České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra řídicí techniky

DIPLOMOVÁ PRÁCE

Model železnice

Praha, 2004 Dušan Havlík

Prohlášení

Prohlašuji, že jsem zadanou diplomovou práci zpracoval samostatně s přispěním vedoucího diplomové práce a používal jsem pouze literaturu v práci uvedenou. Prohlašuji, že nemám námitek proti využití výsledků této práce fakultou ani proti zveřejňování nebo půjčování se souhlasem vedoucího diplomové práce.

Poděkování

Děkuji své vedoucí diplomové práce Ing. Martině Svádové za vedení diplomové práce a za cenné rady při její realizaci. Dále děkuji Ing. Radku Šindelářovi za konzultace, bez kterých by nemohl být realizován model železnice a řízení lokomotiv by bylo podstatně složitější. A v neposlední řadě děkuji i svým rodičů za podporu a trpělivost během diplomové práce.

Abstrakt

Diplomová práce se zabývá stavbou modelu železnice a jeho řízením pomocí PLC Simatic S7-315 2DP. Model železnice se skládá ze tří nádraží s výhybkami a semafory. Ke sledování pohybu lokomotiv po kolejišti jsou použity proudové snímače. Řízení pohybu lokomotiv se provádí pomocí DCC systému. K ovládání výhybek, semaforů a lokomotiv se používá řídicí elektronika, která zajišťuje komunikaci s nadřízeným systémem. Ukázkový řídicí algoritmus v PLC Simatic S7-315 2DP řídí jednak reálný model železnice ale i virtuální model železnice, který je naprogramován ve visualizačním prostředí InTouch. Visualizace jak reálného tak i virtuálního kolejište je provedna ve visualizačním prostředí InTouch.

Abstract

This master thesis presents a railway model built for an assessment of possibilities of model simulations. Imitating a real railway structure the model incorporates three platforms with shunts and semaphores and feedback sensors together with two locomotives with carriages. Through a PIC16F877 microcontroller it is possible to control semaphores and shunt positions, monitor Locomotive model locations and digitally control movements the locomotive models via Digital Control Command standard. To allow a simulation of more complex control processes the microcontroller is managed by Simatic S7-315 2DP PLC, which is able of generating paths between desired points and to transfer the locomotion models between them. Easier to control, the entire control process is visualized via InTouch environment.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Modelová železnice je svět modelů lokomotiv, vagónů, semaforů a nádraží doplněný figurkami výpravčích, průvodčích, strojvůdců a železničářů zasazený do krajiny. Ze strany modeláře a technika je tento svět dán souhrnem pravidel, který zajistí fungování systému dopravy.

Tato diplomová práce měla dva hlavní cíle. Prvním cílem bylo postavit model železnice, který budou v budoucnu využívat studenti ve cvičení v předmětu Řídicí systémy. Druhým cílem bylo naprogramování ukázkové řídicí aplikace. Diplomová práce je rozdělena do sedmi kapitol.

První kapitola nás seznamuje s obsahem diplomové práce a s motivem, proč tato diplomová práce vznikla.

Druhá kapitola se zabývá stavbou modelu železnice. Stavba modelu železnice je rozdělena na dvě na sebe navazující části. První částí je mechanická stavba kolejiště. Do této části patří výběr vhodného druhu kolejiva a přestavníků výhybek. Druhá část kapitoly je věnována řídicí elektronice modelu železnice. Řídicí elektronika modelu zajišťuje komunikační rozhraní mezi modelem a nadřízeným systémem, ovládá všechny výhybky, semafory a řídí pohyb lokomotiv po kolejišti.

Třetí kapitola je určena programování mikrokontroléru PIC16F84, který je použit v DCC zesilovači. V kapitole jsou nejprve uvedeny základní informace o mikrokontroléru a potom je popsán řídicí program mikrokontroléru.

Čtvrtá kapitola se věnuje programování mikrokontroléru PIC16F877, který je jádrem centrální řídicí jednotky. Na začátku kapitoly jsou nejdříve uvedeny základní informace o mikrokontroléru a následně je popsána struktura programu, který jednak generuje příkazy pro ovládaní lokomotiv, ovládá výhybky, semafory a sbírá data z kolejište o pohybu lokomotiv. Dále tento program zajišťuje komunikaci s nadřízeným systémem.

Pátá kapitola se zabývá řídicím algoritmem v PLC automatu Simatic S7-315 2DP. Na začátku kapitoly je popsána hardwarová konfigurace PLC. Následně na to je uve-

den popis komunikace s profibusovým modulem WAGO 750-650, což je převodník profibus-sériová linka RS232. Dále následuje popis programu, který je rozdělen na funkce, funkční bloky a hlavní program.

Šestá kapitola se věnuje visualizaci jak skutečných modelových nádraží tak i softwarově vytvořených nádraží ve visualizačním prostředí InTouch. Na začátku kapitoly je uvedena ukázka konfigurace DDE serveru a základního nastavení prostředí InTouch. Následně je uvedena struktura visualizačních oken a ovládání modelu prostřednictvím visualizace.

Poslední sedmá kapitola obsahuje zhodnocení výsledků diplomové práce s doporučením pro práci s modelem.

Kapitola 2

Stavba modelu

2.1 Způsoby řešení řízení pohybu lokomotiv

Všechny elektrické modely lokomotiv potřebují ke svému pohybu po kolejišti elektrické napětí, které sbírají z kolejí. K řízení pohybu lokomotivy po kolejišti se používají dva základní typy řízení. První typ je tzv. konvenční řízení a druhý typ je DCC řízení. Oba tyto typy řízení budou popsány v následujících kapitolách.

2.1.1 Konvenční řízení

Při konvenčním řízení je celé kolejiště rozděleno na jednotlivé od sebe elektricky izolované kolejové úseky. V každém úseku může být pouze jedna lokomotiva nebo jedna souprava lokomotiv. Připojením stejnosměrného napětí k úseku se uvede lokomotiva do pohybu. Rychlost pohybu je určena velikostí stejnosměrného napětí a směr pohybu je určen polaritou stejnosměrného napětí, které je přivedeno do lokomotivy. Výhodou tohoto způsobu řízení je jednoduchost ovládací elektroniky modelu a nízké náklady na její stavbu při menších rozměrech kolejiště. Další výhodou je, že v lokomotivě není potřeba žádná další elektronika.

Hlavní nevýhodu konvenčního řízení je to, že nelze ovládat nezávisle na sobě dvě a více lokomotiv v jednom kolejovém úseku. V případě rozsáhlých modelů je další nevýhodou velká složitost zapojení ovládací elektroniky v porovnání s druhou metodou.

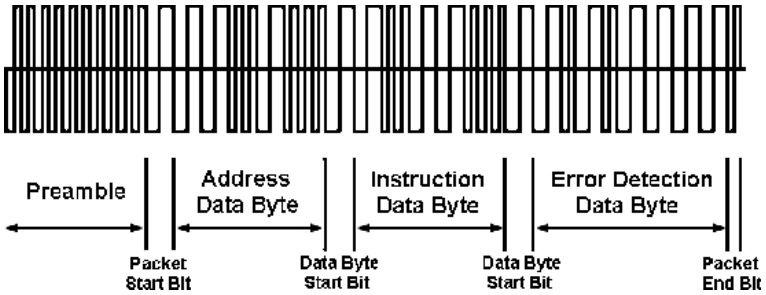
2.1.2 DCC řízení

Při použití DCC[1] řízení není nezbytně nutné rozdělit kolejiště na jednotlivé od sebe elektricky izolované úseky, protože DCC řízení umožňuje na sobě nezávislé řízení dvou a více lokomotiv v jednom kolejovém úseku. DCC řízení je založené na posílání

[1] DCC - Digital Command Control

paketů zařízením, které jsou připojené ke kolejím. Pakety jsou rozděleny do tří základních typů. Prvním typem jsou pakety, které jsou adresovány konkrétním lokomotivám a zařízením, druhým typem jsou pakety, které jsou určeny všem zařízením najednou a třetím typem jsou IDLE pakety, které se posílají v případě, že není co posílat. Pokud byl do kolejí vyslán paket prvního typu, lokomotiva si ho svým dekodérem příjme a přečte. V případě, že byl určen pro ni, začne vykonávat příkaz obsažený v paketu. Pokud přijatý paket nebyl určen pro ni, pokračuje ve své dosavadní činnosti. Zcela stejně by se zachovalo kterékoliv jiné zařízení připojené ke kolejím.

Každý paket (obr.2.1) je složen z preambule, adresy zařízení, dat a kontrolního součtu předcházejících dat paketu.

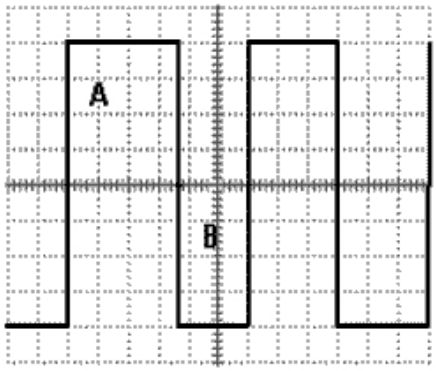


Obrázek 2.1: DCC Paket

Preambule je hlavička paketu, která se skládá minimálně z 11 po sobě jdoucích log.1. Adresa určuje zařízení, kterému je paket určen. Pokud se adresa skládá z osmi po sobě jdoucích log.0, pak je paket určen pro všechny zařízení připojené ke kolejím. Data obsahují příkaz, který má zařízení provést a kontrolní součet je určen pro kontrolu příjmutých dat.

Obsah paketu je reprezentován posloupností log.0 a log.1, které jsou kódovány podle normy NMRA 9.2.1[2]. Kód log.1 je definován jako obdélníkový průběh napětí. V případě modelu železnice se napětí pohybuje v rozmezí +12 V a -12 V. Definice délky jednotlivých části kódu DCC signálu je rozdělená na definici DCC signálu vycházejícího ze zesilovače (tab.2.1) a na definici DCC signálu akceptovaného dekodérem (tab.2.2).

[2]NMRA - National Model Railway Association



Obrázek 2.2: Kódování DCC signálu

Časový průběh pro log.1	Výsledek
$periodaA < 55\mu s$ nebo $periodaA > 61\mu s$	Špatně
$periodaA = periodaB$	Dobře
$\lvert periodaA - periodaB \rvert \Leftarrow 3\mu s$	Dobře
$\lvert periodaA - periodaB \rvert > 3\mu s$	Špatně

Tabulka 2.1: DCC signál na výstupu zesilovače

Časový průběh pro log.1	Výsledek
$periodaA \geq 52\mu s$ nebo $periodaA \leq 64\mu s$	Dobře
$periodaA = periodaB$	Dobře
$\lvert periodaA - periodaB \rvert \Leftarrow 6\mu s$	Dobře

Tabulka 2.2: DCC signál na vstupu dekodéru

Kód log.0 je definován stejně jako kód log.1 s tím rozdílem, že délka jednotlivých částí je minimálně 100 μs a zároveň je menší než 6000 μs. Rychlost přeběhu stejnosměrného napětí musí být minimálně $2,5\ V/\mu s$.

Hlavní výhodou tohoto způsobu řízení je možnost ovládat několik lokomotiv v jednom kolejové úseku nezávisle na sobě, což u prvého způsobu nebylo možné. Další výhodou je menší složitost zapojení ovládací elektroniky při použití na středně velkém modelu železnice oproti prvému způsobu. Hlavní nevýhodou DCC řízení je finanční nákladnost DCC řízení při použití na malém modelu železnice.

Při porovnávání obou základních způsobů řízení pohybu lokomotivy po kolejišti bylo vybráno DCC řízení, protože při zamýšleném rozměru kolejiště a způsobu řízení byl tento způsob levnější a jednodušší než první způsob.

2.2 Mechanická část modelu

2.2.1 Kolejivo

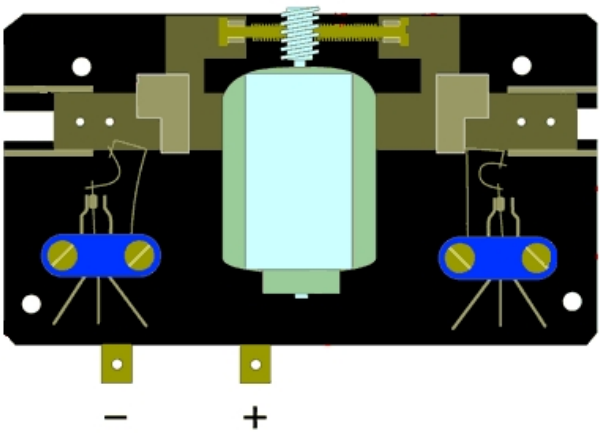
Pro stavbu modelu železnice bylo vybráno "modelové kolejivo", protože je vhodné pro DCC řízení. Kolejivo se skládá z umělohmotných výlisků pražců a z kovových kolejnic. Výsledná kolej se vytvoří složením těchto dvou dílů dohromady. Umělohmotný výlisek pražců tvoří buď rovný úsek o délce 166 mm nebo oblouk o poloměru 315 mm a úhlu 30^0. Profil kovové kolejnice je stejný jako profil skutečné železné kolejnice.

2.2.2 Výhybky

Vzhledem k tomu, že pro stavbu modelu bylo vybráno "modelové kolejivo", byly vybrány výhybky určené pro "modelové kolejivo" s úhlem odbočení $12,5^0$ a délkou rovné koleje 129,5 mm.

2.2.3 Přestavníky výhybek

K přestavování výhybek byl vybrán přestavník FULGUREX (obr. 2.3), který k přestavění výhybky používá posuvný mostek.



