Elektrické zapojení

Při návrhu elektrického zapojení jsem vycházel z datových listů použitých součástek. Zapojení lze rozdělit do těchto hlavních částí:

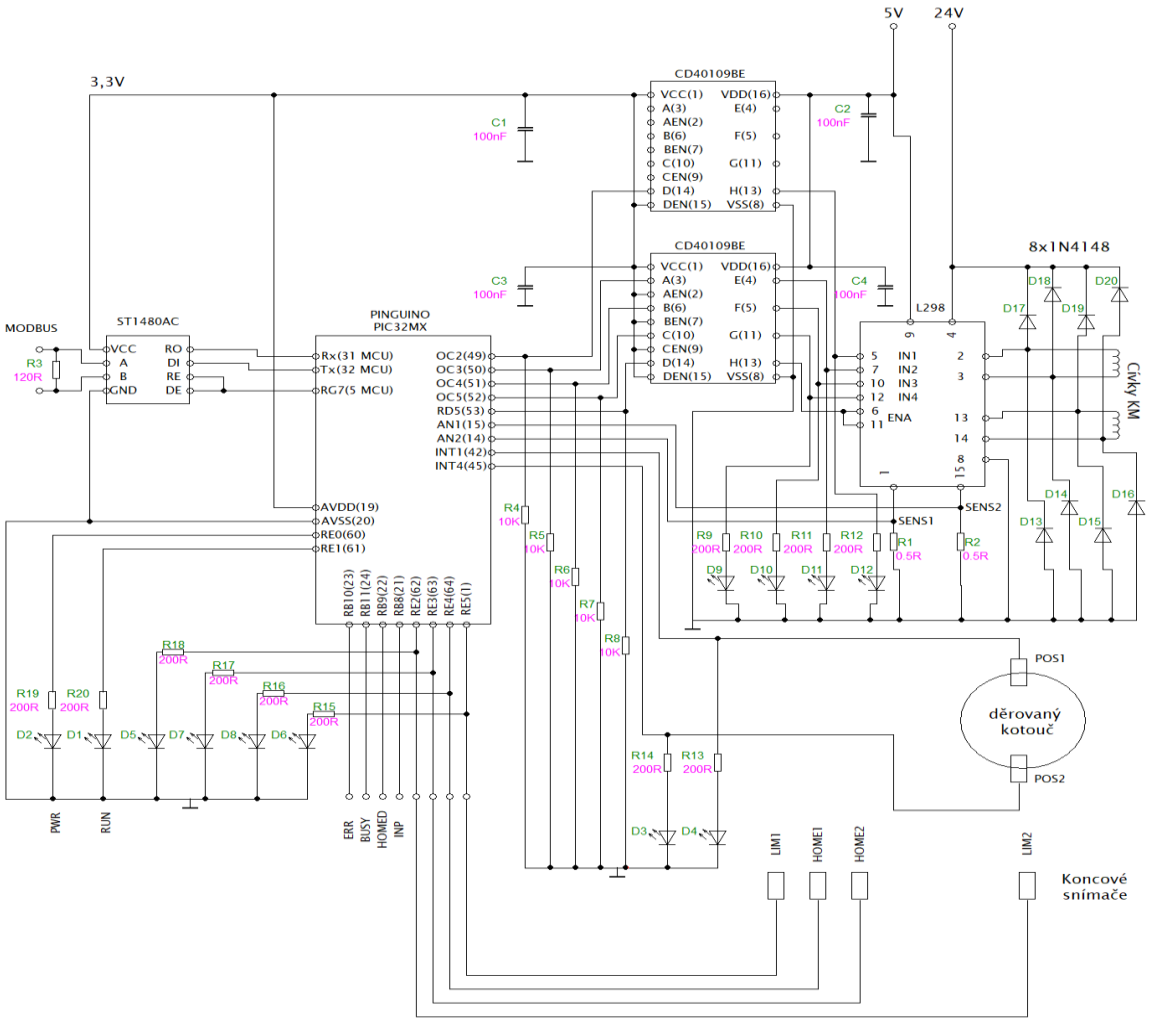
a) Řídicí modul s MCU PIC32MX

b) Převodník RS485<-> TTL

c) Převodník signálových úrovní

d) Výkonový stupeň

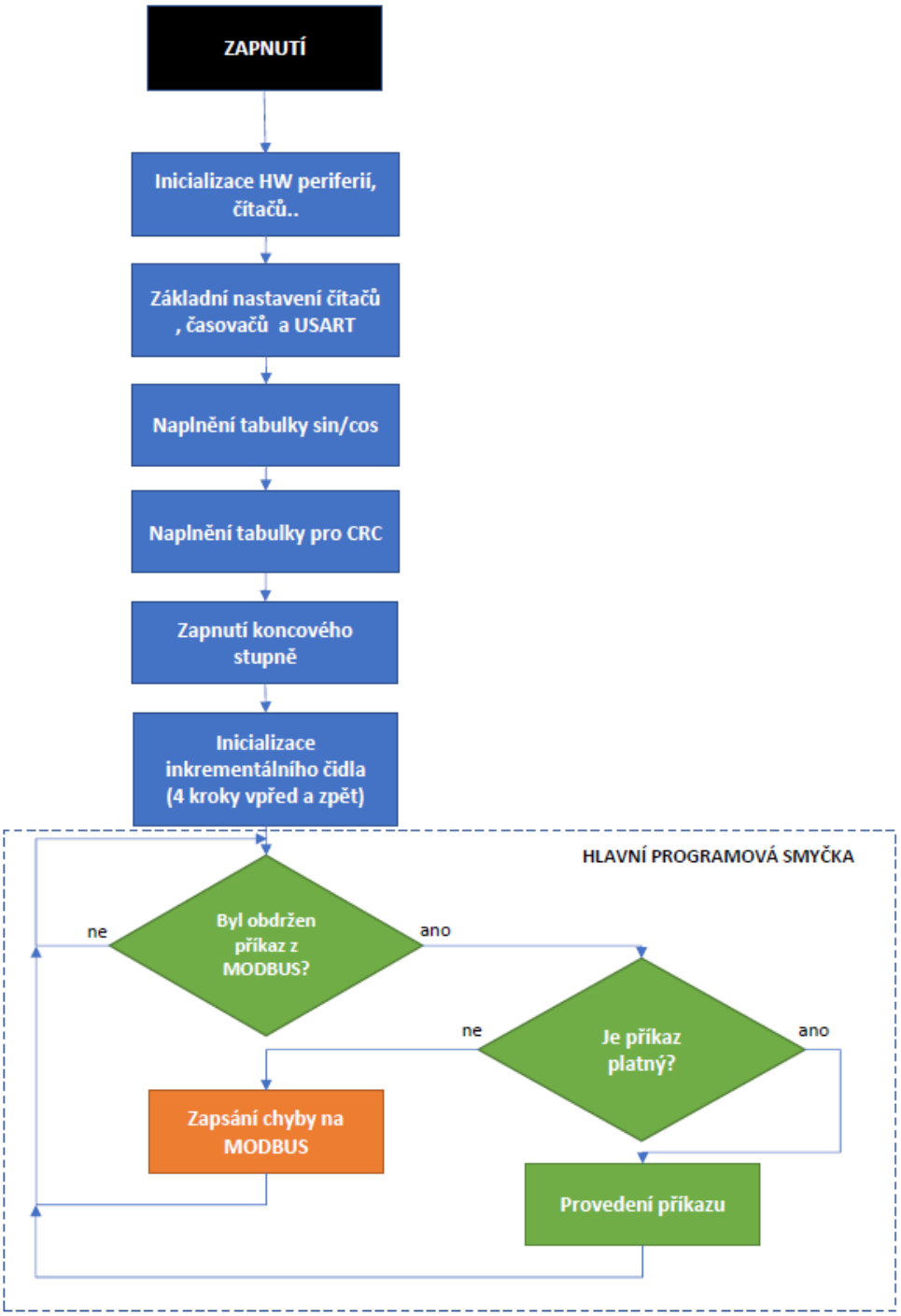
Zdroj napájení je z bezpečnostních důvodů řešen jako externí. Je rozdělen na dvě části: 5 V pro napájení řídicích obvodů a 10–24 V pro napájení výkonové části.



