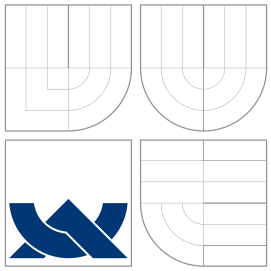
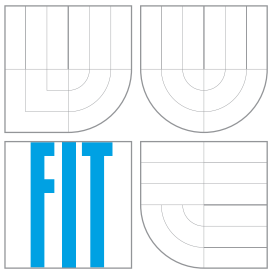


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ŘÍDICÍ JEDNOTKA KROKOVÉHO MOTORU ZALOŽENÁ NA MIKROKONTROLÉRU HC08

STEPPER MOTOR CONTROL UNIT BASED ON MICROCONTROLLER HC08

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JOSEF HÁJEK
AUTHOR

VEDOUCÍ PRÁCE — Ing. JOSEF STRNADEL, Ph.D.
SUPERVISOR

BRNO 2007

Zadání bakalářské práce/5028/2006/xhajek17

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2006/2007

Zadání bakalářské práce

Řešitel: **Hájek Josef**
Obor: Informační technologie
Téma: **Řídící jednotka krokového motoru založená na mikrokontroléru HC08**
Kategorie: Vestavěné systémy

Pokyny:

1. Seznamte se s principy činnosti a řízení krokových motorů.
2. Po dohodě s vedoucím nastudujte parametry zadaného typu krokového motoru.
3. Seznamte se s kitem 908LJ12-G2 a promyslete možnosti řízení motoru pomocí mikrokontroléru HC08 dostupného na tomto kitu.
4. S využitím kitu navrhněte a realizujte jednotku pro řízení krokového motoru z bodu 2. Jednotka nechť je schopna pracovat ve 2 režimech - A a B. V režimu A bude uživatel moci zadávat rychlost a směr otáčení motoru a volit typ řízení. V režimu B bude možno zadat úhel, o který se má pootočit hřídel motoru.
5. Pro komunikaci s uživatelem využijte periferie dostupné na kitu (např. LCD displej a tlačítka). Funkčnost jednotky prakticky ověřte.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Body 1-3.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Strnadel Josef, Ing., Ph.D.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 15. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

L.S.

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

Licenční smlouva

Licenčni smlouva je uložena v archivu Fakulty informačních technologií Vysokého učení technického v Brně.

Abstrakt

Práce je věnována problematice řízení krokových motorů za pomocí jednočipových mikrokontrolérů. Cílem je seznámit se s principy činnosti krokových motorů, s jejich možnostmi a omezeními v případě použití ve vestavěných zařízeních.

Součástí práce je návrh řídící jednotky vybraného krokového motoru a jeho propojení s vývojovým kitem 908LJ12-G2. Implementační část pak zahrnuje program psaný v jazyku C, pro osmibitový mikrokontrolér Motorola MC68HC908LJ12, kterým lze pak za pomocí periférií dostupných na kitu motor ovládat.

Klíčová slova

Krokový motor, řízení, mikrokontrolér

Abstract

This bachelor's work deals with issue of stepper motors and their control using single-chip microcontrollers. Target of the work is to be familiar with method of operation of stepper motors, with their abilities and restrictions in case there are used in built-in devices.

The part of the work is a proposal of control unit of chosen stepper motor and interconnection with developmental kit 908LJ12-G2. Implementation part of the work contains software for eight-bit microcontroller Motorola MC68HC908LJ12 programmed in language C. The stepper motor can be controlled by this program using the available kit's interfaces.

Keywords

Stepper motor, driving, microcontroller

Citace

Josef Hájek: Řídicí jednotka krokového motoru založená na mikrokontroléru HC08, bakalářská práce, Brno, FIT VUT v Brně, 2007

Řídicí jednotka krokového motoru založená na mikrokontroléru HC08

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Josefa Strnadela Ph.D.

.....
Josef Hájek
13. května 2007

Obsah

Kapitola 1

Úvod

Digitalizovaná technika prostoupila již do snad všech lidských odvětví. Počítače se využívají téměř všude, jejich vývoj je stále na postupu a výpočetní výkon roste. Klasické stolní počítače jsou pro použití velmi univerzální a často se s jejich pomocí řídí rozsáhlé systémy, avšak nesou sebou jistá omezení, proto je nelze použít všude. Zde nastupují na scénu vestavěné systémy, které se dnes běžně používají v automatizaci a k řízení periférii, kde není potřeba vysokého výpočetního výkonu, což sebou nese výhody malých rozměrů a nízké ceny řídící jednotky.

Jak už sám název napovídá, tato práce se zabývá návrhem a realizací vestavěného systému pro řízení krokových motorů. Jednotka je založena na vývojovém kitu 908LJ12-G2. S využitím jeho periférii se ve vhodném kódu generují elektrické impulsy, které mohou řídit samotný motor. Tyto impulsy musí být zesíleny, to se děje v tzv. silové části, kde se napěťové úrovně z výstupu mikrokontroléru převádí na napěťové úrovně řízení jednotlivých vinutí samotného motoru SMR 300-100-RI/24. Všechny tyto části jsou podrobněji rozebrány v kapitole 3.

Než se začneme zabývat vlasnim návrhem je nutné se seznámit s nezbytnou teorií a principem činnosti řízení krokových motorů. Každý motor má svá specifika, ale obecně je pro náš návrh důležitých jenom několik zásadních parametrů. S teorií řízení krokových motorů se seznámíme v kapitole 2.

Vlastním návrhem aplikace se zabývá kapitola 4. Jsou zde tedy zváženy možnosti jednotlivých komponent ze kterých se řídící jednotka skládá. Dále pak následuje způsob propojení jednotlivých části do výsledného funkčního celku.

Program pro mikrokontroler je napsán ve vyšším programovacím jazyku C, to sebou nese jistá specifika. Proč tomu tak je a způsob implementace je naznačen v kapitole 5.

Kapitola 2

Krokové motory - teorie

Tato kapitola se zabývá úvodem do principu funkce a řízení krokových motorů. Jsou zde rozebrány metody řízení a základní parametry, které jsou důležité pro další práci s motorem.

2.1 Krokový motor obecně

Krokové motory jsou stroje, které za pomocí elektromagnetismu přeměňují energii ve formě elektrických impulsů na diskrétní točivý pohyb. Jsou speciálním druhem synchronního motoru, který obsahuje mnoho magnetických pólů. Mají široké využití ve výpočetní technice (tiskárny, skenery, disketové mechaniky), zabezpečovací tehnice (elektrické zámky, kamerové systémy), obráběcí stroje (CNC) a v dalších oblastech, kde je potřeba jednoduše a přesně nastavovat polohu nějakého zařízení.

Obecný krokový motor se skládá z rotoru a statoru. Stator krokového motoru je tvořen několika cívkami, které plní funkci elektromagnetů, přitahující zuby rotoru. Rotor je pak složen z vlastní hřídele a prstence permamentních magnetů.

Princip je velmi jednoduchý. Proud procházející cívkou statoru vytvoří magnetické pole, které přitáhne zub rotoru, který je nejblíže. Vhodným spínáním jednotlivých cívek lze pak vytvořit rotující magnetické pole, které otáčí rotorem.

Rotor krokového motoru se otáčí s každým řídícím impulsem o předem daný a přesně definovaný úhel - krok, což je mechanická odezva rotoru, kdy se otočí z jedné klidové polohy do druhé. To je velmi cenná vlastnost, která jej přímo předurčuje pro použití v polohovacích zařízeních. V každé poloze je schopen držet úhel natočení rotoru, potřebuje tedy napájení i v klidovém stavu, tzn. má trvalý odběr proudu ze zdroje. Tento typ synchronního motoru má oproti klasickým rotačním motorům výhodu v tom, že pokud známe počáteční polohu a jestliže neztratíme synchronizaci, máme v každém okamžiku informací o aktuální poloze rotoru.

Existuje více variant metod řízení krokových motorů. Volíme je proto podle toho, jaký chceme mít kroutící moment motoru, přesnost vystavení rotoru nebo odběr ze zdroje.

Podle konstrukce rotoru můžeme krokové motory rozdělit na aktivní a pasivní. Aktivní má rotor s vlastním vinutím nebo pernamentním magnetem. Pasivní typ (často také nazývaný jako reakční), má drážkový rotor z magneticky měkkého materiálu. Dále pak motor pro zmenšení kroku může být jednostatorový nebo vícestatorový. Hlavní řídící vinutí jednotlivých fází může být jednofázové, dvoufázové nebo vícefázové, podle počtu vinutí sepnutých v jednom okamžiku. V každém okamžiku je sepnuta alespoň jedna z fází motoru.

- **Výhody krokových motorů**
  - Přesnost a opakovatelnost natočení rotoru.
  - Citlivost a akcelerace - krokové motory mají malou setrvačnost rotoru, umožňují tedy rychle zvyšovat jejich rychlost, proto se přímo vybízejí pro použití, kde jsou potřeba krátké a rychlé přesuny.
  - Stabilita polohy - na rozdíl od jiných typů motorů jsou schopny držet svoji zastavenou polohu rotoru zcela bez hnutí.
  - Chyba natočení rotoru se s počtem kroků nesčítá.
  - V každém okamžiku známá poloha - krokové motory se otáčejí o předem definovaný úhel v závislosti na vstupních impulsech, známe-li tedy počet impulsů, které jsme motoru předali, známe také celkový úhel o který se rotor otočil. Není potřeba zpětná vazba.
  - Cena a spolehlivost - technologie krokového motoru je spolehlivá a osvědčená. Je to cenové nejefektivnější metoda pro řízení polohy.
- **Nevýhody krokových motorů**
  - Malý kroutící moment v porovnání s klasickými stejnosměrnými motory.
  - Výrazně omezena rychlost rotace - na cca desítky otáček za sekundu. Tato vlastnost je dána především přechodovými magnetickými jevy a při překročení této rychlosti dochází k desynchronizaci kroku s řídícími impulsy.
  - Vibrace motoru během přepínaní jednotlivých fází.
  - Odběr proudu ze droje, i když je motor zastaven v klidové poloze.

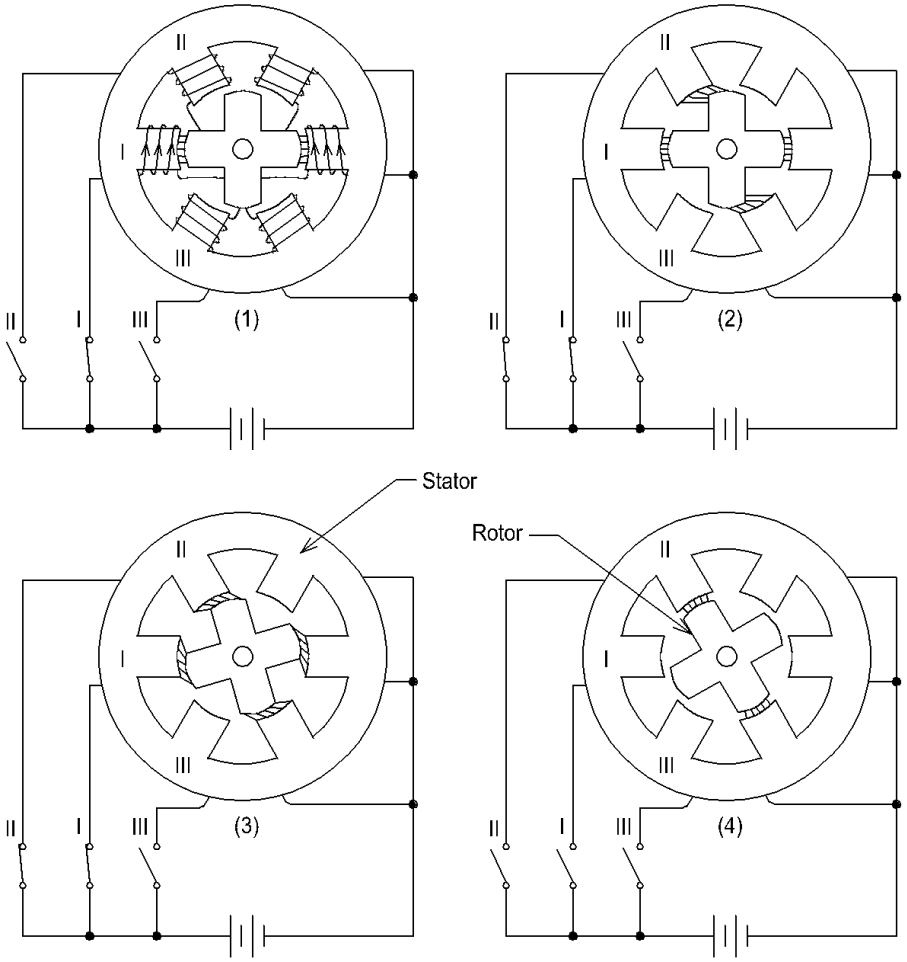
2.2 Příklad činnosti krokového motoru

Abychom si více přiblížili činnost krokového motoru a jeho principu, je zde ukázka funkce převzatá z [3].

Obrázek 2.1 naznačuje příčný řez krokového motoru. Jedná se o klasický reakční motor. Skládá se ze dvou hlavních části a to z rotoru a statoru. Motor na obrázku má jeden rotor se čtyřmi póly (zuby). Pro zjemnění velikosti kroku se často používájí dvou a více rotorové motory, jejichž zuby se v axiálním směru vzájemné nekryjí. Stator je tvořen šesti magnetickými póly, umístěnými po vnitřním obvodu motoru. Póly statoru a rotoru bývají často vroubkované, k docílení větší přesnosti natočení. Stator má tři sady protilehlých elektromagnetů I, II a III. Jedna sada obsahuje vždy dvě cívky zapojené do série a každá z těchto sad se nazývá fáze. Motor na obrázku 2.1 je tedy třífázový a každá fáze může být spínaná stejnosměrným napětím přes spínače I, II, a III.

Samotná funkčnost motoru pak spočívá ve vztahu jednotlivých části při buzení samotných fází. Ve stavu (1) na obrázku 2.1 je sepnuta (vybuzená) fáze I proudem přes první spínač. Šipky kolem cívek naznačují směr magnetického toku, které se vyskytne ve vzduchové mezeře mezi statorem a rotorem během vybuzení. Ve stavu (1) jsou dva póly statoru vybuzeny, zuby rotoru jsou v ose se zuby fáze I statoru, motor se nachází ve vyváženém stavu.