Obrázek 2.3: Přestavník FULGUREX

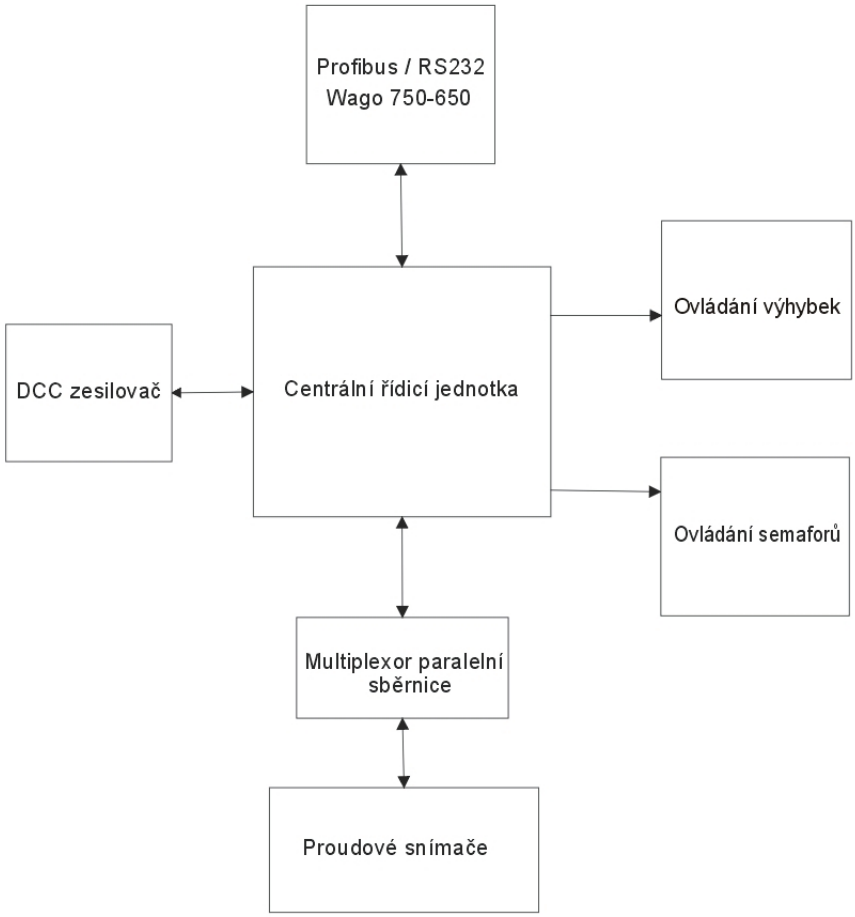
Mostek je posunován přes šnek motorkem. Aby motorek nebyl ve zkratu při přesunutí výhybky do krajní polohy, jsou na každé straně posuvného mostku koncové

spínače, které odpojí motorek od napětí. K opětovnému přesunu výhybky dojde pouze po změně polarity napájecího napětí.

Výhodou přestavníku FULGUREX jsou zpětné kontakty, které se nechají využít ve zpětné vazbě pro řízení modelu.

2.3 Řídicí elektronika

PLC automat Simatic S7-315 2DP použitý pro řízení modelu železnice je příliš pomalý pro generování příkazů použitých při DCC řízení. Proto byl vložen mezi PLC automat a model železnice elektronický systém (obr. 2.4), který generuje příkazy pro ovládaní lokomotiv, ovládá výhybky, semafory a sbírá data z kolejište o pohybu lokomotiv. Dále tento elektronický systém komunikuje s nadřízeným systémem po sériové lince RS232.



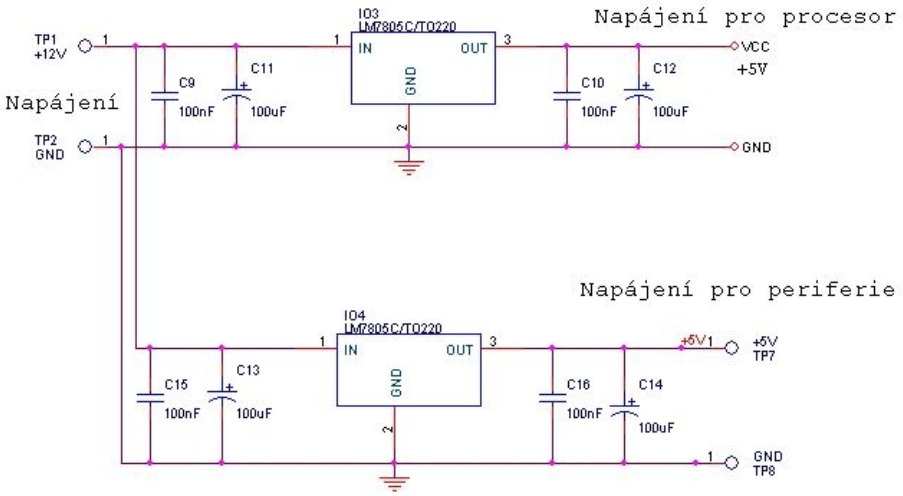
Obrázek 2.4: Blokové schéma řídicí elektroniky

2.3.1 Centrální řídicí jednotka

Centrální řídicí jednotka se skládá z mikrokontroléru PIC16F877 od firmy Microchip, z napájecích zdrojů, které jsou určeny pro napájení jak samotné desky tak i pro napájení periferií připojených k desce. Dále centrální řídicí jednotka obsahuje sériovou linku RS232, sběrnici I^2C, paralelní sběrnici a výstup pro DCC řízení. K indikaci napájecího napětí slouží velká červená LED dioda, k resetování mikrokontroléru slouží červené tlačítko, které je přiděláno k desce stolu. Elektrické schéma je uvedeno v příloze A na str.1. Celá deska je napájena stejnosměrným napětím +12 VDC. Celkový odběr proudu je závislý na počtu připojených periferií, ale nepřesáhne hodnotu 1 A.

2.3.1.1 Nápájení centrální řídicí jednotky

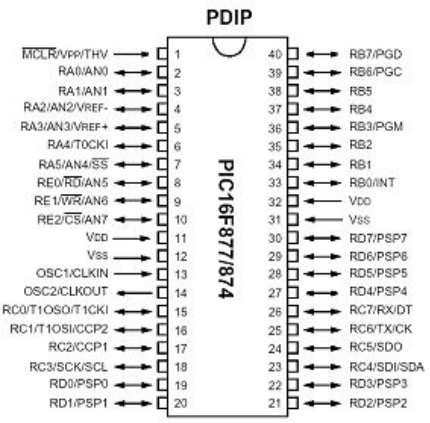
Deska centrální řídicí jednotky je napájena stejnosměrným napětím +12 VDC. Toto napětí je přivedeno na vstup dvou zdrojů stabilizovaného stejnosměrného napětí +5V DC (obr.2.5). První zdroj je určen výhradně pro napájení obvodů na desce, druhý zdroj je určen k napájení zařízení připojených k desce. Jádrem těchto zdrojů jsou stabilizátory napětí LM7805, které jsou schopny s dostatečným chlazením dodat proud 1.5A. Kondenzátory na vstupu stabilizátoru snižují zvlnění vstupního stejnosměrného napětí a kondenzátory na výstupu snižují zvlnění výstupního napětí.



Obrázek 2.5: Napájecí zdroj

2.3.1.2 Mikrokontrolér PIC16F877

Mikrokontrolér je jeden z nejdůležitějších prvků celé ovládací elektroniky modelu železnice. Při výběru vhodného mikrokontroléru bylo nutné splnit dvě hlavní podmínky.



Obrázek 2.6: Mikrokontrolér PIC16F877

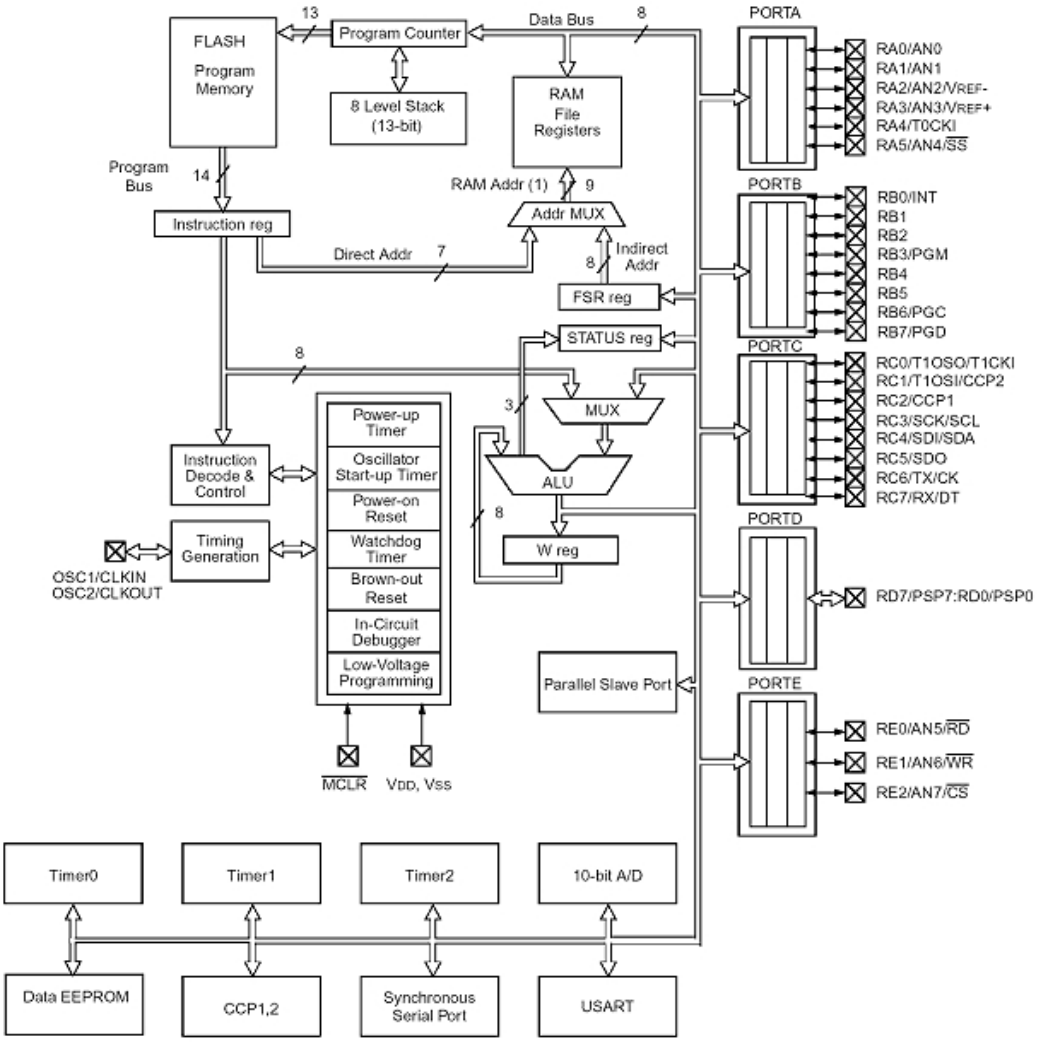
První podmínkou byla implementace podpory rozhraní RS232 a I^2C uvnitř mikrokontroléru. Druhou podmínkou byla snadná dostupnost vývojového prostředí. Obě tyto podmínky splnil mikrokontrolér PIC16F877 (obr. 2.6) od firmy Microchip. V následujícím textu této kapitoly bude mikrokontrolér popsán z hardwarového pohledu a z pohledu programátora bude popsán v kapitole 4, která se zabývá programem v mikrokontroléru.

Mikrokontrolér PIC16F877 má architekturu RISC (Reduced Instruction Set Computer), která je založena na předpokladu, že frekvence používání složitých instrukcí je tak malá, že se nevyplatí pro ně plýtvat plochou na čipu a v případě potřeby jsou nahrazeny posloupností jednoduchých instrukcí. Instrukční sada obsahuje 35 jednoduchých instrukcí. Výkon instrukce s výjimkou komunikace s pamětí je jeden strojový cyklus. Vnitřní struktura mikrokontroléru je zobrazena na obr.2.7.

Mikrokontrolér má celkem 33 vstupně-výstupních pinů rozdělěných do pěti portů A až E. Zápisem do konfiguračních registrů těchto portů se určí který pin jakého portu bude vstup a který pin bude výstup. Piny portu A je možno nakonfigurovat jako vstup A/D převodníku nebo jako digitální vstupy či výstupy. Piny portů B, C, a E je možné nakonfigurovat jen jako digitální vstupy či výstupy. U portu B lze ještě konfigurací určit, zda se budou k pinům připojovat zvyšovací odpory.

Mikrokontrolér má na svém čipu paměť rozdělenou na paměť programu a na paměť dat. Paměť programu je typu FLASH a má velikost 8192B. K tomuto typu mikrokontroléru nelze připojit externí paměť programu jako např. u mikrokontroléru řady '51, protože by to výrazně zpomalovalo výkonávání programu a ztratily by se výhody RISC architektury. Paměť dat je rozdělena na dynamickou paměť RAM o velikosti 368B a na paměť typu EEPROM o velikosti 128B. Dynamická paměť RAM slouží k uchovávaní dat během vykonávání programu. Pokud dojde k výpadku napájecího napětí, data se z této paměti ztratí. Po restartu vynuceném náběhem napájecího napětí se v paměti RAM objeví náhodný obsah log. nul a

Device	Program FLASH	Data Memory	Data EEPROM
PIC16F874	4K	192 Bytes	128 Bytes
PIC16F877	8K	368 Bytes	256 Bytes



Obrázek 2.7: Blokové schéma mikrokontroléru PIC16F877

jedniček. Paměť EEPROM slouží k uchování dat i v případě, že dojde k vypnutí napájecího napětí, protože paměť typu EEPROM nepotřebuje pro uchování dat v sobě zapsaných napájecí napětí.

