

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta strojní

Katedra aplikované kybernetiky

Studijní program: M2301 Strojní inženýrství

Obor: 3902T021 Automatizované systémy řízení ve strojírenství

Zaměření: Automatizace inženýrských prací

Řízení rychlosti a polohy – elektrický pohon ve spojení s modelem řídicího systému

Petr Mervart

Vedoucí práce: Prof. Ing. Miroslav Olehla, CSc., TU Liberec

Počet stran : 52

Počet příloh : 3

Počet obrázků : 22

Počet tabulek : 2

Počet výkresů: -

UNIVERZITNÍ KNIHOVNA
TECHNICKÉ UNIVERZITY V LIBERCI



3146086127



TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta strojní

Katedra aplikované kybernetiky

Studijní rok: 2005/06

ZADÁNÍ DIPLOMOVÉ PRÁCE

Jméno a příjmení: **Petr M E R V A R T**

Studijní program: **M2301 Strojní inženýrství**

Obor: **3902T021 Automatizované systémy řízení ve strojírenství**

Zaměření: **Automatizace inženýrských prací**

Ve smyslu zákona č. 111/1998 Sb. o vysokých školách se Vám určuje diplomová práce na téma:

Řízení rychlosti a polohy elektrický pohon ve spojení s modelem řídicího systému

Zásady pro vypracování:

(uveďte hlavní cíle diplomové práce doporučené metody pro vypracování)

1. Seznamte se s problematikou řízení polohových servomechanismů, proveďte analýzu požadavků na jejich řízení.
Prostudujte možnosti využití jednočipových mikropočítačů pro řízení komunikace mezi modelovou technologií a MATLABem prostřednictvím sériové linky RS232.
2. Navrhněte technické prostředky a programové zabezpečení pro spolupráci modelové technologie s modelem řídicího systému v prostředí MATLAB/SIMULINK.
3. Realizujte řídicí a výkonovou jednotku pro řízení elektrického pohonu a komunikace.
4. Navrhněte a ověřte řídicí algoritmy pro řízení navržené modelové technologie polohového servomechanismu.

+CD

KKY/AIP

52 s.

ph, tab, grafy

ANOTACE

TÉMA:

Řízení rychlosti a polohy – elektrický pohon ve spojení s modelem řídicího systému

ANOTACE:

Diplomová práce se zabývá návrhem řídicí a výkonové jednotky ovládající elektrický pohon a komunikaci po sériové lince z prostředí MATLAB/SIMULINK.

Vytvořený model v prostředí SIMULINK umožňuje ve spojení se zhotovenou řídicí a výkonovou jednotkou řízení rychlosti a polohy elektrického pohonu, s důrazem na využití celé modelové technologie pro výukové účely v laboratoři.

ANOTATION

THEME:

Control of speed and position – electric drive in connection with control model

ANOTATION:

This thesis deals with design of the electronic control and power unit driving the electric drive and the serial communication from MATLAB/SIMULINK.

The model created in toolbox SIMULINK is capable in connection with designed electronic control and power unit to control speed and position of the electrical drive, emphatically to use this complex model technology for education purpose in the laboratory.

OBSAH

1.	ÚVOD	9
2.	ANALÝZA ŘÍZENÍ POLOHOVÝCH SERVOMECHANISMŮ	10
2.1.1.	Hlavní způsoby regulace pohonů	10
2.1.2.	Základní typy řízení servomechanismů	11
3.	JEDNOČIPOVÉ MIKROKONTROLÉRY (MCU)	12
3.1.	Výběr vhodného typu MCU	12
3.2.	MCU řady PIC 16F87xA fi Microchip	13
3.2.1.	CCP jednotky - PWM režim	15
3.2.2.	10bitový A/D převodník	18
3.2.3.	Jednotka sériové komunikace USART (RS232C)	19
3.2.4.	Obsluha přerušení	21
3.2.5.	Pomocné obvody	22
3.2.6.	Programování paměti FLASH	22
3.2.7.	Vývojové prostředí (IDE) SourceBoost IDE	24
4.	NÁVRH TECHNICKÝCH PROSTŘEDKŮ A SOFTWARE MODELU	24
4.1.	Návrh řídicí jednotky – hardware (HW)	26
4.1.1.	Vstupní signály a jejich zpracování	26
4.1.2.	Výstupní signály a jejich generování	27
4.1.3.	Technické parametry řídicí jednotky	29
4.2.	Návrh řídicí jednotky – software (SW)	29
4.3.	Návrh výkonové jednotky (VJ)	31
4.3.1.	Obvod L298	31
4.3.2.	Průběhy napětí a proudu motoru	33
4.3.3.	Mechanická konstrukce výkonové jednotky	34
4.3.4.	Technické parametry výkonové jednotky	35

4.4.	Mechanické uspořádání a propojení řídicí a výkonové jednotky	35
4.5.	Návrh modelu v prostředí MATLAB/Simulink	36
4.5.1.	Řízení rychlosti – model ‘motor_PID.mdl’	38
4.5.2.	Subsystem ‘Modelová technologie s PID regulátorem	38
4.5.3.	S-Funkce ‘smotor.m’	40
4.5.4.	Simulace za použití P, PD, PI a PID regulátoru	41
4.5.5.	Řízení polohy – model ‘motor_poloha.mdl’	44
4.6.	Ověření funkčnosti a verifikace navrženého modelu	45
5.	ZÁVĚR	47
	POUŽITÁ LITERATURA	49

PŘÍLOHY

PŘÍLOHA 1 - SCHÉMA ZAPOJENÍ ŘÍDICÍ JEDNOTKY	50
PŘÍLOHA 2 - SCHÉMA ZAPOJENÍ VÝKONOVÉ JEDNOTKY	51
PŘÍLOHA 3 - SEZNAM SOUČÁSTEK	52

1. Úvod

Žádné moderní stroje a zařízení se dnes neobejdou bez přesného a spolehlivého řídicího systému. Téměř v každém odvětví průmyslu i spotřební techniky se tyto systémy hojně používají. V současné době je již drtivá většina těchto řídicích systémů založena na bázi mikroprocesorového řízení (počínaje jednodušími, mikrokontroléry řízenými, zařízeními a konče složitými PLC systémy či průmyslovými PC řídicí celé výrobní linky s možností vizualizace).

Kde dříve pro řízení postačovala pevně naprogramovaná logika, dnes už by byla pro složité a rychlé regulační a řídicí obvody nevyhovující; a když ano, tak za cenu vysoké složitosti a ceny. Neméně podstatnou výhodou číslicově řízených řídicích systémů je i flexibilita při změně obslužného programu a tudíž i daleko větší univerzálnost použití oproti pevně programované logice. Bez těchto mikroprocesorově řízených systémů by ani nebyl možný prudký rozvoj nejrůznějších servomechanismů.

Především při stavbě moderních obráběcích strojů se již pomalu začíná narážet na hranice možností mechanických převodů a proto není vývoj v této oblasti tak překotný, jako ve vývoji elektrických a elektronických systémů. Na mechanické součásti jsou kladeny vysoké požadavky na tuhost konstrukce, odolnost proti vnějším silám apod. Řešení těchto problémů je proto daleko složitější a finančně náročnější než vývoj elektronických dílů. Stále více se proto výrobci strojů snaží nahrazovat mechanické převody elektricky regulovanými jako jsou např. lineární či krokové motory, umožňující přesné nastavení polohy obrobku, zrychlení posuvu apod.

Tato diplomová práce si dala za cíl vytvořit kompletní řídicí systém ovládající elektrický pohon (SS motor) z prostředí Matlab/Simulink, který bude sloužit v laboratoři pro názornou výuku servomechanismů a číslicových počítačů. Praktická část práce sestává z vytvoření řídicí a výkonové jednotky a vytvoření zkušebního řídicího modelu v prostředí Matlab/Simulink.

Řídicí jednotka byla navržena tak, aby ji bylo možné použít i na výuku programování mikrokontrolérů rodiny PIC 16F87x (vývojová deska).

2. Analýza řízení polohových servomechanismů

Polohový servomechanismus je zařízení, jehož úkolem je přeměna vstupní energie na mechanický pohyb. Pohon servomechanismu zajišťuje motor, přeměňující vstupní energii (elektrickou, chemickou atd.) na mechanický pohyb (otáčivý či posuvný). Tato mechanická energie se dále pomocí vhodného transformačního prvku (spojka, hřídel, řemen atd.) přenáší na pohyblivou část pohybového ústrojí mechanismu.

Nejčastějším pohonem dnešních strojů je elektrický motor napájený výkonovým měničem, který je řízen pomocí povelů, které posílá řídicí jednotka stroje na základě dat, zjištěných z různých čidel polohy, rychlosti, proudu atd. Často již pohon obsahuje výkonový měnič i řídicí jednotku a jako takový je řízen povely přímo z CNC stroje.

2.1.1. Hlavní způsoby regulace pohonů

Pro regulaci pohonů se používají tyto tři základní typy regulací:

- proudová
- rychlostní
- polohová

Proudová a rychlostní regulace (smyčka) se nacházejí přímo v bloku pohonu (jsou tedy v jeho rámci uzavřené). Proudová regulace slouží k co nejrychlejšímu nárůstu/poklesu napájecího proudu motoru při náhlé (skokové) změně vstupní veličiny. Zároveň umožňuje kontrolu maximálního povoleného proudu, tekoucího měničem resp. motorem a dokáže tak zamezit případnému poškození motoru či měniče v důsledku proudového přetížení. Pohon je pak odolný i vůči zkratu. V této práci nebyla proudová regulace použita především díky použití pouze nezatíženého motoru, kterým pak protéká relativně malý proud (do 1A).

Rychlostní regulace vyhodnocuje signál z čidla rychlosti a porovnává ho s rychlostí požadovanou. Tato odchylka je rychlostním regulátorem převedena na požadovaný proud a tento signál je zaveden do výkonového měniče, který patřičně změní otáčky motoru. Tato vlastnost rychlostní regulace proto pohonu umožňuje být nezávislým na zatížení a výrazně ovlivňuje i statickou tuhost pohonu. Propustné pásmo této smyčky též výrazně ovlivňuje dynamickou tuhost a plynulost pohybu při malých rychlostech. Dobře fungující rychlostní regulátor by měl být při skokové změně rychlosti co nejrychlejší s maximálním překmitem 15 až 25% a pak rychlým útlumem.

Vstupním signálem pro polohovou regulaci je okamžitá odchylka skutečné a žádané polohy. Opět je požadována mj. absolutní statická tuhost polohy vůči působení vnějších sil. Jelikož jsou poloha i rychlost matematicky svázány, stačí použití pouze P regulátoru (není nutný regulátor typu PI). Jeho hlavním parametrem je tzv. polohové zesílení K_v . Tato rychlostní konstanta proto vyjadřuje požadovanou rychlost při jednotkové odchylce žádané a skutečné polohy. Kvalitu funkce polohové smyčky lze proto ověřit měřením odezvy na skok rychlosti a příslušné regulační odchylky polohy Δ . Její časový průběh by měl být bez překmitu s minimem kmitavých složek.

2.1.2. Základní typy řízení servomechanismů

Servomechanismy jsou řízeny nejčastěji těmito systémy řízení:

- CP řízení – v každém časovém okamžiku se kontroluje rychlost a poloha pohybových os pro dosažení požadované pohybové trajektorie; dále se toto řízení dělí na:
 - Multi point řízení - řídicí veličiny jsou odebírány a ukládány do paměti jako sled žádaných hodnot
 - Dráhové řízení – používá menší počet tzv. uzlových bodů, které je doplněno výpočtem prostorové křivky interpolací nebo inkrementálními procesy
- Point to Point řízení – používá omezený počet řízených bodů. Rychlost mezi těmito body je daná předběžným nastavením regulačních prvků. Obvykle se pohybuje pouze jedna pohybová jednotka, při současném pohybu více pohybových jednotek nelze přesně určit polohu a vzniká tak tzv. nebezpečná zóna

Pro dráhové řízení se dříve používaly analogové či analogově digitální regulátory. V současné době jsou již tyto nahrazovány plně digitálními typy. Výhodou analogových regulátorů byla především rychlost zpracování, ovšem nevýhody jako teplotní nestabilita parametrů či seřizování parametrů přímo na regulátoru převážily.

Proto se dnes pro řízení pohonů již téměř výhradně používají jednoúčelové mikrokontroléry či speciální průmyslové počítače na bázi architektury Intel x86 . Tyto mikrokontroléry jsou vhodné především díky speciálním obvodům v podobě čítačů, časovačů, PWM jednotek, přesných A/D převodníků (10-12bitů), moderních komunikačních sběrnic (CAN, USART, SPI...), hlídacích obvodů (watchdog, brown-out

detektor) atd. Nejpoužívanějšími mikrokontroléry byly 8 a 16bitové typy založené na jádrech 8096, 8051 či 68HC11. Tyto MCU byly dostatečně výkonné pro jednodušší aplikace. Pro složitější úlohy lze použít moderních 32bitových mikrokontrolérů či speciálních signálních procesorů (DSP).

Rychlý vývoj v oblasti procesorů umožnil výrazně zvýšit možnosti využití PLC automatů pro řízení servomechanismů. Tyto moderní systémy disponují vysokým výpočetním výkonem a obsahují řadu funkcí, umožňující řízení i v několika osách najednou. PLC automaty jsou často schopny mimo samotné řízení servopohonu ovládat komplexně celý technologický proces.

3. Jednočipové mikrokontroléry (MCU)

Jednočipové mikrokontroléry (mikropočítače) jsou polovodičové součástky dnes používané již téměř ve všech výrobcích spotřební elektroniky a mnoha oblastech průmyslu. Každé moderní zařízení měřicí, automatizační a regulační techniky si lze dnes už těžko bez těchto jednoúčelových mikrokontrolérů představit.

MCU používají z pohledu struktury dvě různé topologie: harwardskou – paměť dat a programu je oddělená a nebo Von Neumannovu – paměť programu a dat je v jednom společném adresním prostoru. Stejně tak se MCU dělí dle počtu instrukcí, používaných pro jejich programování; buď na CISC – s kompletním souborem instrukcí (velký počet instrukcí, málo registrů) a RISC – s redukováným souborem instrukcí (jen základní instrukce, mnoho registrů). Základním vybavením MCU je procesor (CPU), paměť RAM, paměť pro uložení programu - dnes už nejčastěji typu FLASH (umožňující opakované přeprogramování), sběrnice propojující CPU s paměťmi a další pomocné a periferní obvody. Těmi jsou především A/D převodníky, komunikační sběrnice (I2C, UART, CAN...), PWM jednotky, komparátory atd.

3.1. Výběr vhodného typu MCU

Nejjednodušší MCU jsou 8bitové (vhodné pro méně náročné aplikace) např. od firem Microchip (např. řady PIC10,12,14,16) či základní rodina procesorů x51 (Intel, Atmel...). Ovšem i přes malou šířku zpracovávaných dat (8bitů), dosahují některé typy výkonu 5-10 MIPS (milionů instrukcí za sekundu), což jejich použití posouvá i do oblastí náročnějších úloh. Velkou výhodou těchto MCU je především relativní jednoduchost při

programování, dostupné vývojové a ladící prostředky a v neposlední řadě i příznivá cena samotného MCU. Především z těchto důvodů byl pro tuto diplomovou práci vybrán 8bitový MCU fi Microchip, typ 16F876A, který bude podrobněji popsán v následujících kapitolách. Sám mikrokontrolér v této práci nezastává funkci regulátoru polohy či rychlosti; pouze zprostředkovává komunikaci mezi výkonovou jednotkou (měničem) a osobním počítačem, který vykonává funkci regulátoru. Z tohoto důvodu nejsou požadavky na výkon MCU tak důležité.