Pokud zároveň sepneme spínač II, bude navíc vybuzená i fáze II a magnetický tok začne na rotor působit způsobem naznačeným na obrázku ve stavu (2). Vytvoří se kroutící moment působící na rotor, který se pootočí tak, aby magnetické síly působící na něj z obou



Obrázek 2.1: Princip činnosti krokového motoru

fází byly v rovnováze. Motor tak dosáhne stavu (3), proti původnímu stavu je tedy pootočen o 15°. Jestliže v tomto stavu rozepneme spínač I, motor se dále pootočí o úhel 15° a zaujme vyvážený stav (4). Postupným spínáním fází I, II a III můžeme roztočit motor proti směru hodinových ručiček. Potřebujeme-li zpětný chod, spínáme fáze v opačném pořadí.

Vhodným spínáním jednotlivých fázi tedy můžeme vytvořit rotující magnetické pole, které při každém přepnutí fází otočí rotorem o daný úhel. Rozlišení velikosti kroku závisí na vzájemném vztahu statoru s rotorem a je tedy pevně dáno mechanickou konstrukcí. To je také důvod, proč se řízení obejde bez jakéhokoliv čidla polohy či zpětné vazby. Pokud neporušíme některou z podmínek bezchybné synchronizace vstupních impulsů s motorem, tak v každém okamžiku máme informaci o tom v jaké poloze se rotor nachází, protože pokud známe počet impulsů předaných motoru k pootočení o jeden krok, známe i celkový úhel o který se rotor otočil.

Přesnost vystavení zubů rotoru vůči pólům statoru závisí na několika hlavních faktorech. A to především na preciznosti konstrukce motoru, otáčivé zátěží (momentu), která může vychylovat osy jednotlivých zubů rotoru vůči zubům statoru. Tato vlastnost se nazývá přídržný moment, je závislá na síle magnetu a tedy i na stejnosměrném odporu cívek.

2.3 Základní vlastnosti krokových motorů

Pokud chceme používat ve svých konstrukcích krokové motory, je nutno se v závislosti na aplikaci motoru seznámit s jejich statickými a dynamickými vlastnostmi. Následující podkapitoly 2.3.1 a 2.3.2 podávají informaci o několika hlavních parametrech krokových motorů, se kterými budeme muset počítat při konstrukci řídící jednotky. Kapitola 2.3.3 pak informuje o dalších pojmech udávaných v souvislosti s krokovými motory.

2.3.1 Statické parametry

Statické parametry udávají závislost výstupní veličiny na vstupní v ustáleném stavu. V případě krokových motorů se za ustálený stav považuje okamžik, kdy je motor zastaven a jsou vybuzeny magnety rotoru.

Úhel kroku

Základní úhel o který se hřídel motoru pootočí při přepnutí fází. Určuje úhlové rozlišení s jakým je motor schopen pracovat. Úhel kroku Θ lze vypočítat následovně (výpočet je uveden na [3]):

$$\Theta_s = \frac{360^\circ}{S}$$
$$S = mN_r$$

kde

m = počet fází

N_r = počet zubů rotoru

Příklad výpočtu pro motor uvedený v podkapitole 2.2:

$$m = 3, N_r = 4$$
$$S = m \cdot N_r = 12$$
$$\Theta_s = \frac{360}{12} = 30^\circ$$

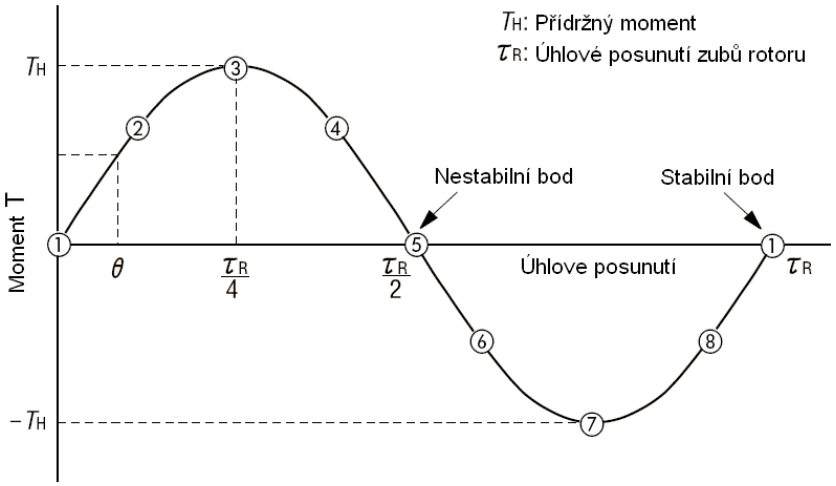
Motor z příkladu má tedy základní úhel kroku 30°. Krok je možno dále zjemnit technologií řízení jednotlivých fází motoru viz podkapitola 2.4.

Přídržný moment

Tento parametr je důležitý pro určení maximální velikosti vnější zátěže. Je to maximální kroutící moment, kterým může být staticky zatížena hřídel vybuzeného motoru, aniž by se začala otáčet. Jedná se o moment při nulových otáčkách motoru, tedy při $f = 0$, kdy je rotor v dané poloze držen právě vybuzenými magnety statoru.

Graf na obrázku 2.2 charakterizuje vztah mezi úhlovým posunutím rotoru a momentem působícím na jeho hřídel vnější zátěží při sepnuté fázi. Charakteristika je uvedena na [2].

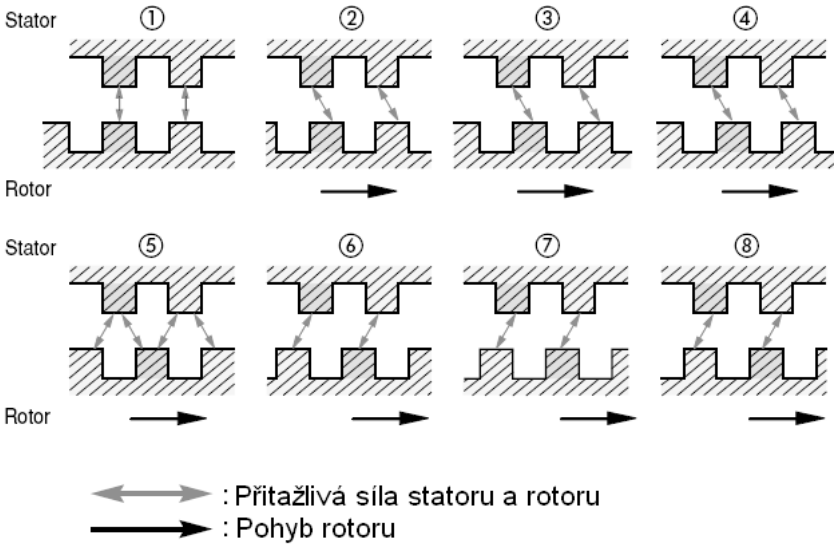
Horní řada zubů na obrázku 2.3 vyznačuje stator, dolní řada pak rotor. Na začátku se motor nachází v bodě (1), což je stabilní poloha, kdy jsou zuby statoru přímo zarovnány se zuby rotoru. Pokud nyní na hřídel začne působit vnější točívá síla, magnetické pole vytvoří kladný moment T, který má tendenci zuby posunout zpět do stabilní polohy (1), kde by



Obrázek 2.2: Závislost kroutícího momentu na velikosti úhlu natočení

se hřídel znovu zastavila. Jestliže vnější síla pootočí hřídelí až do polohy (3), nachází se v úhlu, kde na hřídel působí maximální přídržný moment T_h.

Při překonání tohoto momentu se dostáváme do polohy (5), kdy jsou zuby rotoru a statoru vzájemně posunuty právě o šířku jednoho zubu. Jedná se o nestabilní polohu mezi kladným a záporným točivým momentem T, kdy stačí sebemenší síla na to, aby se rotor otočil doprava nebo doleva, zpět do stabilní polohy. Po překonání této polohy se motor otáčí do další stabilní polohy (1), kde jsou zuby proti sobě znovu přesně zarovnány.



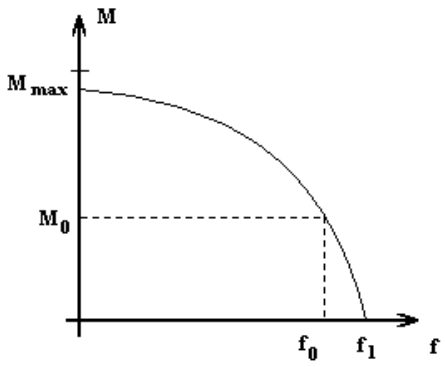
Obrázek 2.3: Pozice zubů rotoru a statoru v jednotlivých fázích natočení

Na tuto vlastnost musíme vždy pamatovat, protože příliš velkou vnější zátěží, na kterou není motor konstruován, může docházet k desynchronizaci kroku s přepínáním jednotlivých fázi, tzn. ztrácíme informací o poloze rotoru.

2.3.2 Dynamické parametry

Týkají se především rychlosti a jejího momentu, když se motor roztáčí nebo již běží. Sada charakteristik vyjadřující provedení krokových motorů je dánn konkrétním motorem a typem jeho řízení viz podkapitola 2.4.

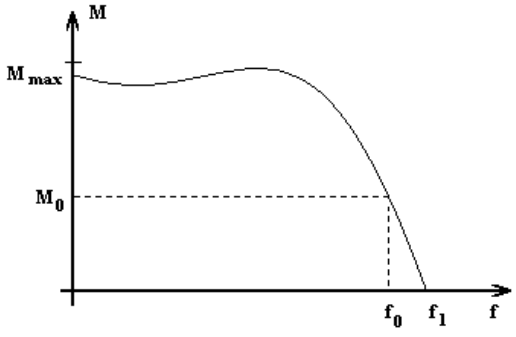
Rozběhový moment



Obrázek 2.4: Rozběhová charakteristika

Graf rozběhové charakteristiky na obrázku 2.4 znázorňuje stavy do kterých se může krokový motor dostat z klidu bez ztráty jediného kroku. Například při zatěžovacím momentu M_0, můžeme skokově zvýšit řídící kmitočet z nuly na f_0, bez ztráty kroku. Chceme-li motor z klidu roztočit kmitočtem f_1, je nutno motor spustit bez zátěže. Je to maximální kmitočet, který lze skokově pro roztočení motoru použít. Použitím vyšších kmitočtů při rozběhu se mohou objevit ztráty kroku.

Momentová charakteristika



Obrázek 2.5: Kroutící moment motoru

Graf na obrázku 2.5 znázorňuje provozní charakteristiku motoru. Je to závislost kroutícího momentu na řídícím kmitočtu. Jedná se o tzv. oblast omezené řiditelnosti. V této oblasti je nutno plynule, nikoliv skokově zvyšovat řídící kmitočet. Tímto způsobem tak musíme měnit otáčky motoru, aby nedocházelo ke ztrátám kroku. Jenom plynulým zvyšováním otáček po

rozběhu motoru můžeme bezchybně dosáhnout hodnoty f_0. Kmitočet f_1 je maximální frekvence, na které lze motor provozovat bez zátěže. M_{max} je pak nejvyšší provozní moment, který může na motor působit.

2.3.3 Další definice parametrů krokových motorů

Několik parametrů často užívaných v souvislosti s krokovými motory. Jejich popis je převzán z [9].

- **Řídicí kmitočet** - kmitočet, kterým je řízen motor. Souhlasí s kmitočtem kroku, jestliže motor běží bez chyby kroku.
- **Magnetická klidová poloha** - poloha, kterou zaujme rotor vybuzeného motoru, jestliže chyba statistického úhlu se rovná nule.
- **Tolerance úhlu kroku** - největší kladná nebo záporná statická odchylka úhlu vůči jmenovitému úhlu kroku, která může nastat jestliže se rotor motoru pootočí o 1 krok z jedné magnetické polohy do druhé, pokud se vychází ze vztažné magnetické klidové polohy. Měří se v průběhu celé otáčky rotoru.
- **Vlastní přídržný moment** - maximální moment, kterým může být staticky zatížena hřídel nevybuzeného motoru aniž by se začala plynule otáčet.
- **Zatěžovací moment** - moment charakteru pasivního tření, kterým je zatížena hřídel motoru.
- **Stabilizační zatěžovací moment** - zatěžovací moment nutný pro funkci krokového motoru v nestabilní oblasti řídicího kmitočtu.
- **Nestabilní oblast řídicího kmitočtu** - část charakteristiky rozběhového momentu a rozběhového momentu setrvačnosti, která má bez stabilizačního zatěžovacího momentu inflexní bod.
- **Rozběhový moment setrvačnosti** - vnější moment setrvačnosti, se kterým se motor může rozběhnout start-stop bez chyby kroku, bez zatížení zatěžovacím momentem při definovaném řídicím kmitočtu.
- **Použitelný zatěžovací moment a moment setrvačnosti zátěže** - pro výpočet platí rovnice:

$$M_p = M_1 \cdot (1 - \frac{Jp}{J_1})$$

$$J_p = J_1 \cdot (1 - \frac{Mp}{M_1})$$

kde

M_p – použitelný zatěžovací moment při určitém řídícím kmitočtu f_1
M_1 – rozběhový moment při řídícím kmitočtu f_1 odečtený z rozběhové charakteristiky dodávané v dokumentaci k motoru
J_p – použitelný moment setrvačnosti zátěže při řídícím kmitočtu f_1
J_1 – rozběhový moment setrvačnosti při řídícím kmitočtu f_1 odečtený z rozběhové charakteristiky dodávané v dokumentaci k motoru

2.4 Možnosti řízení krokových motorů

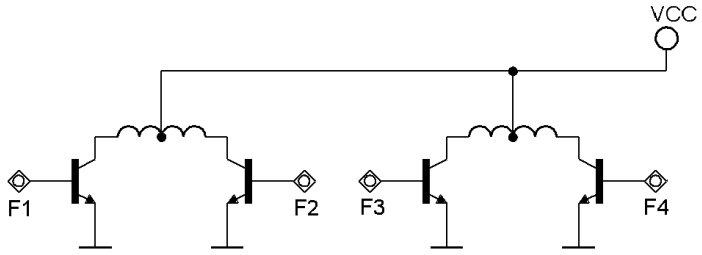
Kapitola rozebírá možnosti a metody řízení motoru. Kapitola čerpá ze zdroje [15].