Obrázek 23: Schéma zapojení ovládání krokového motoru

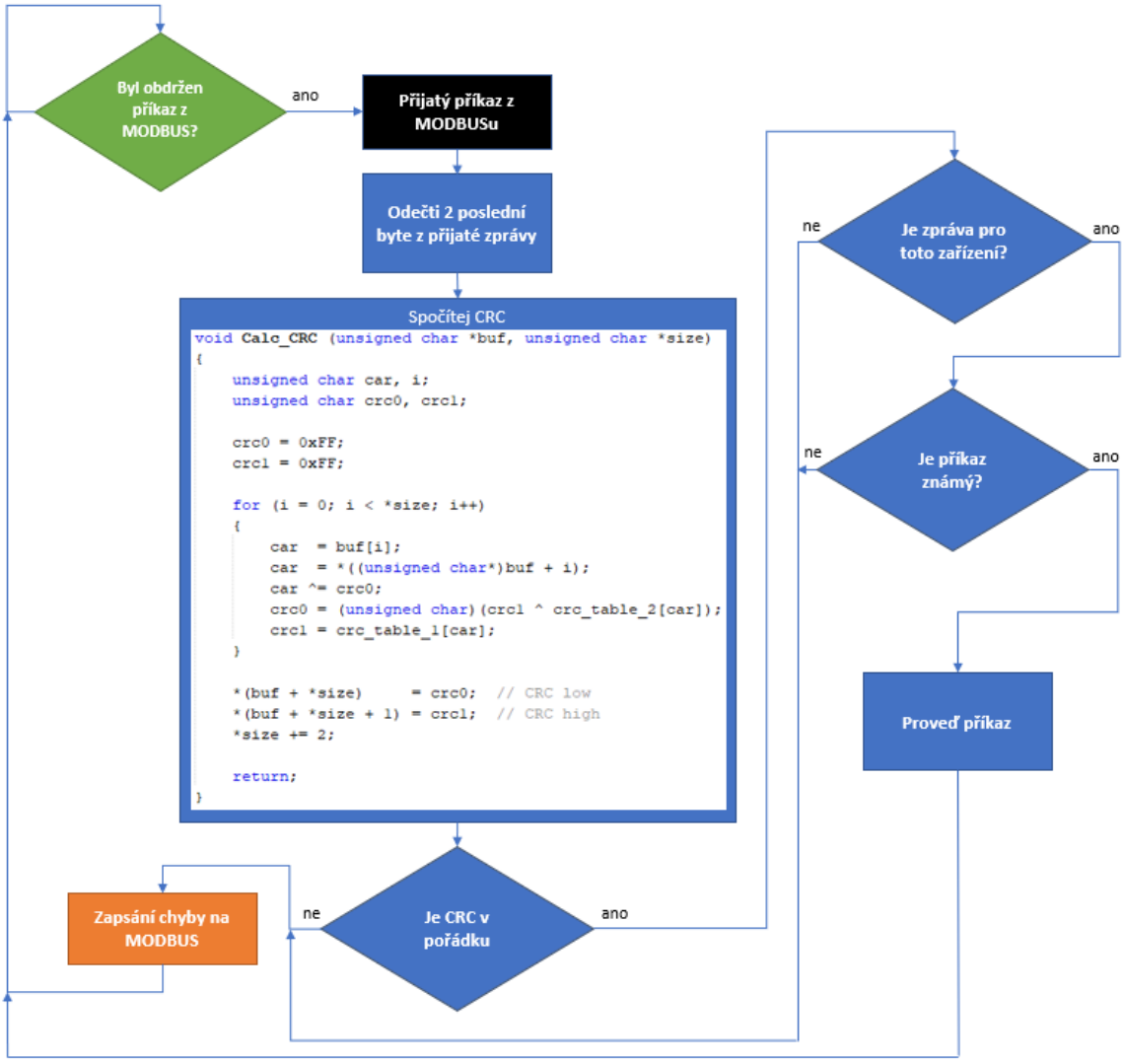
3.4 Software

Celý program je postaven na maximálním využití vestavěných periferií a na řízení událostmi. Mojí snahou bylo, aby hlavní programová smyčka byla velice krátká a tím pádem rychlá. V této smyčce čeká systém na události od externích přerušení pro počítání otáček, přerušení od časovačů pro řízení profilu a také od přerušení modulu USART.