Pro složité aplikace řízení v reálném čase spolu se zpracováváním mnoha vstupních a výstupních dat je už nezbytné použití 16/32bitových MCU, jako jsou např. typy MPC500/MPC5500 fi Motorola, ARM MCU (16/32bit), dsPIC, (DSP MCU) atd. Použití těchto vysoce výkonných MCU je však nutné dobře zvážit s ohledem na již poměrně drahé vývojové prostředky (kompilátory, IDE, programátory), větší složitost při programování a v neposlední řadě i komplikace spojené s povrchovou montáží (např. moderní TQFP pouzdra s až 144 vývody).

3.2. MCU řady PIC 16F87xA fi Microchip

Tyto moderní 8bitové RISC MCU se vyznačují především výhodným poměrem cena/výkon, snadným programováním a velkou škálou vývojových a ladících prostředků dostupným i v ČR.

Device	Program Memory		Data SRAM (Bytes)	EEPROM (Bytes)	I/O	10-bit A/D (ch)	CCP (PWM)	MSSP		USART	Timers 8/16-bit	Comparators
	Bytes	# Single Word Instructions						SPI	Master I ² C			
PIC16F873A	7.2K	4096	192	128	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F874A	7.2K	4096	192	128	33	8	2	Yes	Yes	Yes	2/1	2
PIC16F876A	14.3K	8192	368	256	22	5	2	Yes	Yes	Yes	2/1	2
PIC16F877A	14.3K	8192	368	256	33	8	2	Yes	Yes	Yes	2/1	2

Obrázek 3-1

Z tabulky na obr. 3.1 jsou patrné hlavní rozdíly jednotlivých typů této řady MCU. Především je to počet vstupně/výstupních (I/O) pinů, a to 22 (pouzdro 28pin) a nebo 33 (pouzdro 40/44pin). Dalším rozdílem je velikost datové (SRAM), programové (FLASH) a EEPROM paměti. Ostatní parametry, jako je počet vstupů A/D převodníku, počet časovačů atd. jsou již totožné. Přehled nejdůležitějších vlastností MCU 16F876A shrneme níže:

- pouze 35 instrukcí

- všechny instrukce vykonány v jednom cyklu (mimo větvení programu – dva cykly)
- pracovní frekvence až 20Mhz (instrukční cyklus 200ns) => výkon až 5 MIPS
- 14.3kBytu (8192 instrukčních slov) programové paměti typu FLASH
- 368 x 8 Bytu paměť RAM
- 256 bytu paměť EEPROM
- pinově kompatibilní s 28/40-44 pin typy PIC 16Cxxa 16Fxxx
- 15 vnitřních nebo vnějších zdrojů přerušení
- 8-mi úroňový STACK (pro návratové adresy přerušení a podprogramů)
- přímé, nepřímé a relativní adresování
- ochrana programového kódu
- Timer 0 (časovač/čítač) – 8bit – s programovatelným předděličem
- Timer 1 (časovač/čítač) – 16bit – může být zvětšován v SLEEP módu
- Timer 2 (časovač/čítač) – 8bit (časová základna pro PWM) s předděličem i děličem po
- 2x CCP jednotky s módy: Capture - 16bit, Compare - 16bit a PWM - 10bit
- Synchronní Sériový port (SSP)s módy SPI (master) a I2C (master/slave)
- Univerzální Synchronní Asynchronní Přijímač Vysílač – USART/SCI módy
- 10bitový AD převodník s 5ti vstupy
- 2x analogové komparátory
- instrukce SLEEP pro programové uvedení MCU do úsporného režimu
- Watchdog Timer (WDT) – s vlastním RC oscilátorem pro zvýšení spolehlivosti funkce
- Brown-Out detektor – obvod vyvolávající Reset při poklesu napájení
- prodleva při zapnutí napájení (Power UP Timer) a prodleva pro start oscilátoru (Oscillator Start-Up Timer)
- programování a ladění dvouvodičovou linkou v koncové aplikaci (ICSP)

- 100 000 cyklů mazání/zápis programové paměti FLASH
- 100 000 000 cyklů mazání/zápis paměti EEPROM
- doba zachování dat v EEPROM > 40 let
- proud vývodu až 25mA (ven/dovnitř)
- široký rozsah napájecího napětí: +2.0V až +5.5V
- nízká spotřeba díky CMOS technologii (např. < 2mA při 5V a 4Mhz)
- teplotní rozsahy COMMERCIAL, INDUSTRIAL a AUTOMOTIVE

Jak již bylo napsáno v odstavci 3.1, pro použití jako hlavního MCU v této DP byl vybrán typ 16F876A. Tento MCU má dostatek I/O pinů, velkou paměť programu (14.3KBytes), paměť SRAM o velikosti 368Bytes, dělenou do čtyř paměťových bank, kde v první je k dispozici spolu se speciálními registry také 96 volně přístupných registrů a v dalších třech pak po dalších 80 volných registrech. Paměť EEPROM se může použít pro uložení dat, které v ní zůstanou zachovány i po vypnutí napájení.

3.2.1. CCP jednotky - PWM režim

Velice důležitou periférií použitou v této DP na změnu a regulaci otáček SS motoru je jednotka CCP – Capture/Compare/PWM. Jeden z jejích pracovních režimů je generování signálu s proměnnou střídou – tzv. pulsně šířkovou modulací (Pulse Width Modulation – PWM). Jelikož je PWM jednotka přímo součástí MCU, je generování tohoto signálu poměrně jednoduché a sestává se víceméně pouze ze zápisu hodnot do příslušných systémových registrů ('x' v názvu registrů nabývá hodnot 1 nebo 2, podle zvolené CCP jednotky; PIC 16F876A obsahuje 2 jednotky CCP) :

1. Zapsáním hodnoty do registru PR2 (0-255) - určit periodu PWM signálu
2. Zapsáním hodnoty do registrů CCPRxL (8bit) a CCP1CON<5:4> (2bit) - určit střidu sepnutí (Duty Cycle)
3. Zajistit aby pin CCPx (sdílený s portem C) byl nastaven jako výstupní
4. Určit hodnotu předděliče časovače TIMER2 a povolit tento časovač
5. Přepnout CCPx jednotku do režimu PWM v systémovém registru CCP1CON

Jak z principu PWM modulace vyplývá, střida může nabývat hodnot 0 (trvalé vypnutí) až 100% (trvalé sepnutí). PWM jednotka použitá v MCU 16F876A dosahuje rozlišení až

10bitů (v závislosti na zvolené periodě). Číselně je to reprezentováno rozsahem 0 – 1023. Výpočet periody (frekvence spínání) PWM signálu se dle katalogového listu [7] vypočte:

$$T_{PWM} = [(PR2) + 1] \cdot 4 \cdot T_{OSC} \cdot pd_{TMR2}$$

Frekvence výstupního PWM signálu je:

$$f_{PWM} = \frac{1}{T_{PWM}}$$

T_{PWM} – perioda PWM

f_{PWM} – frekvence PWM

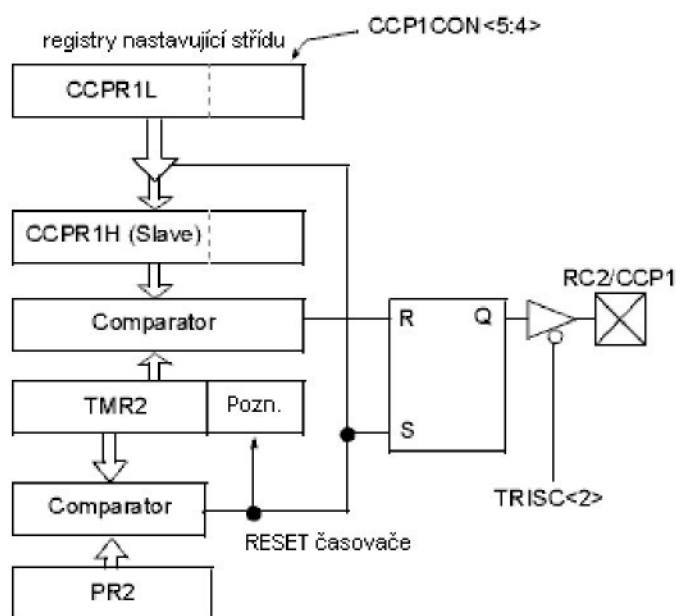
PR2 – registr PR2

pd_{TMR2} – predellic casovace TIMER 2

T_{OSC} – perioda oscilátoru

Když dojde ke shodě obsahu registru PR2 a TMR2, dojde k následujícím událostem při následujícím cyklu zvyšujícím obsah registru TMR2 (instrukčním cyklu) :

- obsah registru TMR2 je vynulován
- výstupní vývod CCPx je nastaven do stavu “1” (kromě případu, kdy je nastavena střída v registru CCPRxL = 0)
- velikost střídý sepnutí je přenesena z registru CCPRxH do registru CCPRxL



Pozn. 8bitový čítač je spojen s 2bitovým vnitřním CLK nebo 2bitovým předděličem vytvářejícím 10bitovou základnu

Obrázek 3-2

Podobně lze určit i střidu sepnutí. Tato je určena v registru CCPRxL (8 bit) a dvěma bity systémového registru CCPxCON<5:4>. Tyto 2 bity (CCPxX a CCPxY) mohou být trvale ve stavu 0, pak je rozlišení PWM pouze 8 bitů. Následující vztah je použit pro výpočet střidy sepnutí :

$$S_{PWM} = [(CCPRxL : CCPxCON < 5 : 4 >) \cdot T_{OSC} \cdot pd_{TMR2}$$

S_{PWM} – strida sepnutí PWM

pd_{TMR2} – preddelice casovace TIMER 2

T_{OSC} – perioda oscilatoru

Hodnota do registru CCPRxL a bitů CCPxX a CCPxY může být zapsána kdykoliv, ale k přenosu do registru CCPRxH dojde pouze při shodě obsahů registrů PR2 a TMR2, to je na začátku periody. V režimu PWM je registr CCPxH přístupný pouze pro čtení jeho hodnoty. Tím je zaručeno že nedojde k zapsání střidy do registru CCPRxH kratší, než je současná hodnota střidy (obsah registru TMR2). Jinak by mohlo dojít v dané periodě (špatně časovaného zápisu) k plnému sepnutí a chybné funkci PWM. V dané periodě by již nedošlo k vypnutí, stavu 0, na výstupním vývodu, neboť by nedošlo ke shodě obsahu registru CCPRxH s registrem TMR2.

Když je obsah registru CCPRxH a 2 bitového vnitřního čítače shodný s obsahem TMR2, který je spojen s hodinovou frekvencí vydělenou 2x a 2 bity předděliče TMR2, je stav na výstupním vývodu shoden na 0. Maximální rozlišení PWM pro danou frekvenci je dáno vztahem :

$$PWM_{res} = \frac{\log(\frac{F_{OSC}}{F_{PWM}})}{\log(2)}$$

F_{PWM} – frekvence PWM

F_{OSC} – frekvence oscilátoru

Frekvence PWM	1.22 kHz	4.88 kHz	19.53 kHz	78.12 kHz	156.3 kHz	2108.3 kHz
Nastavení předděliče (1,4,16)	16	4	1	1	1	1
Hodnota PR2 registru	255	255	255	63	31	23
Maximální rozlišení	10 bitů	10 bitů	10 bitů	8 bitů	7 bitů	5.5 bitů

Tabulka 3.1 - Frekvence a rozlišení PWM při 20Mhz

Pro konkrétní SS motor použitý v testovací modelové technologii byla na základě praktických testů zvolena frekvence PWM asi 488Hz (PR2 = 255, předdělič 16, 8Mhz). Tato poměrně nízká hodnota byla zvolena s ohledem na co nejvyšší a nejplynulejší

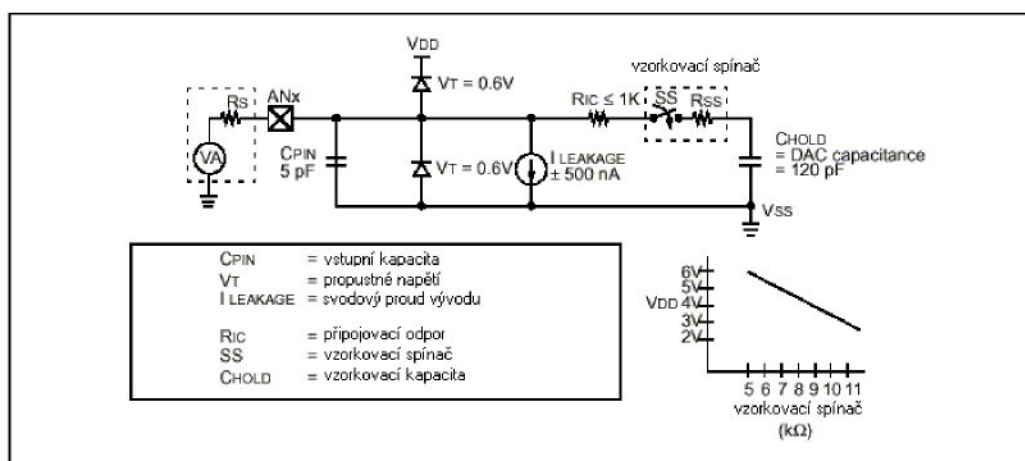
rozsah regulace otáček; motor se při nulovém zatížení a napájecím napětí 20V roztočí již při střídě asi 15% a dosahuje při tom kolem 250 ot/min.

3.2.2. 10bitový A/D převodník

Neméně důležitou periférií je i převodník analogové úrovně na digitální hodnotu. Zde použitý převodník je typu Sample and Hold. To znamená, že analogová hodnota (napětí) je přivedena na vzorkovací kondenzátor obvodu SAMPLE and HOLD. Výstupem tohoto obvodu je vstup samotného A/D převodníku, který metodou aproximace převede tuto hodnotu vstupního napětí na 10bitové číslo.

A/D převodník je schopen i provozu v režimu SLEEP, což omezuje digitální šumy od ostatních částí MCU a zpřesňuje proto samotný převod. Podmínkou je pouze volba vnitřního RC oscilátoru v modulu A/D převodníku jako zdroje hodinového signálu.

K dosažení požadované přesnosti A/D převodu musí být kondenzátor C_{Hold} plně nabit na úroveň vstupního napětí zvoleného kanálu. Náhradní schéma analogového vstupu je na obrázku 3.3. Impedance zdroje R_s a vnitřního spínače R_{ss} přímo ovlivňují čas potřebný k nabití kondenzátoru C_{Hold} . Odpor spínače R_{ss} je závislý na napájecím napětí (viz obrázek 3.3). Maximální doporučená hodnota odporu na analogovém vstupu je 10 kOhm. Poté co je spuštěn A/D převodník, musí být dodržen minimální čas pro nabití C_{Hold} dříve než je spuštěn převod analogového signálu.



Obrázek 3-3

Doba potřebná pro převod 1bitu je definována jako T_{AD} . Pro převod 10bitů je zapotřebí času minimálně $12T_{AD}$. Hodinový signál pro A/D převodník je možné vybrat z těchto možností: 2x, 4x, 8x, 16x, 32x a 64x T_{OSC} nebo vnitřní RC oscilátor (2-6us). Pro

správnou funkci A/D převodu musí být vybrán hodinový signál tak, aby byl dodržen minimální čas $T_{AD} = 1.6\mu s$.

Vstupní napětí A/D převodníku by nemělo překročit hodnotu napájecího napětí MCU o více než 0.3V, pak není zaručena přesnost A/D převodu dle specifikace. Na vstupní vývody je někdy vhodné zapojit vnější RC filtr. Odpor by neměl přesáhnout 10 kOhm a žádná ze součástek by neměla mít svodové parametry horší než daný obvod.