2.4.1 Bipolární a unipolární řízení

Jak již bylo popsáno, činnost krokového motoru zajišťuje proud procházející cívkami fází, který generuje stejnosměrné magnetické pole. Bipolární a unipolární řízení závisí na konstrukci příslušného motoru.

Unipolární řízení

Je-li motor řízen unipolárně, proud prochází vždy jednou cívkou, která přitahuje zuby rotoru. Rotor unipolárního motoru je tvořen z magneticky měkkého materiálu. Unipolární řízení je z hlediska ovládání nejjednodušší, napájecí napětí je zde spínáno ve sledu jednotlivých fází. Motor pracuje s minimálním odběrem, protože v každém okamžiku je sepnuta právě jedna fáze, z toho také vyplývá menší kroutící moment.



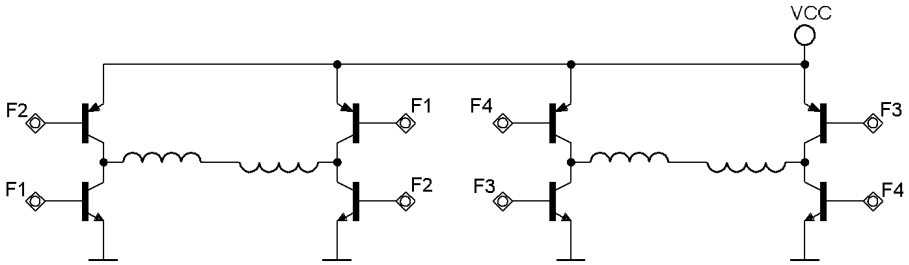
Obrázek 2.6: Unipolární řízení

Schéma na obrázku 2.6 naznačuje unipolární řízení motoru. Ve funkci spínačů fází jsou zde 4 tranzistory, které mají v kolektorech zapojeny jednotlivé cívky statoru. Přivedeme-li na bázi některého z tranzistorů kladnou napěťovou úroveň, tranzistor zajistí spojení fáze s nulovým potenciálem zdroje a cívkou tak může procházet proud. Výhodou takového zapojení je i jednoduchá elektronika potřebná pro řízení.

Bipolární řízení

Cívky jsou zapojeny tak, že mají navzájem opačně orientované magnetické pole. Rotor bipolárního krokového motoru bývá standardně vyroben z permanentního magnetu, nebo z vlastního vinutí. V jednom okamžiku jsou sepnuty vždy dvě protilehlé cívky, každá s jinou orientací magnetického pole. Nevýhodou je tak větší odběr proudu ze zdroje než u unipolárního řízení. Jedna cívka přitahuje severní pól a druhá jižní pól permanentniho magnetu rotoru což přináší výhodu až o 40% většího kroutícího momentu než u unipolárního řízení.

Obrázek 2.7 znázorňuje schéma zapojení řízení bipolárního motoru. Fáze jsou zapojeny v tzv. H-můstku, který na nich umožňuje měnit polaritu napětí a tím i smysl magnetického pole jednotlivých fází. Pro každou dvojici cívek musí být sepnuty vždy dva tranzistory. Je zde větší složitost zapojení, k řízení je potřeba 8 tranzistorů a tedy i více řídících signálů.



Obrázek 2.7: Bipolární řízení

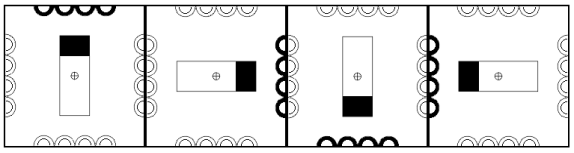
2.4.2 Jednofázové a dvoufázové řízení

Při jednofázovém řízení, je v každém okamžiku vždy sepnuta právě jedna z fází statoru. U dvoufázového řízení jsou sepnuty dvě sousední fáze, které generují magnetické pole ve stejném smyslu. Výhodou je tak větší kroutící moment, ale na druhou stranu je potřeba vyššího napájecího proudu.

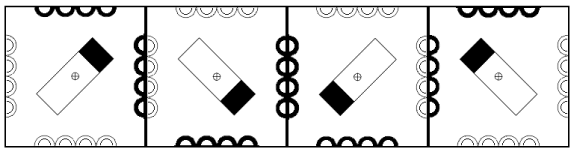
2.4.3 Krokování motoru

Řízení s plným krokem

Řízení s plným krokem říká, že na jednu otáčku je potřeba tolik kroků, kolik má stator motoru magnetických pólů.



Obrázek 2.8: Řízení jednofázové s plným krokem

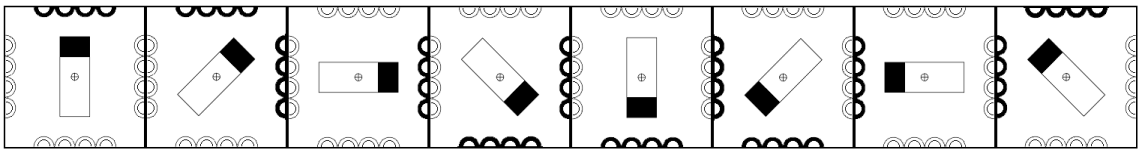


Obrázek 2.9: Řízení dvoufázové s plným krokem

Na obrázcích 2.8 a 2.9 je naznačeno jednofázové a dvoufázové řízení s plným krokem. Magnet uprostřed představuje rotor, cívky po obvodu stator, černě vybarvené cívky pak vybuzené fáze. Jsou spínány pouze vinutí v jednom smyslu magnetického toku, proto se jedná o řízení unipolární. Pro bipolární řízení by musely být spínány i protější cívky a to s opačnou orientací magnetického pole, které by přitahovaly opačný pól magnetu rotoru. V dalším textu budeme označovat řízení s plným krokem jako *čtyřtaktní*.

Řízení s poloviční krokem

Řízení krokového motoru s poloviční krokem umožňuje dosáhnout dvojnásobného rozlišení úhlu na krok, než řízení s plným krokem.



Obrázek 2.10: Řízení s poloviční krokem

Obrázek 2.10 ilustruje řízení, kdy jsou střídavě spínány jedna a dvě fáze statoru. Jedná se tedy o jakousi kombinaci jednofázového a dvoufázového řízení. To sebou také nese některé odlišnosti, např. různé přídržné momenty v okamžicích, kdy rotor střídavě přitahuje jedna nebo dvě vybuzené fáze. V dalším textu budeme řízení s poloviční krokem označovat jako *osmitaktní*.

Řízení s mikrokrokem

Je-li požadováno opravdu jemné rozlišení úhlu natočení, technikou mikrokrokování můžeme krok ještě zmenšit. Při klasickém řízení teče do fází vždy proud o stejné velikosti. Technika mikrokrokování je založena na vhodné volbě velikosti proudu v jednotlivých fázích, která umožňuje dosáhnout téměř libovolné polohy mezi dvěmi sousedními klidovými polohami. Nejefektivnější pro řízení proudu jednotlivými fázemi je pulsně šířková modulace, s jejíž pomoci můžeme dosáhnout velmi velké přesnosti vystavení rotoru. Více informací k teorii řízení pomocí technologie mikrokroku je na [4].

Kapitola 3

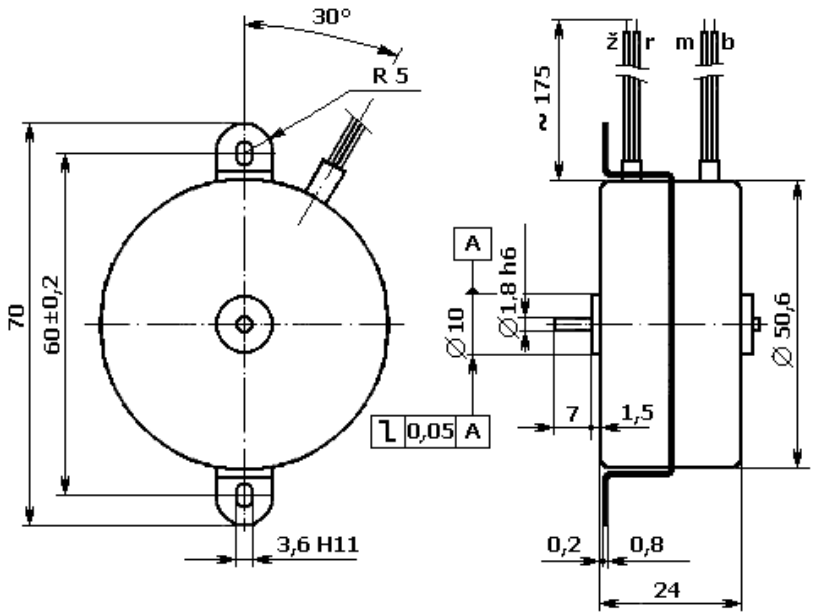
Části řídící jednotky

Kapitola rozebírá parametry jednotlivých částí pro návrh samotné řídící jednotky. Sekce 3.3 se pak zabývá návrhem konstrukce silové části jednotky, pro spínání jednotlivých fází motoru.

3.1 Kroková pohonná reverzační jednotka SMR 300-100-RI/24

Jednotka SMR 300-100-RI/24 je krokový motor, kontrukčně řešen jako čtyřfázový s rotorem z permanentního magnetu a se statorem s vyniklými póly. Počet kroků na jednu otáčku je určen počtem pólů a způsobem řízení. Podle rozdělení vstupních impulsů do jednotlivých cívek může motor pracovat buď v čtyřtaktním nebo osmitaktním režimu. Tento motor vyrábí firma REGULACE – AUTOMATIZACE BOR, spol. s r.o [10].

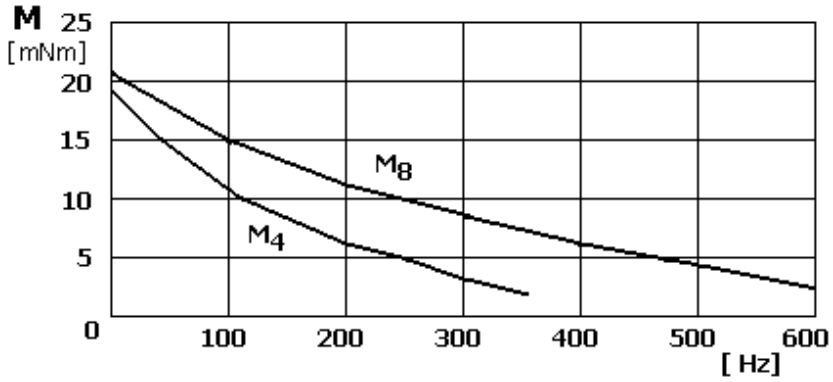
3.1.1 Rozměrový náčrt



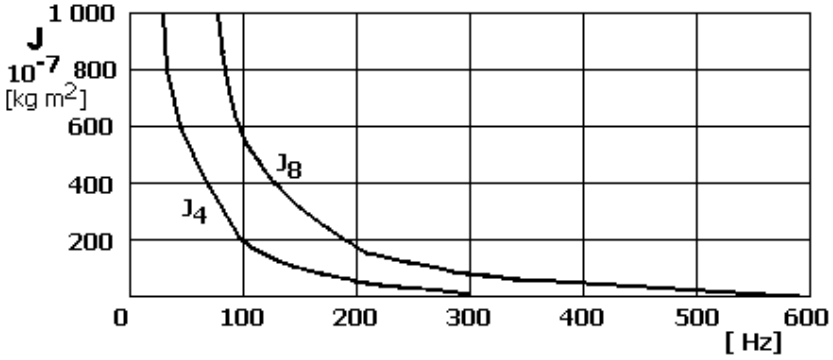
Obrázek 3.1: Rozměry motoru

3.1.2 Momentové charakteristiky

Grafy popisující dynamické vlastnosti krokového motoru SMR 300-100-RI/24. Budeme z nich vycházet při návrhnu a konstrukci jednotky.



Obrázek 3.2: Rozběhový moment



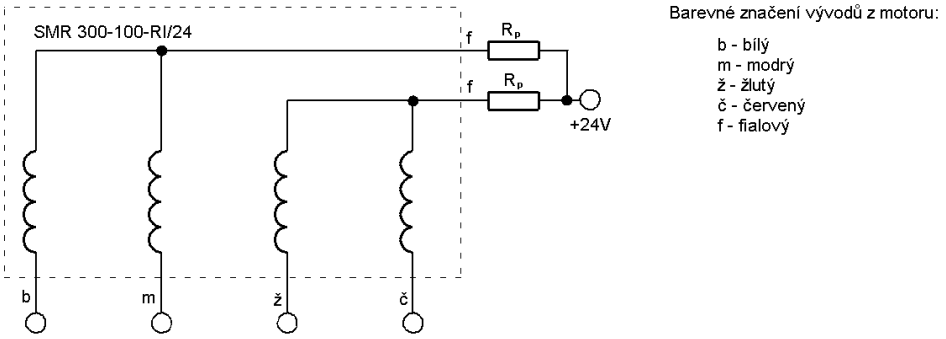
Obrázek 3.3: Rozběhový moment setrvačnosti

Tyto charakteristiky jsou uvedeny v katalogovém listu k motoru [8]. Momentové charakteristiky jsou pro nás velmi důležité. Rozběhový moment je popsán v podkapitole 2.4, rozběhový moment setrvačnosti se týka nesymetrické zátěže připojené na hřídel rotoru, viz podkapitola 2.3.3.

3.1.3 Zapojení motoru

Na obrázku 3.4 je vnitřní zapojení motoru SMR 300-100-RI/24. Vývody příslušných fází motoru jsou značeny barevně. Přilehlé fáze motoru mají jeden fialově označený společný vodič, na který se připojuje předřadný odpor R_p, který slouží k převedení tepelné ztráty ven z motorku. Jeho hodnota je doporučená výrobcem viz tabulka 3.1.

Motor je napájen stejnosměrným napětím 24V a jeho magnetické cívky jsou spínány na záporný pól zdroje.



Obrázek 3.4: Zapojení motoru

3.1.4 Technické údaje

Další informativní údaje převzané z katalogového listu k motoru viz [8].