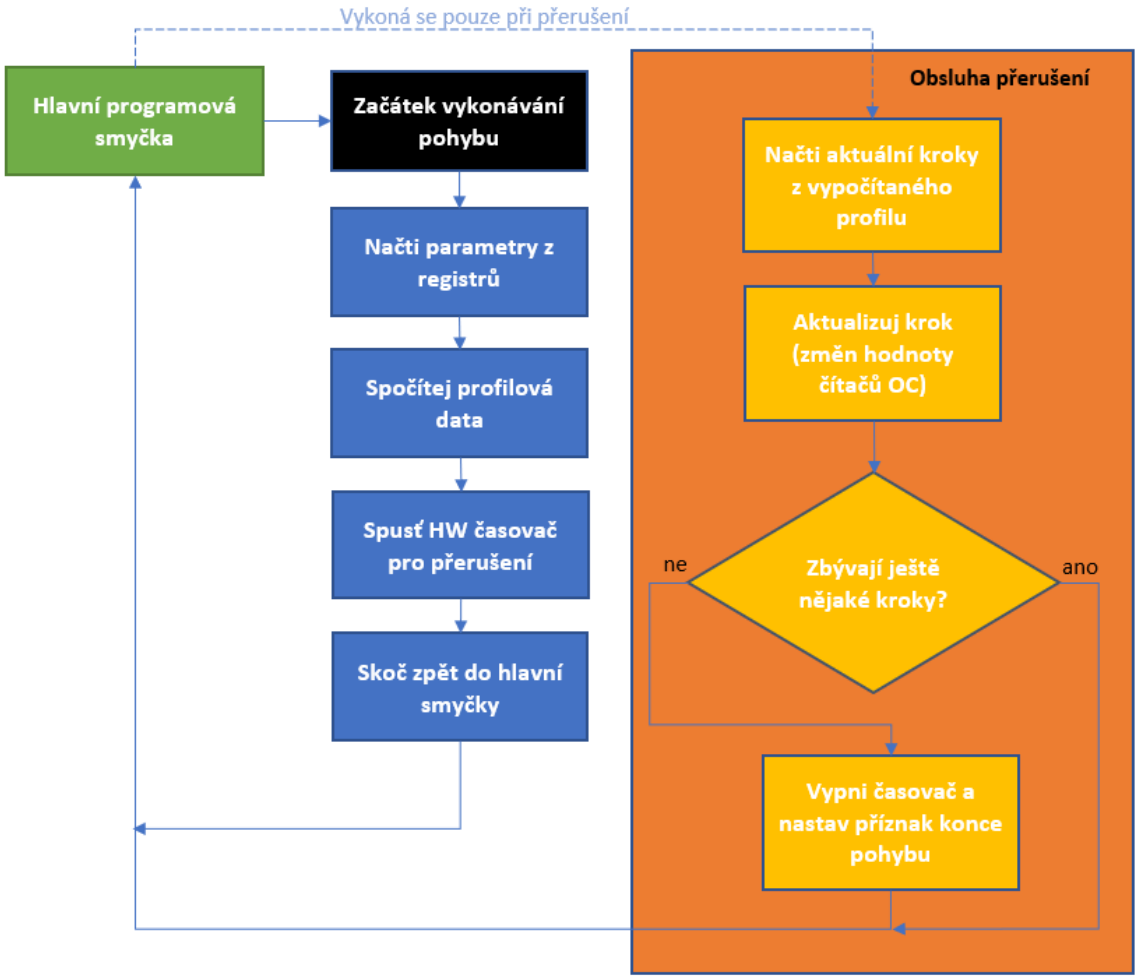
Obrázek 24: Základní algoritmus programu

Princip zpracování zpráv z MODBUSu je opět založen na událostním řízení. Pokud tedy modul USART vyvolá událost, v obsluze přerušení se vyčtou přijatá data z bufferu a nastaví se příznak přijatých dat. Ten je poté zpracován již v hlavní programové smyčce s povolením obsluh dalších událostí. Vše je naznačeno na vývojovém diagramu níže.



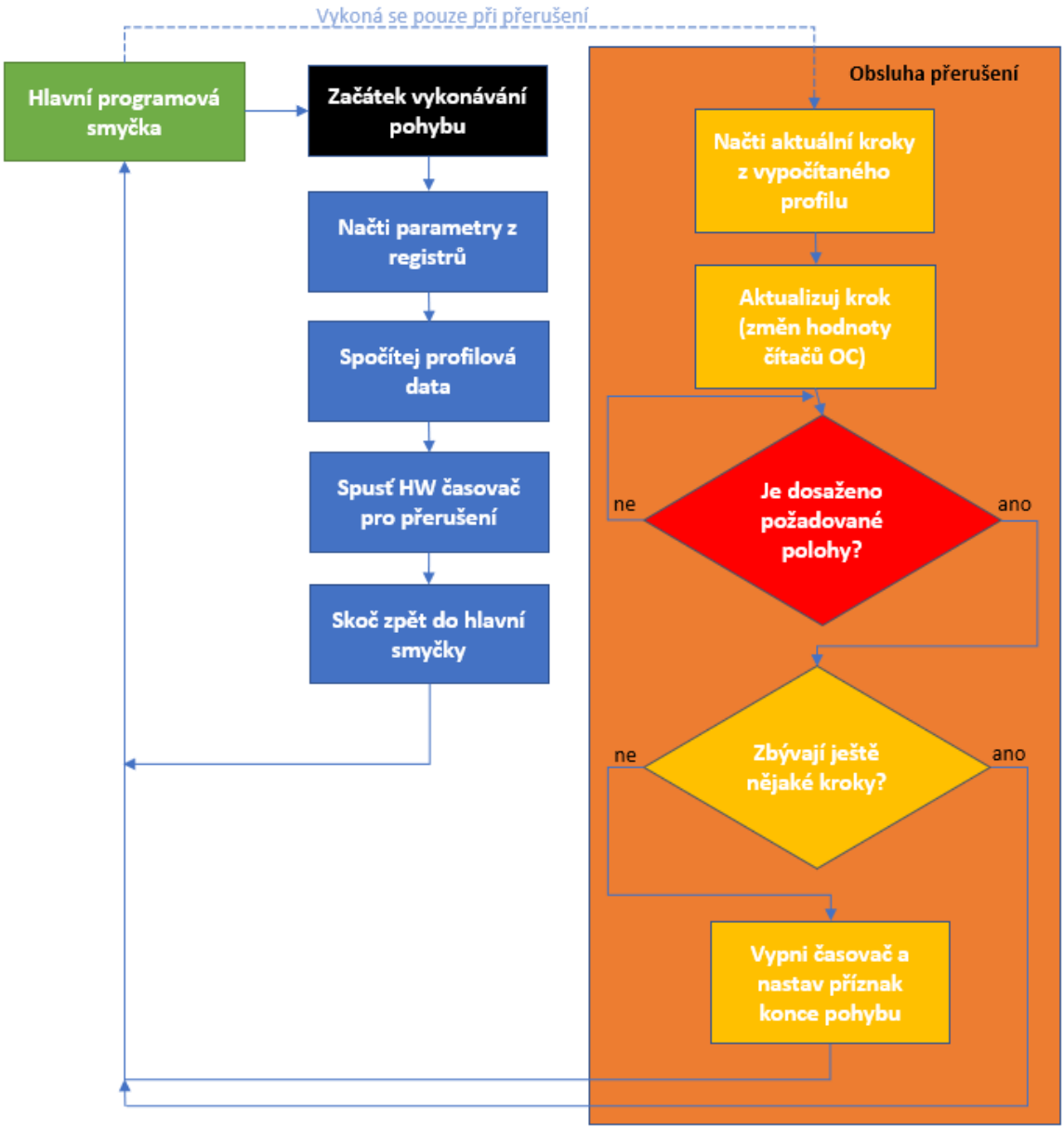
Obrázek 25: Algoritmus zpracování zprávy z MODBUSu

Samotné řízení pohybu se skládá z předem vypočítaného profilu, který se potom vykonává v obsluhách přerušení časovače určeného pro řízení profilu. V těchto obsluhách se nastavuje nový krok, aktualizuje se čítač vykonané dráhy a nastaví se nový čas pro další přerušení. Během čekání na nové přerušení může program obsloužit další události, aniž by to ovlivnilo plynulost pohybu motoru. Níže je naznačen princip ovládání pohybu.



Obrázek 26: Algoritmus vykonání pohybu

Pohyb v uzavřené smyčce se vykonává stejně jako pohyb bez ní. Rozdíl je v tom, že po každém ixtém aktualizování kroku se kontroluje počet programově vykonaných celokroků s počtem skutečně vykonaných celokroků, napočítaných inkrementálním snímačem. Kroky z tohoto snímače jsou opět načítány/odčítány v nezávislé obsluze přerušení. V rutině pro kontrolu polohy se tedy skutečně porovnávají pouze načtená čísla. Ixtým krokem je míněn takový krok, jenž odpovídá násobku prováděného mikrokroku na celý krok. Na obrázku níže je ukázán způsob, jakým se kontroluje skutečná poloha natočení hřídele motoru.