Mikrokontrolér PIC16F877 má dvě komunikační rozhraní. První rozhraní je univerzální synchronní – asynchronní vysílač – přijímač v katalogovém listu označované jako USART [3]. Druhé rozhraní je synchronní sériový port, který lze na konfigurovat jako komunikační rozhraní pro sběrnici SPI [4] nebo I^2C. Rozhraní USART lze nastavit pomocí konfiguračních registrů jako plně duplexní asynchronní systém, jehož prostřednictvím mikrokontrolér může komunikovat se zařízeními jako je osobní počítač. Dále lze rozhraní USART nakonfigurovat jako poloviční duplexní synchronní systém, kterým mikrokontrolér může komunikovat se zařízeními jako jsou A/D a D/A převodníky, nebo sériové paměti EEPROM apod. Rozhraní synchronního sériového portu lze nakonfigurovat pomocí konfiguračních registrů buď jako rozhraní pro sběrnici SPI nebo jako rozhraní pro sběrnici I^2C. V obou možnostech se mikrokontrolér může chovat na sběrnici jako nadřízené či podřízené zařízení. Obě sběrnice se používají ke komunikaci s pamětmi, ovladačemi displejů a podobnými zařízeními. V centrální řídicí jednotce bylo použito rozhraní pro sběrnici I^2C, jehož konfigurace je popsána v kapitole 4.2.5. Při použití sběrnice I^2C je nutné připojit mezi piny RC3, RC4 a napájení zvyšovací rezistory s hodnotou 1kΩ.

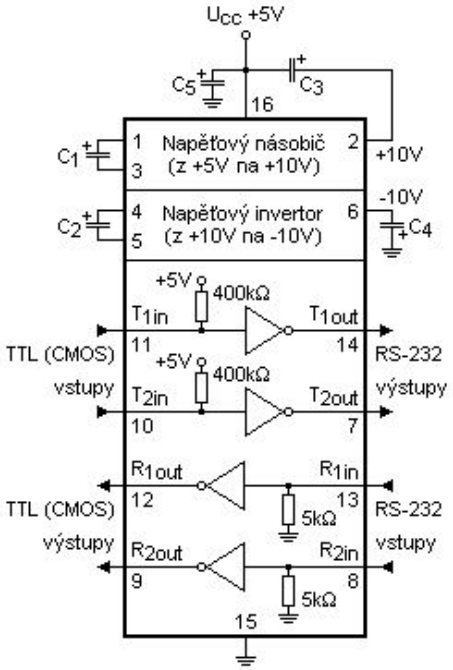
2.3.1.3 Obvod MAX232

Pro převod napěťových úrovní z TTL na V24 je použit obvod MAX232 v typickém zapojení (obr. 2.8), které je uvedeno v katalogovém listu tohoto obvodu. U tohoto obvodu je velice důležité si dávat pozor na typ obvodu, protože každý typ obvodu potřebuje ke své správné funkci externí kondenzátory s různými hodnotami. V modelu byl použit obvod MAX232CPE s externími kondenzátory 1 μF.

2.3.1.4 DCC výstup

DCC signál je generován mikrokontrolérem na pinech RC0 a RC1. Odtud je signál přiveden přímo do galvanicky izolovaného vstupu zesilovače, který tento signál výkonově zesílí. Pokud na pinu RC0 je log.1, tak na pinu RC1 je log.0 a naopak. Odebíraný proud z pinů RC0 a RC1 je cca 2 mA. DCC signál generovaný mikrokontrolérem je v napěťové úrovni TTL.

[3] USART - Universal Synchronous Asynchronous Receiver Transmitter
[4] SPI - Serial Peripheral Interface



Obrázek 2.8: Typické zapojení obvodu MAX232

2.3.1.5 Paralelní sběrnice

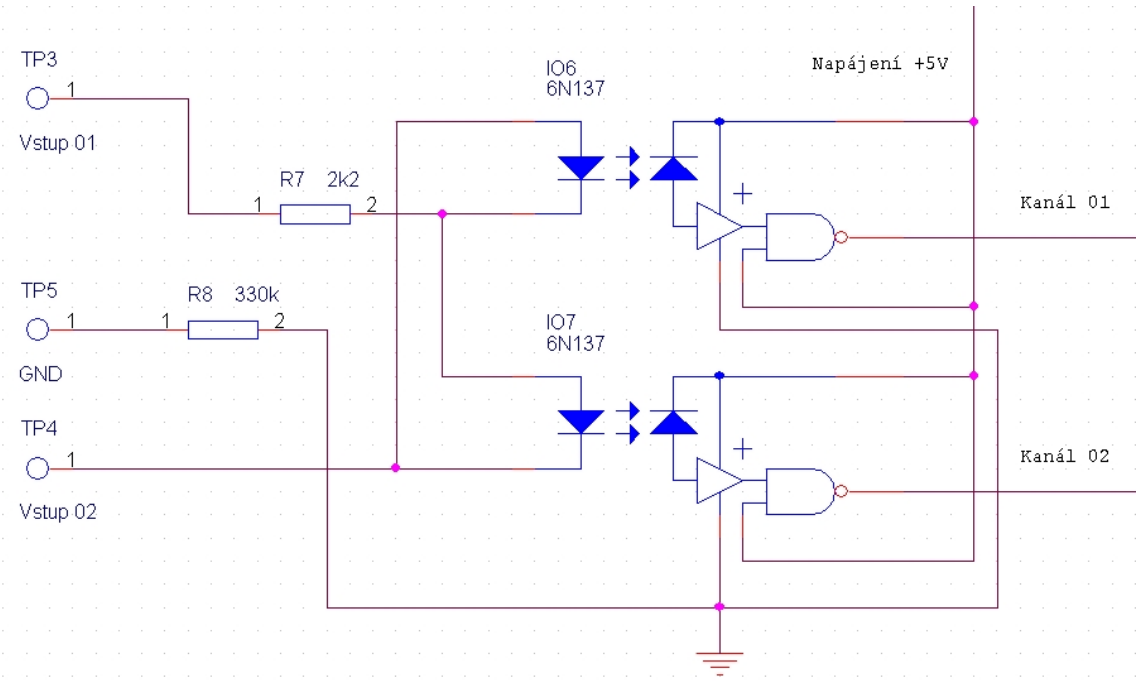
Paralelní sběrnice připojuje proudové snímače k centrální řídicí jednotce. Výběr proudových snímačů, které jsou připojeny ke sběrnici, se provádí pomocí obvodu 74HCT154, což je multiplexer 1 z 16. Multiplexer 74HCT154 ovládá sběrnicové budiče 74HCT541 na deskách proudových snímačů. Rychlost přenosu dat po paralení sběrnici je cca 10kHz. Větší frekvence není jednak žádoucí, protože by mohlo docházet k rušení vedlejších vodičů a jednak není třeba tak často sbírat data z modelu.

2.3.2 DCC zesilovač

DCC zesilovač je jednou z nejdůležitější částí ovládací elektroniky modelu, bez které by nebylo možné ovládat lokomotivy pohybující se po kolejišti. DCC zesilovač výkonově zesiluje DCC signál generovaný v centrální řídicí jednotce a napájí jím celé kolejiště. Protože se po kolejišti budou pohybovat pouze dvě lokomotivy stačí, aby minimální dostatečný výkon dodávaný do kolejište byl 24 W tj. aby při 12 V DC mohl být odběr proudu 2 A. Vzhledem k tomu, že na internetu bylo nalezeno hned několik zajímavých volně šířitelných zapojení DCC zesilovačů, nebylo třeba vymýšlet úplně nové zapojení. Jako nejvíce vyhovující zapojení bylo vybráno elektrické schéma, které vytvořil Dipl. Ing. Stefan Haack.

Toto zapojení je složeno ze čtyř funkčních částí. První částí jsou galvanicky oddělené signálové vstupy (obr. 2.9) DCC signálu , který přichází z centrální řídicí

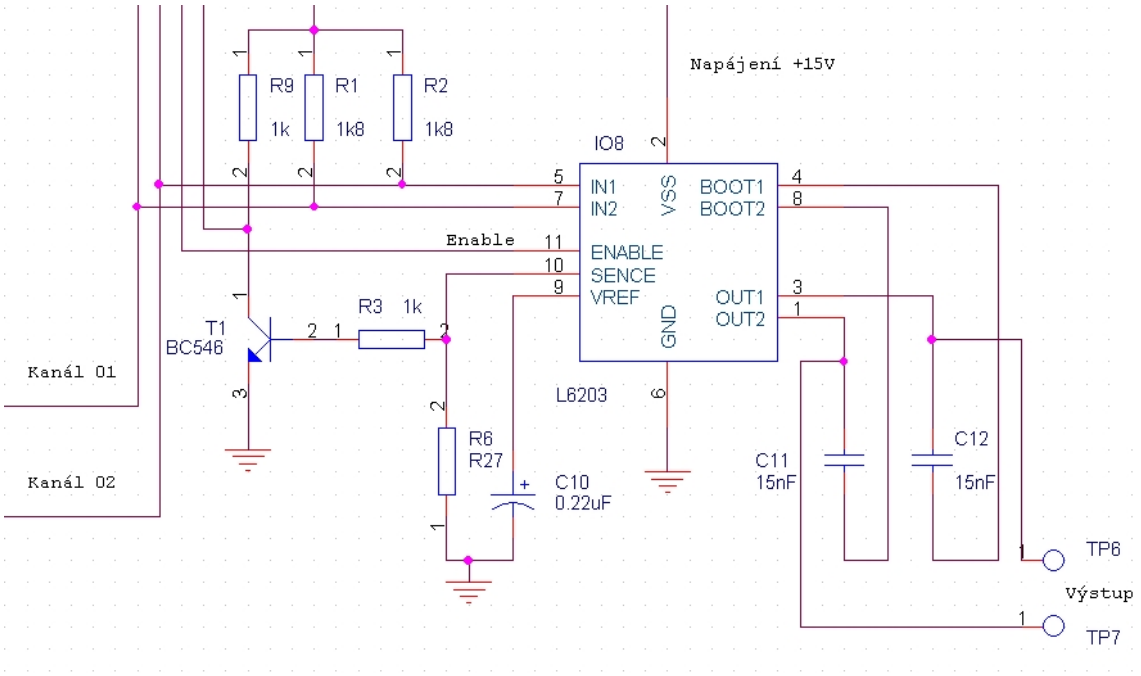
jednotky. Pro galvanické oddělení signálu byly použity optočleny 6N137. Výhodou těchto optočlenů je, že výstup je přímo v napěťové úrovni TTL a jsou schopny přenášet signál o frekvenci několika jednotek MHz.



Obrázek 2.9: Vstupy DCC zesilovače

Další funkční částí DCC zesilovače je můstkový zesilovač L6203 (obr. 2.10) spolu s externími kondenzátory C1 až C3 a odporem R5. Obvod L6203 je tvořen čtveřicí unipolárních výkonových tranzistorů zapojených do můstku a logickými obvody, které spínají dané tranzistory podle signálů na vstupech IN1 a IN2. Součástí obvodu L6203 je i výstup SENSE, který je ve skutečnosti výkonovým vstupem tranzistorového můstku. Mezi tento výstup a zem je připojen rezistor R5 sloužící k snímaní velikosti odebíraného proudu obvodem L6203. Pokud je hodnota proudu větší než 2,5 A, vytvoří se na rezistoru napětí, které sepne tranzistor T1. Sepnutý tranzistor T1 přivede na pin RB3 mikrokontroléru PIC16F84 zem tj. log.0 a tím signalizuje mikrokontroléru PIC16F84 výkonové přetížení výstupu zesilovače. Mikrokontrolér okamžitě odpojí výstup obvodu L6203 od zdroje napětí pomocí vstupu ENABLE.

Třetí funkční částí DCC zesilovače je mikrokontrolér PIC16F84. Mikrokontrolér má v zapojení DCC zesilovače dvě základní funkce. První základní funkcí je kontrola přítomnosti DCC signálu na vstupu zesilovače. V případě, že na vstupu zesilovače není žádný signál, tak pomocí vstupu ENABLE obvodu L6203 odpojí výstup zesilovače od napájecího zdroje. Druhou základní funkcí je odpojení výstupu zesilovače od napájecího zdroje v případě výkonového přetížení výstupu DCC zesilovače. Program uvnitř mikrokontroléru se skládá pouze z detekce hran na vstupu



Obrázek 2.10: Můstkový zesilovač L6203

RB0, v případě detekce DCC signálu a detekce hran vstupu RB3 v případě detekce výkonového přetížení zesilovače. Pokud DCC zesilovač pracuje správně, je rozsvícena pouze LED č.1, pokud není signál na vstupu DCC zesilovače svítí LED č.1 i LED č.2. V případě, že nastalo výkonové přetížení výstupu zesilovače rozsvítí se LED č.2 a LED č.1 zhasne.

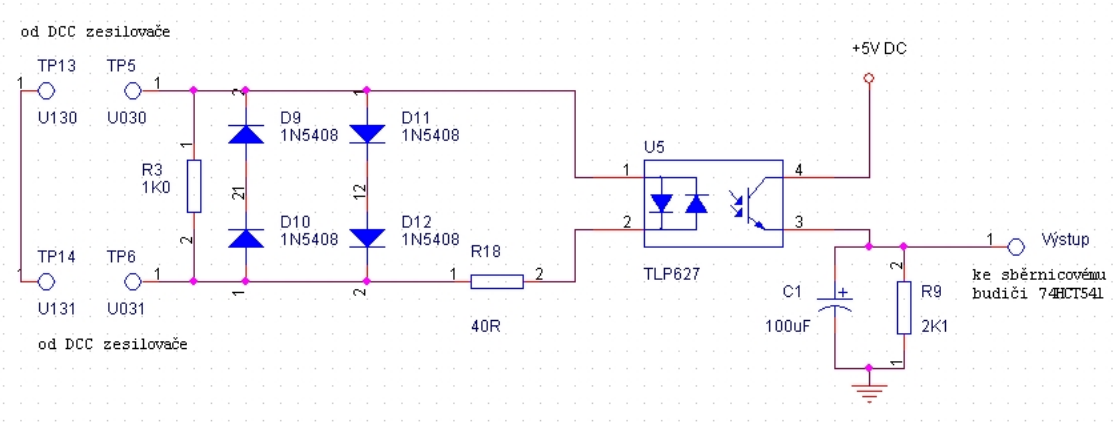
Poslední funkční částí jsou dva stabilizované napěťové zdroje. První napěťový zdroj, na jehož výstupu je $+15\ V\ DC$, slouží k napájení obvodu L6203 a druhý zdroj, jehož výstupní napětí je $+5\ V\ DC$, slouží k napájení mikrokontroléru PIC16F84 a optočlenů 6N137.