Technické údaje	čtyřtaktní řízení	osmitaktní řízení
Počet kroků na 1 otáčku	40	80
Úhel kroku	9°	4, 5°
Tolerance úhlu kroku	0, 27°	0, 75°
Max. rozběhový kmitočet	280Hz	560Hz
Max. provozní kmitočet	800Hz	1700Hz
Statický zatěžovací úhel	2, 25°/10m Nm	2, 25°/5m Nm
Přídržný moment	28m Nm	22m Nm
Vlastní přídržný moment	2m Nm	
Amplituda proudu jedné fáze	0,25A	
Napájecí napětí	$24V$ ss ±10%	
Odpor vinutí fáze	30Ω ± 10%	
Indukčnost vinutí fáze	52mH	
Předřadný odpor Rp	$62\Omega/6W$	
Max. radiální zatížení ložiska	1N	
Maxi. axiální zatížení ložiska	1,5N	
Váha	0,65Kg	
Stabilizační zatěžovací moment	2m Nm	
Moment setrvačnosti rotoru	$13,8 \cdot 10^{-7}$Kg · m^2	
Rozběhový moment	10m Nm, 100Hz	5m Nm, 400Hz
Rozběhový moment setrvačnosti	100 ·10^{-7} Kg · m^2, 100Hz	100·10^{-7} Kg · m^2, 200Hz

Tabulka 3.1: Technické údaje

Provozní podmínky:

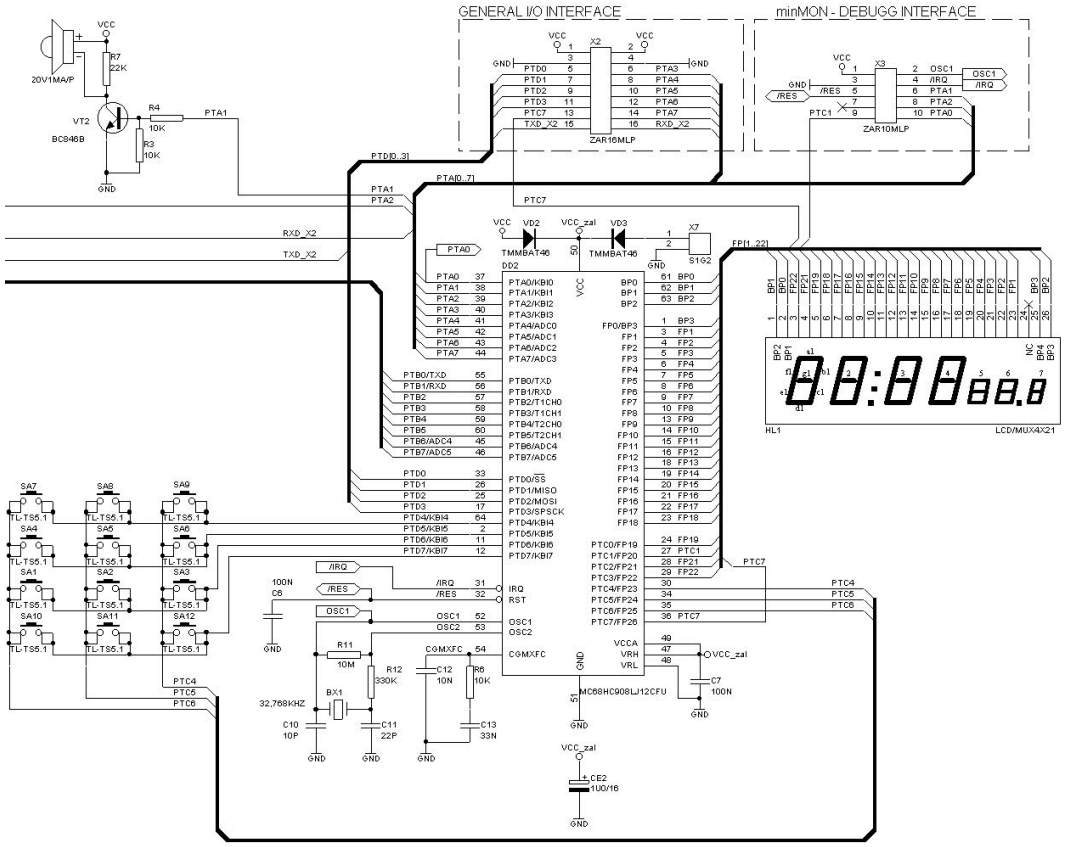
Prostředí	IE 33 dle ČSN EN 60721-3-3 v prachotěsném krytu
Odolnost chvění	0,1mm/50Hz
Teplota okolí	-20°C až +55°C
Pracovní poloha	libovolná

3.2 Vývojový kit 908LJ12-G2

V podkapitole je popsán kit 908LJ12-G2 na kterém je řídící jednotka vyvíjena. Jsou zde pouze základní charakteristiky a údaje nutné pro vývoj jednotky. Více je pak možno nalézt přímo v manuálu ke kitu na [16]. Samotnou prací s jednotlivými periferiemi se zabývá kapitola 4 a 5.

3.2.1 Popis periferií kitu

Srdcem vývojového kitu 908LJ12-G2 je mikrokontrolér MC68HC908LJ12 od firmy Motorola. Deska je napájena napětím 5V. Na fragmentu ze schématu zapojení kitu na obrázku 3.5, jsou zachyceny základní periferie, které jsou použity pro vývoj jednotky, proto se na toto schéma při návrhu jednotky budeme často odkazovat. Použité periferie jsou tedy klávesnice, displej a vstupní/výstupní porty (v dalším textu budou označovány jako I/O porty).



Obrázek 3.5: Schéma zapojení

Kit má samozřejmě mnohem více dalších možností. Jejich popis není předmětem této práce, ale jen ve stručnosti uvedeme základní vlastnosti:

- Levná vývojová platforma s rozhraním miniMON08 pro sériové programování a ladění
  - Kompatibilní s vývojovým prostředím Metrowerks CodeWarior pro HC08
  - Kompatibilní s vývojovým balíkem P&E ICS software tools

- SCI rozhraní
- Rozhraní RS232 pro komunikaci s PC
- LCD displej
- 2 LED diody
- Maticová klávesnice 3x4 tlačítek + dvě klasická tlačítka
- Teplotní čidlo
- Spínací výstup s relé
- Zvukový výstup v podobě malého reprodukorku
- Externí krystalový oscilátor 32,768KHz
- Velikost desky 122x60mm

Vstupní/výstupní rozhraní

Kit 908LJ12-G2 je vybaven několika vstupně/výstupními rozhraními, pomocí kterých může komunikovat s okolním světem. K připojení vnějších uživatelských periferií je určen především port s označením X2 na obrázku schématu 3.5 jako *GENERAL I/O interface*, viz tabulka 3.2. Na tento konektor jsou vyvedeny piny mikrokontroléru.

VCC	GND	PTA3	PTA4	PTA5	PTA6	PTA7	RXD-TX2
2	4	6	8	10	12	14	16
1	3	5	7	9	11	13	15
VCC	GND	PTD0	PTD1	PTD2	PTD3	PTC7	RXD-RX2

Tabulka 3.2: Popis signálů vyvedených na konektor X2

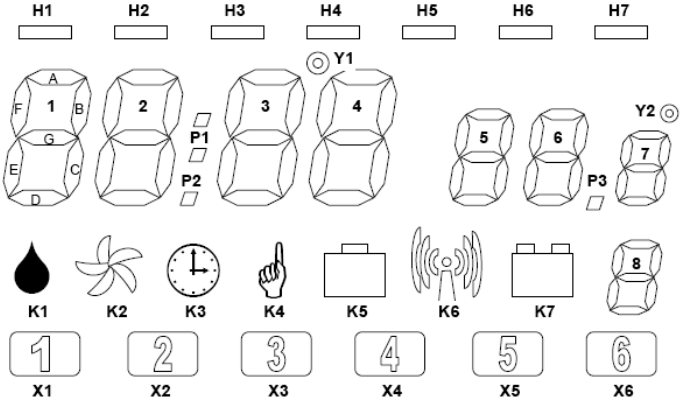
V naší konstrukci budeme využívat PTA[7:3] (pin 3 – pin 7 portu A mikrokontroléru). Připojení vnějších periferií je do značné míry omezeno maximálním výstupním proudem, který je mikrokontrolér schopen do daného signálu dodat viz 3.2.2. V jednodušších případech, kdy připojujeme na výstup zátěž s menším odběrem (např. LED dioda), není potřeba žádného zásahu do zapojení. V případě, kdy připojujeme zařízení jehož odběr je vyšší (krokový motor) než maximální výstupní proud, který je mikrokontrolér schopen na výstup dodat, musíme mezi procesor a zařízení umístit zesilovací prvek. Ten pak pro zařízení zajistí potřebné proudové a napěťové úrovně.

Klávesnice

Jako uživatelský vstup pro komunikaci s řídící jednotkou je použita klávesnice. Patří k nejjednodušším prostředkům pro vstup údajů do digitálních zařízení. Schéma na obrázku 3.5 ukazuje klávesnici dvanácti tlačítek zapojenou do matice 3x4 tlačítka (SA1–SA12). Vývody jsou pak zapojeny přímo na I/O porty mikrokontroléru, sloupce k PTC[6:4] a řádky k PTD[7:4]. Detekce konkrétní stisknuté klávesy je pak dána souřadnicí aktuálního řádku a sloupce klávesy.

LCD Displej

Kit 908LJ12-G2 obsahuje i displej z tekutých krystalů, připojený přímo na vývody mikrokontroléru. Ze schématu na obrázku 3.5 je zřejmé, že řízení displeje probíhá po čtyřech tzv. *backplane* (BP) a dvaadvaceti *frontplane* (FP) signálech. Má 81 segmentů a je zapojen v módu *1/4 duty cycle* což je poměr doby vyhrazené k řízení jednoho segmentu k době trvání jednoho celého rámce řízení. Životnost tekutých displejů závisí na střídavém elektrickém poli, kterým musí být napájeny.



Obrázek 3.6: LCD displej.

Obrázek 3.6 představuje jednotlivé segmenty displeje na kitu. Každý z nich má svoje označení, segmenty cifer se pak označují číslem a písmenem (např. 2B – druhá cifra, segment B). Popisovaný displej neobsahuje žádný podpůrný obvod (řadič), který by zajišťoval střídavé elektrické pole pro napájení jednotlivých segmentů. Tuto fukci plní mikrokontrolér obsažený v kitu, který je tímto LCD řadičem vybaven. Řízení samotných signálů pro displej je tak pro programátora mikrokontroléru zcela transparentní, proto se zde nebudeme zabývat princip řízením LCD displejů. Více k úvodu do řízení LDC displejů můžete najít v [6]. Pokud řadič displej obsahuje, hovoříme o tzv. *LCD modulu*, který je pro řízení jednodušší, ovšem jeho cena je díky podpůrným obvodům mnohem vyšší.

3.2.2 Mikrokontrolér MC68HC908LJ12CFU

Kit 908LJ12-G2 je založen na mikrokontroléru MC68HC908LJ12CFU. Zde jsou uvedeny hlavní rysy tohoto mikrokontroléru od firmy Motorola a popsány některé vestavěné moduly, které byly použity pro návrh řídící jednotky. Bližší popis je možno najít přímo v dokumentaci k mikrokontroléru [7].

Základní charakteristiky

Použitý mikrokontrolér patří do rodiny mikroprocesorů motorola HC08. Tyto osmibitové mikrokontroléry plně postačují pro menší aplikace. Výhodná je jednodušší struktura a relativně malý počet vývodů. Mikrokontrolér je určen pro všeobecné použití.

- 12KB FLASH paměti umístěné přímo na čipu s podporou programování přímo v obvodu (ISP)
- 512B paměti RAM umístěné přímo na čipu

- CISC instrukční sada
- Architektura Von Neumann
- Periferie jsou mapovány do paměti
- Řadič displeje (4 backplanes s maximálně 27 frontplanes řídících signálů)
- SCI, SPI
- Zabudovaný modul hodin reálného času (RTC)
- ADC převodník, 6 kanálů, 10-ti bitové rozlišení
- Dva 16-ti bitové dvoukanálové čítače/časovače
- Až 32 vstupně/výstupních pinů
- 64-pinové plastikové pouzdro QFP

Elektrické parametry

Základní a krajní elektrické parametry důležité při konstrukci řídící jednotky, které musíme explicitně zajistit. Zde popisovaná varianta MC68HC908LJ12 je určena pro napájecí napětí 5V.

Charakteristika	Symbol	Hodnota	Jedn.
Napájecí napětí	V_{DD}	od -0.3 do $+6.0$	V
Vstupní napětí	V_{IN}	od $V_{SS} - 0.3$ do $V_{DD} + 0.3$	V
Max. proud na I/O pin	V_{IN}	± 25	mA
Max. proud z V_{SS}	I_{MVSS}	100	mA
Max. proud do V_{DD}	I_{MVSS}	100	mA
Skladovací teplota	T_{STG}	-55 do $+150$	$C°$

Tabulka 3.3: Krajní elektrické parametry MC68HC908LJ12

Charakteristika	Symbol	Hodnota	Jedn.
Provozní nap. napětí	V_{DD}	$+5.0 \pm 10\%$	V
Provozní teplota	T_A	-40 do $+85$	$C°$

Tabulka 3.4: Provozní elektrické parametry MC68HC908LJ12

Poznámka: všechna napětí se vztahují k potenciálu V_{SS}.

3.3 Výkonový budič fází motoru

Srovnáme-li elektrické parametry krokového motoru SMR 300-100-RI/24 z tabulky 3.1 a elektrické parametry I/O portů mikrokontroléru MC68HC908LJ12 z tabulky 3.3, jsou na první pohled zřejmé jejich odlišnosti v napěťových a proudových úrovních. Motor při napájecím napětí 24V má při jedné sepnuté fázi odběr 250mA. Mikrokontrolér je však na jeden výstupní pin schopen poskytnou napěťovou úroveň 5V a výstupní proud maximálně 25mA což je naprosto nedostačující. Mezi mikrokontrolér resp. kit a motor je nutno vsadit výkonový prvek, který bude převádět napěťové a proudové urovně z 5V na 24V.