Obrázek 27: Algoritmus vykonání pohybu v uzavřené smyčce

4 Dosažené výsledky

V následující kapitole bude popsáno, jak probíhala stavba celého zařízení, co a proč se muselo změnit a také co se zcela nepodařilo.

4.1 Testování

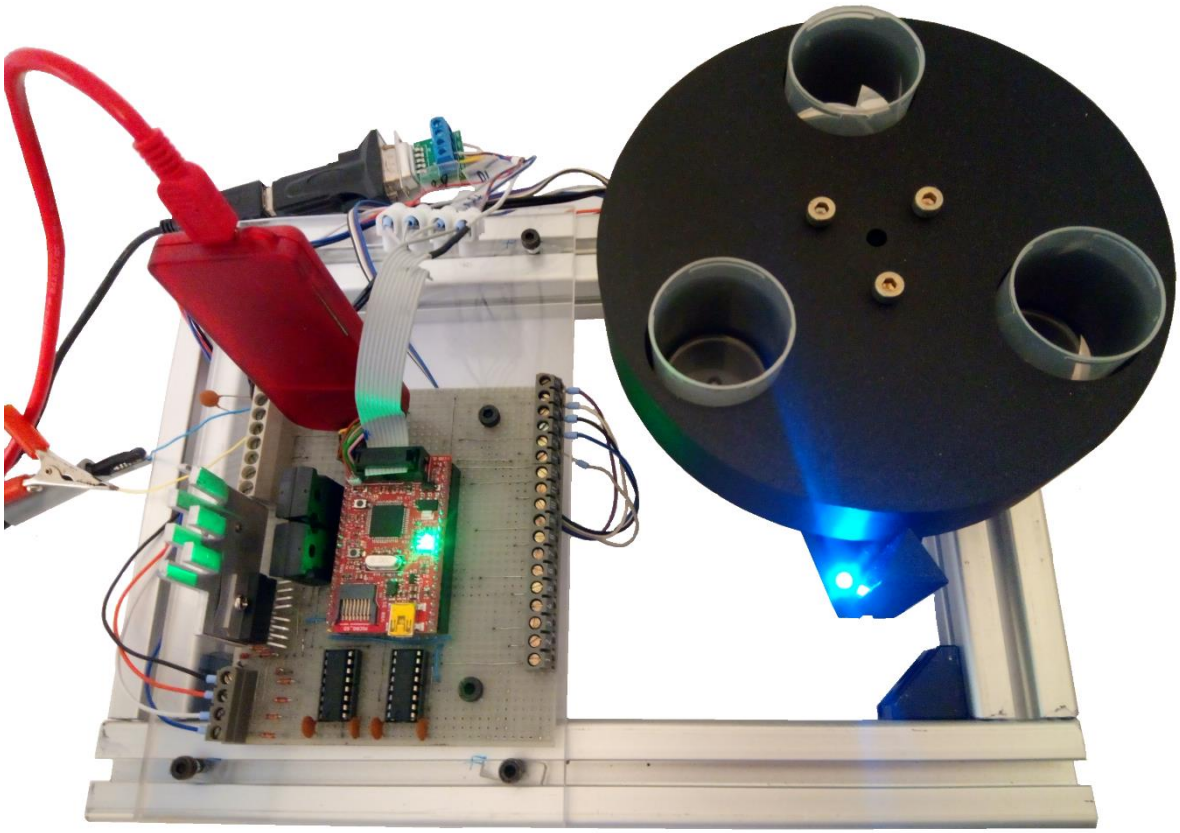
Projekt takového rozsahu se samozřejmě neobešel bez testování. Nejvíce času mně zabralo testování mechanické koncepce pohonu, což jak již bylo zmíněno, vyústilo v kompletní změnu pohonu. Od prvního nápadu vznikly tři návrhy pohonu.

První byl vertikální pohon s řetězem. Na jednom článku byl držák s okem, na který by se pověsilo například závaží. Na tom samém článku byl také jazýček, který by zajížděl do snímačů (optozávor). Pohon by měl koncové snímače a snímač na „domácí“ pozici. Řetěz měl zajistit to, že se pohon bude moci točit neustále dokola v případě nějaké poruchy. Během různých úprav tohoto pohonu jsem zjistil, že při pověšení závaží na oko se řetěz značně prohne a tím pádem se nevejde jazýček pro snímač do prostoru štěrbiny optozávory. Když jsem to chtěl vyřešit větším napnutím řetězu, citelně vzrostl odpor otáčení. To silně a zbytečně zatížilo motor. Tato varianta se tedy stala nepoužitelnou.

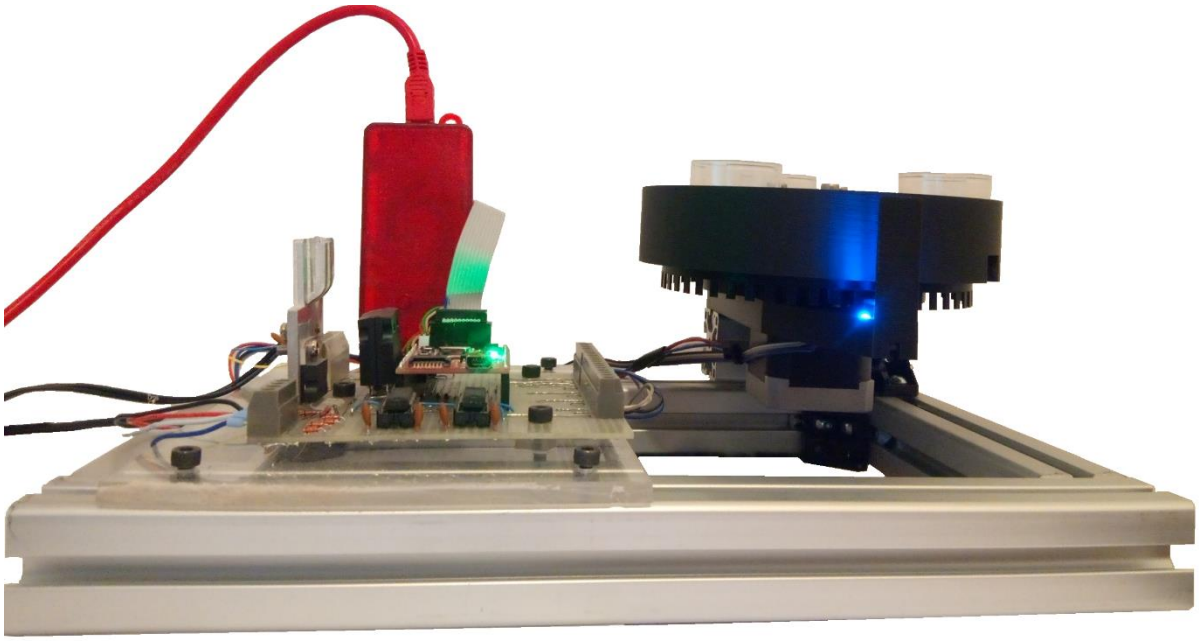
Obrázek 28: Varianta pohonu s řetězovým pohonem

Druhým pokusem o lineární pohon byl pohon se šroubovicí a vozíkem. Díky důvodům zmíněným v kapitole 3.2 jsem tento pohon také nerealizoval. S tímto pohonem jsem se nedostal ani do fáze sestavení. Pouze jsem testoval rychlosti šroubovice na vrtačce a sehnal jsem staré lineární vedení.