Z registrů výsledků ADRESH a ADRESL se vhodným způsobem získá po převodu výsledná číselná hodnota v rozsahu 10bitů, tj. 0 – 1023. Pro zjednodušení je možné používat pouze jeden z výše uvedených registrů výsledku (podle zarovnání výsledku vpravo či vlevo) ale tím zároveň klesá rozlišení A/D převodníku na 8bitů. Při použití vnitřního referenčního napětí 5V je pak rozlišení převodníku 4.88mV. Pro zpřesnění výsledku A/D převodu je možné pomocí softwarové rutiny opakovat tento převod několikrát za sebou (převzorkování - běžně 8-32x) a pak zpětně ho vydělit počtem měření. Tato metoda převzorkování je použita i v této DP.

Chyba A/D převodníku je absolutním součtem všech chyb (aproximační chyba, integrační chyba, rozdílové chyby, chyby rozsahu, chyby monotónosti...) a je definována jako rozdíl současného ideálního převodu od současného skutečného. Tato absolutní chyba A/D převodníku je $< \frac{1}{2} \text{ LSB}$ pro napájecí napětí rovné referenčnímu ($U_{DD} = U_{ref}$). Tato přesnost se zmenší, když se referenční napětí liší od napájecího.

Chyba kvantizace je typicky $\frac{1}{2} \text{ LSB}$ a je podstatnou chybou v procesu převodu. Jediný způsob jak tuto chybu zmenšit je zvětšení rozsahu A/D převodníku nebo převzorkování. Chyba offsetu je měřena jako rozdíl ideálního převodu proti prvnímu provedenému převodu daného napětí. Tato chyba posouvá celou přenosovou funkci převodníku. Chyba offsetu však může být zkalibrována systémem a tak potlačena. Chyba zesílení je chyba mezi posledním provedeným převodem a posledním ideálním převodem stejného signálu při odstranění chyby offsetu. Chyba zesílení je v chybě rozsahu proměnná, ale neobsahuje chybu offsetu. Tato chyba může být zkalibrována v programu. Chyba linearity nemůže být kalibrována systémem.

3.2.3. Jednotka sériové komunikace USART (RS232C)

Mikroprocesory řady 16F87x obsahují přímo na čipu modul Univerzálního Synchronního/Asynchronního Přijímače/Vysílače – USART, někdy je též nazýván jako Serial Communications Interface – SCI (sériový komunikační interface). Tento modul

může pracovat v plně duplexním asynchronním režimu a může komunikovat se systémy jako terminály CRT a nebo PC. Druhou možností je práce v poloduplexním synchronním režimu, který najde využití při umožňuje komunikaci např. s A/D, D/A převodníky, sériovými pamětmi atd.

USART modul je možné provozovat v následujících režimech:

- asynchronní (plně duplexní)
- synchronní – řídicí (MASTER, poloduplexní)
- synchronní – řízený (SLAVE, poloduplexní)

Pro potřeby řídicí jednotky konstruované v této DP se pro komunikaci s PC prostřednictvím sériové linky RS232 používá asynchronní režim modulu USART. Samotná komunikace proto používá klasického schématu 1-8-1 – (1 start bit – 8 datových bitů – 1 stop bit) a rychlost přenosu 19200 b/s. Modul USART přenáší nejméně významný bit (LSB) jako první. Vysílač a přijímač jsou při přenosu dat vzájemně nezávislé, ale používají stejnou frekvenci a datový formát. Pro volbu rychlosti přenosu se používá Generátor rychlosti přenosu (BRG). Tento generátor vytváří hodinovou frekvenci, která je buď 16 nebo 64násobkem rychlosti, s jakou jsou data vysouvána posuvným registrem a to v závislosti na hodnotě řídicího bitu BRGH v systémovém registru TXSTA<2>. Vnitřním zapojením není přímo zajištěna kontrola parity, ale je možné ji ošetřit softwarově použitím 9bitového přenosu, kde je uložena právě v posledním devátém bitu. V této DP není kontrola parity použita. Asynchronní režim je zastaven při vyvolání stavu SLEEP.

Asynchronní modul sestává z následujících částí:

- generátor rychlosti přenosu (BRG)
- vzorkovací obvod
- asynchronní vysílač (transmitter)
- asynchronní přijímač (receiver)

Asynchronní přijímač (obr. 3.4.) je schopný přijmout do registru RCREG až 2 byty (2byte FIFO) aniž by se přijatá data ztratila. V případě, že v RSR registru je detekován STOP bit třetího bytu a RCREG FIFO je stále naplněn (2 byty), je vyvolána chyba přetečení (Overrun Error) – bit OERR je nastaven na 1. Zároveň jsou data v příjmovém posuvném registru RSR ztracena. Po vyvolání této chyby je příjmová jednotka dále

3.2.5. Pomocné obvody

Mezi pomocné obvody, které zjednodušují návrh aplikace, zlepšují spolehlivost funkce, snižují cenu atd. patří tyto:

- výběr oscilátoru
- RESET systém - POR, PWRT, OST, BOR
- systém přerušení
- WatchDog obvod ("hlídací pes")
- SLEEP funkce (úsporný režim napájení)
- ochrana programového kódu
- uživatelem programovatelný ID řetězec
- ICSP programování
- LVP programování (programování nízkým napětím)
- In-Circuit Debugger (ladění programu přímo v aplikaci)

Mikrokontrolér PIC 16F87xA může pracovat se čtyřmi různými typy oscilátorů :

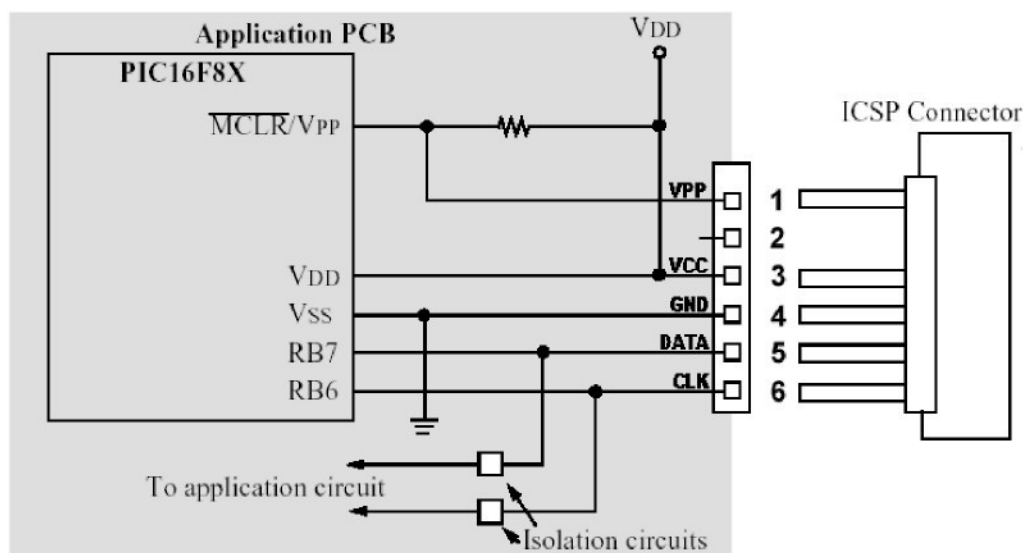
- LP – vhodný pro nižší frekvence a malou spotřebu
- XT – pro standardní frekvence krystalu a normální spotřebu (do 4Mhz)
- HS – pro maximální frekvenci krystalu (do 20Mhz))
- RC – vnější RC oscilátor

3.2.6. Programování paměti FLASH

Další velkou výhodou tohoto typu MCU je možnost jeho programování přímo v aplikaci, tedy bez vyjímání z patice a nutnosti použití klasického externího programátoru. Způsob, který toto programování umožňuje, se nazývá ICSP (In-Circuit Serial Programming – obr. 3.5). Z toho plyne hned několik výhod, jako je daleko rychlejší ladění/vývoj aplikací, nehrozí nebezpečí ulomení vývodů MCU, levné a jednoduché ICSP programátory (např. Piccolo či ASIX).

Jedinou podmínkou pro použití tohoto způsobu programování je zajištění oddělení uživatelských obvodů připojených na piny RB6 / RB7 (ICSP CLK / ICSP DATA) a pin 1 –MCLR/VPP musí být oddělen od resetovacích obvodů (např. rezistorem 10kOhm).

Během programování je na tento pin přivedeno napětí 13V a náběžná hrana a napětíová úroveň Vpp nesmí být aplikací ovlivněna.



Obrázek 3-5 – Schéma zapojení při ICSP programování

Neméně výhodnou vlastností tohoto MCU je i možnost programování sebe sama, přímo z programu uloženého ve FLASH paměti. Tato vlastnost proto například umožňuje užití tzv. BOOTLOADERu (který byl s úspěchem používán i při vývoji této DP). BOOTLOADER je malý program (zde o velikosti pouhých 256 byte), který se pomocí např. výše zmíněného rozhraní ICSP nahraje do mikrokontroléru. Další nahrávání uživatelských programů do MCU již probíhá prostřednictvím tohoto BOOTLOADERu.



Obrázek 3-6

K tomu stačí pouze spustit příslušný program v PC (viz obr. 3.6) a na MCU vyvolat Reset (tlačítko TL1). Po tomto jednoduchém úkonu se provede samotné nahrání programu do FLASH paměti MCU a ihned po jeho dokončení se právě nahraný program

rozeběhne. Použití BOOTLOADERu proto významně snižuje čas potřebný pro nahrávání a ladění programu a umožňuje i laickému uživateli provést jednoduchou změnu programu bez nutnosti vlastnit a umět používat programátor.

3.2.7. Vývojové prostředí (IDE) SourceBoost IDE

V této DP byl jako hlavní vývojový prostředek pro tvorbu programu pro MCU použit sw SourceBoost IDE (vývojové prostředí) spolu s C-kompilátorem C2C-Plus či www.sourceboost.com. Tento vývojový balík umožňuje pohodlné psaní a ladění aplikací ve vyšším programovacím jazyku C, spolu s možností použití rutin, psaných přímo v assembleru (pro časově kritické případy).

4. Návrh technických prostředků a softwaru modelu

Pro zajištění spolupráce modelové technologie s prostředím Matlab/Simulink bylo nutné zajistit tyto hardwarové a softwarové úkony:

- realizovat řídicí a výkonovou jednotku ovládající elektrický pohon (SS motor)
- navrhnout obslužný program pro mikrokontrolér řídicí jednotky
- navrhnout funkce zajišťující komunikaci z prostředí Matlabu s řídicí jednotkou
- sestavit zkušební model s regulátorem rychlosti/polohy (otáček) v Simulinku
- navrhnout a realizovat vhodné mechanické řešení celé modelové technologie

Při návrhu modelové technologie jako celku byly kladeny následující požadavky:

- jednoduchost a opakovatelnost zapojení – určeno pro výuku v laboratoři
- názornost a přehlednost (LCD displej, komentovaný zdrojový kód atd.)
- použití běžně dostupných ele. součástek
- univerzálnost použití ŘJ (vhodné i pro výuku programování MCU PIC)
- možnost přeprogramovat MCU pouze s využitím linky RS232 (BootLoader)

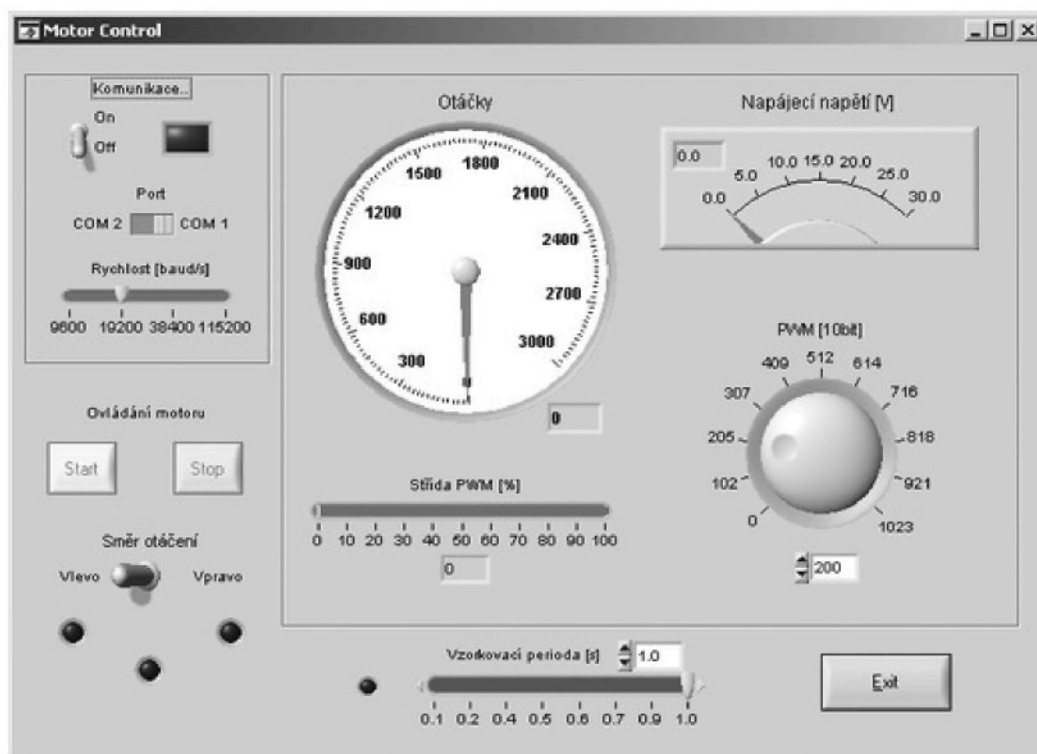
S ohledem na výše uvedené požadavky bylo sestaveno zapojení HW části tak, že obsahuje pouhé dva integrované obvody (včetně výkonové části), řídicí mikrokontrolér a dále už jen nezbytné pasivní součástky. HW část modelové technologie fyzicky sestává ze dvou desek plošných spojů – řídicí části s MCU a výkonové části vzájemně propojené

plochým kabelem. Bylo zkoušeno i jednodeskové řešení, to se ovšem neosvědčilo z důvodu elektromagnetického rušení vznikajícího při generování PWM signálu.

Pro zajištění komunikace z prostředí Matlabu s HW částí modelu byly nejprve vytvořeny základní M-funkce, zajišťující inicializaci a přenos dat po sériové lince. Tyto funkce byli následně použity (především kvůli názornosti) přímo z modulu Simulink. Zde bylo s úspěchem použito tzv. S-funkcí, které umožňují použít (volat) již existující elementární M-funkce z modelu vytvořeného v prostředí Simulinku.

Dále bylo nutné vytvořit obslužný program pro mikrokontrolér, který má za úkol zajistit komunikaci po sériovém kanále s osobním počítačem z prostředí Matlab.

Pro usnadnění oživování a seřizování ele. pohonu byl též vytvořen program pro PC MotorControl. Tento software, napsaný ve zkušební verzi vývojového prostředí LabWindows/CVI 8.0 umožňuje i bez nainstalovaného programu Matlab otestovat komunikaci mezi PC a řídicí/výkonovou jednotkou, umožňuje sledovat napájecí napětí motoru, měnit výkon a směr otáčení, sledovat aktuální střidu atd. Tento program je jako instalační balíček obsažen na CD přiloženém k této práci.