Firma, která vyrábí popisovaný krokový motor [10], k němu dodává i hotový rozdělovač impulsů RI 250-24-4/8. Tento rozdělovač umožňuje nastavení všech základních funkcí pro řízení motoru – čtyřtaktní nebo osmitaktní režim řízení, reverzaci a zastavení v libovolné poloze. Jako vstupní signál mu slouží obdélníkové impulsy v napěťových úrovních TTL (rozdělovač je sestaven z hradel a klopných obvodů). Na výstup tohoto rozdělovače již můžeme připojit samotné fáze motoru, které pracují s napěťovou úrovní 24V. Výstup má čtyři svorky, stejně jako počet fází. Výstupním signálem jsou obdélníkove impulsy vhodné pro řízení motoru v nastaveném módu, tzn. s každým dalším impulsem na vstupu motor udělá právě jeden krok.

Do návrhu řídící jednotky byl původně tento rozdělovač zahrnut. Nastaly však potíže při konstrukci jednotky, kdy rozdělovač nastavený na 8-mi taktní řízení otáčel motorem střídavě po kroku $4,5°$ a $9°$, což je nepřípustné. Rozdělovač byl vyzkoušen na více generátorech impulsů, ale vždy se stejným výsledkem. Zřejmě se jednalo o pokus vyrovnání se rozdělovače s nestabilitami v řídícím kmitočtu, nebo jen o špatný kus. Proto byla vyvinuta silová část popsaná dále, která tímto neduhem netrpí.

Více k rozdělovači RI 250-24-4/8 se můžete dočíst v jeho katalogovém listu v [11].

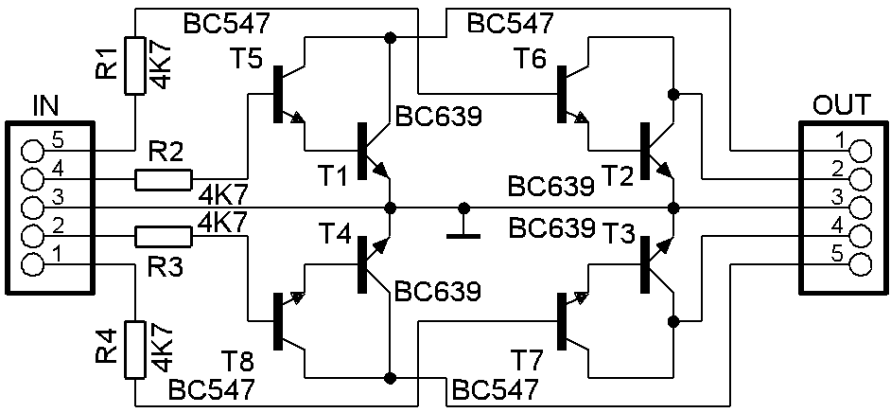
3.3.1 Požadavky na budič fází

Výkonová část musí být schopna pro každou fázi spínat proudy o velikosti amplitudy proudu do jedné fáze, s maximální frekvencí řízení motoru (viz tabulka 3.1). Ovládání spínání musí pracovat na úrovních 5V a vstupní proud do rozdělovače musí být menší než maximální výstupní proud, který je schopen výstup z kitu resp. z mikrokontroléru dodat. Každá fáze tedy bude řízena zvlášť jedním signálem.

Všem těmto požadavkům musí být podřízeny i hodnoty součástek použitých pro konstrukci budiče.

3.3.2 Návrh budiče

Celý výkonový budič fází motoru je sestaven z diskrétních součástek. Na obrázku 3.7 je zobrazeno navržené schéma zapojení.



Obrázek 3.7: Schéma zapojení budiče

Pro vybuzení každé fáze jsou zde vždy dvojice tranzistorů v spojené do tzv. darlingtonového zapojení. Tranzistory T4–T8 slouží jako budiče koncových stupňů T1-T4. Jednou

z vlastností darlingtonova zapojení je, že má velké proudové zesílení, které se rovná součinu proudových zesílení obou tranzistorů. Tzn. pro vybuzení stačí již velmi malý proud na vstupu.

Funkce zapojení dále bude popsána na první větvi tranzistorů T1 a T5. Zapojení funguje tak, že pokud výstupní svorku 1 konektoru OUT připojíme na kladné napětí zdroje (napájení motoru) $+U$ přes zátěž (fáze motoru), objeví se na kolektorech T1 a T5 napětí U_{CET1} rovné tomuto kladnému potenciálu. Záporný pól tohoto napájení musí být samozřejmě připojen na svorku 3 konektoru OUT.

Pokud nyní přivedeme kladné napětí U_{IN} z výstupu kitu. resp mikrokontroléru na svorku 4 konektoru IN, začne přes odpor R2 do bazé tranzistoru T5 procházet proud I_{BT5}. T5 se otevře, čímž z kolektoru na bázi T1 přivede napájecí napětí vnější zátěže (motoru) $+U$. Tím se plně otevře i tranzistor T1, jeho napětí mezi emitorem a kolektorem U_{CET1} výrazně klesne a na zátěž je přivedeno napajecí napětí $+U$ ochuzené o úbytek na U_{CET1}.

Tranzistory jsou běžně používané, jejich parametry byly voleny s ohledem na zapojení. Tranzistory T1–T4 slouží jako koncové stupně. Katalogový list k BC639 uvádí jeho maximální kolektorový proud I_C 1A a jeho maximální U_{CE} 100V, což pro náš případ bohatě postačuje, viz tabulka parametrů motoru 3.1. Úbytek napětí na přechodu emitor–kolektor při plně otevřeném tranzistoru je asi 0,5V. Parametry budících stupňů T5 až T8 už nejsou tolik kritické, lze zde použít téměř libovolné tranzistory pro menší výkony.

Spínací proud odebíraný z kitu resp. mikrokontroléru pak můžeme určit následovně:

$$I_{BT5} = \frac{U_{IN} - U_{BET5} - U_{BET1}}{R2} = \frac{5 - 0,7 - 0,7}{4700} \doteq 7,65 \cdot 10^{-4} A$$

Hodnoty U_{BET5} a U_{BET1} jsou vyčteny z katalogových listů tranzistorů BC547 [12] a BC639 [13]. Vezmeme-li v úvahu, že maximální proud který dokáže jeden I/O pin mikrokontroléru do vnější zátěže dodat je 25mA, tak vypočtená hodnota je cca 32krát menší.

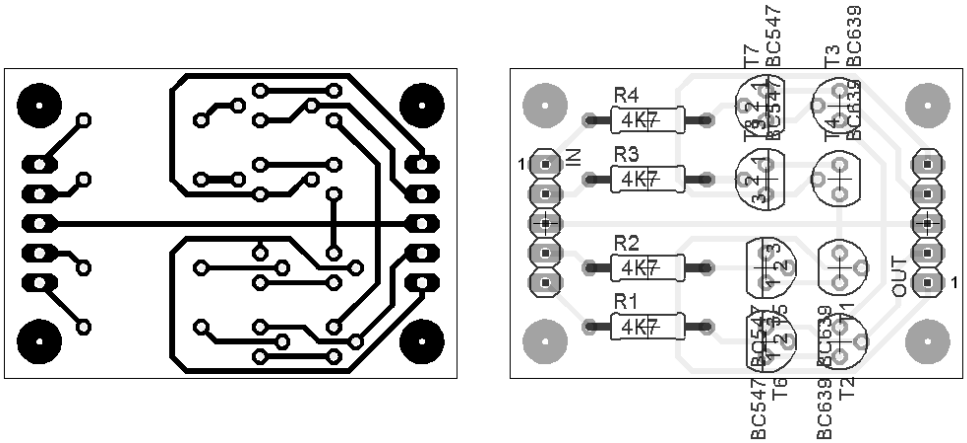
3.3.3 Konstrukce budiče

Pro použití budiče v praxi je vhodné součástky umístit na desku plošného spoje. Tato deska je navržena jako jednostranná, nenáročná na výrobu viz obrázek 3.8. Schéma a obraz desky plošného spoje ve formátu programu *Eagle Layuout Editor* jsou umístěny v příloze na datovém nosiči k této práci.

	Konektor	
Popis	IN	OUT
Fáze 1	1	4
Fáze 2	2	5
GND	3	3
Fáze 3	4	1
Fáze 4	5	2

Tabulka 3.5: Vstupní a výstupní svorky budiče

Tabulka popisuje vždy vstupní a k ní příslušnou výstupní svorku ovládané fáze na jednom řádku. Tento popis a popis výstupů vývojového kitu a krokového motoru budou potřebné při propojování komponent do funkčního celku při návrhu kontrukce.



Obrázek 3.8: Deska plošného spoje a její osazení

Seznam součástek:

R1–**R4** 4K7
T1–**T4** BC639
T5–**T8** BC547
IN, OUT lámací konektorové kolíky RM2,54mm

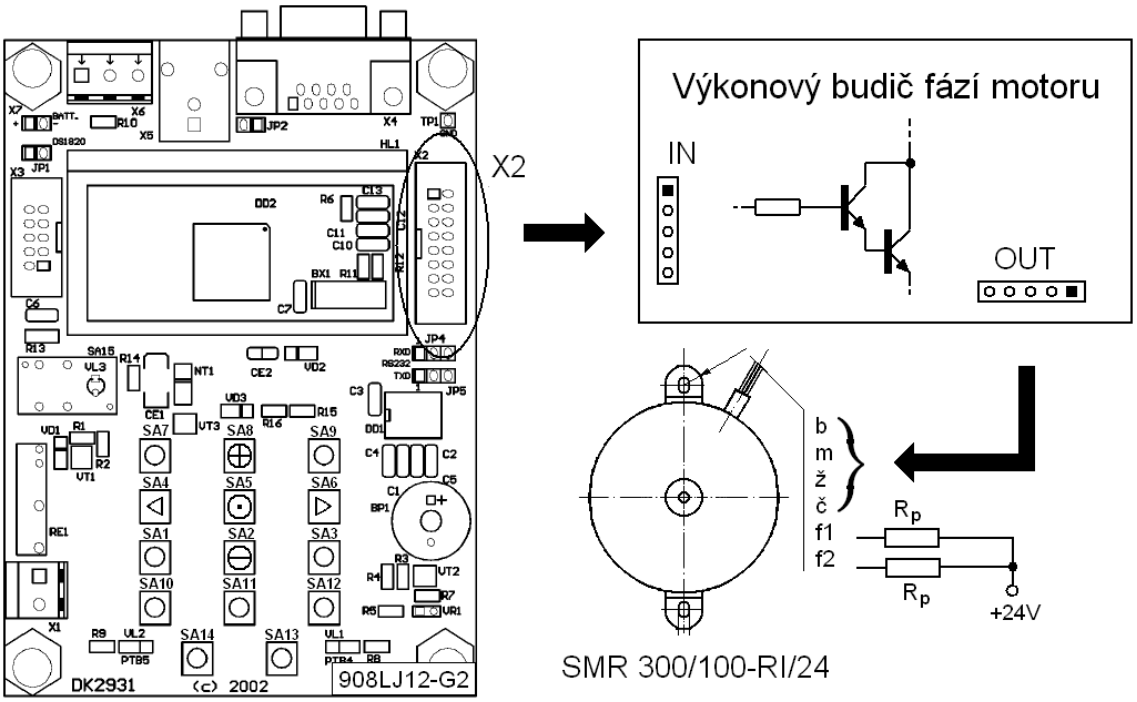
Kapitola 4

Návrh řídící jednotky

Zadání této práce specifikuje jednotku, která umožňuje řídit krokové motory s využitím popsaných součástí v kapitole 3 a jejich případných periferií. Tato kapitola se zabývá vlastním návrhem a propojením jednotlivých komponent. Dále pak vývojem, komponenty potřebnými k vývoji a z toho vyplývajícími specifiky.

4.1 Blokové schéma a propojení do funkčního celku

Blokové schéma na obrázku 4.1 popisuje způsob propojení jednotlivých komponent.



Obrázek 4.1: Blokové schéma řídící jednotky krokového motoru

Pro výstup řídících signálů byly zvoleny piny PTA[6:3] mikrokontroléru. Jejich popis a vyvedení na konektor kitu je popsán v tabulce 3.2. V následující tabulce 4.1 je popsáno

fyzické propojení očíslovaných pinů konektorů jednotlivých komponent řídící jednotky do funkčního celku.

	Kit	Budič		Motor
Popis	X2	IN	OUT	Vodiče
Fáze 1	6 (PTA3)	1	4	modrý
Fáze 2	8 (PTA4)	2	5	bílý
Fáze 3	10 (PTA5)	4	1	červený
Fáze 4	12 (PTA6)	5	2	žlutý
GND	2 (GND)	3	3	

Tabulka 4.1: Fyzické propojení jednotlivých částí

Motor je napájen přes předřadné odpory R_p ze zdroje 24V, podle schématu na obrázku 3.4. Zaporný pól zdroje napájení motoru (zem) je pak připojen na pin 3 výstupního konektoru OUT budiče fází.

4.2 Ovládání jednotky

4.2.1 Klávesnice

Jednotka využívá pro uživatelský vstup maticovou klávesnici dostupnou na kitu. Ta nabízí celkově dvanáct tlačítek, což je pro tuto aplikaci poměrně zbytečné. Pro zajištění jednoduchého a intuitivního ovládání je jich využito pouze pět. Jsou to následující tlačítka s označením (viz schéma kitu na obrázku 3.5 a blokový diagram na 4.1) a ve funkcích uvedených v tabulce 4.2.

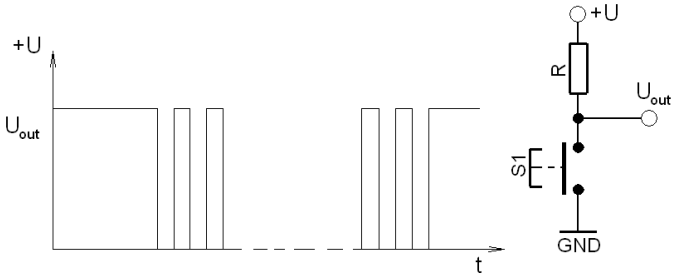
Název	Sch. označení	Popis	Symbol
KB_SET	SA5	potvrzovací tl.	⊙
KB_LEFT	SA4	pohyb v menu	▷
KB_RIGHT	SA6	pohyb v menu	◁
KB_PLUS	SA8	nastavení parametrů	⊕
KB_MINUS	SA2	nastavení parametrů	⊖

Tabulka 4.2: Funkce tlačítek

Symbolická značka tlačítka slouží jako zkratka jeho identifikaci v dalším textu. Podle blokového schématu na obrázku 4.1 si můžete všimnout, že tlačítka jsou upořádána do kosočtverce, s potvrzovacím tlačítkem uprostřed, tohle je poměrně častý způsob používaný ve vestavěných zařízeních. Název tlačitek je stejný jako jejich symbolické názvy používané ve zdrojovém textu programu.