2.3.3 Proudové snímače

Proudové snímače (obr.2.11) jsou jedním z nejdůležitějších prvků zpětné vazby modelu železnice. Poskytují totiž informaci o tom, který úsek kolejiště je obsazen a který není. V modelu železnice bylo použito celkem 54 proudových snímačů.

Každý sledovaný úsek kolejiště má svoji vnější kolej elektricky izolovanou od sousedního úseku. Každá tato vnější kolej je extra napájena z DCC zesilovače přes dva páry usměrňovacích diod, které jsou vůči sobě antiparalelně zapojeny. Pokud např. lokomotiva vjede do hlídaného úseku, uzavře se přes motor elektrický obvod tvořený motorem lokomotivy, kolejemi, snímacími usměrňovacími diodami a DCC zesilovačem. Na snímacích usměrňovacích diodách vznikne úbytek napětí. Paralelně k těmto usměrňovacím diodám je připojen vstup optočlenu TLP627-4, který

při vzniku úbytku napětí na usměrňovacích diodách sepne svůj výstupní tranzistor. Sepnutím výstupního tranzistoru optočlenu se na vstup sběrnicového budiče 74HCT541 přivede napětí +5 V znamenající log.1.



Obrázek 2.11: Schéma proudového snímače

Vzhledem k vysokému počtu použitých snímačů v modelu železnice byly proudové snímače rozděleny do skupin po osmi snímačích. Každá skupina snímačů je na jedné desce, která je připojena paralelní sběrnicí k centrální řídicí jednotce a zároveň je z ní napájena. Jeden proudový snímač se skládá ze tří částí, snímací části: převodní části a výstupní části.

Snímací část je tvořena dvojicí páru antiparalelně zapojených usměrňovacích diod 1N5408, na kterých při průchodu proudu vzniká úbytek napětí cca 1,2 − 1,4 V. K těmto antiparalelně zapojeným usměrňovacím diodám je paralelně přes rezistor 39 Ω připojen vstup optočlenu TLP627-4. Vstup toho optočlenu tvoří dvě antiparalelně zapojené LED diody, kterými protéká proud cca 20 mA. Maximální propustný proud usměrňovacími diodami 1N5408 je 3 A a úbytek napětí na jedné usměrňovací diodě při tomto proudu je 1,5 V. Pokud by požadovaný propustný proud usměrňovacími diodami byl větší, lze tyto usměrňovací diody zaměnit za jiný výkonější typ.

Převodní část se skládá z optočlenu a RC článku připojeného na výstup optočlenu. Když vstupem optočlenu začne protékat proud cca 20 mA, sepne se výstupní tranzistor optočlenu. Protože vstupní napětí má obdélníkový průběh a prochází oběma polaritami, dochází na výstupu ke kmitání výstupního napětí optočlenu. Toto kmitání je nepřípustné. Mohlo by se totiž stát, že přesně v momentu, kdy bude napětí na snímacích usměrňovacích diodách procházet nulou, bude se odečítat hodnota na výstupu snímače. Odečítaná hodnota bude v tento moment log.0, protože úbytek napětí na snímacích usměrňovacích diodách bude nulový a vstupem optočlenu nepoteče žádný proud. Následkem toho bude výstup optočlenu zavřený a vstup sběrnicového budiče bude uzeměn přes rezistor na zem. Proto je na výstup dán ještě kon-

denzátor 50 μF, který vyfiltruje tyto kmity a zabezpečí tak log.1 po celou dobu, kdy na snímacích usměrňovacích diodách je úbytek napětí. Časová konstanta RC článku je 5 ms.

Výstupní částí proudového snímače je sběrnicový budič 74HCT541, jehož výstup je třístavový tj. log.0 , log.1 a stav vysoké impedance. Log.0 je reprezentována napětím do 0,8 V a log.1 napětím od 3,5 V - 5 V. Všech osm proudových snímačů je připojeno přes sběrnicový budič na paralelní sběrnici o šířce 8 bitů. K paralelní sběrnici je připojeno celkem sedm desek proudových snímačů. Výběr, ze které desky se budou číst výstupy proudových snímačů se provádí pomocí multiplexeru 74HCT154, který buď připojí výstup sběrnicového budiče ke sběrnici nebo ho odpojí tj. ho uvede do stavu vysoké impedance. Frekvence připojování jednotlivých desek ke sběrnici je cca 100 μs.

2.3.4 Ovládání výhybek

Deska ovládání výhybek je principiálně velice jednoduchá. Skládá se ze vstupně-výstupního expandéru PCF8574P, který přes galvanické oddělení ovládá výstupní relé. Relé svým sepnutím či rozepnutím kontaktů pak přivede buď kladnou nebo zápornou polaritu napětí na vstup přestavníků výhybek. Schéma desky ovládaní výhybek je uvedeno v příloze A na str.4.

Ovládání výhybek funguje tak, že centrální řídicí jednotka po sběrnici I^2C pošle příkaz vstupně-výstupnímu expandéru PCF8574AP, které výstupy má nastavit do stavu log.1 a které do stavu log.0. Nastavením výstupu expandéru do stavu log.1 nastane situace, kdy vstup optočlenu TLP627-4 je připojen mezi výstupní napětí expandéru a napájecí napětí +5 V. Protože výstupní napětí expandéru je +5 V, vstupem optočlenu neprochází žádný proud a výstup optočlenu a relé jsou v základní pozici. Nastavením výstupu expandéru do stavu log.0 začne vstupem optočlenu procházet proud, protože vstup optočlenu je sepnutím výstupního tranzistoru expandéru připojen mezi napájecí napětí +5 V a zem. Proud protékající vstupem optočlenu je omezen rezistorem na cca 10 mA. Průchodem proudu vstupem optočlenu se sepne výstupní tranzistor optočlenu a tím začne cívkou relé procházet proud. Cívka sepne reléové kontakty a změní polaritu napětí na přestavníku výhybky a tím polohu výhybky. Výstupním tranzistorem optočlenu prochází proud 50 mA, který je daný pouze odporem cívky relé. Výkonová ztráta na výstupním tranzistoru optočlenu není zanedbatelná, ale nevyžaduje v laboratorních podmínkách žádné přídavné chlazení. Kontakty relé jsou dimenzovány až na proud 10 A, ale tento proud jimi nebude kromě zkratu protékat. Při normální provozu přestavníků je nejvyšší možný proud 1 A a to v případě, že motorek přestavníku bude zabržděn a nebude

se moci roztočit.

Deska ovládání výhybek má dvě nezávislá napájení. První napájecí napětí je $+5\ V$, které je přivedeno z centrální řídicí jednotky. Tímto napětím je napájen expandér PCF8574P a vstupní část optočlenu TLP627-4. Celkový maximální odbíraný proud je cca 200 mA. Při tomto proudu jsou všechny výstupy expandéru ve stavu log.0. Druhým napájecím napětím je $+12\ V$, které slouží k napájení reléových cívek a přestavníků výhybek. Celkový odebíraný proud ze zdroje $+12\ V$ je závislý na počtu sepnutých relé a na momentu, jestli se některá z výhybek nepřestavuje. Maximální průměrný naměřený proud při přestavování všech výhybek byl 5 A.

2.3.5 Semafory

V modelu železnice je použito celkem 12 semaforů, které mají čtyři světla. Vždy jedna dvojice semaforů je ovládána jedním vstupně-výstupními expandérem PCF8574P. Každé světlo je reprezentováno jednou LED diodou, ktrérá je připojena přes ochranný rezistor mezi napájecí napětí expandéru a výstup expandéru. Maximální proud diodou je cca 2mA. Komunikace mezi centrální řídicí jednotkou a deskou ovládání semaforů probíhá po sběrnici I^2C. Napájení semaforů se provádí přímo z centrální řídicí jednotky, kde je umístěn stabilizovaný zdroj +5V sloužící právě pro napájení periferií připojených k centrální řídicí jednotce.

2.3.6 Napájení modelu železnice

Protože model se nachází v laboratoři bylo rozhodnuto, že se využije napěťových zdrojů umístěných ve stolech, které jsou v laboratoři. Model je tedy napájen stejnosměrným napětími $+15V$, $+12V$ a $+24V$.

Kapitola 3

Programování PIC16F84

3.1 Programování PIC16F84

Tato kapitola je věnována popisu programu, který je v mikrokontroléru PIC16F84 umístěném v DCC zesilovači. Kromě popisu programu jsou tu uvedeny i potřebné informace k jeho napsání. K naprogramování mikrokontroléru PIC16F84 byl použit programátor PICCOLO od firmy ASIX a programovací prostředí UIP od firmy Microchip. Kompletní informace o mikrokontroléru PIC16F84 jsou uvedeny v katalogovém listu uloženém na CD, které je součástí této diplomové práce. Na CD je rovněž uloženo i programovací prostředí UIP.

3.1.1 Mikrokontrolér PIC16F84

Mikrokontrolér PIC16F84 je vhodný pouze na jednoduché funkce jako je generování PWM signálu, časovací funkce nebo pro sériovou komunikaci. Má 13 vstupně výstupních pinů, kde každý pin lze nakonfigurovat buď jako vstup nebo výstup a to zápisem do registru $Trisa$ u portu A a u portu B do registru $Trisb$. Paměť mikrokontroléru je rozdělena na paměť programu a na paměť dat. Paměť programu, která je typu FLASH, má velikost 1 kB. Paměť dat je dále rozdělena na pameť typu RAM a paměť typu EEPROM. Paměť RAM slouží k dočasnému ukládaní dat, tj. když dojde k výpadku napájecího napětí, ztratí se všechna data. Paměť typu EEPROM se používá k trvalému uložení dat tj. když dojde k výpadku dat, tak se všechna uložená data uchovají v paměti. Velikost paměti RAM je 68 B a velikost paměti EEPROM je 64 B. Maximální hodinový kmitočet mikrokontroléru je 10MHz, ale v zapojení DCC zesilovače je hodinový kmitočet 4MHz. Délka jednoho strojového cyklu je rovna 1/4 převrácené hodnoty hodinového kmitočtu. Při kmitočtu 4MHz je délka strojového cyklu rovna 1 μs. U mikrokontroléru se vyskytuje operační registr, který je označován písmen W. Přes tento registr, který nemá definovanou

adresu, se provádějí veškeré aritmeticko-logické operace a přesuny dat mezi registry a pamětí a opačně. Velice důležitou součástí mikrokrontroléru je registr *Status*, kde se uchovávají příznaky charakterizující stav programu. Dále mikrokontrolér obsahuje jeden watchdog a jeden časovač TMR0. Protože časovač TMR0 je pouze 8-bitový, je součástí mikrokontroléru předdělička hodinového kmitočtu, která se nechá buď připojit před časovač nebo před watchdog.

3.1.2 Programování PIC16F84

Program v mikrokontroléru PIC16F84 slouží k hlídaní přítomnosti DCC signálu na vstupu DCC zesilovače. V případě, že na vstupu zesilovače není přítomen pulsní signál s minimální frekvencí 1kHz, mikrokontrolér změní stav pinu $RB3$ z log.1 na log.0 a tím odpojí výstup můstkového zesilovače L6023 od napájecího napětí. Tento stav je signalizován dvojicí LED, které jsou rozsvíceny. Dále program slouží k odpojení výstupu zesilovače v případě, že výstup zesilovače je výkonově přetížen. Tento stav je opět signalizován dvojicí LED, přičemž LED č.2 je rozsvícena a LED č.1 je zhasnuta. V případě, že DCC zesilovač pracuje správně, je rozsvícena LED č.1 a LED č.2 je zhasnuta. Pokud na vstupu DCC zesilovače není signál, jsou rozsvíceny obě dvě LED diody.

Vzhledem k tomu, že program v mikrokontroléru využívá přerušení od náběžné hrany na pinu RB0, bude nejdříve popsán způsob obsluhy přerušení a jeho povolení. Pokud nastane situace, která vyvolá přerušení, mikrokontrolér provede rozpracovanou instrukci. Potom si uloží do zásobníku adresu momentálního místa v programu, kde se zrovna nachází, a skočí na adresu $0x04h$ v paměti programu. Na této adrese se nachází počátek programu, který obsluhuje vzniklé přerušení. Při přechodu mikrokontroléru na obsluhu události, jež vyvolala přerušení, se neuchovává obsah pracovního registru W a registru *Status*. Proto je nutné hned na začátku programu obsluhujícího přerušení si obsah těchto registrů uchovat v paměti dat jak ukazuje níže uvedená část kódu.

```
INTSIG
    movwf MEM_W
    movf  STATUS,W
    movwf MEM_STAT
          .
```

Vzhledem k tomu, že mikrokontrolér má pouze jednu adresu, na kterou přechází v případě vzniku jakéhokoliv přerušení, je dále nutné na začátku určit o jaký typ přerušení se jedná tj. musí se pomocí testů jednotlivých bitu v registru *Intcon*

zjistit, která událost vyvolala přerušení a tuto událost zpracovat. Prioritu událostí si programátor stanovuje sám, podle toho v jakém sledu seřadí testovací podmínky jednotlivých bitů. Na konci obsluhy přerušení se musí opět vrátit uschovaný obsah registrů W a $STATUS$ z doby před přerušením do registrů W a $STATUS$, což se provede následujícím kódem.

```
        .
movf   MEM_STAT,W
movwf STATUS
swapf MEM_W,F
swapf MEM_W,W
retfie
```

Program v mikrokontroléru využívá obsluhy přerušení vyvolané náběžnou hranou na pinu RB0. Toto přerušení se nastaví pomocí několika důležitých bitů v registrech *Option_reg* a $Intcon$. V registru *Option_reg* nastavíme bit 6 na hodnotu log.1, tj. bude povoleno přerušení od náběžné hrany na pinu RB0. V registru INTCON nastavíme bity GIE, INTE na hodnotu log.1. Při inicializaci je nutné bit INTF v registru $Intcon$ vynulovat. Ostatní bity registru $Intcon$, které se netýkají přerušení od náběžné hrany, se taky vynulují, protože jejich funkce nejsou v programu použity.