Obrázek 4-1 Prostředí programu Motor Control

4.1. Návrh řídicí jednotky – hardware (HW)

Řídicí jednotka zajišťuje komunikaci po sériové lince, ovládá výkonovou jednotku a zajišťuje další pomocné funkce (obsluha LCD displeje, přepínání relé tachodynamu, obsluha LED diod signalizující sériovou komunikaci atd.). Základem ŘJ je mikrokontrolér spolu s obvodem zajišťujícím převod napěťové úrovně TTL (0 a 5V) na RS232 kompatibilní (-12V a +12V). Dále deska ŘJ obsahuje znakový dvouřádkový (2x16 znaků) LCD displej, který zobrazuje nejdůležitější provozní údaje (napájecí napětí motoru, otáčky, střidu PWM a směr otáčení). Tlačítko RESET resetuje mikrokontrolér a usnadňuje především nahrávání nového programu do MCU pomocí BootLoaderu. Obsahuje též 9pin Cannon konektor, který slouží k propojení s PC. Tento konektor je typu Female (zásuvka) a tudíž nehrozí záměna kabelu (na straně PC je Male – vidlice). Kompletní schéma zapojení vytvořené programem Eagle v. 4.14 je uvedeno v příloze č.1, seznam součástek je uveden v příloze č.3.

4.1.1. Vstupní signály a jejich zpracování

ŘJ zpracovává následující vstupní signály :

- napětí z tachodynamu - analogový signál – 0 až 5V – pin RA0
- napájecí napětí motoru - analogový signál – 0 až 5V – pin RA3
- RX (příjmový) signál z RS 232 kanálu – digitální signál (0/5V) – pin RC7

Pro první dva uvedené analogové vstupní signály je nutné zajistit, aby napětí bylo vždy v rozsahu 0 až 5V (kvůli A/D převodníku MCU). Napětí z tachodynamu je proto zmenšováno odporovým děličem, tvořeným trimrem TR1 a pevnými odpory R1 a R2. Odporový trimr byl zvolen záměrně především pro usnadnění nastavování vstupního napětí pro A/D převodník, resp. jeho převodu na číselnou hodnotu. Obslužný program pro mikrokontrolér je nastaven tak, že výstupnímu napětí tachodynamu o velikosti 3V odpovídá číselná hodnota 1000 (použité konkrétní tachodynamo na SS motoru mělo převod právě 3V/1000 ot/min). Při prvním oživení ŘJ je proto nutné nastavit trimr TR1 do takové polohy, aby číselná hodnota otáček odpovídala napětí vystupujícímu z použitého tachodynamu při daných otáčkách. Pro tuto kalibraci je vhodné použít již dříve zmiňovaný program MotorControl (MC), kde při nastavování postupujeme následovně:

- nastavit trimr TR1 na maximální odpor
- v programu MC nastavit otáčky (PWM 10bit) tak, aby tachodynamo generovalo napětí, odpovídající např. 1000 ot/min
- trimrem TR1 otáčet doleva tak dlouho, až otáčkoměr v MC ukazuje správnou hodnotu zvolených otáček

Tento postup je díky zvoleným hodnotám odporového děliče použitelný pro tachodynamo generující napětí max. 12V. V opačném případě je nutné zvolit hodnotu součástek děliče tak, aby vstupní napětí A/D převodníku nepřesáhlo 5V (i přesto je jako ochrana A/D vstupů použita externí Zenerova dioda 5,1V).

Napájecí napětí motoru je pro A/D převodník upraveno podobným odporovým děličem, tentokrát bez použití odporového trimru. Příslušné vstupní napětí převedené na číselnou hodnotu se proto musí programově upravit tak, aby odpovídalo skutečnému napětí ve voltech. V našem případě je nutné výsledek A/D převodu vydělit konstantou 147.5 (tato konstanta vychází z převodu odporového děliče a počtu převzorkování při A/D převodu). Maximální napájecí napětí motoru je proto asi 30V (opět závisí na hodnotách odporů děliče). Výsledná hodnota napájecího napětí motoru se zobrazuje jak v programu MC tak i na LCD displeji ŘJ.

Posledním vstupním signálem je datový tok, přicházející z převodníku úrovně RS232/TTL. Jeho zpracování je díky použitému mikrokontroléru velice snadné a bylo již popsáno v kapitole jednotky sériové komunikace.

Na desce ŘJ se dále nacházejí konektory, pomocí nichž jsou přístupné některé nevyužité vývody (RB0,RB1,RC4,RC5) mikrokontroléru. Je jich proto možné využít např. pro připojení inkrementálního čidla otáčení (čítání impulsů), digitálního čidla pro měření teploty (např. SMT160) atd. Podobně je na desce ŘJ umístěna i dvojice konektorů s napětím 5V (spolu s GND) pro možnost napájení těchto externích čidel či dalších uživatelských aplikací.

4.1.2. Výstupní signály a jejich generování

ŘJ generuje a obsluhuje následující signály:

- PWM signál pro řízení výkonové jednotky (VJ) – pin RC1 a RC2
- Enable signál – zapíná/vypíná výkonovou část obvodu L298 – pin RC0
- přepínání polarit napětí z tachodynamu – pomocí relé – pin RC3

- TX (vysílání) signál pro RS232 komunikaci – pin RC6
- obsluha LCD displeje (4bitová komunikace) – piny RB2-RB7

10bitový PWM signál je generován řídicí jednotkou na základě dat přijatých z PC po sériové lince. Pro umožnění změny směru otáčení se využívají obě PWM jednotky použitého MCU. Která PWM jednotka se má v danou chvíli použít, program v MCU rozpozná dle odpovídajícího příkazu, zaslaného z PC. Právě aktivní PWM jednotka generuje signál s proměnnou střídou dle hodnot přijatých z PC, zatímco druhá je vypnutá a výstupní signál z ní má úroveň logické 0.

Pro aktivaci výkonové jednotky slouží signál Enable, generovaný mikrokontrolérem na pinu RC0. Pokud je Enable signál v log. 0, motor stojí i když MCU posílá z PWM jednotek signál na změnu rychlosti.

Přepínání polarity výstupního napětí tachodynamu je řešeno pomocí relé s dvojicí přepínacích kontaktů. Tento způsob přepínání byl zvolen s ohledem na skutečnost, že při změně směru otáčení motoru se obrací i polarita napětí vystupujícího z tachodynamu, dále vedeného do A/D převodníku MCU. Tento převodník by pak nebyl schopen zpracovat záporné napětí. Jednodušším řešením by bylo použití např. diodového usměrňovače. Tento způsob byl v první verzi ŘJ vyzkoušen, ale neosvědčil se díky nelinearitě diod a měření otáček tak bylo velice nepřesné. Použití relé pro změnu polarity napětí z TD sebou přináší i jisté úskali. Především je nutné zajistit správné přepnutí relé při roztočení motoru tak, aby se na vstupu A/D převodníku objevilo vždy pouze kladné napětí. Podobně i při vypnutí motoru nesmí relé přepnout dříve, než se motor zastaví. Tyto požadavky proto zajišťuje program v MCU. Připojení motoru k VJ je realizováno čtyřpinovým konektorem se zámkem, který zabraňuje nesprávnému připojení (přepólování vůči TD). Podobně je řešeno i připojení tachodynamu třípinovým zámkovým konektorem.

TX signál (odesílání dat) je podobně jako RX signál velice jednoduché generovat a byl taktéž popsán v jednotce sériové komunikace. Pro vizuální kontrolu, zda-li mikrokontrolér odesílá/přijímá data, jsou na desce ŘJ zapojeny dvě LED diody. Při nečinnosti na sériové lince jsou tyto LED diody zhaslé (na pinech RX a TX je trvalá logická 1), při přenosu dat blikají dle frekvence odchozích/příchozích dat (reagují na logickou 0).

4.1.3. Technické parametry řídicí jednotky

Hlavní parametry modulu řídicí jednotky:

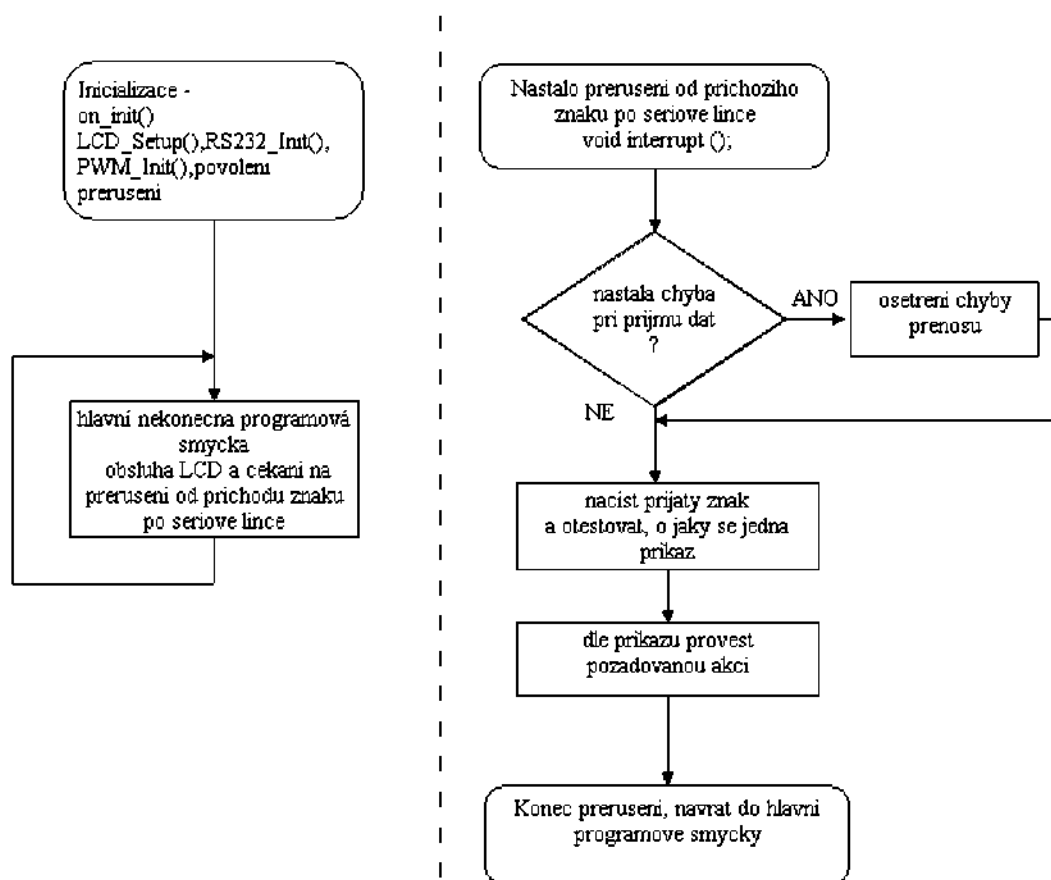
- napájecí napětí – 9-12V (max. 20V)
- odběr proudu max. 60mA (při 9V)
- rozsah provozních teplot 0 – 60°C
- RS232: komunikační schéma: 8-1-1 při rychlosti 19200 b/s
- LCD displej 2x 16 znaků

4.2. Návrh řídicí jednotky – software (SW)

Návrh ovládacího softwaru spočíval ve vytvoření vhodného programu pro použitý mikrokontrolér PIC16F876A. Na tento program byly kladeny následující požadavky:

- zajistit komunikaci po RS232 s PC
- řídit výkonovou jednotku (PWM)
- měřit otáčky ze signálu z tachodynamu
- měřit napájecí napětí motoru
- ovládat LCD displej

Základem obslužného programu je moment vyvolání přerušení od přijatého příkazu (znaku) po sériové lince. Nastane-li tato situace, program na základě přijatého příkazu provede požadovanou akci (roztočí motor, změří otáčky atd.). Funkci programu popisuje vývojový diagram na obrázku 4.2.



Obrázek 4-2

O tom, jaký úkon má MCU na základě přijatého příkazu provést se rozhoduje ve funkci interrupt(). Seznam jednotlivých příkazů je uveden v tabulce 4.1.

příkaz ASCII	vrací [byte]	popis
O	2	vrací aktuální hodnotu otáček (16bit)
P	2	vrací aktuální hodnotu napětí motoru (16bit)
Lxxxx	-	nastaví PWM výstup pro chod vlevo (xxxx = 0 až 1023)
Rxxxx	-	nastaví PWM výstup pro chod vpravo (xxxx = 0 až 1023)
E	-	zapíná výkonovou jednotku
l	-	zastavuje chod motoru vlevo
r	-	zastavuje chod motoru vpravo
e	-	vypne výkonovou jednotku (i motor)
t	2	test komunikace, do PC vrátí "OK"

Tabulka 4.1 – Popis příkazů

Ovládací program pro mikrokontrolér byl vytvořen ve vývojovém prostředí SourceBoost IDE, pomocí jazyka C a assembleru. Výpis zdrojového kódu zde vzhledem k jeho délce není uveden; veškeré zdrojové kódy jsou umístěny na CD. Hlavní program se nachází v souboru main.c, který používá soubory:

- P16f876a.h – definiční soubor použitého MCU
- display.c – obsahuje funkce zajišťující obsluhu LCD displeje (4bit komunikace)
- RS232_hw.c – obsahuje funkce pro komunikaci po sériové lince
- pwm_lib.c – obsahuje funkce pro generování PWM signálu

Po zkompilování zdrojového kódu vygeneruje program SourceBoostIDE soubor typu *.HEX, který se pomocí utility TinyBootLoader nahraje do mikrokontroléru.

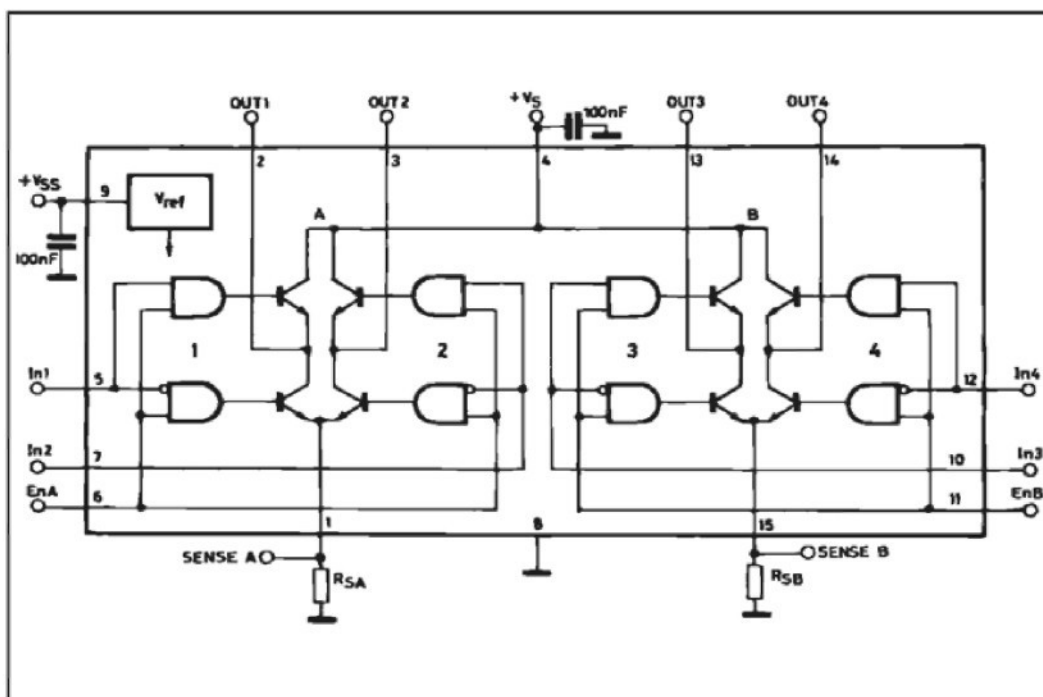
4.3. Návrh výkonové jednotky (VJ)

Výkonová jednotka slouží k samotnému řízení stejnosměrného motoru. Umožňuje otáčení oběma směry (reverzace), dokáže motor zastavit dvěma způsoby - zrychleně (brzdění pomocí protiproudu) nebo volným dotočením. Výkonová jednotka je řízena přímo mikrokontrolérem a to dvojicí PWM jednotek a signálem Enable. Veškerou činnost obstarává monolitický výkonový integrovaný obvod L298 [8] určený především právě pro řízení SS nebo krokových motorů. Schéma zapojení VJ je uvedeno v příloze č.2, seznam součástek je uveden v příloze č.3.