Třetím a finálním řešením byl již diskutovaný rotační pohon s prohlubněmi na malé skleničky. Celá sestava je na obrázcích 29 a 30.



Obrázek 29: Sestava rotačního pohonu s řídicí deskou



Obrázek 30: Sestava rotačního pohonu. Pohled z boku

Po vyřešení mechanické části pohonu jsem se věnoval testování funkcí MODBUSu. Zajímala mě použitelnost a chování při nestandardních stavech, např. poslání příkazu při vykonávání pohybu. Více o tom uvádím v kapitole 4.2.

V kapitole 4.3 jsou popsány funkce pohonu. Je tím myšleno chování zařízení při běžném provozu, při vyšších rychlostech a také chování různých rychlostních profilů.

Poslední testování jsem věnoval samotnému hardwaru ve smyslu dimenzování komponent, odolnosti vůči nadproudu a podobně. Vše je popsáno v kapitole 4.4.

4.2 MODBUS

Komunikaci na lince MODBUS jsem u tohoto projektu vystavěl na pěti základních funkcích:

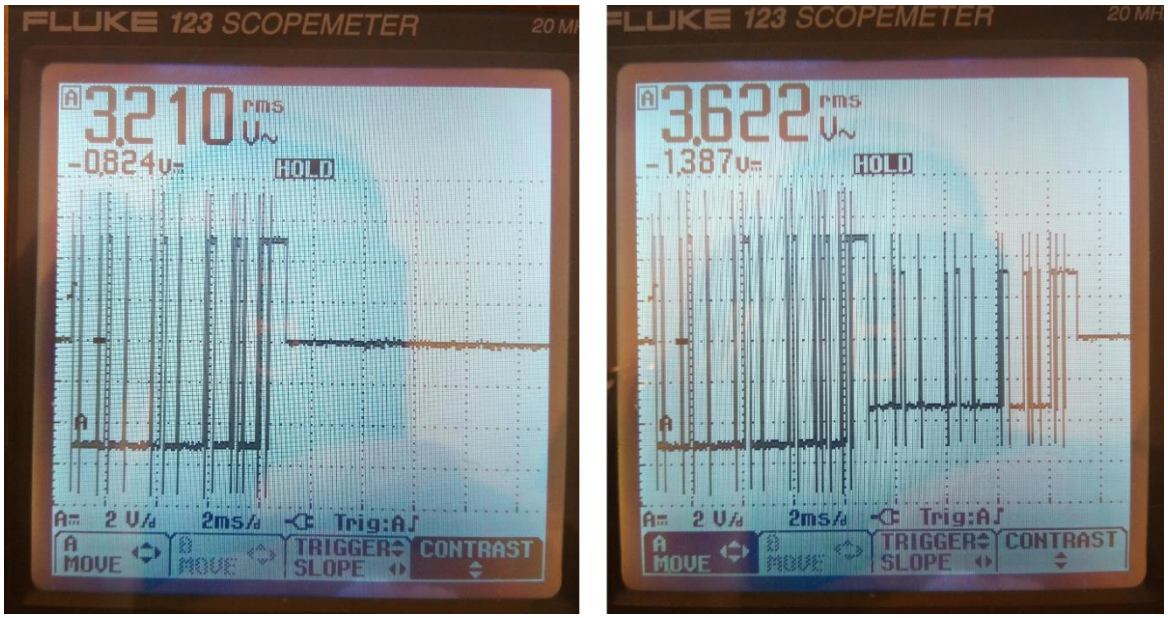
a) Funkce 0x05 – zápis diskrétního výstupu. Tuto funkci například využívám na spuštění pohybu, nebo na určení směru otáčení.
b) Funkce 0x06 – zápis jednoho výstupního registru. Tato funkce zapíše jeden 16bit registr. Pokud potřebujeme zapsat více než 16 bitů, použijeme tutéž funkci na následující adresu, případně adresy. V této aplikaci využívám tuto funkci na zápis všech parametrů potřebných pro spuštění pohybu.
c) Funkce 0x02 – čtení diskrétního výstupu. Touto funkcí lze vyčíst například status pohonu, jako jsou BUSY, ERROR, HOME nebo také požadavek na spuštění pohonu po zapsání funkcí 0x05.
d) Funkce 0x03 – čtení výstupních registrů. Touto funkcí lze vyčíst zapsané hodnoty parametrů pohonu.
e) Funkce 0x04 – čtení vstupních registrů. S touto funkcí se zatím vyčítá pouze aktuální dráha.

Jelikož jsem nikdy předtím nemusel řešit komunikaci MODBUS na úrovni součástek, byl jsem při návrhu zapojení několikrát zaskočen. Od začátku jsem věděl, že budu potřebovat nějaký integrovaný převodník z TTL na RS485, ale nevěděl jsem, co použiji a co bude dostupné. Zadal jsem tedy požadavek do vyhledávače Google a ten mi vrátil několik integrovaných obvodů, deklarovaných jako převodníky pro RS485. První z nich, který byl skladem v ČR, byl obvod ST1480AC.

I když jsem si prohlížel datový list, nevšiml jsem si, že musím řídit také směr toku dat. Na tento logický problém jsem narazil při testování odpovědí z MCU. Naštěstí na vývodu konektoru UEXT desky PINGUINO bylo ještě pár volně konfigurovatelných vstupů/výstupů, a tak jsem mohl využít výstup RG7, který se nastaví na 1, když se zapisuje na MODBUS.

Po vyřešení a otestování hardwarových problémů jsem naprogramoval nejdříve výpočty CRC, poté nejjednodušší funkci 0x05 a poté 0x06. Tím už bylo možno nastavit mód profilu a spustit pohon. Poté už jsem doprogramoval ostatní požadované funkce.

Parametr, který mě velice zajímal, byla doba odezvy na zprávu zaslanou z nadřazeného systému. V našem případě to bylo PC. Na zjištění tohoto času jsem použil dvoukanálový osciloskop FLUKE 123 SCOPEMETER. Měřil jsem časy jak při klidovém stavu, kdy program běží v hlavní programové smyčce a pouze vyčkává na příkaz, tak během spuštěného pohybu. Naměřené hodnoty je nutno brát s velkou rezervou, protože použitý osciloskop nemá funkci lupy. Zpráva i odpověď se tedy musely vejít na jednu obrazovku. I tak je odezva očividně velice rychlá a s rezervou se vejde pod 1 ms. Na obrázku níže je to dobře vidět.