Vlastní program se skládá z inicializace, hlavní programové smyčky a z přerušení. Při inicializaci se porty A a B vynulují. Piny $RB0$, $RB1$, $RB6$ a $RB7$ portu B se nastaví zápisem hodnoty $0x63h$ do registru $Trisb$ jako vstupy. Ostatní piny portu B jsou nastaveny jako výstupy. Piny portu A se nastaví zápisem hodnoty $0x00$ do registru $Trisa$ jako výstupy. Dále se v inicializaci nastavením příslušných bitů registrech *Option_reg* a $Intcon$ nastaví a inicializuje přerušení od náběžné hrany na pinu $RB0$. Hlavní program je tvořen smyčkou, která se vykonává neustále dokola. Během ní se testuje pin RB3 na portu B a bity v paměťové registru $Intb$. V případě pozitivního testu se odskočí na obsloužení daného stavu, který je reprezentován pozitivním testem. V paměťovém registru $Intb$ se testuje zda bit 0 není ve stavu log.0. Pokud je bit je ve stavu log.0, tak je vše v pořádku, protože na pinu $RB0$ se objevila náběžná hrana od doby poslední kontroly. Mikrokontrolér přeskočí následující instrukci a neguje testovaný bit. V případě, že se tak nestalo, je bit v log.1 a to způsobí odskok na rozsvícení obou LED na pinech $RB3$ a $RB4$. Vynulovaní bitu 0 v registru $Intb$ se provádí v obsluze přerušení, které vzniklo na základě náběžné hrany na pinu $RB0$.

Kapitola 4

Programování PIC16F877

4.1 Mikrokontrolér PIC16F877

Z hlediska hardwaru byl mikrokontrolér již popsán v kapitole 2.3.1.2. Mikrokontrolér PIC16F877 byl programován pomocí programátoru MPLAB-ICD od firmy ASIX a vývojového prostředí MPLAB 5.7. Vývojové prostředí je uloženo na CD, které je součástí diplomové práce.

4.2 Programování PIC16F877

V programu se využívá přerušení, asynchronní sériové komunikace a sériové synchronní komunikace po sběrnici I^2C. Princip obsluhy přerušení je stejný jako u mikrokontroléru PIC16F84.

4.2.1 Organizace paměti RAM

Velikost paměti RAM je 368B, ale tato paměť nejde adresovat celá najednou. Proto je datová paměť rozdělena do čtyř registrových bank v nichž jsou již jednotlivá paměťová místa přímo adresovatelné. Pomocí bitu RP1 a RP0 v registru *Status* se určí, která registrová banka se bude adresovat. Protože je nutné, aby některé řídicí registry byly přístupné bez nutnosti přepínat registrové banky, mají tyto registry několik absolutně adresovatelných adres.

4.2.2 Nepřímé adresování

K přístupu do paměťi dat typu RAM lze kromě přímého adresovaní použít i nepřímé adresování. K nepřímému adresování se používají dva registry - registr *Indf* a *Fsr* a bit 7 v registru *Status*. Registr *Indf* je fiktivní tj. nelze ho nijak adresovat.

Prostřednictvím registru *Indf* se zapisuje hodnota do paměťového místa daného obsahem registru *Fsr* a bitem 7 v registru *Status* (viz. následující část kódu). Stejný způsobem se i čte hodnota z paměti.

```
    MOVLW  0x20    ;pocatecni adresa v pameti dat
    MOVWF  FSR     ;vlozeni do FSR
NEXT
    CLRFR  INDF    ;nulovani pameti
    INCF   FSR,F   ;inkrementace FSR
    BTFSS  FSR,4   ;test 4.bitu FSR
    GOTO   NEXT
CONTINUE
        .
```

4.2.3 Konfigurace přerušení

Počáteční adresa programu obsluhující přerušení je 0x04h. Povolení přerušení se provede nastavením bitu GIE v registru *Intcon* na hodnotu log.1. Dále se je nutné povolit přerušení periferií, což se provede nastavením log.1 v bitu PEIE v registru *Intcon*.

```
bsf INTCON,GIE
bsf INTCON, PEIE
```

4.2.4 Konfigurace USART

Před konfigurací asynchronního sériového přenosu známeho především jako RS232 je vhodné vědět, v jakém formátu vysílač příjmá data a jakou rychlostí. V případě modelu železnice komunikace probíhá ve formátu 1 start-bit, 8 datových bitů, 1 stop-bit a bez parity. Rychlost přenosu je 9600 $baud/s$. Z určené rychlosti přenosu se podle tabulky v katalogovém listu určí hodnota časovače. Časovač slouží k určení rychlosti vysílání bitu po sériové lince. V případě modelu železnice je to hodnota $0x81h$ a tato hodnota se zapíše do registru *Spbrg*. Dále je nutné nastavit vysílání dat. Zápisem hodnoty $0x24h$ do registru *Txsta* se nastaví osmibitová šířka dat, asynchronní vysílání, povolí se vysílání a inicializuje se vysílací buffer. Po nastavení vysílaní se nastaví přijímač asynchronního sériového kanálu. Zápisem hodnoty $0x90h$ do registru *Rcsta* se povolí příjem a vysílání dat na pinech RC6 a RC7, nastaví se příjem osmi datových bitů, povolí se kontinuální příjem dat po sériové lince a inicializuje se přijímač. Jelikož lze dokončením příjmu dat ze sériové linky či ukončením vysílání dat na sériovou linku vyvolat přerušení, musí se nastavit ještě bity TXIE

a RCIE v registru *Pie*1. Bit TXIE se nastaví do stavu log.0, protože se nepoužívá přerušení k vysílaní dat po sériové lince a bit RCIE se nastaví na hodnotu log. 1, protože se používá přerušení vznikající příjmem 1B dat. Pokud by se nepoužívalo přerušení u přijmu dat, došlo by buď k zablokování celého řízení modelu nebo by došlo ke ztrátě dat při přenosu z nadřízeného systému, což není přípustné.

```
movlw  0x81         ; nastaveni rychlosti - 9600 Baud
movwf  SPBRG        ;
movlw  0x24         ;  nastaveni vysilani
movwf  TXSTA        ;
bcf    STATUS,RP0   ; nastaveni registrove banky 0
movlw  0x90         ; nastaveni prijimani
movwf  RCSTA        ;
bsf    STATUS,RP0   ; nastaveni prvni registrove banky
bcf    PIE1,TXIE    ; zakazani preruseni vysilani
bsf    PIE1,RCIE    ; povoleni preruseni prijimani
bsf    INTCON,GIE   ;
bsf    INTCON,PEIE  ; povoleni periferijni preruseni
bcf    STATUS,RP0   ;
```

4.2.5 Konfigurace I^2C

Sběrnice I^2C (Inter Integrated Circuit) je dvouvodičovou sériovou sběrnicí. Tuto sběrnici vyvinula firma Philips pro komunikaci integrovaných obvodů. Standardní komunikační rychlost je do 100kb/s. V režimu fast lze dosáhnout rychlosti až 400kb/s. V režimu high-speed lze dosáhnout rychlosti až 3.4Mb/s (verze 2.0).

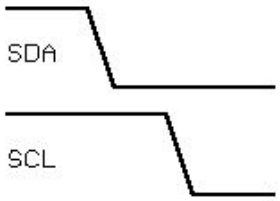
Sběrnice se skládá ze dvou vodičů - SDA (Serial Data) a SCL (Serial Clock). V nejjednodušším případě je na sběrnici jedno zařízení MASTER a jedno nebo více zařízení SLAVE. Zařízení, které pracuje jako MASTER, řídí sběrnici. Pokud si zařízení MASTER přeje zahájit komunikaci, vyšle nejprve 7-bitovou adresu cílového zařízení a bit režimu komunikace. Zařízení SLAVE poslouchají na sběrnici. Pokud některé zařízení SLAVE zjistí, že byla zařízením MASTER vyslána jeho adresa, potvrdí přijetí pomocí signálu -ACK (přidrží SDA ve stavu L) po dobu devátého hodinového impulzu. Pokud je bit režimu komunikace nastaven na 1, bude se ze zařízení SLAVE číst. V opačném případě bude proveden zápis.

Na začátku každé komunikace vysílá zařízení MASTER start sekvenci. Na konci komunikace potom zařízení MASTER vyšle stop sekvenci. Stav vodiče SDA je možné měnit pouze tehdy, pokud je vodič SCL ve stavu L. Pokud je vodič SCL ve stavu H, považuje se stav na vodiči SDA za platná data. Výjimku z tohoto pravidla tvoří

pouze start sekvence a stop sekvence.

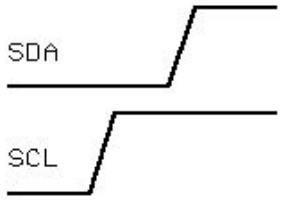
Zařízení jsou na sběrnici připojena pomocí elektronických. obvodů v zapojení open-collector. Úroveň H je na sběrnici v klidovém stavu tvořena upínacími rezistory, které jsou připojeny na napájecí napětí. To znamená, že kterékoliv zařízení může ovlivňovat stav na vodičích SDA a SCL bez nebezpečí zničení jiného zařízení. Této přednosti se také využívá pro řízení toku dat. Pokud dojde při komunikaci dvou zařízení k tomu, že jedno zařízení je zaneprázdněno a nemůže pokračovat v komunikaci, přidrží stav vodiče SCL na úrovni L a tím signalizuje stav wait. Komunikace je pozastavena a pokračuje se až po uvolnění vodiče SCL.

V klidovém stavu je na sběrnici napětí $+5\ V$. Komunikace na sběrnici I^2C se zahájí start podmínkou (obr. 4.1), kdy se signál SCL ponechá na napětí $+5\ V$ a provede se stažení signálu SDA k nule, nakonec se stáhne k nule i signál SCL.



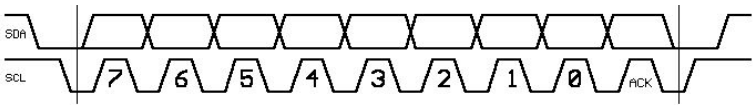
Obrázek 4.1: Počáteční podmínka komunikace na sběrnici I^2C

Komunikace na sběrnici I^2C se ukončí nastavení signálu SDA na $+5\ V$ při nastaveném signálu SCL na $+5\ V$ (obr. 4.2). Některé obvody fungují i když se stop podmínka nepoužívá.



Obrázek 4.2: Koncová podmínka komunikace na sběrnici I^2C

Samotný přenos dat (obr. 4.3) probíhá tak, že 8bitová data se přenáší od nejvýznamnějšího bitu k nejméně významnému. Stav signálu SDA se smí měnit pouze pokud je signál SCL v nule. Nastavení signálu SCL oznamuje obvodu platnost dat. Při čtení ze sběrnice I^2C se signál SDA nastaví na $+5\ V$ a data se načítají při nastaveném signálu SCL na $+5\ V$. Na konci přenosu každého bytu se ještě přenese 9. kontrolní bit ACK. Vynulováním bitu ACK obvod provede potvrzení příjmu. Při příjmu dat nuluje tento bit řídící obvod a to v případě, že má dále probíhat komunikace mezi nadřízeným a podřízeným obvodem. Pokud bude po příjmu následovat podmínka stop, pak se tento poslední bit nastaví do stavu log.1.



Obrázek 4.3: Přenos jednoho bytu na sběrnici I^2C

Ke konfiguraci modulu synchronního sériového portu se používají registry $Sspcon$, $Sspcon2$, $Sspstat$ a $Sspadd$. Zapsáním hodnoty $0x18h$ do registru $Sspcon$ se nastaví I^2C mód synchronního sériového portu, který bude na sběrnici vystupovat jako nadřízený systém s frekvencí hodinového signálu danou následujícím vztahem:

$$Fosc = (SSPAD + 1) \div 4$$

Dále se v registru $Sspstat$ vynulují příznaky SMP a CKE, které se používají při samotné komunikaci po sběrnici I^2C. Registr $Sspad$ je ve skutečnosti časovač určující frekvenci hodinového signálu sběrnice I^2C. Zápisem hodnoty $0x31h$ do registru $Sspad$ nastavíme frekvenci 100kHz při 20MHz kmitočtu mikrokontroléru PIC1F877.

```
Config
      bcf     STATUS,RP0      ; registrova banka 0
      bsf     SSPCON,SSPEN    ; povoleni I2C mode
      bsf     SSPCON,SSPM3    ; nastaveni I2C
      bcf     SSPCON,SSPM2    ;
      bcf     SSPCON,SSPM1    ;
      bcf     SSPCON,SSPM0    ;
      bsf     STATUS,RP0      ; registrova banka 1
      clrf    SSPCON2         ;
      bcf     SSPSTAT,SMP     ;
      bcf     SSPSTAT,CKE     ; nastaveni I2C Levels
      movlw   0x31            ; nastaveni  rychlosti I2C
      movwf   SSPADD          ; 100k at 20Mhz kmitočtu
      bcf     STATUS,RP0      ; registrova banka 0
      return
```

Komunikace po sběrnici I^2C probíhá podle daného protokolu. Ke zlepšení obsluhy komunikace po sběrnici I^2C byly napsány podprogramy, které zahajují komunikaci, posílají potvrzení příjmu dat a ukončují komunikaci s podřízeným systémem. Prvním z těchto podprogramů je podprogram pro zahájení komunikace. Zahájení komunikace se provede nastavením bitu SEN v registru SSPCON2 do stavu log.1 a následným čekáním na potvrzení od podřízeného zařízení.

```
I2CStart
        bsf     STATUS,RP0        ;registrova banka 1
        bsf     SSPCON2,SEN       ;inicializace START podminky
        bcf     STATUS,RP0        ;registrova banka 0
        call    I2CWait           ;cekani
        return
```