4.3.1. Obvod L298

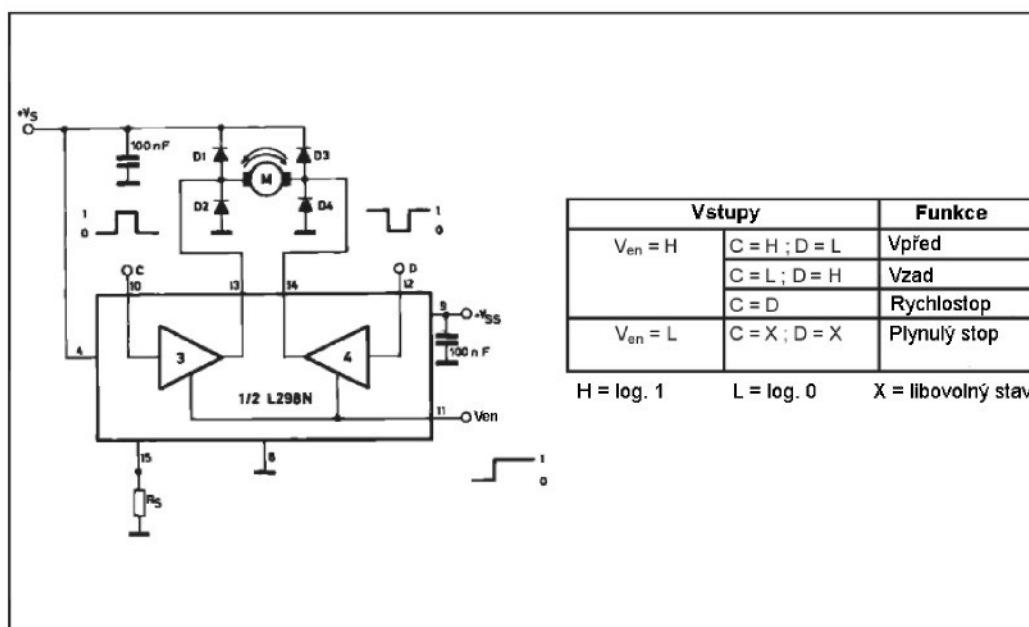
Základem VJ je integrovaný obvod L298 v pouzdře Multiwatt 15. Tento obvod obsahuje především dvojici nezávislých H-budičů (viz obrázek 4.3) umožňujících ovládat dvojici motorů; každý je schopen zpracovat proudy do 2A. Pro zvětšení proudové zatížitelnosti na 4A je možné paralelně spojit oba výstupy a ovládací vstupy. Díky TTL logice je možné obvod připojit přímo na výstupní piny mikrokontroléru. IO L298 též obsahuje obvody, které potlačují šum na logických vstupech a je proto odolný proti rušení (logickou 0 obvod rozpozná ještě při napětí 1.5V, logická 1 je od 2.3V). Celkem se k řízení tohoto obvodu používají tři signály: (jejichž přítomnost na desce VJ je signalizována pomocí trojice LED diod)

- IN 1(4) – řízen pinem RC1 MCU (PWM jednotka CCP1) – červená LED
- IN 2(3) – řízen pinem RC2 MCU (PWM jednotka CCP2) – zelená LED
- En A(B) – řízen pinem RC0 MCU – zapíná/vypíná výkon. část – žlutá LED



Obrázek 4-3 – Vnitřní zapojení obvodu L298

Obvod L298 se v této práci používá ve stejném zapojení, jaké doporučuje výrobce (obr. 4.4). Použitím tohoto obvodu rovněž odpadá starost s potlačením death-time (doby, kdy jsou sepnuty oba páry tranzistorů v H-můstku a teče jimi zkratový proud).



Obrázek 4-4 – Doporučené zapojení L298

Obvod L298 může mít několik provozních režimů, tak jak je uvedeno v tabulce na obrázku 4.4. Otáčení vpřed (vlevo) či vzad (vpravo) bylo již popsáno v kapitole výstupních signálů. Zajímavé jsou i dvě možnosti zastavení motoru. První z nich je tzv.

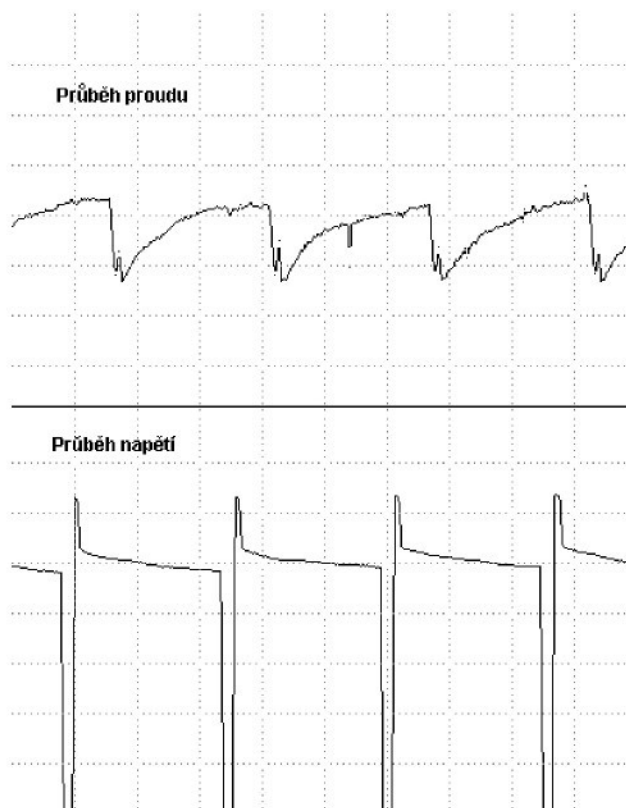
rychlostop. Při tomto zastavení motoru se používá brzdění protiproudem a docílí se ho tak, že Enable signál (V_{en}) zůstane v log. 1 a vstupy C i D se nastaví na stejnou logickou úroveň (tzn. aktivní PWM vstup se “vynuluje“ – příkazy z PC ‘l’ příp. ‘r’). Tohoto způsobu zastavení je nutné použít uvážlivě tak, aby nedošlo k překročení mezního odběru proudu obvodu. V případech, kde není kritická doba zastavení a nevadí volné “dotočení” motoru, je vhodné použít plynulý stop. Ten se aktivuje přivedením log. 0 na Enable (V_{en}) vstup (příkaz ‘e’ z PC).

4.3.2. Průběhy napětí a proudu motoru

Díky řízení výkonové jednotky PWM signálem jsou i průběhy napětí a proudu závislé na aktuální střídě řídicího signálu. Průběh proudu byl snímán na rezistoru R_s (viz obr. 4.4), průběh napětí byl spínám přímo na svorkách motoru.



Obrázek 4-5 - Průběhy napětí a proudu při 1000ot/min (střída cca 40%)



Obrázek 4-6 Průběhy napětí a proudu při 2700ot/min (střída cca 97%)

4.3.3. Mechanická konstrukce výkonové jednotky

Podobně jako řídicí jednotka je i výkonová jednotka zhotovena na desce plošných spojů o rozměrech 160x110mm. Obsahuje dva výkonové konektory pro připojení napájecího napětí motoru a pro samotný motor. Spolu s nimi deska obsahuje i konektor pro připojení tachodynamu a 14pinový systémový konektor, propojující VJ s deskou ŘJ. Jak čtyřpinový napájecí, třípinový konektor tachodynamu i 14pin konektor jsou typy se zámkem, zabraňujícím nesprávné zapojení. Hlavní vstup napájecího napětí je řešen klasickou šroubovací svorkovnicí. Případnému přepólování je zde zamezeno použitím ochranné diody s malým úbytkem napětí.

Deska VJ obsahuje též relé pro přepínání polaritu výstupního napětí tachodynamu. Princip jeho funkce byl již popsán v kapitole vstupních signálů.

Nezbytnou součástí výkonové jednotky je i tavná pojistka 2.5A, zamezující překročení max. povoleného proudu obvodu L298.

Pro chlazení obvodu L298 slouží vhodný hliníkový profil, dimenzovaný na nepřetržitý provoz použitého nezatíženého motoru. V případě potřeby je možné použití aktivního chlazení pomocí 40mm ventilátorem

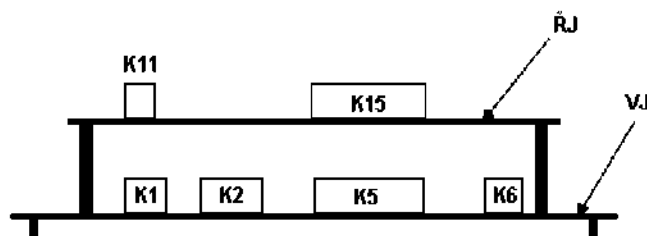
4.3.4. Technické parametry výkonové jednotky

Hlavní parametry modulu řídicí jednotky:

- max. vstupní napětí: 46 V
- provozní napětí: 20V (dle použitého motoru)
- max. proud obvodem: 2 (4) A
- max. napětí na logických vstupech: 7 V
- max. zatížení kontaktů relé: 0.3A při 60V

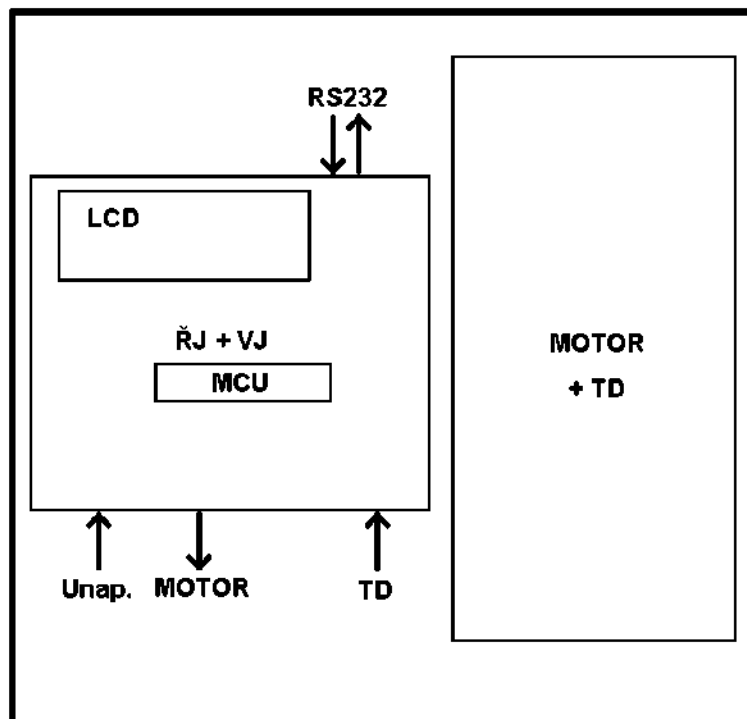
4.4. Mechanické uspořádání a propojení řídicí a výkonové jednotky

Deska řídicí jednotky je pomocí distančních sloupků délky 40mm umístěna nad desku výkonové jednotky. Obě desky jsou propojeny pomocí plochého 14žilového kabelu (7 signálů + 7 GND). Schéma uspořádání výkonové a řídicí jednotky s popisem jednotlivých konektorů je uvedeno na obr. 4.7.



Obrázek 4-7

Použitý motor PIVTM6/25-2 je uchycen do plastového držáku, použitého z šasi speciálního radiálního ventilátoru, používaného pro odvětrávání prachu. Spolu s tímto plastovým držákem je motor přichycen ke dřevotřískové desce o rozměrech 250x350mm. Na této desce je umístěn i blok řídicí a výkonové jednotky spolu s propojovacími kabely. Celá modelová technologie proto tvoří jeden kompaktní celek (obr. 4.8).



Obrázek 4-8

4.5. Návrh modelu v prostředí MATLAB/Simulink

Pro zajištění komunikace po sériové lince mezi řídicí jednotkou a prostředím Matlab/Simulink bylo nutné vytvořit základní M-funkce, které slouží pro inicializaci COM portu a pro odesílání/příjem dat :

- COM_Config.m – nastavuje základní parametry přenosu po RS232
- w8_pic.m – posílá 1byte příkaz a int(8bit) hodnotu do MCU
- w16_pic.m – posílá 1byte příkaz a uint(16bit) hodnotu do MCU
- r_pic.m – přijímá 16bit. hodnotu zaslanou mikrokontrolérem

Výpis zdrojového kódu M-funkce COM-Config.m, sloužící pro inicializaci COM portu:

```
function s=COM_Config(port,rychlost)
warning off MATLAB:serial;:fread:unsuccessfulRead
s = serial(port,'BaudRate',rychlost,'Parity','none','DataBits',8,'StopBits',1);
s.DataTerminalReady='off'; %DTR=0
s.FlowControl='none';
s.InputBufferSize=32;
s.OutputBufferSize=32;
```

```
s.TimeOut=5;
```

```
fopen(s);
```

Příklad volání funkce COM_Config.m:

```
s=COM_Config('COM1',19200);
```

Výpis zdrojového kódu M-funkce w_pic.m:

```
function w8_pic(s,k,x)
```

```
fwrite(s,k,'uchar'); % pošle řídící byte (příkaz)
```

```
fwrite(s,x,'uint8'); % pošle 1-byte hodnotu (8bit unsigned integer)
```

Příklad volání funkce w_pic.m:

```
w8_pic(s,L,212); % odešle příkaz O s hodnotou 212
```

Výpis zdrojového kódu M-funkce w16_pic.m:

```
function w16_pic(s,k,x)
```

```
fwrite(s,k,'uchar'); % pošle řídící byte (příkaz)
```

```
fwrite(s,x,'uint16'); % pošle 2-byte hodnotu (16bit unsigned integer)
```

Příklad volání funkce w16_pic.m:

```
w16_pic(s,L,980); % odešle příkaz L s hodnotou 980
```

Výpis zdrojového kódu M-funkce r_pic.m:

```
function data=r_pic(s,c)
```

```
fwrite(s,c,'uchar'); % poslán požadavek (příkaz) - 1byte
```

```
data=fread(s,1,'uint16'); % data přijatá z MCU do proměnné data
```

Příklad volání funkce r_pic.m:

```
otacky=r_pic(s,O); % odešle příkaz O a čeká na příjem dat z MCU
```

Jelikož veškeré řízení polohy a rychlosti musí obstarávat model vytvořený v Simulinku, byly navrženy a vytvořeny následující modely:

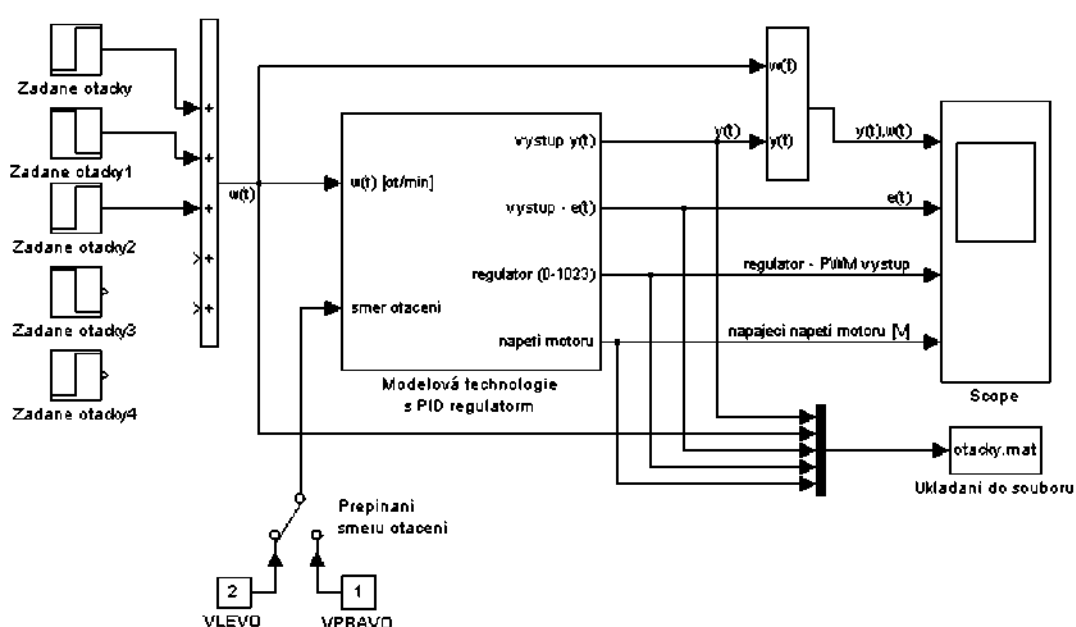
- pro rychlostní regulaci – ‚motor_PID.mdl‘
- pro polohovou regulaci – ‚motor_poloh.mdl‘

Oba výše uvedené modely budou popsány v nadcházejících kapitolách.