Obrázek 31: Odezva na zprávu MODBUS z PC. Pro názornost je levý obrázek bez odezvy – pouze vyslaná zpráva

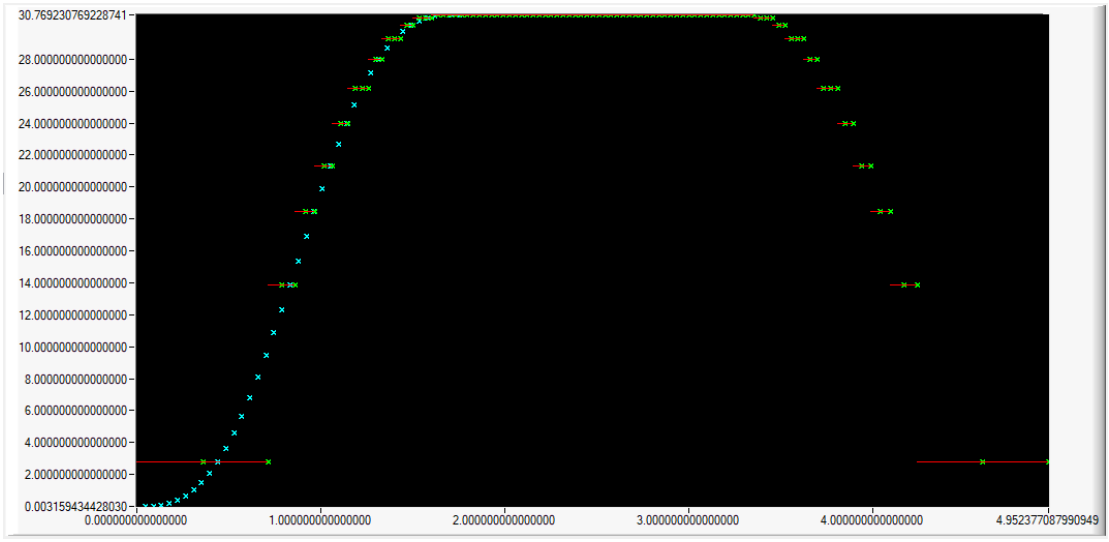
4.3 Funkce pohonu

Již od začátku tohoto projektu byly určeny tři hlavní cíle:

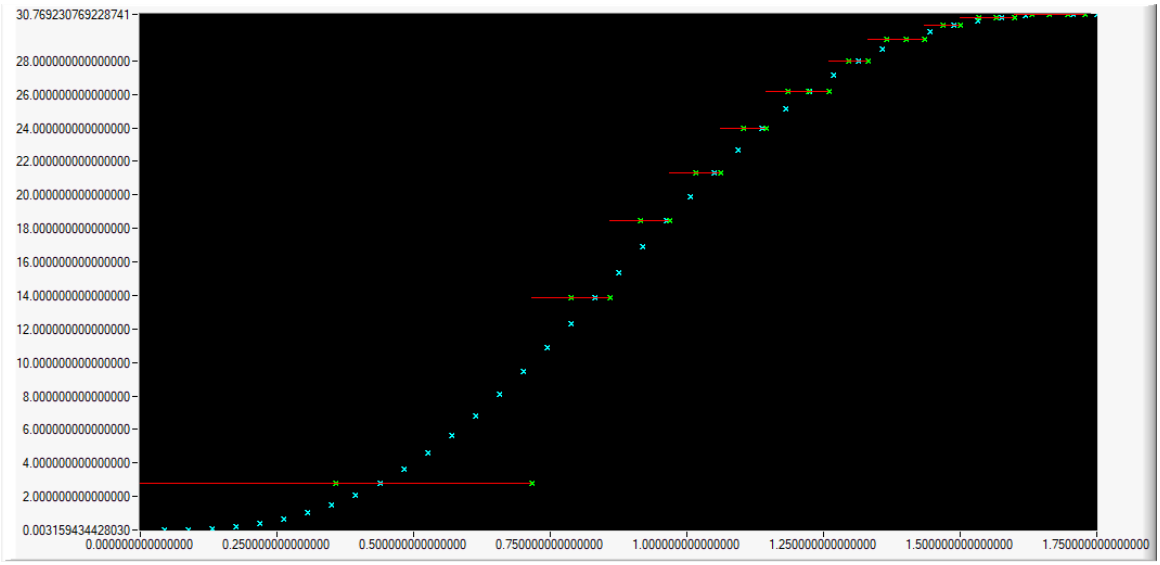
a) Na základě zadaných parametrů vypočítat a provést požadovaný pohyb, a to buď obdélníkovým, trapezoidním nebo S-křivkovým rychlostním profilem.
b) Ovládání, nastavování a kontrola vybraných dat a parametrů bude prováděna výhradně přes MODBUS.
c) Možnost spustit pohon v režimu uzavřené smyčky.

Matematické vypočítání rychlostních profilů až tak složité není, ale implementace do ovládání krokového motoru se ukázala být jako poměrně komplexní problém. Vzhledem k tomu, že jsem nechtěl tupě kopírovat triky z internetu, a také vzhledem k tomu, že jsem chtěl mít pod kontrolo u nejen ujetou dráhu, ale také celkový čas pohybu s danou akcelerací a decelerací, narazil jsem na spoustu problémů. Jedním z hlavních parametrů při ovládání krokového motoru je závislost času na úhlu otočení hřídele na jeden krok. Je nutno si uvědomit, že s každým vykonaným krokem v určitém čase jsme přišli nejen o jeden krok ze zbývající dráhy, ale také o určitý čas z celkového daného času. A v tomto zbývajícím čase a dráze musíme správně vykonat zbytek prováděného profilu. Po mnoha slepých uličkách jsem vymyslel metodu, jak toho dosáhnout. Tato metoda je popsaná v kapitole 2.3. A po úspěšné implementaci do PIC32MX jsem si prakticky ověřil, že tyto výpočty skutečně fungují, motor dosahuje opakovaně cílové polohy a jsou patrné rozdíly mezi jednotlivými rychlostními profily.

Na obrázku 32 je vidět nasimulovaný časový profil S-křivkového pohybu. Modré křížky vyznačují matematicky vypočítaný profil, červené horizontální linky vyznačují dobu trvání pohybu konstantní rychlosti. Zelené křížky potom vyznačují jednotlivé kroky.



Obrázek 32: Simulace pohybu s S-křivkou (celkový čas 5 s, akcelerace/decelerace 1,75 s) Legenda: Modré křížky ukazují vypočítaný profil, zelené křížky jednotlivé kroky, červené trvání pohybu