Protože jsou tlačítka mechanického původu, je třeba počítat s odskakováním jejich kontaktů. Tyto zákmity trvají řádově jednotky až desítky milisekund, rychlý procesor však pracuje v řádu mikrosekund a proto se pro něj všechny tyto odskoky jeví jako několikanásobný stisk/uvolnění.

Tento jev se může objevovat jak při stisku, tak při uvolnění a je potřeba s ním počítat. Na obrázku 4.2 je naznačen časový průběh zákmitů. V této konstrukci ošetřuje zákmity tlačítek program v mikrokontroléru viz kapitola implementační část 5.



Obrázek 4.2: Průběh zákmitů při stisku nebo uvolnění tlačítka

4.2.2 Uživatelské vstupy

Uživatelské vstupy jsou parametry otáčení motoru, které uživatel zadává přes klávesnici řídící jednotce a podle kterých pak motor pracuje. Podle zadání je nutná možnost nastavení následujících parametrů:

- Režim otáčení
  - Trvalé otáčení
  - Možnost zadat úhel o který se motor pootočí
- Úhel otočení - úhel ve stupních o který hřídel motoru při dalších nastavených parametrech pootočí
- Rychlost otáčení - bude zadávána v otáčkách za sekundu (ot/s)
- Směr otáčení - možnost reverzace chodu motoru
- Typ řízení - čtyřtaktní nebo osmitaktní režim

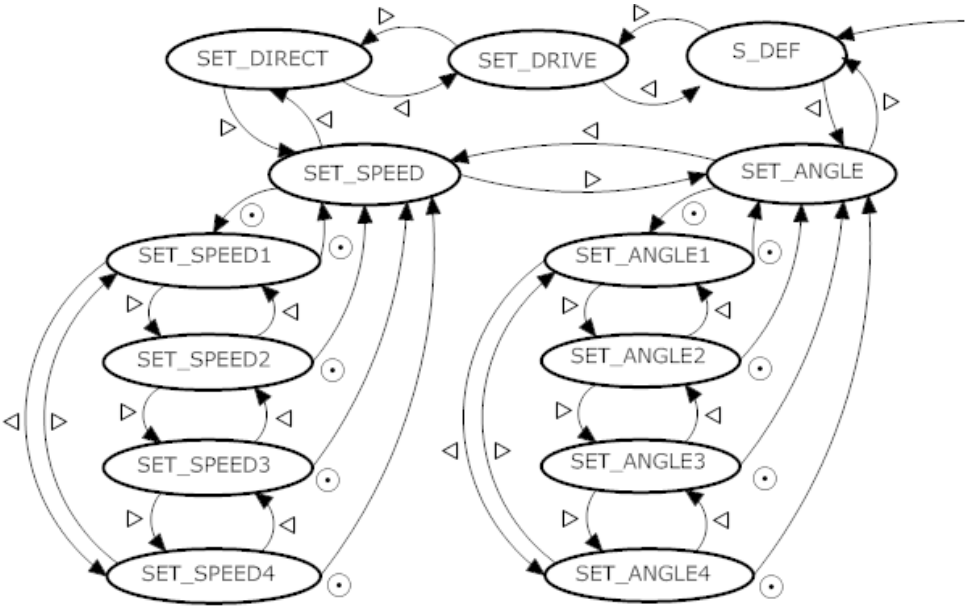
Tyto zadávané hodnoty jsou omezeny parametry motoru, proto musí být uživatelské vstupy programem kontrolovány a případně upravovány, aby byly korektní s vlastnostmi daného motoru. Zadávání těchto hodnot probíhá podle schématu stavového automatu, viz následující podkapitola.

4.2.3 Stavový automat

Pro snadné ovládání jednotky a nastavování vstupních parametrů bylo zvoleno jednoduché menu. Pohyb v menu je znázorněn pomocí stavového automatu. Pro přechod z jednoho stavu do dalšího je nutná vnější akce, tzn. stisk jednoho z pěti ovládacích tlačítek popsaných v podkapitole 4.2.1. Na obrázku 4.3 odpovídají názvy stavů označením těchto stavů ve zdrojových textech programu (kapitola 5).

Je zde pět základních stavů menu (S_DEF, SET_DRIVE, SET_DIRECT, SET_SPEED, SET_ANGLE) a vedlejší stavy pro nastavení vstupních hodnot rychlosti a úhlu otáčení (SET_SPEEDx, SET_ANGLEx). Přechody mezi stavy jsou označeny symbolem příslušného tlačítka podle tabulky 4.2. Pro přehlednost nejsou v automatu na obrázku 4.3 zakresleny přechody tlačítek $\oplus$ a $\ominus$, které v daném stavu pouze nastavují zvolený parametr motoru. V tabulce 4.3 je význam jednotlivých stavů popsán i s funkcí tlačítek $\oplus$ a $\ominus$.

Výchozí stav je S_DEF ve kterém lze navíc pomocí tlačítka $\odot$ spustit nebo zastavit motor. Zvláštní roli zde hrají stavy nastavující čtyřmístná čísla rychlosti a úhlu SET_SPEEDx a SET_ANGLEx, ve kterých se stiskem $\oplus$ resp. $\ominus$ k cifře x přičítá resp. odečítá jednička.



Obrázek 4.3: Stavový automat pohybu v menu

Stav	Popis	⊕	⊖
S_DEF	výchozí stav	režim otáčení	režim pootočení o úhel
SET_DRIVE	řízení	osmitaktní	čtyřtaktní
SET_DIRECT	směr	vpravo	vlevo
SET_SPEED	rychlost		
SET_ANGLE	úhel		
SET_SPEEDx	x-tá cifra rychlosti	inkrementace	dekrementace
SET_ANGLEx	x-tá cifra úhlu	inkrementace	dekrementace

Tabulka 4.3: Popis stavů

4.3 LCD displej

V každém stavu displej zobrazuje příslušné údaje podle automatu na obrázku 4.3 jako zpětnou vazbu k uživateli řídící jednotky. Popis segmentů displeje je na obrázku 3.6 v podkapitole 3.2.1. Jen ve stručnosti význam jednotlivých segmentů:

X1-X6 – indikace jednoho z pěti základních stavů nebo stavy při nastavování cifer

K2 – mód stálého otáčení

K3 – mód pootočení o zadaný úhel

Číslovka 5 – chod vpravo/vlevo

Číslovka 8 – čtyřtaktní nebo osmitaktní řízení

Číslovky 1-4 – ve stavu S_DEF zobrazuje aktuální úhel natočení rotoru (0-360°), ve stavech SET_SPEEDx a SET_ANGLEx pak hodnotu nastavované hodnoty, v ostatních případech pak rychlost otáčení

4.4 Rozhranní MON08 a jeho specifika

Mikrokontrolér MC68HC908LJ12 poskytuje možnost poměrně komfortně ladit programy. Pro samotné odladění programu bylo použito rozhraní kitu miniMON Debug Inerface vyvedené na jeho konektor X3 (schéma 3.5), sloužící pro pohodlné ladění aplikací přímo v běžícím obvodu. Mikrokontrolér pak funguje v tzv. módu *monitor*, kdy je možno pomocí počítače krokovat jednotlivé instrukce programu, měnit obsah registrů procesou atd. a to přímo za běhu programu. Všechna tato data se pak zobrazují v počítači, kde je lze analyzovat a pracovat s nimi. Ladění je tedy stejné jako simulace práce mikrokontroléru v počítačovém programu.

Pro ladění programu pomocí MON08 je třeba nastavit speciální napětí na některých pinech procesoru, a zavést vnější hodinový signál. O všechny tyto věci se postará vnější obvod, který se připojí na MiniMON Debug Inerface. Rozhraní MiniMON, které kit 908LJ12-G2 je standardizované. Jako vývojový a programovací adaptér připojený na X3 je použit vývojový kit Janus. Jedná se o jednoduchý nástroj pro seznámení se s problematikou mikrokontrolérů HC08, konkrétně nejmenší zástupce rodiny HC08 tzv. Nitrony. Více k tomuto vývojovému prostředku lze nalézt v [14]. Ke kitu Janus je pak přes sériové rozhraní RS232 připojeno PC, pomocí kterého lze ladění provádět.

Rozhranní MON08 je kompatibilní s vývojovým prostředím Freescale Code Warior, které bylo použito při překladu ladění kódu.

4.5 Časování systému

Jak již bylo řečeno, při ladění programu pomocí módu MON08 je nutno mikrokontroléru nastavovat vnější hodinový signál, ten zajišťuje právě kit Janus, obsahuje jednoduchý oscilátor s kmitočtem 9,8304MHz. Tato frekvence se v modulech CGM a SIM procesoru, na kterých je závislá frekvence sběrnice BUSCLK dále dělí v našem případě čtyřmi viz [7]. Frekvence sběrnice procesoru BUSCLK je stanovena na **2,4576MHz**. Tím pak bude dáno veškeré časování v programu, posílání řídících impulsů do krokového motoru apod.

Pokud pak program budeme používat v samostatné aplikaci bez módu MON08, je nutno upravit časování mikrokontroléru. O principech časování se můžete více dočíst v knize [5] ze které tato podkapitola čerpá.

Hodinový signál má na starosti modul CGM (Clock Generator Module) mikrokontroléru. Skládá se z vlastního oscilátoru a obvodů, které s touto frakvencí pracují a upravují ji. Je zde několik možností jak dosáhnout požadované frekvence hodinových impulsů a protože mikrokontrolér MC68HC908LJ12 obsahuje variantu použí smyčky fázového závěsu (PLL), stačí použít krystal 32KHz dostupný na přímo kitu a tuto frekvenci pomocí PLL „vytáhnout“ na požadovanou úroveň.

PLL může generovat hodinový signál poměrně ve velkém rozsahu. K tomuto je použit vnější referenční hodinový signál. Avšak čím větší je požadovaná frekvence, tím chyba generovaného kmitočtu roste i s minimálníi odchylkami od referenčního taktu. Frekvenci generovaného signálu lze vypočítat pomocí vzorce:

$$f_{VCLK} = \frac{2^P \cdot N}{R} \cdot f_{RCLK}$$

Zde je f_{VCLK} naše požadovaná frekvence. Kde f_{RCLK} je referenční frekvence, v našem případě krystal 32KHz umístěný na kitu. Hodnoty P, N a R se nastavují v registrech modulu CGM. Hodnota dělícího poměru bínárního děliče P je nastavována na bitech 2 a

3 registru *PTCL* (bity *PRE0* a *PRE1*). Násobitel frekvence N se nastavuje v registrech *PMSH* a *PMSL*, hodnota může být v rozsahu 0-4095. Dělící poměr předděliče referenčního kmitočtu R je stanovena ve spodních čtyřech bitech registru *PMDS*.

Pro naš případ, kdy potřebujeme zvýšit frekvenci na 9,8304MHz (tato je pro frekvenci sběrnice BUSCLK ještě modulem CGM a SIM vydělena čtyřmi), tedy na na frekvenci stejnou kterou generuje signál kitu Janus při ladění v módu MON08, může být stanovení hodnot pro registry *PTCL*, *PMDS*, *PMSH* a *PMSL* hodnoty $R = 1$, $P = 0$, $N = 300$.

Rozsah použitelnosti řídící jednotky je omezen především frekvencí hodin s jakou je mikrokontrolér taktován. S tím souvisí maximální řídící výstupní kmitočet. V našem případě, kdy je maximální frekvence řízení motoru SMR 300-100-RI/24 v osmitaktním režimu 1700Hz, musí být hodinový takt mikrontroléru řádově vyšší, režijní náklady (i z důvodu psaní programu v jazyku C) jsou poměrně vysoké.

Kapitola 5

Implementační část

Kapitola popisuje implementaci programu pro mikrokontrolér MC68HC908LJ12 na základě návrhu popsaném v kapiptole 4.

5.1 Programování mikrokontrolérů v jazyku C

Technologie mikrokontrolérů prošla za poslední desetiletí bouřlivým vývojem, z toho vyvstaly požadavky na efektivní a pohodlné psaní programů. Pro mikrokontroléry byl především určen *Jazyk symbolických adres* (assembler), který skýtá své výhody především v rychlosti programu, kdy jeden řádek psaného programu odpovídá jedné instrukci procesoru a v paměťové nenáročnosti kdy programátor spravuje využití paměti RAM vlastními prostředky. Kód assembleru bývá přímo přeložen do strojového kódu mikrokontroléru. Tento způsob je velmi rychlý a efektivní ale nehodí se pro větší projekty. Přehlednost kódu v assembleru je nízká a bývá složité takový kód analyzovat.

Zde přicházejí na scénu vyšší programovací jazyky, konkrétně jazyk C, ve kterém je také program pro řídící jednotku napsán. Programování ve vyšším programovacím jazyku pro mikrokontrolér není zcela identické s psaním programu pro klasický stolní počítač. Stejně jako má mikrokontrolér odlišnosti od klasických mikroprocesorů užívaných v PC, tak má i program pro mikrokontrolér svá specifika, která je nutno dodržovat. Jde především o paměťové nároky a rychlost, kdy velikost pamětí a rychlost taktování mikrokontrolérů bývá řádově nižší.

Nicméně v dnešní době technologií poměrně výkonných mikropočítačů jsou tyto argumenty neopodstatněné a překladače vyšších programovacích jazyků pro mikrokontroléry jsou běžně dostupné. Tyto překladače bývají velmi výkonné jsou schopné optimalizovat kód na maximální možnou míru. Překladač musí realizovat i režijní část programu (rozmisťování dat a kódu, volání podprogramů, adresování apod.) o kterou se však programátor již nemusí starat. Má-li však programátor k dispozici mikropočítač s omezenými zdroji (např. malá paměť RAM), je vhodné použití vyššího programovacího jazyka pro vývoj aplikace zvážit.