4.5.1. Řízení rychlosti – model 'motor_PID.mdl'

Model zastává funkci regulátoru rychlosti (otáček) a byl vytvořen tak, aby maximálně využíval vlastností zkonstruované řídicí a výkonové jednotky a zároveň byl svou názorností vhodný pro použití při výuce v laboratoři. Zvolené řešení tohoto modelu není jediným možným, uživatel si může sestavit model jiný, dle svých specifických požadavků. Jedinou podmínkou je nutnost dodržet komunikační protokol řídicí jednotky, případně si naprogramovat jiný. Schéma kompletního modelu 'motor_PID' je uvedeno na obrázku 4.9.

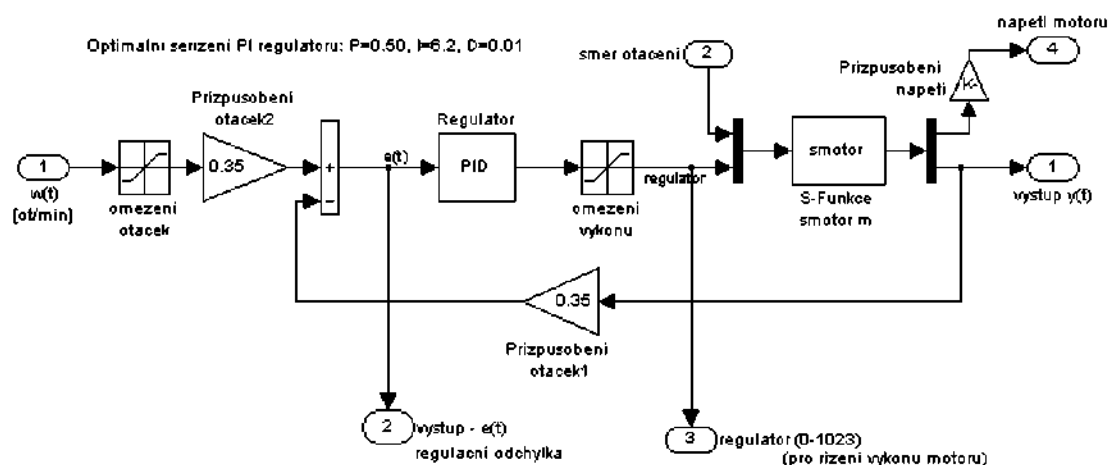
Optimalní parametry simulace: Step time 0.05, fixed step (Ode1)



Obrázek 4-9

4.5.2. Subsystém 'Modelová technologie s PID regulátorem'

Ze schématu na obr. 4.9 vyplývá, že v modelu je použit subsystém (obr. 4.10), který obsahuje rychlostní smyčku s PI(D) regulátorem a modelovou technologii tvořenou řídicí a výkonovou jednotkou spolu s motorem. Tento subsystém byl vytvořen především pro usnadnění použití a zlepšení přehlednosti modelu. Všechny výstupní veličiny jsou zobrazovány pomocí bloku Scope a zároveň ukládány do souboru pro další zpracování.



Omezení otacek je nastaveno dle zvoleného motoru, standardně jsou povolené otáčky v mezích 200-2500

Omezení výkonu je nastaveno na 0-1023 (tj. stáda 0-100%). Odpovídá to 10bit. rozsahu PWM řídicí jednotky.

Konstanty přizpůsobení otacek jsou určeny z měření na konkrétním motoru, kde např. výkonu 500 ~ asi 1400 ot/min, 1023 ~ cca 2930 ot/min atd. Z toho konstanta $K=1023/2930=0.35$.

Obrázek 4-10

Subsystem obsahuje následující vstupy :

- žádaná hodnota otáček - ,w' [ot/min]
- přepínač směru pohybu (otáčení)

Ze subsystému jsou vyvedeny následující výstupy:

- okamžitá hodnota otáček ,y' – [ot/min]
- regulační odchylka ,e'
- okamžitá hodnota výkonu (PWM) – akční veličina 'u'
- napájecí napětí motoru [V]

V subsystému jsou použity dva bloky Saturace (Saturation). První z nich (omezení otacek) zamezuje zadání nepřipustných či nesmyslných údajů o požadovaných otáčkách 'w'. Ještě důležitější je druhý blok saturace (omezení výkonu), který omezuje velikost akční veličiny (PWM hodnota odesílaná do ŘJ). Jelikož PWM jednotka použitého MCU dokáže zpracovat max. 10bit hodnotu, je horní hranice saturace nastavena na hodnotu 1023 a spodní na 0. Z toho také vyplývá "jemnost" samotné regulace; tzn. minimální změna napětí na motoru je rovna asi 19,6mV (to odpovídá změně střidy PWM signálu o 1 při napájecím napětí výkonové jednotky 20V).

Konstanty subsystému ,prizpusobeni otacek1 až 3' jsou použity pro přepočet otáček na výkon. Tento přepočet je nutný pro správnou funkci PID regulátoru, jehož výstupem

je PWM signál, který dále zpracovává ŘJ. Konstanta „prizpusobeni napeti“ přepočítává přijaté data z ŘJ na hodnotu napětí ve voltech.

Pro regulaci otáček je v tomto subsystému obsažen PI(D) regulátor. Změna jeho parametrů se ihned projeví na chování celého modelu a studentům tak umožní lepší porozumění jednotlivých souvislostí této regulace.

4.5.3. S-Funkce 'smotor.m'

Nejdůležitějším blokem v subsystému je S-Funkce 'smotor.m'. Tento blok zajišťuje vlastní komunikaci mezi ŘJ a Simulinkem. K tomu používá již dříve popsané základní komunikační M-Funkce. Princip S-Funkcí spočívá právě v možnosti volání již dříve vytvořených funkcí v jazyku C, C++, Matlab, Ada a Fortan v prostředí Simulinku. S-Funkce použitá v této práci se jmenuje 'smotor.m' a musí být umístěna v adresáři spolu s hlavním modelem 'motor_PID.mdl' ('motor_poloha.mdl').

S-Funkce má přesně definovaný tvar, kde se jednotlivé akce provádějí na základě příznaků, které se začnou generovat po spuštění simulace. Nejdůležitějšími jsou tyto příznaky:

- 0 – inicializace – provede se pouze jednou ihned po spuštění simulace
- 2 – aktualizace (update) – provede se v každém kroku simulace
- 3 – výstup (output) – vrací výstupní hodnoty do vektoru výsledků (v každém kroku simulace)
- 9 – ukončení (termination) – vyvolá se po ukončení simulace

Jak vyplývá z výše uvedeného, po spuštění simulace se volá funkce COM_Config (nastavení COM portu) a nulují se proměnné, viz zdrojový kód níže :

```
case 0,  
    [sys,x0,str,ts]=mdlInitializeSizes;  
    disp('Inicializace COM portu a promennych');  
    disp('-----');  
    akt_otacky=0;  
    n=1; % pomocna promenna pro zamezeni prepnuti smeru za chodu motoru  
    vystup=zeros(2,1); % inicizalizace vektoru vystupu  
    s=COM_Config('COM1',19200);  
    fwrite(s,'E','uchar'); % posle prikaz ridici byte pro zapnuti RJ (otacky = 0)
```

vystup(1,1)=r_pic(s,'P'); % aktualni napeti motoru ulozit do vektoru

Po inicializaci již začíná samotný běh simulace, který se opakuje po vzorkovací periodě, jejíž hodnota se nastavuje v parametrech simulace. V každém kroku je proto vyvolán příznak Update, kde se provedou následující akce (dle zdrojového kódu):

case 2,

if (n==1) smer = u(1,1); n=0; end; % zjistí smer otáčení ze vstupního vektoru

if (smer == 1) w16_pic(s,'R',u(2,1)); % PWM výstup - VPRAVO

else w16_pic(s,'L',u(2,1)); end; % PWM výstup - VLEVO

vystup(2,1)=r_pic(s,'O'); % aktualni otacky motoru ulozit do vektoru vystupu

vystup(1,1)=r_pic(s,'P'); % aktualni napeti motoru ulozit do vektoru vystupu

Zároveň je v každém kroku simulace vyvolán příznak 3 – výstup, který zajišťuje zasílání výstupních hodnot z S-Funkce do modelu :

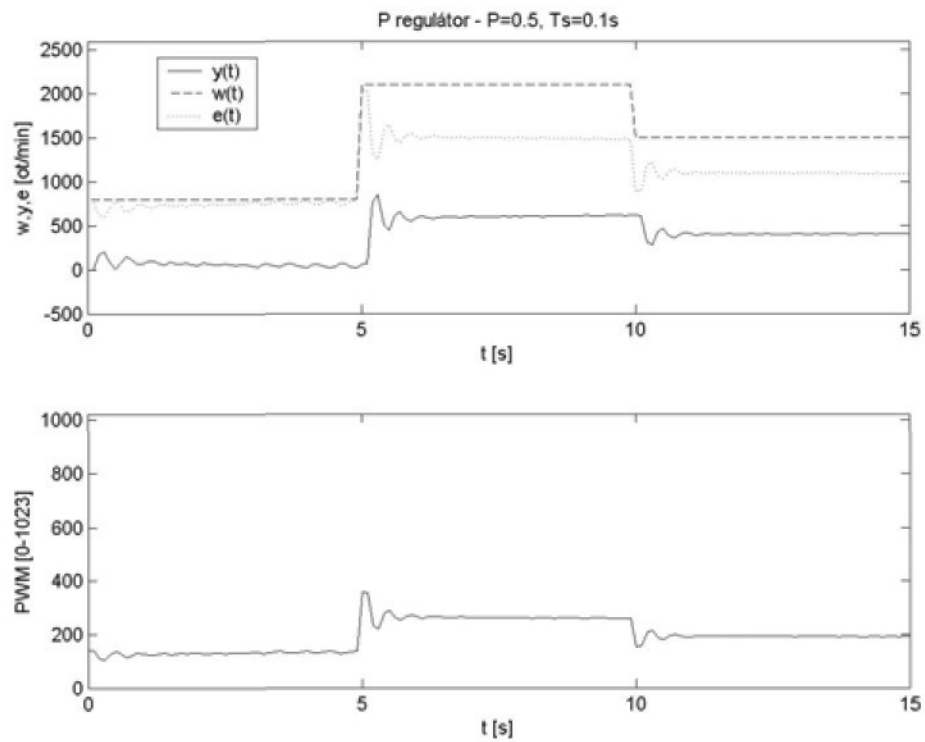
case 3,

sys = vystup; % odesle vektor vystupu do modelu

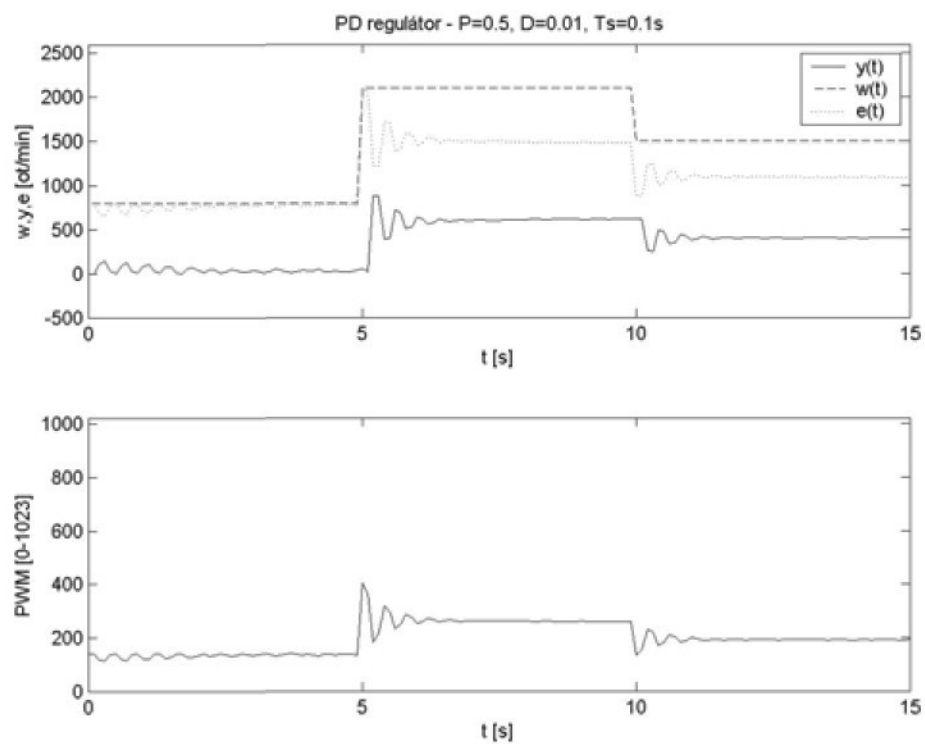
Výsledné otáčky a napětí, vystupující z S-funkce coby vektor, je nutné rozdělit pomocí bloku Demux na jednotlivé skalární veličiny. Tyto jsou pak vyvedeny ze subsystému k dalšímu zpracování.