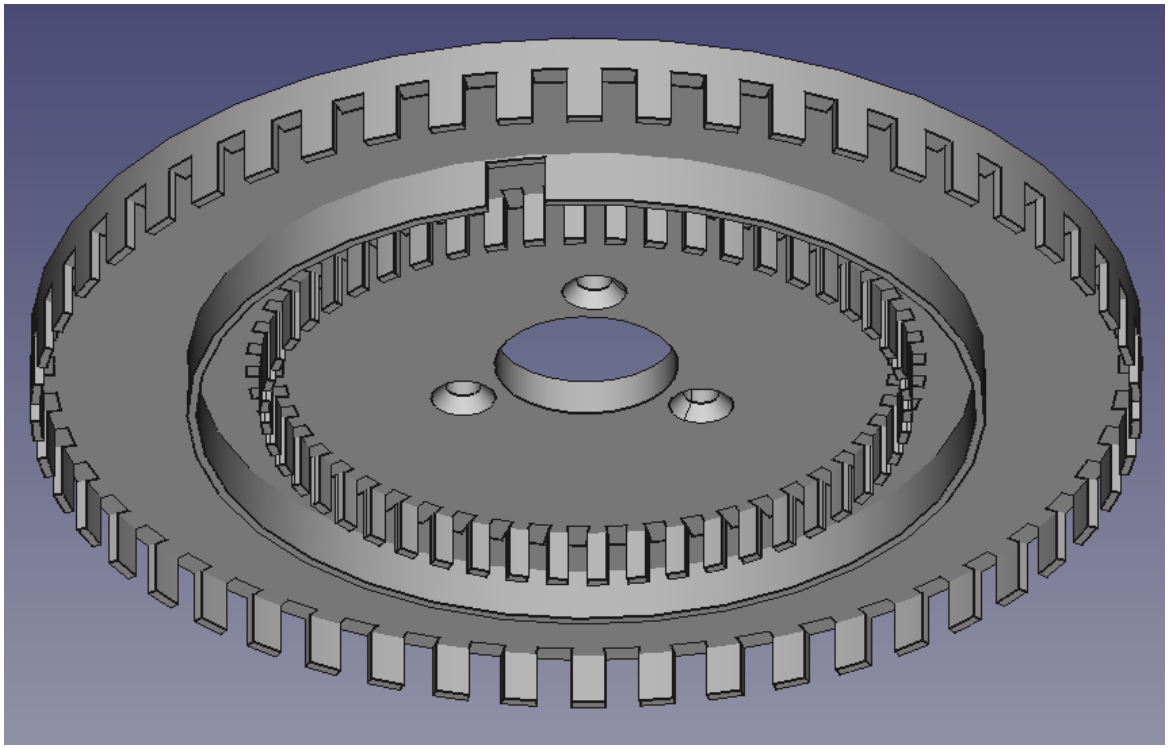


Obrázek 33: Detailní pohled na akceleraci simulovaného pohybu s S-křivkou podle obrázku 32

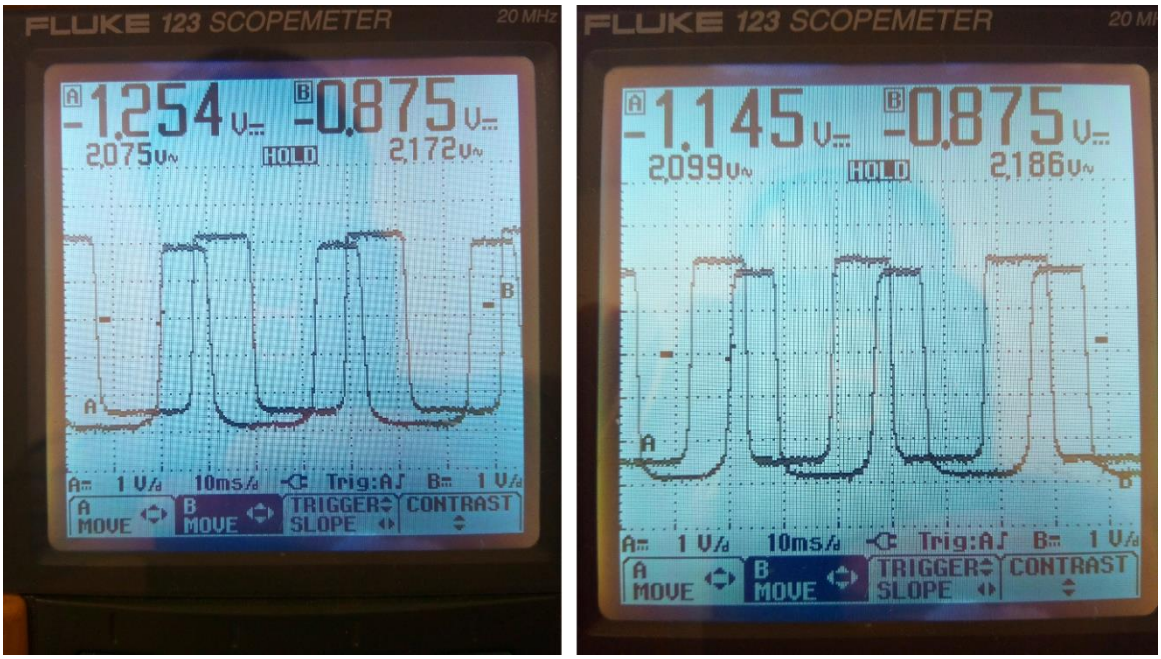
Jak již bylo popsáno v kapitole 4.2, funkce MODBUSu se povedla implementovat dobře a po odladění s ní nejsou žádné další potíže.

Výše uvedený výsledek se však nedá tvrdit o implementaci řízení v uzavřené smyčce. I když se podařilo dobře ošetřit načítání polohy a při testech se to několikrát potvrdilo, tak se mi do dnešního dne nepodařilo řízení v uzavřené smyčce uspokojivě zprovoznit. Při režimu uzavřené smyčky totiž motor v určitých pozicích začne kmitat. Pozitivní je, že i přes nedůvěryhodný pohyb dojede pohon vždy stejné polohy. V tuto chvíli stále hledám možnosti, jak dosáhnout plné funkčnosti režimu v uzavřené smyčce.

Podle mé teorie je problém v enkodéru. I když je precizně vytisknutý na 3D tiskárně Průša a je i dostatečně velký, tak je asi mechanicky na limitu tiskárny. Druhý problém může být i v tom, že ozubená kola enkodéru jsou ve dvou řadách s různým průměrem. Z toho důvodu musí být šířky zubů a děr na obou kolech různé. Na vnějším kole jsou zuby širší než na vnitřním. Jejich počet totiž musí být shodný na obou kolech. Další věcí bylo to, že kolo s enkodérem nešlo nasadit na hřídel motoru. Proto jsem upravil díru 5mm vrtákem. Naneštěstí jsem to udělal na stojanové, ne úplně přesné vrtačce. Kolo enkodéru při otáčení kolísá nahoru a dolů. To by mohlo vysvětlovat, proč se motor rozechvěje vždy ve stejných polohách. Na obrázku 33 je vidět konstrukce enkodéru.



Obrázek 34: Konstrukce kola enkodéru



Obrázek 35: Naměřené průběhy na enkodéru – otáčení doleva a otáčení doprava

4.4 Hardware

Hardware jsem navrhoval podle datových listů jednotlivých komponent. Takže jsem zapojil jak ochranné diody na koncový H-můstek, tak blokovací kondenzátory na převodnících úrovní signálů. Ovšem jak už to tak bývá, něco jsem podcenil. Tím je chlazení koncového můstku L298. Během jednoho z mnoha testování jsem se přepsal v programu a výsledkem byl poměrně rychlý konec H-můstku. Bohužel se zničil tím nejhorším možným způsobem, a to do zkratu. Díky tomu se zničily i výstupy MCU na desce PINGUINO. Po výměně součástek už jsem pro jistotu napájel výkonovou část s proudovým omezením. To je také důvod, proč si myslím, že by mělo být napájení této sestavy laboratorním zdrojem s proudovým omezením. Upsat se v programu je totiž velice snadné. Co se týče desky plošných spojů, je celé zařízení postaveno na univerzální desce. Vzhledem k nízkým frekvencím, které se v této aplikaci vyskytují, je tato deska vyhovující a za celou dobu se v tomto ohledu nevyskytl jediný problém.