Kompilátory pro jazyk C zpravidla nabízí možnost řídit optimalizaci a její stupeň. Programátor si tak může efektivnost kódu řídit v závislosti na tom, zda potřebuje rychlý program, nebo malou velikost výsledného kódu. Je také velmi důležité znát architekturu mikrokontroléru pro který kód píšeme. Architekrura a konstrukce psaného kódu jsou na sobě tedy úzce závislé a tentýž kód kompilovaný pro dvě různé architektury může byt velikostně a rychlostně značně odlišný.

Není v silách této práce rozebírat problematiku psaní programů pro mikrokontroléry ve

vyšších programovacích jazycích, nicméně velmi zajimavý úvod do psaní programů v C pro mikrokontroléry můžete najít v [1].

V kapitole jsou často pro přehlednost návrhu implementace zobrazeny segmenty kódu. V příloze 1 lze nalézt programovou dokumentaci a veškeré soubory se zdrojovými kódy na které se budeme v této kapitole odkazovat.

5.2 Struktura programu

Vlastní program pro řídící jednotku je rozdělen do několika funkčních modulů, které spolu vzájemně komunikují.

5.2.1 Hlavní programová smyčka

Hlavní programová smyčka je umístěna v souboru *Main.c*, kde se po inicializaci všech potřebných programových částí zacyklí a odchytává zprávy od modulu klávesnice nebo motoru (velmi zjednodušená obdoba WinAPI).

Počáteční inicializace

Po startu programu proběhne prvotní inicializace mikrokontroléru. Běh samotného jádra nastavíme v jeho registrech *CONFIG1*, kde vypneme modul COP (watchdog), který pro náš účel není důležitý, protože se jedná o interakivní aplikaci.

Dále je pak potřeba inicializovat komponenty systému zasílání zpráv, modulu KBI, LCD displeje a krokového motoru. Tyto části jsou popsány v dalším textu.

Nekonečná smyčka

V hlavní programové smyčce se odchytávají zprávy, které se potom vyhodnocují a provádějí se podle nich příslušné akce. Všechny zdrojové kódy jsou dostupné v příloze 1.

```
for(;;) {
  ptr=PopMessage(); // vyzvednutí zprávy
  if (ptr){   // jestliže je fronta neprázdná
    if (ptr->Src == KEYB){    // zpráva od klávesnice
      stateAutomaton((unsigned char)ptr->Data);
    }
    else if (ptr->Src == MOTOR && state==S_DEF) {  // zpráva od motoru
    if (MCon.Mode==MODEA)      // zobrazení aktuálního úhlu natočení
        displayInSet(ptr->Data,X1,K2);
      else if (MCon.Mode==MODEB)
        displayInSet(ptr->Data,X1,K3);
    }
  }
}
```

Fragment kódu ukazuje hlavní smyčku, kde se čeká na příchod zprávy. Viz systém zpráv v následující podkapitole.

5.2.2 Systém zpráv

Aby bylo možné jednoduše zachytávat a následně vyhodnocovat informace z více komunikujících periferií, je použito událostní programování. Jednotlivé komponenty tak můžou kdykoliv zaslat zprávu, která se uloží do fronty zpráv kde bude čekat na vyhodnocení. Výhodou tohoto systému je, že komponenta může zaslat zprávu v jakémkoliv okamžiku a jejich pořadí je jasně dáno pořadím zpráv ve frontě. Tohoto systému se využívá především při přerušeních, kdy podprogramy přerušení ukládají do fronty příslušné zprávy a hlavní programová smyčka je pak ve volném čase odchytává. Není tak problém např. v momentu, kdy právě probíhá obsluha displeje a motor zašle zprávu, že se pootočil o jeden krok. V tomto případě se dokončí obsluha periferie a jakmile se řízení vrátí zpět do hlavní smyčky, zprávu si vyzvedne a vyhodnotí.

Inicializace modulu zpráv

Prvotní nastavení modulu zpráv zahrnuje alokaci fronty v paměti a nastavení ukazatelů potřebných k operacím s polem pomocí kterého je fronta implementována.

Struktura zasílaných zpráv

Zde je naznačena struktura zprávy. Jedná se fragment kódu ze souboru *interface.h*.

```
typedef struct {
  unsigned char Src;    // kód zařízení, které zprávu zaslalo
  unsigned int Data;    // zaslaná data ke zpracování
} MESS;
```

Fronta zpráv je implementována pomocí pole a její velikost je pevně daná. Pokud je fronta zpráv již zaplněná, je nově příchozí zpráva jednoduše zahozena bez možnosti návratu. To není samoúčelné, zamezuje to přehlcení programu. Rozhraní pro systém zpráv je implementováno v souboru *interface.c*.

5.2.3 Modul klávesnice – KBI

Mikrokontrolér MC68HC908LJ12 obsahuje systém pro řízení maticové klávesnice, tzv. KBI modul. Ten poskytuje osm nezávislých vnějších maskovatelných přerušení, které jsou přístupné přes piny PTA[3:0] a PTD[7:4]. Na schématu kitu v obrázku 3.5 je vidět, že klávesnice je připojena právě na PTD[7:4], z toho taky musí vycházet veškeré nastavení modulu KBI. Více ve zdrojovém souboru *kbi.c*, kódy jednotlivých kláves lze nalézt v *kbi.h*.

Inicializace

Inicializace klávesnice lze provést nastavením příslušných bitů v registru *KBSCR*, kde vynulujeme bit *IMASKK* aby přerušení nebylo maskované a bit *MODEK* pro reakci modulu KBI na sestupnou hranu. V registru *KBIER* nastavíme bity KBIER[7:4], které připojí pull–up rezistory k pinům procesoru PTD[7:4] a povolí přerušení. Výstupní piny PTC[6:4] se vynulují. Tak se při stisku některého z tlačítek dostane na jeden z pinů PTD[7:4] logická nula, což vyvolá přerušení. Další podrobnosti k nastavení KBI modulu se lze dočíst v [7].

Ošetření mechanických zákmitů

V podkapitole 4.2.1 je zmínka o neduhu mechanických kontaktů při spínání nebo rozpínání, kdy kontakt před tím než se ustálí do klidové polohy ještě několikrát odskočí. Tohle je ošetřeno funkcí *Flasback_delay()* v souboru *kbi.c*, která provádí několik desítek čtení portu klávesnice v určitých intervalech za sebou. Pokud během toho dojde ke změně signálů na portu (zákmitu), začne se čtení provádět znovu tak dlouho, dokud nebude stav portu ustálený.

Detekce stisknuté klávesy

Při stisku některého z tlačítek klávesnice dojde k přenesení logické nuly na jeden z pinů procesoru PTD[7:4]. Modul KBI, který je nastaven na aktivaci přerušení při sestupné hraně vyvolá podprogram přerušení. Ten po ošetření zákmitů kontaktů tlačítek začne vlastní identifikaci stisknuté klávesy. Více je zřejmé z fragmentu zdrojového kódu ze souboru *kbi.c*, funkce *KBI_Keypressed()*.

```
  col = 0b01000000;
  if (Flasback_delay()==TRUE) {  // ošetření zákmitů
    for (i=0; i<COLS; i++){
      PTC |= 0b01110000;
      PTC &= ~col;              // přepínání sloupců
      if ((PTD & 0xF0) != 0xF0) {
        kbcode = (col & 0xF0) | ((PTD>>4) & 0x0F);  // uložení kódu klávesy
        break;
      }
    col=col>>1;               // další sloupec
    }
}
```

Přivedem-li tedy na jeden ze sloupců matice klávesnice logickou nulu, pak po přečtení všech řádků lze snadno zjistit která klávesa byla v daném sloupci stisknuta. Postupně se testují všechny sloupce. Jejich změnu ilustruje proměnná *col*, ve se které v každém průchodu rotací posouvá bit, který se pak neguje a nuluje tak jeden z bitů PTC[6:3]. Všechny kódy kláves z detekovaného řádku a sloupce jsou uloženy v *kbi.h*, kde později probíhá jejich identifikace v programu. Jak již bylo zmíněno, klávesnice je řízena přerušením, tzn. při každém stisku se vyvolá podprogam, který data z klávesnice dekóduje a uloží je do fronty zpráv.

5.2.4 Modul řadiče displeje – LCD

Jak bylo podrobněji popsáno v kapitole 3.2.1, použitý mikrokontrolér obsahuje vlastní řadič displejů z tekutých krystalů. Řízení je tedy z pohledu programátora naprosto transparentní, kdy se zápisem příslušného bitu do LDAT registru mikrokontroléru de/aktivuje příslušný segment displeje.

Inicializace

LCD displej v kitu 908LJ12-G2 je zapojen pro použití v *1/4* duty cycle. Toto a časování displeje je nutno nastavit v registru LCDCLK. Bit PCEL v registru CONFIG2, pak aktivuje piny PTC[7:4] jako další řídící frontplane signály. V řídícím registru LCDCR je ještě potřeba

nastavit samotné povolení řadiče displeje a jeho kontrast. Monitorovací režim, ve kterém byla jednotka odlaďována neposkytuje dostatečnou frekvenci pro řízení displeje a tak tyto hodnoty jsou nastaveny jinak, než by byly v reálné aplikaci.

Metodika zobrazování

Každému segmentu na displeji na obrázku 3.6 odpovídá jeden bit registrů LDAT pro řízení displeje. K převodu segmentů na bity těchto registrů slouží následující tabulka, je převzata z manálu ke kitu [16].

BPx	FP22	FP21	FP19	FP18	FP17	FP16	FP15	FP14	FP13	FP12	FP11
BP0	1F	1A	1B	**2F**	2A	2B	3F	3A	3B	4F	4A
BP1	1E	1G	1C	2E	2G	2C	3E	3G	3C	4E	4G
BP2	K1	1D	K2	K3	2D	P1	K4	3D	K5	K6	4D
BP3	X1	H1	X2	X3	H2	P2	X4	H3	X5	Y1	H4

BPx	FP10	FP9	FP8	FP7	FP6	FP5	FP4	FP3	FP2	FP1
BP0	4B	5F	5A	6F	6A	7F	7A	Y2	8F	8A
BP1	4C	5G	5B	6G	6B	7G	7B		8G	8B
BP2	K7	5E	5C	6E	6C	7E	7C		8E	8C
BP3	X6	5D	H5	6D	P3	7D	H6		8D	H7

Tabulka 5.1: Tabulka segmentů

Chceme-li např. aktivovat segment F druhé číslovky displeje, v tabulce 5.1 si jej najdeme se souřadnicemi FP18 a BP0. Z dokumentace k mikrokontroléru [7] lze v sekci věnované LCD řadiči pak vyčíst, že označení F18B0 náleží nultému bitu registru LDAT10. Zápisem jedničky do tohoto bitu se pak segment 2F „rozsvítí“.

Kódy cifer, které jsou zobrazovány na displeji je vhodné mapovat do převodní tabulky. Ta na indexu odpovídající cifře kterou chceme zobrazit vrací data ve formátu vhodném přímo k zápisu do registru LDAT. Tyto převodní tabulky jsou uvedeny v souboru *lcddrive.c*.

5.2.5 Stavový automat

Menu programu je implementováno stavovým automatem popsaným v kapitole 4.2.3. Změna stavu je závislá na kódu zprávy, která se zašle při stisku klávesy. V hlavní smyčce se tato zpráva zachytí a zavolá se funkce *stateAutomaton()*, která dékóduje stisknuté tlačítko. V případě, že je v daném stavu tento kód klávesy platný, provede se příslušná akce.

Automat je implementován ve formě přepínače *switch*, ve kterém se porovnávájí stavy popsané v kapitole 4.2.3.

```
case SET_DIRECT:                    // stav nastavení směru otáčení
  if(messCode==KB_RIGHT)            //přechod na stav SET_SPEED
    state=SET_SPEED;
  else if(messCode==KB_LEFT)        // přechod na stav SET_DRIVE
    state=SET_DRIVE;
  else if(messCode==KB_PLUS)        // otáčet vpravo
    MCon.Revers=TRUE;
  else if(messCode==KB_MINUS)       // otáčet vlevo
    MCon.Revers=FALSE;
```

```
    else if(messCode==KB_NOCODE)       // pomocný stav
      displayInSet(MCon.Turn,X3,NOK);  // zobrazení údajů na displeji
break;
```

Fragment kódu zobrazuje stav ve kterém se nastavuje směr otáčení motoru. V proměnné *messCode* se nachází kód stisknuté klávesy na jehož základě se provede akce. Stiskneme-li např. klávesu s kódem KB_PLUS, automat stav nemění, změní se pouze nastavení příslušného směru rotace motoru. Po stisku např. KB_RIGHT se stav již mění, ale je pouze přepsána hodnota stavu v proměnné *state*, stav není aktivovaný. Proto se stromem stavů prochází dvakrát, kdy se v prvním průchodu změní stav a v druhém průchodu se kód zprávy *messCode* přepíše na KB_NOCODE a požadovaný stav se tak aktivuje, v našem případě se zobrazí určitý údaj na displeji.

Stavový automat je implementován ve stejném souboru jako hlavní smyčka, tedy v *Main.c*.

5.3 Řízení chodu motoru

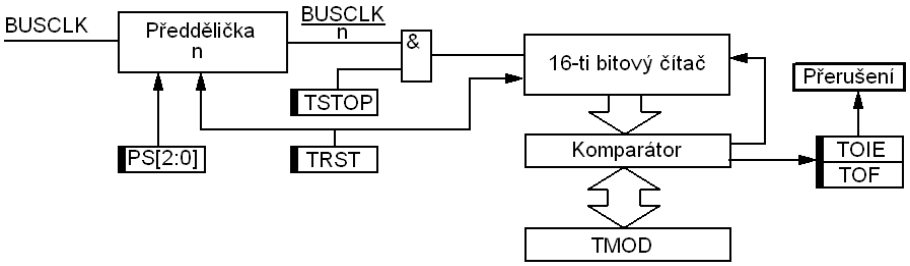
Samotné zasílání řídících impulsů do motoru je řízeno modulem čítače/časovače. Je zde nutno počítat s parametry použitého motoru, především s momentovými charakteristikami.

5.3.1 Modul čítače/časovače – TIM

Pro přesné generování frekvence impulsů pro řízení motoru je použit jeden ze dvou časovačů, které jsou součástí mikrokontroléru. Řízení probíhá, stejně jako u klávesnice, pomocí přerušení. Pro ujasnění je zde uveden princip funkce časovače MC68HC908LJ12 v módu použitém v konstrukci.