4.5.4. Simulace za použití P, PD, PI a PID regulátoru

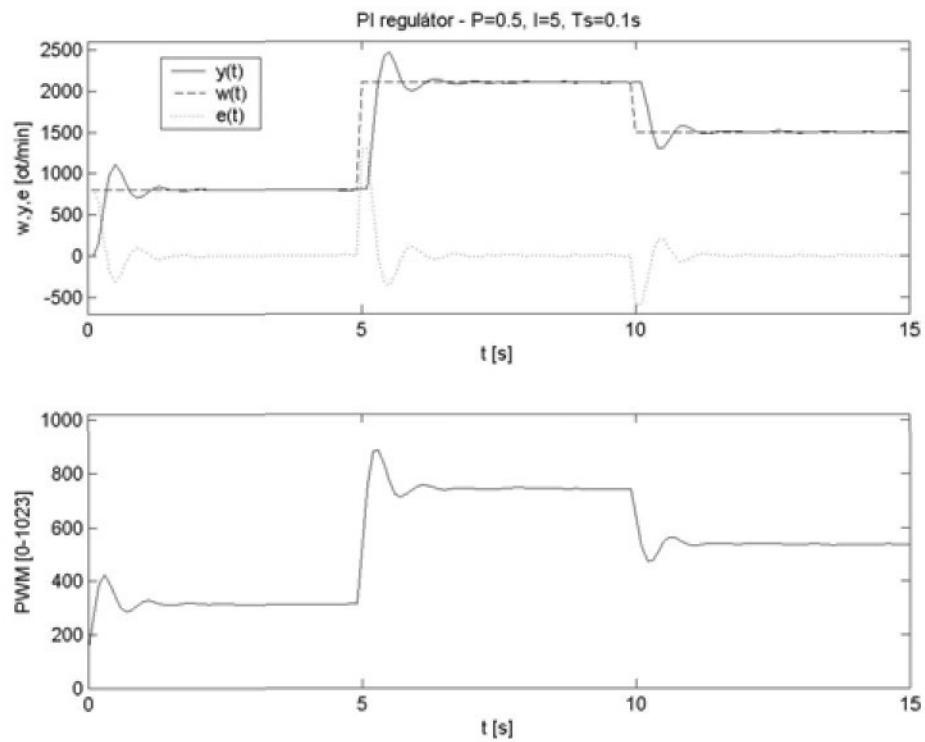
Pro ověření funkčnosti regulace rychlosti byly postupně měněny parametry v bloku PID regulátoru a sledoval se vliv změny jednotlivých složek na chování celé soustavy. Na obrázcích 4.11 až 4.14 jsou uvedeny průběhy žádané ,w', výstupní ,y' a akční (PWM) ,u' veličiny spolu s regulační odchylkou ,e' pro jednotlivé typy regulátorů v reakci na skokovou změnu žádané veličiny (otáček).



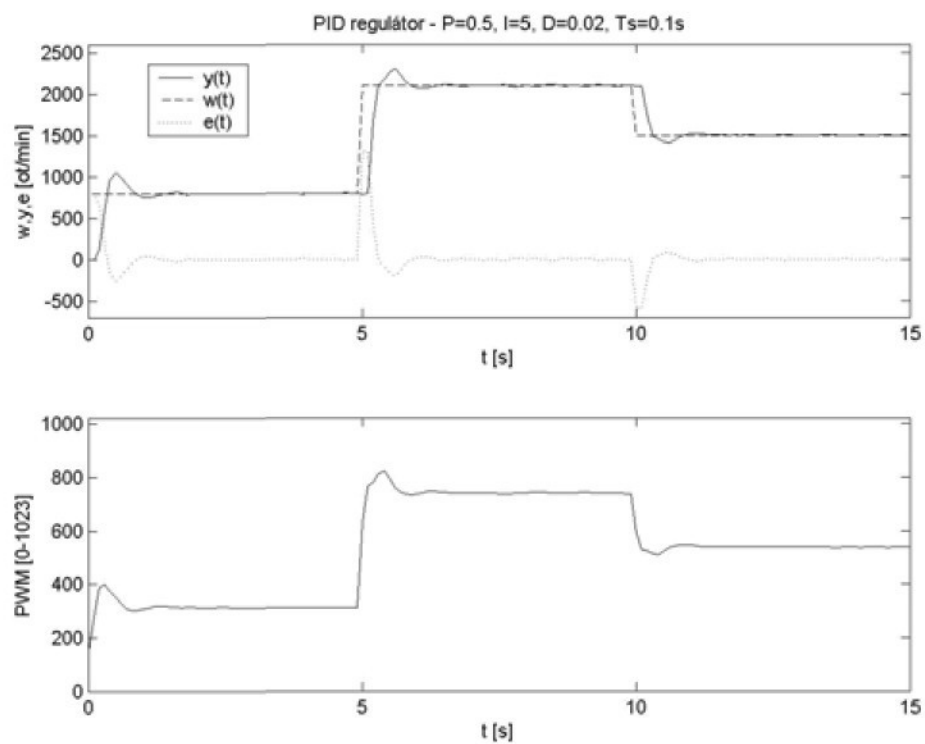
Obrázek 4-11 - P regulátor



Obrázek 4-12 - PD regulátor



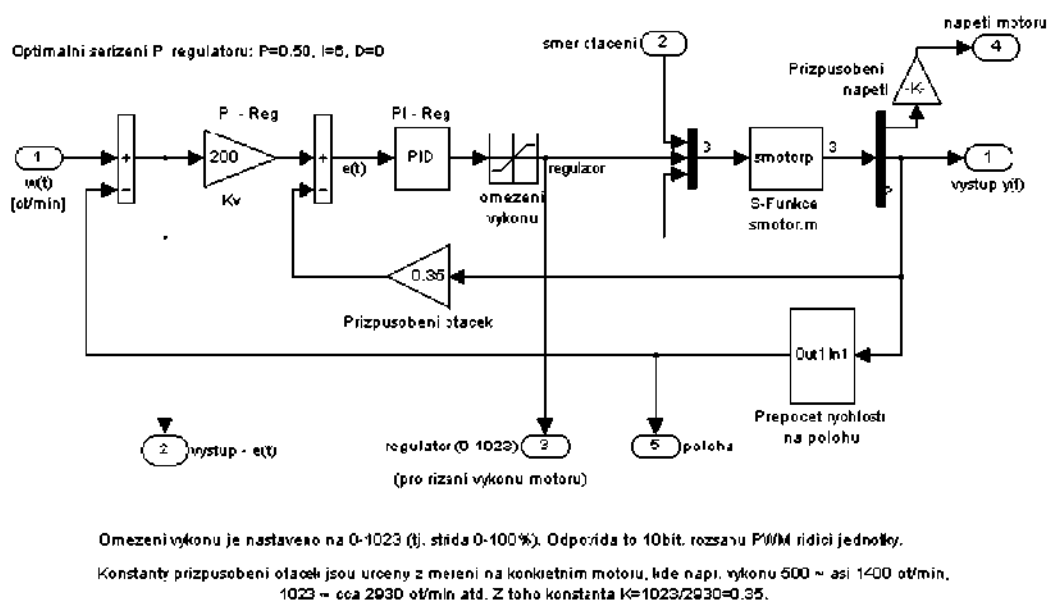
Obrázek 4-13 - PI regulátor



Obrázek 4-14 - PID regulátor

4.5.5. Řízení polohy – model 'motor_poloha.mdl'

Podobně jako pro regulaci rychlosti byl sestaven i model pro řízení polohy. Základy obou modelů jsou totožné, jiný je pouze subsystém, který obsahuje mimo rychlostní smyčky i smyčku polohovou. Schéma subsystému modelu je uvedeno na obrázku 4.15.



Obrázek 4-15

Jak vyplývá z obr. 4.15, nejdůležitějšími bloky subsystému řízení polohy jsou polohový a rychlostní regulátor. Polohový regulátor je tvořen P-regulátorem se zesílením K_v (rychlostní konstanta). Hodnota konstanty byla pro dané podmínky simulace a daný SS motor zjištěna experimentálně. Pomocný subsystém 'Prepocet rychlosti na polohu' zajišťuje výpočet dráhy (polohy) v závislosti na otáčkách a čase. Jelikož použitý SS motor nepoháněl žádné další zařízení, je dráha počítána v radiánech. Proto se musí i žádaná hodnota polohy zadávat v radiánech.

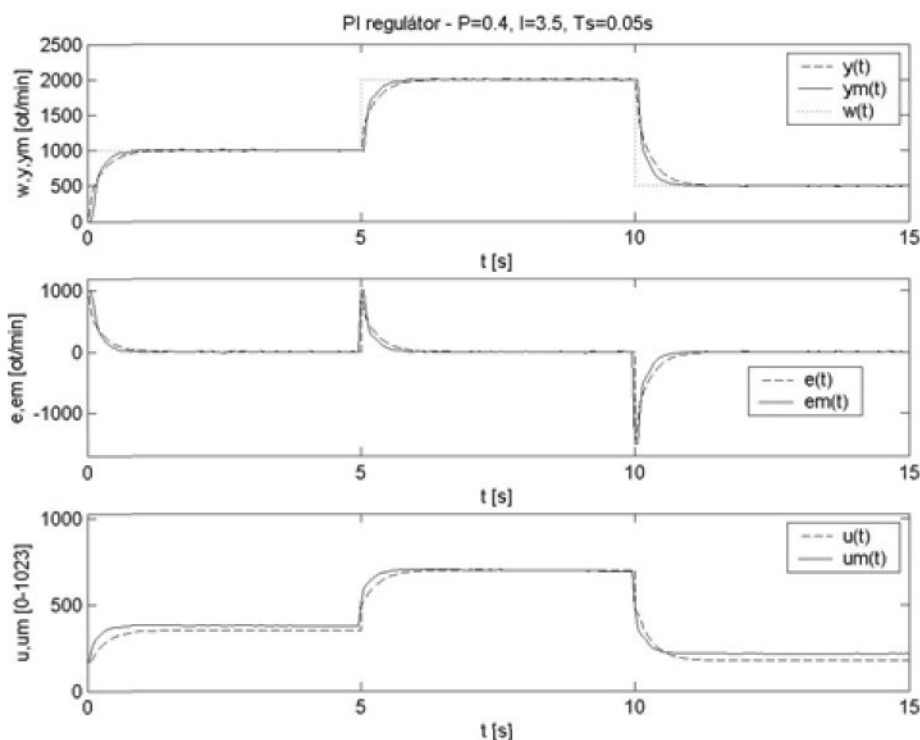
Rychlostní regulátor je totožný jako v modelu řízení rychlosti, pouze D-složka je nulová.

4.6. Ověření funkčnosti a verifikace navrženého modelu

Pro ověření funkčnosti celé modelové technologie byl pomocí M-funkce 'idT.m' (soubor přiložen na CD) zjištěn obrazový přenos soustavy (SS motoru) a to ve tvaru:

$$F(s) = \frac{1}{0.0005826 s^2 + 0.04827 s + 1}$$

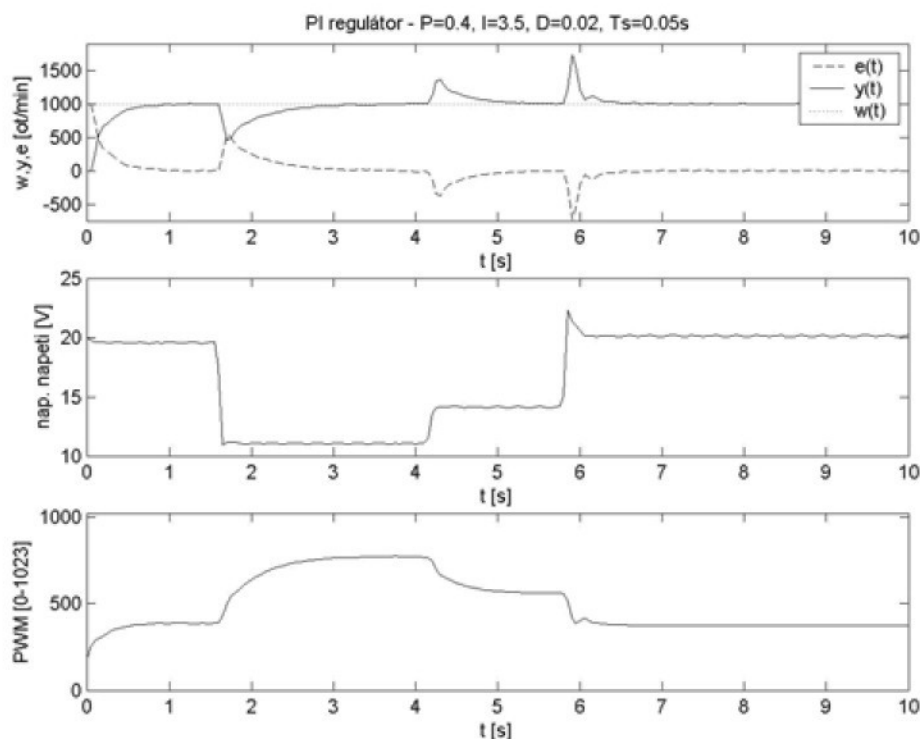
Tímto obrazovým přenosem byla nahrazena S-funkce 'smotor' v subsystému hlavního modelu. S takto upraveným modelem byly spuštěny simulace, které se předtím se stejnými žádanými hodnotami otáček a polohy provedly i na skutečné soustavě (motoru). Získané výsledky ze všech simulací byly zpracovány pomocí programu ,verifikace.m' (resp. ,verifikace_poloha.m'); výstupy z těchto programů ve formě grafů jsou uvedeny na obr. 4.16 a 4.18.



Obrázek 4-16

Při obou simulacích byl použit pouze PI regulátor s nastavením jednotlivých složek: $P=0.4$, $I=3.5$, vzorkovací perioda $T_s = 0.05$ s. Jak je vidět z grafů na obrázku 4.16, průběhy simulace na reálném (plná čára) a teoretickém (přerušovaná čára) modelu se liší jen velmi nepatrně. Díky tomu můžeme považovat navrženou modelovou technologii za dobře fungující.

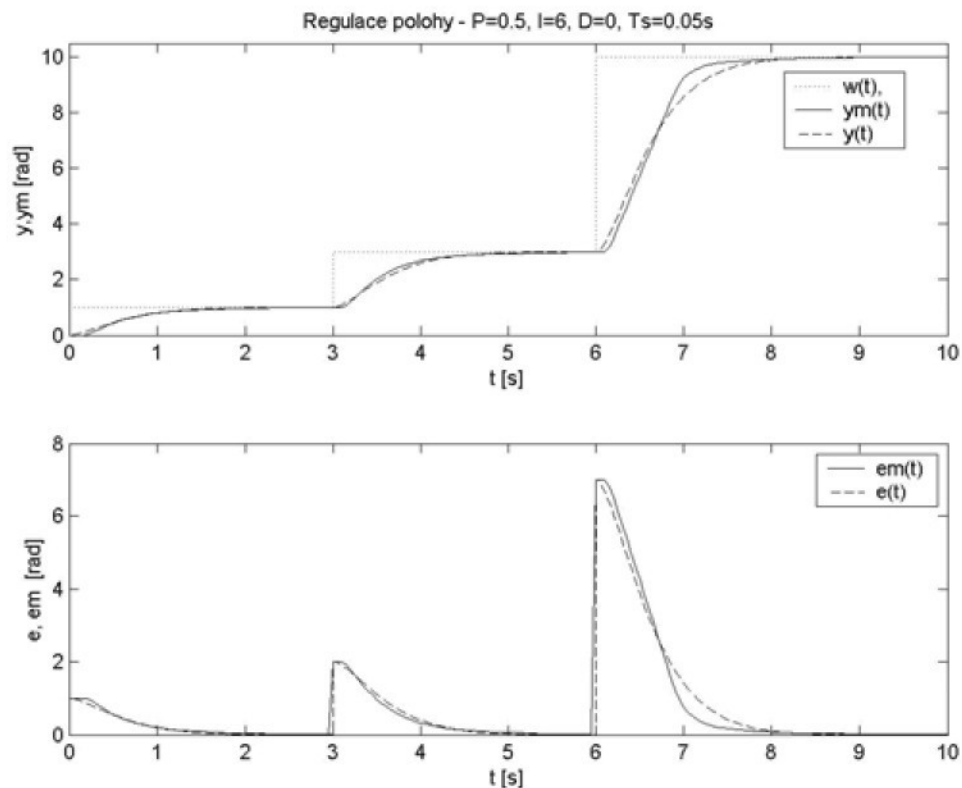
Funkce použitého PI regulátoru byla též ověřena při simulaci náhlého poklesu napájecího napětí výkonové jednotky (ověření reakce na poruchu). Na laboratorním zdroji bylo skokově zmenšeno napětí 20V na 11V, pak zvýšeno na 14V a nakonec zpět na 20V. Průběhy, zpracované programem ,verifikace_napeti.m', z této simulace jsou uvedeny obrázku 4.17



Obrázek 4-17

Z grafu výkonu (PWM) na obrázku 4.17 jsou vidět akční zásahy regulátoru při náhlém poklesu napájecího napětí (při vzniku poruchy). Regulační odchylka $e(t)$ se po těchto zásazích vrací zpět k nulové hodnotě. Tímto se potvrzuje dobrá funkce navrženého modelu řízení rychlosti (otáček).