Obrázek 36: Vytištěné díly na 3D tiskárně

5 Závěr

Hned na úvod této kapitoly musím pokorně uznat, že toto téma bylo složitější, než jsem si původně myslel. Zabýval jsem se touto prací téměř rok, ale i tak mi zbyl jeden nevyřešený cíl. Mou motivací bylo a je ukázat studentům principy řízení krokového motoru. To, co jsem chtěl, bylo ukázat, že motor lze řídit i v režimu mikrokrokování jenom s MCU, bez dalších pomocných obvodů. Toto se podařilo. Také se mi podařilo vyřešit metodu, jak počítat rychlostní profily libovolných vzdáleností i na mikrokontrolérech s omezenou kapacitou paměti RAM.

Jako student kombinované formy studia jsem v trvalém zaměstnání a profesně se věnuji mimo programování PLC (Programmable Logic Controller) také pohonům, řízením motorů a jejich programování. I když mám poměrně bohaté zkušenosti s navrhováním řízení strojů, navrhování „stroje“ na úrovni součástek pro mě byla zcela nová kapitola. Při dlouhé cestě k točícímu se motoru jsem musel překonávat spoustu neznámých překážek. Příkladem může být kompletně vpálená deska po překlepu v programu, zdvihání napětí pullup rezistory na vstupech kvůli rušení a spousty dalších věcí.

Jednou z věcí, která výrazně překonala mé představy o náročnosti, byl návrh mechanické konstrukce celého zařízení. Díky konfliktu mezi bezpečností a funkčností jsem po mnoha pokusech a omylech dospěl k rotačnímu pohonu. Kvůli nekonečným slepým uličkám při snahách o výpočty rychlostních profilů jsem si musel napsat simulační aplikaci v LabWindows/CVI. Na samotném MCU32MX bylo debugování programu a hledání správných řešení pomalé, ale hlavně velice složité.

V současné době ještě pracuji na dokončení provozu v uzavřené smyčce. Stále ještě věřím tomu, že důvody nefunkčnosti uvedené v kapitole 4.3 lze do jisté míry vyřešit softwarově. Všechny mé pokusy však zatím končily neúspěchem. O tom, jestli se mi podaří, nebo nepodaří tento problém vyřešit, nechci polemizovat. Nechci ovšem zbaběle vyměnit amatérsky vyrobený enkodér za nějaké průmyslové řešení, případně nechat vyrobit enkodér třískovým obráběním z duralu, kde by byl sice úspěch téměř zaručen, ale chyběl by mi pocit vítězství.

Ten je k životu také potřeba. Díky tomu, že projekt je plně otevřený, může se v případě mého neúspěchu pokusit o řešení někdo jiný. Také mám objednanou novou desku plošných spojů, ale vzhledem k těžké době nevím, jestli ji budu mít v době obhajoby této práce. Doufám, že ano.

Pokud bych měl shrnout výsledky této práce, tak mi přinesla hodně nových zkušeností, které jistě využiji. Také mi dala motivaci nebát se vývoje dalších zařízení na této úrovni v budoucnu. Mám radost z toho, že se mi povedlo spoustu věcí dotáhnout do konce, ale také trochu cítím zklamání z nedokončeného řízení v uzavřené smyčce.

Seznam literatury

[1] NEW JAPAN RADIO CO. Microstepping. *Njr.com* [online]. © 2020 [cit. 2021-03-03]. Dostupné z: https://www.njr.com/electronic_device/PDF/application_notes/Microstepping_APP_E.pdf

[2] TUCKER, K. *Marissa. Basic Machine Design and the Physics of Motion* [online]. Charlotte: Parker Hannifin [cit. 2021-03-03]. Dostupné z: https://www.google.com/url?client=internal-element-cse&cx=007853267621696440593:t27sg9jgpks&q=https://www.automateshow.com/filesDownload.cfm%3Fdl%3DMCMACMCP_BasicMachineDesignandthePhysicsofMotion_REVD.pdf&sa=U&ved=2ahUKEwjmlLv93aLtAhWQ4YUKHbxjAuUQFjAAegQIARAB&usg=AOvVaw1yh40Rw-_Sh8fsV91DVH09

[3] COLLINS, Danielle. How to generate the motion profile for a linear system. In: Linearmotiontips.com [online]. 22. 4. 2019 [cit. 2021-03-03]. Dostupné z:https://www.linearmotiontips.com/how-to-generate-motion-profile-for-linear-system/

[4] RS-485. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2020 [cit. 2021-04-12]. Dostupné z: https://cs.wikipedia.org/wiki/RS-485

[5] IPC2U. Modbus RTU v kostce. *Ipc2u.cz* [online]. © 2021 [cit. 2021-02-02]. Dostupné z: https://ipc2u.cz/articles/simple-decisions/protokolu-modbus-rtu-v-kostce/

[6] SCHNEIDER ELECTRIC. Modbus. Se.com [online], Dostupné z: https://www.se.com/us/en/product-range-presentation/574-modbus/

[7] *Microchip* [online]. © 2021 [cit. 2021-02-02]. Dostupné z: https://www.microchip.com/

[8] RS COMPONENTS. Přírůstkový kodér HTL, DF60B-S4PA10000. Cz.rs-online.com [online]. © 2021 [cit. 2021-02-02]. Dostupné z: https://cz.rs-online.com/web/p/snimace-rizeni-pohybu/7125424/?cm_mmc=CZ-PLA-DS3A-_-google-_-CSS_CZ_CZ_Automatizace_a_řídicí_zařízení_Whoop-_-(CZ:Whoop!)+Snímače+řízení+pohybu-_-7125424&matchtype=&aud-826607885227:pla-341215817503&gclid=Cj0KCQjw0caCBhCIARIsAGAfuMxubtAwpzQ_UZVtfia8cqOautPvfY31Ya83fJFjuSjWQLckyLj1jqAaAlMEEALw_wc

[9] *PINGUINO* [online]. © 2021 [cit. 2021-02-02]. Dostupné z: https://pinguino.cc

[10] *STMicroelecronics* [online] © 2021 [cit. 2021-02-02]. Dostupné z: https://www.st.com

[11] GM ELECTRONIC. Optická závora TCST2300. Gme.cz [online]. © 2021 [cit. 2021-02-02]. Dostupné z: https://www.gme.cz/opticka-zavora-tcst2300

[12] STMicroelecronics. Datasheet ST1480AC. *St.com* [online]. © 2021 [cit. 2021-02-02]. Dostupné z: https://www.st.com/resource/en/datasheet/st1480ac.pdf

Přílohy

- STL modely pro 3D tisk,
- 3D modely sestavy ve formátu FreeCad,
- PC aplikaci pro ovládání výukové sestavy (zdrojový kód a .exe distribuci),
- Zdrojový kód MCU PIC 32,
- Stručný návod k ovládání se seznamem ovládacích registrů,
- Schéma zapojení ve vysokém rozlišení.