K přenosu dat slouží podprogram I2CSend. Před zavoláním tohoto podprogramu se načte do pracovního registru W jeden byte posílaných dat. Zavolaný podprogram přesune obsah pracovního registru W do vysílacího bufferu $Sspbuf$. Tímto přesunem je hardwarově inicializováno vysílání dat. Odeslání obsahu bufferu je potom signalizováno nastavením bitu $SSPIF$ v registru $Pir1$ do stavu log.1. Tento příznak je nutné softwarově mazat, aby mohl byt vysílací buffer opětovně použit.

```
I2CSend
        movwf   SSPBUF            ;presun do SSPBUF
        btfss   PIR1,SSPIF        ;cekani na odeslani
        goto    $-1               ;
        bcf     PIR1,SSPIF        ;cekani na potvrzeni
        return
```

Pro příjem dat ze sběrnice I^2C slouží podprogram I2CReceive. V podprogramu se nejdříve provede test na skončení vysílaní dat, které by ješte v době přijímání mohly být ve vysílacím bufferu. K přijímaní a vysílaní se totiž používá stejný registr $Sspbuf$. Po úspěšném testu tj. není nic ve vysílacím bufferu, se povolí příjem dat nastavením bitu RCEN do stavu log.1 a čeká se do doby než bude přijímací buffer $Sspbuf$ naplněn příchozími daty. Naplnění přijímacího bufferu je signalizováno nastavením bitu SSPIF do stavu log.1, který je potom nutné softwarově nulovat. Podprogram nakonec vyzvedne obsah přijímacího bufferu a zapíše ho do paměti.

```
I2CReceive
        bsf     STATUS, RP0       ; registrova banka 1
        btfsc   SSPSTAT,2         ; test na stav vysilani
        goto    $-1               ;
        bsf     SSPCON2, RCEN     ; povoleni prijimu dat
        bcf     STATUS, RP0       ; registrova banka 0
        call    I2CWait           ; cekani
        movf    SSPBUF,0          ; vyzvednuti dat z SSPBUF
        movwf   TMP_SSPBUF        ;
        return
```

K ukončení komunikace na sběrnici slouží podprogram I2CStop, který nastavením bitu PEN v registru $Sspcobn2$ do stavu log.1 vygeneruje ukončení komunikace.

```
I2CStop
     bsf     STATUS,RP0      ;registrova banka 1
     bsf     SSPCON2,PEN     ;generovani STOP komunikace
     bcf     STATUS,RP0      ;registrova banka 0
     call    I2CWait         ;cekani
    return
```

V některých případech může dojít k uváznutí komunikace na sběrnici a pro tento stav je připraven podprogram I2CRestart, který provede reinicilizaci komunikace. Podprogram je stejný jako podprogram pro inicializaci komunikace, pouze se liší v tom, že nastavuje do stavu log.1 v registru $Sspcon2$ bit $RSEN$ místo bitu SEN.

```
I2CRestart
      bsf     STATUS,RP0      ;registrova banka 1
      bsf     SSPCON2,RSEN    ;inicializace START podminky
      bcf     STATUS,RP0      ;registrova banka 0
      call    I2CWait         ;cekani
      return
```

Protože komunikace po sběrnici I^2C je komunikací potvrzovanou, jsou vytvořeny i podprogramy pro posílání potvrzení při příjmu dat.

```
I2CAck
     bsf  STATUS,RP0        ;registrova banka 1
     bcf  SSPCON2,ACKDT     ;nastaveni  ACK
     bsf  SSPCON2,ACKEN     ;poslani ACK
     bcf  STATUS,RP0        ;registrova banka  0
     call I2CWait           ;cekani
     return

I2CNak
     bsf  STATUS,RP0        ;registrova banka  1
     bsf  SSPCON2,ACKDT     ;nastaveni NAK
     bsf  SSPCON2,ACKEN     ;poslani NAK
     bcf  STATUS,RP0        ;registrova banka  0
     call I2CWait           ;cekani
    return
```

Posledním často využívaným podprogram je čekání na dokončení operace na sběrnici I^2C. Dokončení operace je vždy signalizováno hardwarovým nastavením bitu $SSPIF$ v registru $Pir1$ do stavu log.1, které se musí potom softwarově nulovat.

```
I2CWait
    btfss  PIR1,SSPIF
    goto    $-1            ;cekani na dokonceni operace
    bcf     PIR1,SSPIF     ;nulovani priznaku
    return
```

4.2.6 Konfigurace časovače TMR0

Norma NMRA9.21 klade na časový průběh signálu DCC velice přísné požadavky, které byly uvedeny v kapitole 2.1.1. a vzhledem k rozsahu programu nelze generovat DCC signál jinak než pomocí přerušení od časovače TMR0. Toto přerušení se nastaví v registru INTCON pomocí bitu 5, který se uvede do stavu log.1, a v registru OPTION_REG pomocí bitu 5, který určí, že registr TMR0 bude plnit funkci časovače.

```
bcf  OPTION_REG,5      ;nastaveni TMR0 jako casovace
bsf INTCON,5          ;povolujeme preruseni od TMR0
```

4.2.7 Popis programu

Program je rozdělen do tří částí - inicializace mikrokontroléru, hlavní program s podprogramy a obsluha přerušení. Během inicializace se provede konfigurace obsluhy přerušení, konfigurace sběrnice I^2C a USART, konfigurace časovače $TMR0$ a základní nastavení obsahu používaných registrů. Hlavní program je tvořen cyklem, ve kterém se testují jednotlivé bity v příznakovém registru *Control*. Každý testovaný bit reprezentuje požadavek na obsloužení určité události. Bit 1 v registru *Control* reprezentuje požadavek na aktualizaci výstupů ovládající polohu výhybek. Pokud je bit ve stavu log.1, zavolá se podprogram *Switchs*. V případě, že vznikl požadavek na aktualizaci výstupů spínající LED diody v semaforech, je bit 2 v registru Control nastaven na log.1. Nastavením bitu 2 registru *Control* na log.1 se vyvolá podprogram *Light* určený k aktualizaci stavu semaforů. Bit 5 registru *Control* signalizuje požadavek nadřízeného systému, aby mu byla poslána data o pohyby lokomotivy. V případě, že bit 4 registru *Control* je ve stavu log.1 je zavolán podprogram Transmit, který odešle data po RS232. Pokud mikrokontrolér příjme data od nadřízeného systému, nastaví se bit 5 registru *Control*. Nastavením bitu 5 se zavolá podprogram *Sorting* provádějící aktualizaci dat periferií. V každém cyklu

hlavní programové smyčky se provádí sbíraní dat o pohybu lokomotiv po kolejišti. Sběr dat provádí podprogram MEASURE. K příjmu dat po RS232 a ke generování DCC signálu se využívá přerušení mikrokontroléru.

4.2.7.1 Podprogram Sorting

Podprogram *Sorting* na základě dat přijatých od nadřízeného systému aktualizuje data jednotlivých periferií. Příkazy posílané po sériové lince mají pevně danou strukturu (tab. 4.1). Na začátku je vždy adresa zařízení, které je určen následující jeden byte data. Potom je poslán jeden byte dat a za ním kontrolní součet adresy zařízení a dat.

adresa	data	CRC	popis
0x10h	DDDDDDDDD	CCCCCCCC	rychlost lokomotivy BR221
0x11h	DDDDDDDDD	CCCCCCCC	rychlost lokomotivy V180
0x20h	0000DDDDD	CCCCCCCC	výhybky 1 − 4
0x21h	0000DDDDD	CCCCCCCC	výhybky 4 − 8
0x22h	0000DDDDD	CCCCCCCC	výhybky 9 − 12
0x23h	0000DDDDD	CCCCCCCC	výhybky 13 − 15
0x40h	0000DDDDD	CCCCCCCC	semafor č.6
0x41h	0000DDDDD	CCCCCCCC	semafor č.8
0x42h	0000DDDDD	CCCCCCCC	semafor č.7
0x43h	0000DDDDD	CCCCCCCC	semafor č.11
0x44h	0000DDDDD	CCCCCCCC	semafor č.3
0x45h	0000DDDDD	CCCCCCCC	semafor č.4
0x46h	0000DDDDD	CCCCCCCC	semafor č.12
0x47h	0000DDDDD	CCCCCCCC	semafor č.5
0x48h	0000DDDDD	CCCCCCCC	semafor č.1
0x49h	0000DDDDD	CCCCCCCC	semafor č.2
0x4Ah	0000DDDDD	CCCCCCCC	semafor č.10
0x4Bh	0000DDDDD	CCCCCCCC	semafor č.19

Tabulka 4.1: Struktura příkazů

Všechny data přijaté po sériové lince se ukládají do zásobníku, ze kterého se příjmou první tři byte podprogramem *Sorting* k dalšímu zpracování. Před zpracováním se nejdříve provede kontrola správnosti přijatých dat. Kontrola se provádí součtem XOR adresy zařízení a dat. Výsledek se potom porovná s třetím bytem. Pokud se výsledek XOR součtu rovná třetímu bytu, přijatá data jsou platná. Pokud se sobě nerovnají, přijatá data nejsou platná a zahodí se. Po kontrole platnosti dat

se zjistí, kterému zařízení jsou data určena. V případě, že jsou určeny lokomotivě BR221, skočí se do části programu obsluhující aktualizaci dat lokomotivy BR221. Aktualizace probíhá tak, že se zapíší data do připravené datové struktury a následně se provede přepočet obsahu datové struktury, který je závislý na měnících se datech. Tento postup je stejný i u ostatních zařízení, pouze se mění místo, kam se data zapisují. Po zapsání dat se hodnota ukazatele zásobníku, sníží o hodnotu 3 a provede se další načtení tří bytů. Pokud hodnota ukazatele klesne pod $0xA0h$, je třídění dat zastaveno, protože se celý zásobník přijatých dat zpracoval. Vzhledem k tomu, že komunikace mezi modelem a nadřízeným systémem není tolik intenzivní, postačilo toto řešení vyrovnávacího bufferu, protože do příchodu dalších dat se přijatá data zpracují.

4.2.7.2 Podprogram Switchs

Podprogram *Switchs* aktualizuje stavy na výstupu vstupně-výstupního expandéru PCF8574, který přepíná polohu výhybek. Stavy výstupů expandéru PCF8574 se ovládají přes sběrnici I^2C. První skupina výhybek má na sběrnici I^2C adresu $0x44h$, druhá skupina adresu $0x42h$. Do první skupiny patří všechny výhybky v nádraží "Marketta" a první vstupní výhybka nádraží "Hawkey". Ve druhé skupině jsou všechny ostatní výhybky. V následující tabulce 4.3 je znázorněno který bit odpovídá který výhybce.

Podprogram funguje tak, že se voláním podprogramu I2CStart zahájí komunikace na sběrnici I2C. Potom se uloží do pracovního registru adresa 0x44 expandéru PCF8574, který obsluhuje první skupinu výhybek a zavolá se podprogram I2CSend. Ten si vezme adresu z pracovního registru a pošle ji na sběrnici I2C. Potom se do pracovního registru přesune obsah registru $Vyhy01$ a opět se zavolá podprogram I2CSend, který pošle po sběrnici data uložená v pracovním registru. V registru $Vyhy01$ jsou uložené informace o poloze výhybek první skupiny. Poslání dat se ješte jednou zopakuje a následně se zavolá podprogram I2CStop, který ukončí komunikaci s expandérem PCF8574 adresy $0x44h$. Stejným postupem se komunikuje i s expandérem PCF8574, který obsluhuje druhou skupinu výhybek, pouze se změní adresa $0x44h$ na adresu $0x42h$ a registr $Vyhy01$ na registr $Vyhy02$. Před návratem do hlavního programu se ještě vynuluje bit 1 registru *Control*, čímž se signalizuje obsloužení požadavku na aktualizaci stavu výstupů ovládají polohu výhybek.

4.2.7.3 Podprogram Light

Princip funkce podprogramu *Light* je podobný jako princip funkce podprogramu *Switchs*. Jediný rozdíl je v adresách expandérů PCF8574, které spínají jednotlivé

Adresa na I2C	Číslo bitu	Číslo výhybky
0x44	bit 0	výhybka 01
0x44	bit 1	výhybka 02
0x44	bit 2	výhybka 03
0x44	bit 3	výhybka 04
0x44	bit 4	výhybka 05
0x44	bit 5	výhybka 06
0x44	bit 6	výhybka 07
0x44	bit 7	výhybka 08
0x42	bit 0	výhybka 09
0x42	bit 1	výhybka 10
0x42	bit 2	výhybka 11
0x42	bit 3	výhybka 12
0x42	bit 4	výhybka 13
0x42	bit 5	výhybka 14
0x42	bit 6	výhybka 15
0x42	bit 7	

Tabulka 4.2: Rozřazení výhybek k jednotlivým výstupům expandéru

LED diody v semaforech a v registrech, ve kterých jsou uloženy informace o stavech jednotlivých světel semaforů. Bity 7 a 3 odpovídají LED diodám přestavující nejvýše umístěné světlo semaforu.