Podobná verifikace byla použita i pro model regulace polohy. Opět byly za stejných podmínek provedeny simulace na skutečném a teoretickém modelu. Získané data byly zpracovány programem ,verifikace_poloha.m'. Průběhy z této verifikace jsou uvedeny v grafu na obr. 4.18.



Obrázek 4-18

Také zde je vidět dobrá shoda u obou průběhů simulace (čárkovaná čára představuje teoretický model, plná skutečný). Proto i model řízení regulace polohy můžeme považovat za správný.

5. Závěr

V této diplomové práci byl zhotoven mikrokontrolérem řízený řídicí systém polohového servomechanismu ovládaný prostřednictvím sériové linky z prostředí Matlab/Simulink. Mikrokontrolér přitom v této práci zajišťuje pouze komunikaci po RS232C a ovládá výkonovou jednotku. Samotné řízení polohy (rychlosti) zajišťuje model, vytvořený v toolboxu Simulink, který ve vizualizované formě ovládá a monitoruje důležité parametry, nutné pro správnou funkci celé modelové technologie. Především díky prostředí Simulink je tento systém velice názorný a umožňuje tak snadnější pochopení problematiky regulace polohy (rychlosti); změna parametrů jednotlivých složek regulátoru se ihned projeví na chování celého systému.

Řídicí jednotka byla navržena s ohledem na co největší jednoduchost a univerzálnost použití i v jiných úlohách či jako zkušební deska pro výuku programování jednočipových

mikrokontrolérů PIC 16F87x. Velké urychlení při vývoji nových uživatelských aplikací nabízí možnost nahrávat tyto programy do MCU pouze prostřednictvím sériové linky z PC. Mikrokontrolér PIC 16F876A byl vybrán především kvůli velkému počtu periférií (UART, 10bit A/D převodník, 2x PWM jednotky atd.), dostatečně velké paměti programu FLASH a neméně důležitým kritériem byla i jeho cena, dostupnost a vhodné vývojové prostředky.

Podobný přístup byl zvolen i při návrhu výkonové jednotky. Obsahuje pouze minimum aktivních součástek – stěžejním prvkem je integrovaný obvod L298. Tento umožňuje díky TTL logice přímé připojení k vývodům mikrokontroléru, dokáže zpracovávat velké napájecí napětí (až 46V) a především již obsahuje dvojici H-můstků vhodných pro řízení bipolárních motorů. Zvolená metoda řízení polohy a rychlosti motoru pomocí pulsně šířkové modulace (PWM) představuje moderní a rozšířený způsob řízení motorů.

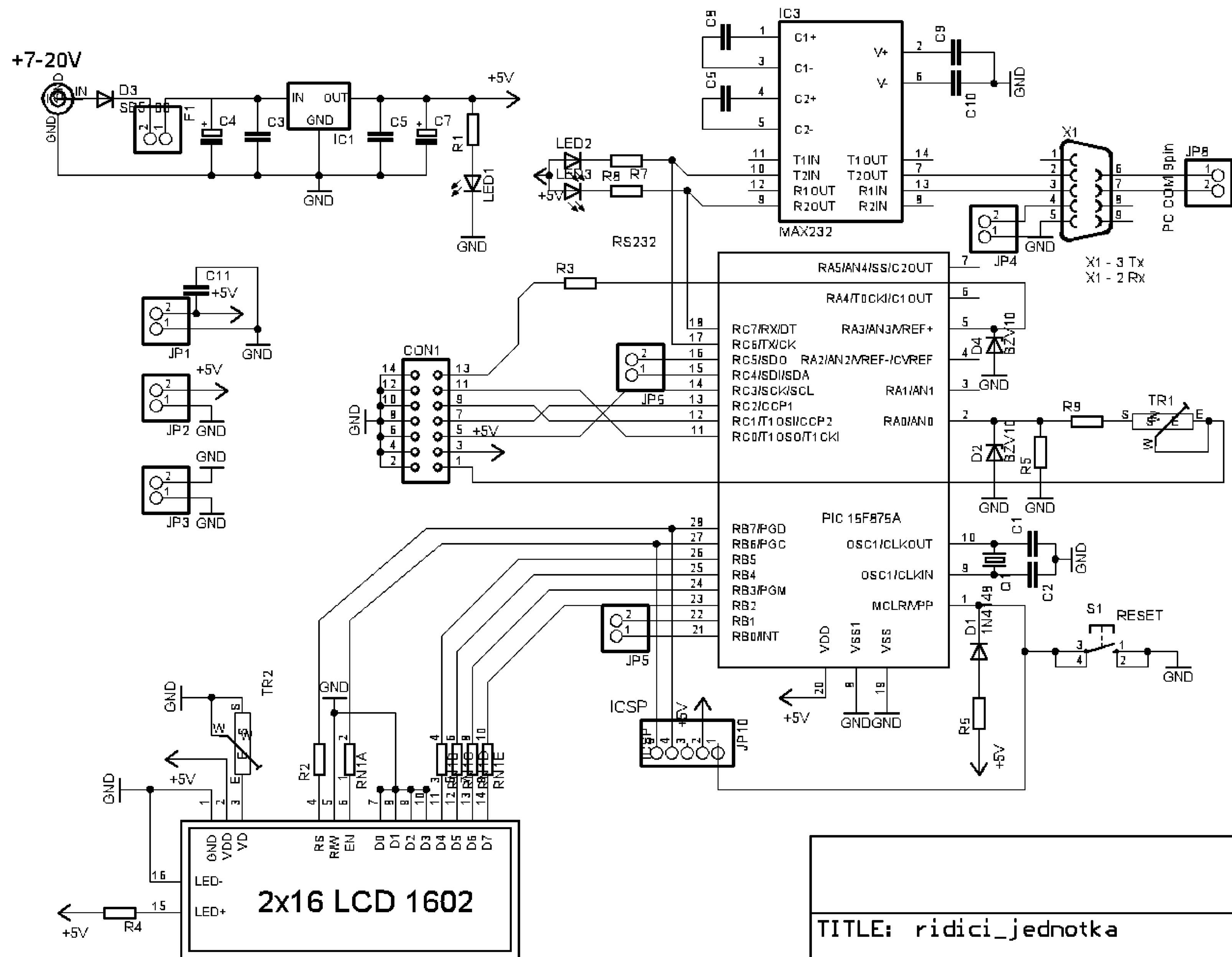
Velice důležitou součástí této práce je i programové zabezpečení. Modely řídicího systému rychlosti a polohy v toolboxu Simulink byly vytvořeny především s ohledem na využití při výuce studentů v laboratoři. Tyto modely názorně demonstují použití rychlostní a polohové smyčky (usnadňují odpověď na otázku: “Co že to vlastně je rychlostní a polohová smyčka ? “) při řízení elektrického pohonu. Velkou výhodou tohoto řešení je především velice propracovaná vizualizaci řízení celého technologického procesu a jednoduchá změna jednotlivých parametrů. Tyto vlastnosti velice usnadňují pochopení základních problémů regulace a řízení.

Výsledky této diplomové práce zjištěné při simulaci na skutečném modelu byly porovnány s teoretickým modelem, používající zjištěný obrazový přenos skutečného systému (motoru). Až na drobné odchylky se výsledky velice podobají a pro výukové účely je můžeme považovat za správné.

Doufám, že touto prací pomohu usnadnit studium řídicí techniky budoucím studentům a motivovat je k tvorbě dalších podobných projektů, využívajících moderních metod dnešní vědy a techniky.

Použitá literatura

- [1] SOUČEK, P. : Servomechanismy ve výrobních strojích
- [2] DUŠEK, F., KLÁN, P. : Laboratorní jednotka styku s prostředím pro počítače PC. Automatizace (36) 10, 1993, s. 294-296.
- [3] DUŠEK, F., HONC, D. : Využití sériové linky pod MATLABem verze 6. Sborník příspěvků 10. ročníku konference MATLAB 2002.
- [4] Hardwarové emulátory MU Alpha, MU Beta a vývojové prostředí ARTEMIS. Uživatelská příručka.
- [5] MATLAB/SIMULINK Uživatelské příručky.
- [6] NOVÁK, M. : Řízení polohového servomechanismu jednočipovým mikropočítačem. Diplomová práce, TUL, FS, KKY, 1999
- [7] Microchip Technology Inc., Katalogový list obvodu 16F876A - <http://ww1.microchip.com/downloads/en/DeviceDoc/39582b.pdf>
- [8] STMicroelectronics Inc., Katalogový list obvodu L298 - <http://www.st.com/stonline/products/literature/ds/1773.pdf>
- [9] Peroutka O., : Mikrokontroléry PIC 16F87X



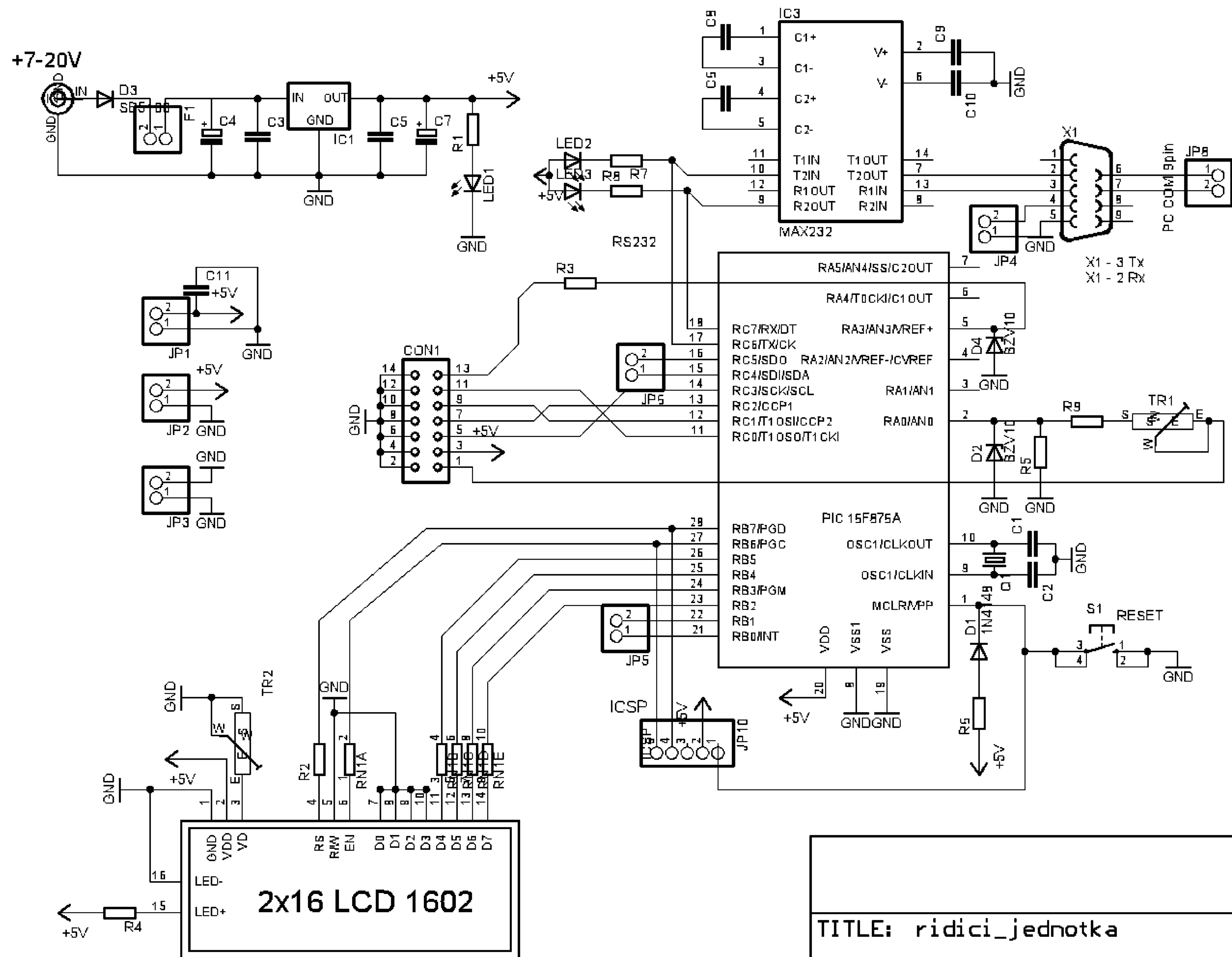
TITLE: ridici_jednotka

Document Number:

REV:

Date: 17.05.2006 10:20:44

Sheet: 1/1



TITLE: ridici_jednotka

Document Number:

REV:

Date: 17.05.2006 10:20:44

Sheet: 1/1

Seznam součástek

Řídící jednotka

R1,R6,R7,R8 - 1,5k
 R2,R3 - 330R
 R4 - 33R
 R5 - 560R
 R9 - 430R
 RN1 - 6x330R (odporová síť)
 TR1,TR2 - 5k (trimr)

C1,C2 - 33p
 C3,C5,C11 - 100n
 C4 - 470uF/35V
 C6,C8,C9,C10 - 1uF/16V (tantal)
 C7 - 47uF/25V

D1 - 1N4148
 D2,D4 - BZX83 5V1 /0.5W
 D3 - SB5100 (Schottky)
 LED1-3 - LED dioda (2mA)

Q1 - krystal 8Mhz

S1 - mikrospínač 5x5mm

IC1 - 7805
 IC2 - PIC 16F876A
 IC3 - MAX232

X1 - konektor Cannon 9pin (samice)
 X2 - napájecí konektor 2.1mm
 CON1 - 14pin MLW14G
 JP1-6,JP8 - 2xpin, rozteč 2.54mm

F1 - vratná pojistka PFRA020 (200mA)

LCD displej 2X16 znaku (řadič HD44780)

Napájecí adaptér 9V/300mA

Propojovací kabel PC-ŘJ

Cannon 9pin samice kabelová
 Cannon 9pin samec kabelový
 2m kabel RG-MD137 (2x0,14mm+stínění)

Výkonová jednotka

R1 - 3,3k
 R2,R3,R4 - 330R
 R5 - 0R47/5W
 R6 - 4k7
 R7,R8,R9 - 1k5
 R10 - 430R
 R11 - 10k

C1,C2 - 100n
 C3,C4 - 2000uF/50V

D1 - 1N4148
 D2-D5,D7 - 1N5908 (Schottky)
 D6 - BZX85/12V 1,3W
 LED1-4 - LED dioda (2mA)

IC1 - L298
 T1 - BC547 (NPN)

RE1 - relé RY12W-K

X1 - svorkovnice 2x, rozteč 5mm
 SL1 - 4xpin konektor PSH02-04WG
 SV1 - 3xpin konektor PSH02-03WG
 CON1 - 14pin MLW14G

F1 - tavná pojistka 2,5A

AL chladič 35x 30x80mm

4x distanční sloupek kovový 40mm

Propojovací kabel ŘJ-VJ

kabel plochý AWG28-14H - cca 7cm
 2x samořezná zásuvka PFL14 (2x7)