Princip modulu TIM

Použitý čítač v konstrukci tvoří 16-ti bitový registr, ke kterému se s každým hodinovým taktem přičítá jednička. Pokud tedy známe frekvenci s jakou se čítač inkrementuje a jeho počáteční stav, lze z obsahu čítacího registru vypočítat dobu, která uběhla od počátku čítání. Umožňuje tedy odměřovat časové úseky, což se nám bude hodit právě pro generování frekvence řídících impulsů pro motor.



Obrázek 5.1: Schéma použitého módu čítače

Na obrázku 5.1 je znázorněn princip funkce čítače TIM1 v módu použitém v konstrukci. Na vstup přicházejí impulsy s frekvencí sběrnice BUSCLK, ty procházejí programovatelnou předděličkou, které můžeme pomocí bitů PS[2:0] registru T1SC určit dělící poměr. Signál

dále prochází skrz povolovací hradlo, kde bitem TSTOP registru T1SC povoluje inkrementaci čítače. Bitem TRST téhož registru pak můžeme čítač i předdělič nulovat. Čítač se každou periodou hodinového signálu inkrementuje a porovnává s registrem TMOD, pokud dojde ke shodě, čítač se vynuluje a vyvolá přerušení ve kterém dojde k obsluze signálů pro řízení motoru.

Inicializace čítače

Inicializace zahrnuje zastavení čítače bitem TSTOP, povolení přerušení od TIM1 bitem TOIE v registru T1SC a zápisem jedničky do bitu TRST, dojde tak k vynulování čítače a předděličky. Zahrnuje také naplnění registru TMOD výchozí hodnotou, viz dále.

Výpočet obsahu čítacích registrů

Aby mohl výpočet obsahu registru TMOD pružně reagovat na změny frekvence sběrnice (použijeme-li např. jiné taktování), je tato frekvence zahrnuta přímo do výpočtu. V případě frekvencí v řádech miliónů Hz výpočet trvá řádově jednotky milisekund, což není kritické. Rychlost otáčení se zadává v otáčkách za sekundu (ot/s), kdy nejmenší rozlišení je 0,01ot/s což je plně postačující.

Výpočet probíhá ve funkci *SetTcnt()* v souboru *motor.c*. Aby byla přesnost generovaného kmitočtu co nejvyšší, je nutno zařídit, aby registr TMOD vždy obsahoval nejvyšší možnou hodnotu, tj. aby měl co nejvyšší rozlišení. Proto se nejdříve nastaví předdělička na ten nejnižší dělící poměr, se kterým se čítací hodnota ještě vejde do 16-ti bitového rozsahu registru TMOD. Výpočet předděličky provedeme následujícím způsobem.

$$ps = \frac{BUSCLK}{CNTMAX \cdot speed \cdot steps}$$

kde

$BUSCLK$ - frekvence sběrnice

$CNTMAX$ - maximální obsah čítacího registru

$steps$ - počet kroků na otáčku

$speed$ - rychlost v ot/s

Pokud je číslo ve výsledku menší než jedna, předdělička se nazařazuje, čítací registr se inkrementuje přímo v taktu BUSCLK. Pokud je větší než jedna, znamená to, že požadovaná hodnota se nevejde do rozsahu TMOD. Frekvence se pak dělí podle binárního řádu vypočtené hodnoty ps. (např. jestliže $ps = 5,1$, leží mezi řády 4-8 a protože $2^3 = 8$, pak bude BUSCLK dělena právě hodnotou vypočtenou vyššího řádu $log_2 8 = 3$). Tato hodnota se zapíše do bitů PS[2:0] registru T1SC.

Samotný obsah registru TMOD se pak vypočítá za pomocí dalšího vzorce.

$$TMOD = \frac{\frac{BUSCLK}{prescaler^2}}{speed \cdot steps}$$

kde

$BUSCLK$ - frekvence sběrnice

$prescaler$ - dělitel vypočtený předděličkou

steps - počet kroků na otáčku

speed - rychlost v ot/s

Číslo vypočtené předěličky by z předchozího případu bylo 3, to odpovídá děliteli $2^3 = 8$, viz bity PS[2:0] registru T1SC [7]. V registru TMOD se tedy objeví číslo schopné po spuštění čítače s vypočtenou předděličkou generovat impulsy o jisté frekvenci, která v daném režimu (čtyřtaktní nebo osmitaktní - dáno počtem kroků na otáčku) otáčí rotorem se zadanou rychlostí.

5.3.2 Rozběhový kmitočet

Abychom předešli ztrátám kroku při roztáčení motoru, je nutno sledovat charakteristiku rozběhového momentu motoru. Ta je uvedena na obrázku 3.2. Jedná se o moment charakteru pasivního tření bez přidané vnější setrvačné hmoty. Pokud na hřídel motoru připojujeme vnější setrvačnou hmotu, pak se jedná o rozběhový moment setrvačnosti, jehož rozběhový kmitočet můžeme vypočítat podle vzorce uvedeném v kapitole 2.3.3 a chrakteristiky 3.3 z deviačního momentu tělesa.

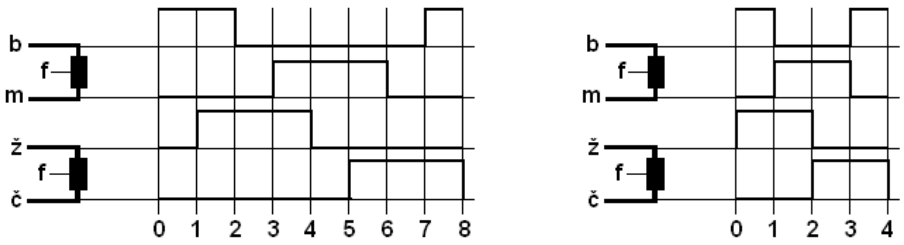
Známe-li tedy zatěžovací moment, který bude na hřídel v době startu působit, můžeme z momentových chrakteristik stanovit s jakou maximální frekvencí se motor bude roztáčet.

Rozběhový kmitočet v Hz lze v programu pevně nastavit v souboru *motor.h* zvlášť pro čtyřtaktní nebo osmitaktní řízení pomocí konstant STARTFREQ8 a STARTFREQ4.

Pokud je požadovaná rychlost vyšší než rozběhová, je nutno ji zvětšovat postupně. To zajišťuje odečítání konstanty od registru TMOD (tedy zvyšování frekvence), při každém kroku motoru, dokud v TMOD není požadovaná hodnota. Tuto konstantu reprezentuje proměnná *accelStep* viz zdrojový soubor *Motor.c* v příloze 1. Jedná se vlastně o zrychlení, které rotor provede než se roztočí na zadanou rychlost.

5.3.3 Metodika řízení

V každém momentu, kdy se vyvolá přerušení od přetečení čítače se tedy provede podprogram, který má za úkol přepínat fáze v závislosti na parametrech nastavených uživatelem. Pro řízení jsou ve strukturách programu uloženy kódy, které přímo odpovídají posloupnosti sepnutých fází při čtyřtaktním nebo osmitaktním řízení. Změna směru otáčení se provede spínáním opačné posloupnosti. Tyto kódy jsou pak přímo posílány na výstupní port.



Obrázek 5.2: Signály řízení

Na obrázku 5.2 vlevo jsou naznačeny signály pro osmitaktní řízení motoru, vpravo pak pro čtyřtaktní. Označení jednotlivých signálů odpovídá barevnému značení fází motoru SMR 300-100-RI/24 viz obrázek 3.4.

Modul *Motor.c* je do jisté míry transparentní v tom smyslu, že veškeré nastavení motoru probíhá přes trukturu *MControl*. Tato struktura, která reprezentuje nastavení módu otáčení motoru je uvedena ve zdrojovém souboru *Motor.h*.

```
typedef struct {
  int Turn;
  unsigned char Steps;  // počet kroků na otáčku
  Bool Revers;          // směr otáčení
  int Angle;            // úhel o který se má motor pootočit
  unsigned char Mode;   // mód otáčení motoru
  int CurAngle;         // aktuální úhel natočení
} MCONTROL;
```

Požadované parametry otáčení se zapíšou do struktury, modul motoru si je pak při aktivaci převezme a řídí podle nich rychlost a sled výstupních signálů. Soubor *Motor.h* pak obsahuje konstanty, kterými lze předat programu informaci o technických parametrech motoru, jeho liminích hodnotách apod. To zaručuje jistou přenositelnost kódu na více typů motorů.

Kapitola 6

Závěr

V této práci je rozvedena problematika řízení krokových motorů a její možné řešení. Podařilo se napsat program, který je schopen se přizpůsobit katalogovým parametrům krokového motoru a podle nich řízení provádět, proto je v praxi velice univerzální. Navíc je napsán v jazyku C, což přináší výhodu jeho přenositelnosti i na jiné platformy.

Požadavky zadání se podařilo beze zbytku splnit, navíc byl vyvinut i jednoduchý tranzistorový budič magnetických vinutí motoru pro unipolární řízení. Funkčnost jednotky byla prakticky ověřena se všemi popisovanými částmi. Velkým přínosem pro vývoj jednotky se stal univerzální vývojový kit 908LJ12-G2 a možnost připojit jej k PC v monitorovacím módu MON08, díky kterému se práce značně urychlí.

Tato práce může být inspirací ke stavbě mnohem složitějších zařízení. Program byl psán velmi univerzálně a princip jeho jádra pro samotné řízení motoru může být lehce přenositelný na celé systémy používající krokové motory, roboty, polohovací zařízení atp..

Další práce by se mohla vyvíjet např. směrem řízení koordinace soustavy krokových motorů v souřadnicovém zapisovači, kdy se ze stolního PC zašle ve vhodném formátu grafický obrazec, který řídící jednotka zpracuje a vykreslí. Jiným velmi zajimavým využitím může být koordinace v prostoru, např. strojový manipulátor nebo CNC obráběcí stroj, kde se v PC vytvoří 3D model a ten se opět zašle k fyzickému zpracování.

Literatura

[1] Mann Burkhard. *C pro mikrokontroléry*. 1. vydání, Nakladatelství BEN, ISBN 80-7300-077-6, 2003.

[2] ORIENTAL MOTOR GENERAL CATALOGUE. The basic of stepping motor. [online], [cit. 01-05-2007], URL: `http://www.azaticaret.com/1062491525_fr.pdf`.

[3] Ken Connor. Lecture 8. stepper motors. [online], [cit. 01-05-2007], URL: `http://hibp.ecse.rpi.edu/~connor/education/IEE/lectures/Lecture_8_Stepper_motors.pdf`.

[4] Douglas W. Jones. Control of stepping motors. [online], [cit. 01-05-2007], URL: `http://www.cs.uiowa.edu/~jones/step/`.

[5] Josef Strnadel Josef Shwarz, Richard Růžička. *Skripta pro přdmět IMP - Mikroprocesorové a vestavěné systémy*. FIT VUT Brno, 2006.

[6] Daniel Malik. Bare lcd display drive in embedded applications. rev. 7 May 2003, [online], [cit. 01-05-2007], URL: `http://dataweek.co.za/article.aspx?pklArticleId=2382&pklIssueId=322&pklCategoryId=31`.

[7] Motorola. Mc68hc908lj12 technical data. Rev. 2/2002, [online], [cit. 01-05-2007], URL: `http://www.motorola.com.cn/semiconductors/mcudsp/forms/databook/devices_for_contest/hc08_contest_selected_%20devices/MC68HC908LJ12.pdf`.

[8] spol. s r.o. REGULACE AUTOMATIZACE BOR. Kroková reverzační pohonná jednotka smr 300-100-ri/24. [online], [cit. 01-05-2007], URL: `http://www.regulace.cz/CZ/KM/km_300-100ri.html`.

[9] spol. s r.o. REGULACE-AUTOMATIZACE BOR. Krokové motory - všeobecné údaje. [online], [cit. 01-05-2007], URL: `http://www.regulace.cz/CZ/KM/km_vu.html`.

[10] spol. s r.o. REGULACE AUTOMATIZACE BOR. Regulace - automatizace bor, spol. s r.o. [online], [cit. 01-05-2007], URL: `http://www.regulace.cz`.

[11] spol. s r.o. REGULACE AUTOMATIZACE BOR. Rozdělovač impulzů ri 250-24-4/8. [online], [cit. 01-05-2007], URL: `http://www.regulace.cz/DOWNLOADS/PDF/kl_km_ri.pdf`.

[12] FAIRCHILD SEMICONDUCTOR. Bc546/547/548/549. [online], [cit. 01-05-2007], URL: `http://www.ortodoxism.ro/datasheets/fairchild/BC547.pdf`.

[13] FAIRCHILD SEMICONDUCTOR. Bc635/637/639. [online], [cit. 01-05-2007], URL: http://www.ortodoxism.ro/datasheets/fairchild/BC639.pdf.

[14] Vladimíř Váňa. *Začínáme s mikrokontroléry HC08 Nitron.* Ben, technická literatura, Praha, 2003. 338 s., ISBN 80-7300-124-1.

[15] Kamil Řezáč. Krokové motory. rev. 2002-10-28, [online], [cit. 01-05-2007], URL: http://robotika.cz/articles/steppers/en.

[16] Beta Control řídící a informační systémy. Starter kit lj12evb. rev. Listopad 2002, [cit. 01-05-2007], URL: http://www.betacontrol.cz/prospekty/vyvojprostredky/908lj12/hc908lj12_en.pdf.

Seznam příloh

Přílohy

Příloha č. 1 Obsah přiloženého datového nosiče

Adresáře

budic - schéma a obraz desky plošného budiče fází spoje ve formátu *Eagle Layuout Editor*

cwproject - projektové soubory pro vývojové prostředí Freescale CodeWarrior for HC08 V5.1

docs - dokumentační soubory

- **progdoc** - programová dokumentace ve formátu HTML
- **stepper.pdf** - tato dokumentace ve formátu PDF

src - obsahuje zdrojové soubory programu (*interface.c*, *kbi.c*, *lcddrive.c*, *main.c*, *motor.c*, *Start08.c*, *derivative.h*, *interface.h*, *kbi.h*, *lcddrive.h*, *main.h*, *motor.h*)