Adresa na I2C	Skupina bitů	Označení semaforu
0x40	0 − 3	semafor č.6
0x40	4 − 7	semafor č.8
0x48	0 − 3	semafor č.10
0x48	4 − 7	semafor č.9
0x46	0 − 3	semafor č.1
0x46	4 − 7	semafor č.2
0x4A	0 − 3	semafor č.7
0x4A	4 − 7	semafor č.11
0x4C	0 − 3	semafor č.12
0x4C	4 − 7	semafor č.5
0x4E	0 − 3	semafor č.3
0x4E	4 − 7	semafor č.4

Tabulka 4.3: Rozřazení semaforů k jednotlivým expandérům PCF8574

4.2.7.4 Podprogram Measure

Podprogram *Measure* zajišťuje sbírání dat z proudových snímačů a přestavníků výhybek připojených přes multiplexor 74HC154 k portu B a jejich ukládaní do paměti. Ukládání dat se provádí pomocí nepřímého adresování. Na začátku podprogramu se provede inicializace podprogramu. Registr *Timer* používaný k vytváření zpoždění mezi jednotlivými čteními portu B je naplněn hodnotou $0xFFh$. Hodnota $0xFFh$ odpovídá 20 μs mezi jednotlivými čteními portu B. Dále je do registru *Countsen*, který je ve funkci počítadla, zapsána hodnota $0x0Bh$. Hodnota odpovídá počtu skupin proudových snímačů a počtu snímačů poloh přestavníků. A nakonec je inicializováno nepřímé adresovaní. Do registru FSR sloužícího jako ukazatel do paměti dat při nepřímém adresování, se zapíše počáteční adresa paměti, kam se budou ukládat přečtená data z portu B. Potom už následuje hlavní cyklus podprogramu. Provede se dekrementace se obsahu registru *Countsen* o hodnotu 1 a výsledek dekrementace se uloží do pracovního registru W, odkud se zapíše na port A. Tím se určí, která skupina snímačů se připojí na paralelní sběrnici připojenou na port B. Počká se dokud obsah registru *Timer* nebude nulový a potom se stav portu B uloží pomocí registru INDF do paměti. Inkrementuje se registr FSR a dekrementuje se registr *Countsen*. Tento cyklus se provádí tak dlouho, dokud není obsah registru *Countsen* nulový tj. dokud se neaktualizují všechny data ze snímačů. Po opuštění cyklu se mikrokontrolér vrátí do hlavního programu.

4.2.7.5 Podprogram Transmit

Podprogram *Transmit* zajišťuje odesílání dat ze snímačů, které jsou uloženy v paměti dat. Počáteční adresa paměti je $0x64h$. Tato adresa se zapíše do registru FSR. Dále se inicializuje registr *Countsen*, který je počítadlem odeslaných bytů dat.

Komunikace mikrokontroléru s nadřízeným systémem probíhá podle předem pevně stanoveného protokolu. Na vyžádání nadřízeného systému mikrokontrolér začne posílat data ve formátu, který je uveden v tabulce (tab. 4.4). Posílaná data jsou zakončena třemi byty s hodnotou $0xFFh$. Velikost posílaných dat je 36B a tato velikost se nemění.

adresa	data	CRC	číslo úseku nebo výhybky
0x10h	DDDDDDDDD	CCCCCCCC	45 47 53 46 06 42 48 44
0x11h	DDDDDDDDD	CCCCCCCC	34 33 32 31 52 51 50 49
0x12h	DDDDDDDDD	CCCCCCCC	22 03 23 24 27 26 28 25
0x13h	DDDDDDDDD	CCCCCCCC	39 XX XX XX 29 04 38 30
0x14h	DDDDDDDDD	CCCCCCCC	09 08 07 01 10 11 13 12
0x15h	DDDDDDDDD	CCCCCCCC	40 43 05 41 14 15 16 17
0x16h	DDDDDDDDD	CCCCCCCC	21 20 19 02 36 18 37 35
0x17h	DDDDDDDDD	CCCCCCCC	SXX SXX SXX SXX SXX SXX SXX SXX
0x18h	DDDDDDDDD	CCCCCCCC	SXX SXX SXX SXX SXX SXX SXX SXX
0x19h	DDDDDDDDD	CCCCCCCC	SXX SXX SXX SXX SXX SXX SXX SXX
0x1Ah	DDDDDDDDD	CCCCCCCC	SXX SXX SXX SXX SXX SXX SXX SXX

Tabulka 4.4: Formát data při odesílání po RS232

Vždy před odesláním dalšího bytu dat se testem na nulu v registru *Countsen* zjistí, zda se už odeslaly všechny data snímačů. V případě, že obsah registru *Countsen* bude nulový tj. už se odeslaly všechny data, provede se skok na odeslání zakončovací značky posílaných dat a potom podprogram *Transmit* skončí. Pokud obsah registru *Countsen* není nulový, načte se do registru *Datre*1 posílaný byte dat. Obsah registru *Datarail*, ve kterém je adresa posílaných dat, se inkrementuje o hodnotu 1 a pošle se podprogramem *Send* po RS232 nadřízenému systému. Potom se podprogramem SEND pošle obsah registru DATRE1. Po odeslání se provede XOR součet obsahů registrů *Datre*1 a *Datarail* a výsledek se uloží do pracovního registru *W* a odešle podprogramem *SEND* na RS232. Po odeslání všech dat podprogram vynuluje bit 4 registru *Control*.

4.2.7.6 Přerušení

Mikrokontrolér při přerušení skočí vždy na adresu $0x04h$ v paměti programu, kde očekává program obsluhují událost, která vyvolala přerušení. Priorita obsloužení událostí se ponechává na uvážení programátora. Dále se ponechává na programátorovi ukládání si stavu registru *Status* a pracovního registru *W*. Na začátku obsluhy přerušení se provede uložení stavů registru *Status* a pracovního registru *W*. Následně se testováním bitu *T0IF* registru *Intcon* a bitu *RCIF* registru *Pir*1 zjistí zdroj přerušení. Pokud by test na log.1 u bitu T0IF registru *Intcon* byl pozitivní, mikrokontrolér přeskočí na obsluhu přerušení, které vyvolalo přetečení časovače *Tmr*0. Tento časovač se používá ke generování DCC signálu. Pokud by test na log.1 u bitu registru *Pir*1 byl pozitivní, bude obslouženo přerušení, které znamená jednoho bytu dat po sériovém kanálu RS232.

Generování DCC signálu probíhá tak, že v paměti dat je připravena pro obě lokomotivy datová struktura přesně odpovídající struktuře DCC paketu posílaného lokomotivám. Tyto struktury jsou v paměti celkem čtyři, prvním dvěma se obsah mění, třetí je stabilně plná log.1 a čtvrtá se používá k vysílání. Všechny tři datové struktury se cyklicky vysílají na DCC výstup. Na začátku generování DCC signálu se nastaví počáteční hodnota registru $Tmr0$. Při vzniku přerušení vyvolaného přetečením registru časovače $Tmr0$ se zjistí, kolik bitů DCC paketu zbývá k vyslání na DCC výstup. V případě, že nezbývá žádný bit, tak se do pracovní struktury nahraje v pořadí další datová struktura DCC paketu a začne se vysílat na DCC výstup. Vysílání jednotlivých bitů se provádí rotací datové struktury přes bit $CARRY$ registru $Status$. Vysílání jednoho bitu je rozděleno na dvě části. V první části je pin $RC0$ ve stavu log.1 a pin $RC1$ ve stavu log.0 a v druhé části je to opačně. Délka setrvání v těchto stavech je dána tím, zda se vysílá log.0 nebo log.1, tj. tím zda bit $CARRY$ registru $Status$ je ve stavu log.1 či log.0. Podle tohoto bitu se do registru $Tmr0$ nahraje konstanta pro odměřování doby pro log.1 nebo log.0. Na konec rotované datové struktury se načítají log.1. Po změně stavu na pinech RC0 a RC1 se skočí na návracení z přerušení.

Při obsluze přerušení od přijetí jednoho bytu poslaného sériovým kanálem RS232 se nejprve přesune obsah přijímacího bufferu $Rcreg$ do registru Rcw. Následně se otestuje, zda posílaný byte je koncová značka $0xFFh$. Pokud posílaný byte je koncová značka, nastaví se bit 4 registru $CONTROL$ do stavu log.1. Tímto bitem se signalizuje, že se má provést zpracování přijatých dat. V případě, že přijatý byte dat není koncová značka, přesune se byte do místa, které je určeno obsahem registru Fsr. Po zapsání tohoto bytu dat se provede inkrementace registru Fsr o hodnotu 1 a skok na navrát z přerušení.

Kapitola 5

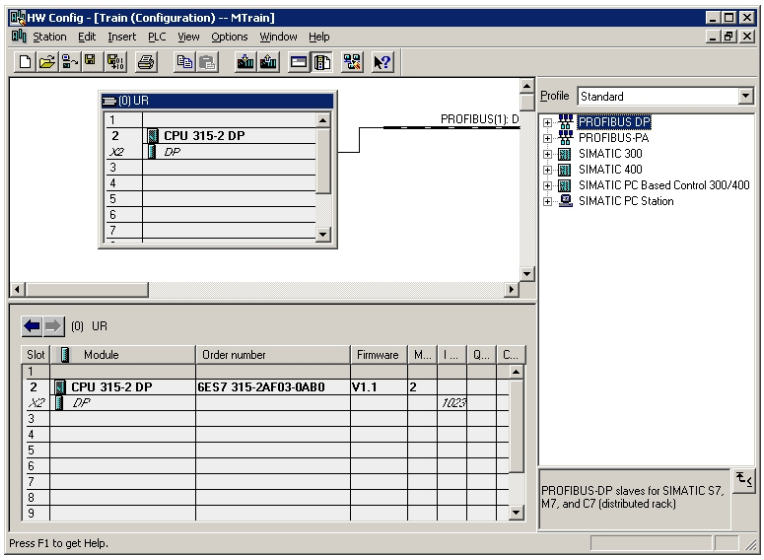
Řídicí algoritmus v PLC

5.1 Hardwarová konfigurace

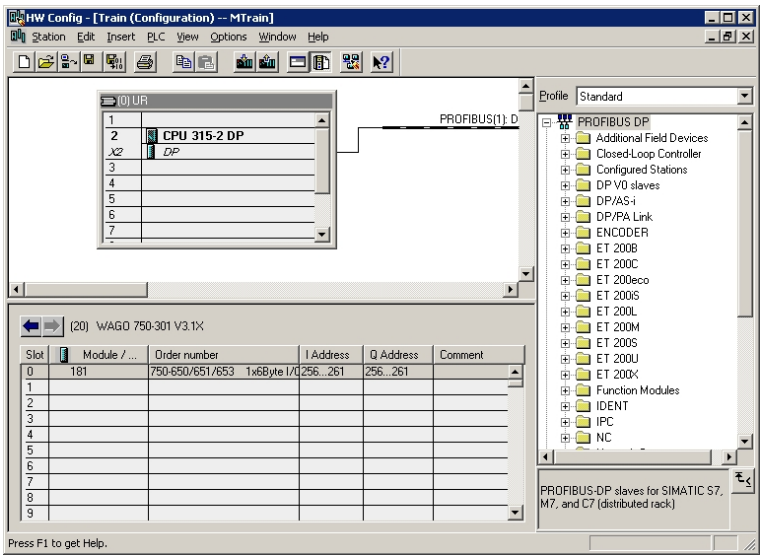
Aby se model mohl řídit pomocí PLC automatu Simatic S7-315 2DP, bylo nutné k PLC připojit buď přímo kartu se sériovým výstupem nebo přes sběrnici profibus připojit převodník Profibus/Sériový kanál. Vzhledem k tomu, že převodník profibus/sériový kanál od firmy WAGO je levnější než karta od firmy Siemens a lépe se s ním pracuje v řídicím programu, byla ke komunikaci vybrána druhá varianta. Od firmy WAGO byl vybrán modul 750-650/003-000 Serial Interfaces RS232.

5.1.1 Hardwarová konfigurace PLC

Hardwarová konfigurace Simaticu se provede podle následujícího obrázků obr. 5.1 a obr. 5.2. Rychlost sběrnice profibus je nastavena 1,5Mb/s.



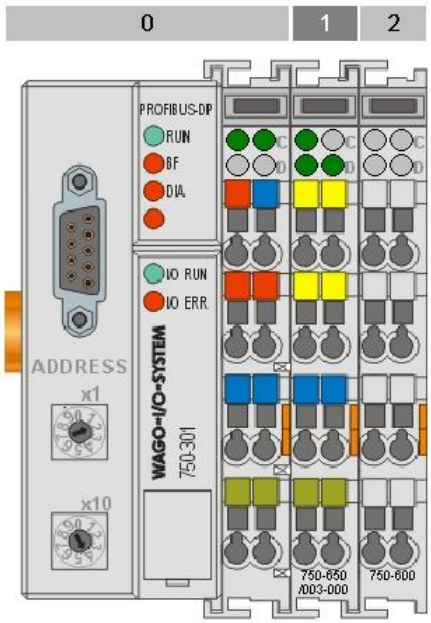
Obrázek 5.1: Konfigurace Simaticu



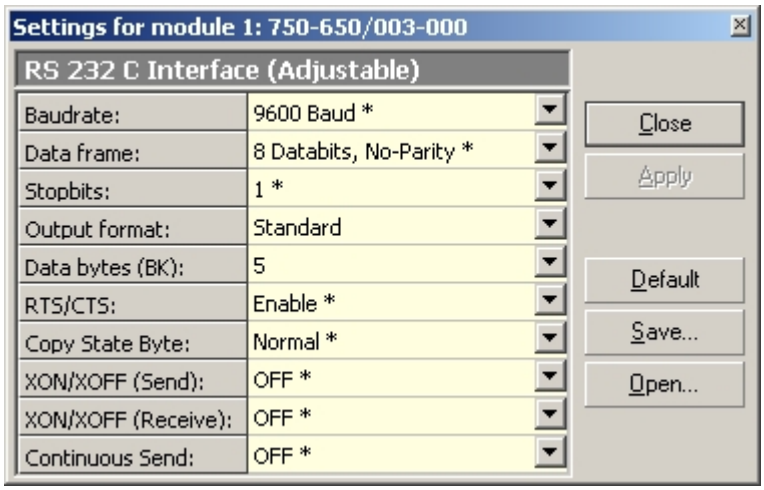
Obrázek 5.2: Konfigurace profibusového modulu WAGO 750-650

5.1.2 Hardwarová konfigurace Wago Serial Interface RS232

Konfigurace modulu WAGO 750-650/003-000 Serial Interface RS232 se provede podle následujícího obrázků. Důležité je zakázat kontinuální vysílání dat. Pokud se tento bit nezakáže, tak nastane situace, že lze přijímat data po RS232, ale nelze je vysílat.



Obrázek 5.3: Nastavení adresy profibusového modulu



Obrázek 5.4: Konfigurace modulu WAGO 750-650/003-000

5.2 Řídicí algoritmus v PLC

Řídicí algoritmus se skládá z hlavního programu umístěném v organizačním bloku OB1 a z funkčních bloků a funkcí volaných z organizačního bloku OB1. Řídicí algoritmus byl rozdělen na několik vzájemně se ovlivňujících se částí . První částí je řízení pohybu lokomotivy po kolejišti, v řídicím algoritmu tuto část představuje funkční blok FB1. Druhou částí je ovládání zhlaví všech nádraží, v algoritmu je reprezentováno funkcemi FC101, FC107. Třetí částí je stavba vlakových cest mezi sousedními nádražími, v algoritmu reprezentovaná funkční bloky FB2, FB4 a FB6 a FB7. Vzhledem k tomu, že lokomotivy mají v sobě dekodér určující směr pohybu lokomotivy, nelze svévolně postavit lokomotivu na koleje. Lokomotiva musí být postavena tak, že přední část lokomotivy BR221 (černý komín) a zadní část lokomotivy V180 (černý komín) směřují k na levou stranu nádraží "Marketta". Pokud lokomotivy budou opačně postavené, algoritmus nebude fungovat.

Celé kolejiště je v paměti PLC popsáno rozsáhlou tabulkou, která se nachází v datovém bloku DB4 - Useky. Tabulka je rozdělena na tři části. V první části tj. od adresy 0 do adresy 480 je popsáno reálné kolejiště, v druhé části tj. od adresy 500 do adresy 980 je popsána virtuální část kolejiště. V obou těchto částech se nejsou popsány traťové úseky, ve kterých jsou výhybky. Tyto úseky jsou popsány ve třetí části, která je od adresy 1500. Jeden fiktivní řádek tabulky představuje jeden úsek. Struktura jednoho řádku vypadá takto:

Položka	Velikost	popis
Číslo úseku	1B	číslo úseku
Adresa01	2B	adresa následujícího úseku
Adresa02	2B	adresa předcházejícího úseku
Stav	1B	stav úseku
Rychlost	1B	hodnota rychlosti pro automatické řízení
Atributy	1B	atributy úseku
Volný byte	1B	
Semafor	1B	adresa semaforu, který má být sledován

Tabulka 5.1: Struktura řádku v tabulce Úseků

První byty je číslo úseku. Další dva byty jsou adresa následujícího úseku tj. úseku, který je od něho na pravé straně při pohledu od hrany stolu kolejiště. Další dva byty tvoří adresu úseku předcházejícího tj. úseku, který je na levé straně úseku při pohledu od strany stolu kolejiště. Další byte, v pořadí šestý, je stavový byte. Tento byte podává informaci o tom, zda a kým je daný úsek je obsazen. Přesná struktura bytu je zobrazena následující tabulkou 5.2.

Bit	Popis
bit 0	
bit 1	v úseku je lokomotiva BR221
bit 2	v úseku je lokomotiva V180
bit 3	úsek je obsazen
bit 4	
bit 5	
bit 6	úsek je zamknut lokomotivou V180
bit 7	úsek je zamknut lokomotivou BR221

Tabulka 5.2: Struktura stavového bytu

Dalším bytem je hodnota rychlosti pohybu lokomotivy po kolejišti. Tato rychlost se používá v případě, že lokomotiva je ovládaná automatem. Osmý byte je u reálného úseku kolejiště volný, u virtuálního úseku se v něm nachází délka virtuálního úseku. V posledním bytu pomyslného řádku tabulky je relativní adresa semaforu, který lokomotiva sleduje v průběhu přejezdu daného úseku. Všechny data semaforů se nacházejí v datovém bloku DB3.

5.2.1 Komunikace modulem WAGO 750-650

Profibusový modul Wago 750-650 je nakonfigurován tak, že na periferijní výstup lze zapsat až 6B a z periferijního vstupu lze číst taky až 6B. Při zápisu je skladba dat následující. První byte dat je řídicí byte (tab. 5.3), který nese informace kolik bytů se bude zapisovat a dále jsou v něm řídicí příznaky pro inicializaci, čtení a zápis. Ostatní byte jsou datové. Při čtení je první byte dat stavový (tab. 5.4). Obsahuje informace o počtu přijímaných bytů a řídicích příznacích pro inicializaci, čtení a zápis. Vzhledem k tomu, že se mezi modulem Wago a PLC přenáší až 6B, bylo nutné k přenosu dat použít funkce SFC15 a SFC14. Funkce SFC14 přenese z modulu 6B do PLC a funkce SFC15 přenese 6B z PLC do modulu Wago.

bit	Označení	Popis
bit 0	TR	požadavek na vysílání
bit 1	RA	potvrzení příjmu
bit 2	IR	požadavek na inicializaci
bit 3	0	vždy je nula
bit 4	OL0	
bit 5	OL1	OL0,OL1 a OL2 udávají celkový počet vysílaných datových bytů
bit 6	OL2	
bit 7	0	vždy je nula

Tabulka 5.3: Control byte

bit	Označení	Popis
bit 0	TA	potvrzení vysílaní
bit 1	RR	požadavek na příjem
bit 2	IA	potvrzení inicializace
bit 3	BUF_F	vstupní buffer je plný
bit 4	IL0	
bit 5	IL1	IL0,IL1 a IL2 udávají celkový počet příjmutých datových bytů
bit 6	IL2	
bit 7	0	vždy je nula

Tabulka 5.4: Status byte

Před samotnou komunikací po RS232 je nutné modul Wago inicilizovat. Inicializace se provede zápisem řídicího bytu, který má v nastaven bit IR na log.1 a ostatní bity jsou nulové. Na zbývajících šesti bytech nezáleží.

Control byte	Output byte	Output byte	Output byte	Output byte	Output byte
0000 0100	0000 0000	0000 0000	0000 0000	0000 0000	0000 0000

Tabulka 5.5: Požadavek na inicializaci

Když modul Wago vrátí v odpověďních 6B dat, které se přečtou funkcí SFC14, následující sekvenci bitu. Je modul inicializován. V opačném případě není a musí se opakovat čtení funkcí SFC14. Písmena "X" znamenají, že na stavu tohoto bitu nezáleží. Po uspěšné inicializaci se může začít s komunikací po RS232.

Status byte	Input byte	Input byte	Input byte	Input byte	Input byte
0XXX 01XX	0xXXh	0xXXh	0xXXh	0xXXh	0xXXh

Tabulka 5.6: Potvrzení na inicializace

Zápis na sběrnici RS232 se provádí pomocí bytu 2 až 6, které se plní daty. Pokud je dat méně než 5 bytů, naplní se jen ty nejnižší. Do řídicího bytu se zapíše kolik bytů bude posílano a invertuje se bit TR v řídicím bytu. Další zápis lze provést až když modul Wago vrátí ve stavovém bytu na pozici bitu 0 stejnou hodnotu jaká mu byla poslána.

Čtení ze sběrnice RS232 je podobné. Nejprve se musí poslat modulu žádost o čtení dat. Při posílání žádosti musí být bit 3 nulový a bit 1 musí být invertovaný oproti poslední operaci čtení z RS232. Pokud modul vrátí stavový byte se stejnou hodnotou bitu 1 jaká mu byla poslána, tak jsou příchozí data platná a v horním niblu stavového bytu je počet platných dat. V případě, že se bude ve čtení pokračovat, je nutné zinvertovat bit 1 v řídicím bytu a poslat ho funkcí SFC15 modulu Wago.

5.2.2 Funkční bloky

5.2.2.1 Funkční blok FB1 - řízení lokomotivy

Funkční blok FB1 ovládá pohyb lokomotivy jak po reálném kolejišti tak i virtuálním. Řídí její rychlost, kontroluje volnost následujících vlakových úseků, kontroluje semafory a uzamčení úseku, stará se o přechod mezi virtuální a reálnou částí kolejiště. Při přechodu z reálné části do virtuální bezpečně odveze skutečnou lokomotivu na odstavné nádraží, kde čeká až virtuální lokomotiva projede celou virtuální část.

Funkční blok FB1 je rozdělen na tři části. První část ovládá reálnou lokomotivu fyzicky přítomnou na modelu železnice. Druhá část ovládá virtuální lokomotivu tj. lokomotivu, která se pohybuje po kolejišti naprogramovaném ve vizualizačním prostředí InTouch. Třeří část zajišťuje přechod mezi virtuální a reálnou lokomotivou. První dvě části jsou z velké části identické, jediný rozdíl je ve způsobu přechodu mezi jednotlivými traťovými úseky.

Před použitím sledování pohybu lokomotivy, se musí nejdříve lokomotiva ručně přivést na synchronizační úsek v reálném kolejišti, kde se do paměti lokomotivy zapíše aktuální poloha lokomotivy a nastaví se výchozí hodnoty registrů. Od této inicializace lokomotiva sleduje svůj pohyb po kolejišti v tabulce traťových úseku. Sledování funguje tak, že lokomotiva si pamatuje adresu řádku popisující traťový úsek, ve kterém se lokomotiva právě nachází. Od této adresy se potom pomocí ukazatele směru pohybu lokomotivy odvíjí i znalost dvou bezprostředně navazujících traťových úseků.

Sledování pohybu lokomotivy probíhá tak, že lokomotiva si do své paměti načte adresy a stavy dvou bezprostředně následujících úseků. Pokud načítaná adresa následujícího úseku, který bezprostředně nesousedí s aktuálním úsekem, je rovna hodnotě $0xFFh$ zapíše, se do paměti pro tento úsek adresa patřící bezprostředně následujícímu úseku. V případě, že načítaná adresa bezprostředně následujícího úseku je rovna hodnotě $0xFFh$, je do místa pro tento úsek nahrána adresa aktuálního úseku a lokomotiva je zastavena. Pokud nastane situace, kdy úsek, který je veden jako bezprostředně následující se stane aktivním, tj. bude se v něm lokomotiva nalézat, nastane přesun ukazatele aktuálního úseku ze současného úseku na úsek následující a aktualizují se stavy těchto úseku. V případě, že lokomotiva před sebou tlačí vagóny, k přesunu ukazatele dojde v momentě, kdy se aktivní úsek stane neaktivní tj. lokomotiva ho opustí. Situace, kdy na každé straně lokomotivy je vagón, není přípustná.

Funkční blok FB1 umožňuje řídit rychlost lokomotivy buď operátorem nebo automaticky. Pokud lokomotiva má nastaveno automatické řízení, hodnota rychlosti je dána načtenou hodnotou z tabulky traťových úseků. V této tabulce nejsou maximální rychlosti, kterou se nechá projet daný úsek, ale rychlosti určené pro automatické řízení. V celém kolejišti lokomotiva může jet maximální nejvyšší rychlostí.

Při pohybu po kolejišti si lokomotiva kontroluje volnost druhého následujícího traťového úseku. V případě, že lokomotiva před sebou tlačí vagóny, tato kontrola se neprovádí a bezpečnost závisí na operátorovi. Případě, že probíhá operátorem řízený posun vlaku, bezpečnost závisí na operátorovi.

Automatické zastavení lokomotivy je možné provést několika způsoby. První způsobem je zastavení lokomotivy je rozsvícení červeného světla na příslušném semaforu v nádraží. Druhým způsobem je chybně přehozená výhybka. V tomto případě, příjde příkaz k zastavení od kontroly návaznosti úseků. Třetím způsobem je zastavení lokomotivy na základě příjezdu lokomotivy do konečného traťového úseku vlakové cesty. Předposledním způsobem je zastavení lokomotivy od kontroly zamknutí úseků a poslední možností je příkaz od kontroly volnosti vlakové cesty. První čtyři uvedené způsoby zastavení lokomotivy uvedou lokomotivu do stavu STOP. Opětovné povo-

lení jízdy musí potom provést operátor. Při posledním způsobu se lokomotivě sníží rychlost na nulu a to do doby než se hlídaný traťový úsek stane volným. Jakmile se úsek stane volným, lokomotiva se rozjede rychlostí jakou jela před zastavením.

Na konci první části se aktualizují data o rychlosti a směru pohybu lokomotivy, která se budou posílat lokomotivě.

Druhá část funkčního bloku FB1 je principiálně stejná jako první, která ovládá skutečnou lokomotivu. Jediný zásadní rozdíl oproti řízení skutečné lokomotivy spočívá v určení momentu, kdy se lokomotiva přesune z jednoho traťového úseku do druhého a s ním související potřebné nastavení registrů. V případě skutečné lokomotivy stačí sledovat stav následujícího traťového úseku, do kterého se lokomotiva má přesunout tj. stačí hlídat, kdy se následující traťový úsek stane obsazený. V případě virtuální lokomotivy toto nelze a proto bylo nutné virtuální lokomotivě implementovat jednoduché vlastní počítadlo odpočítávající dobu, za kterou lokomotiva daný úsek projede. Sledování pohybu lokomotivy po virtuálním kolejišti je stejné jako u skutečné lokomotivy. Přejezd z jednoho traťového úseku do druhého se ale liší. Lokomotiva sleduje, kdy se stane traťový úsek, ve kterém se lokomotiva nachází, neaktivním. V momentě, kdy tento okamžik nastane, se přesune adresa z paměti pro následující traťový úsek do paměti pro aktuální traťový úsek. Po tomto přesunu se z tabulky traťových úseku načte údaj o rychlosti, kterou má lokomotiva daný úsek projet a údaj o délce úseku vynásobený hodnotou $0xFFFFh$. Údaj o délce je potom přesunut do počítadla používaný k určení délce doby, kterou lokomotiva potřebuje k přejezdu tohoto traťového úseku. Dále se zapíše příznak obsazenosti traťového úseku do tabulky v datovém bloku DB4. Potom jsou stejný způsobem jako u skutečné lokomotivy načteny adresy dvou po sobě následujících úseků. K určení se délky doby přejezdu daného úseku se používá 32-bitový registr Timer, do kterého se načte hodnota představující délku traťového úseku. Od tohoto registru se odečítá hodnota uložená v registru Virtual_Autodata. Hodnota registru Virtual_Autodata záleží na situaci, ve které se lokomotiva nachází. V případě, že je řízení přepnuto na automatické řízení, je hodnota registru Virtual_Autodata načítaná z tabulky traťových úseků. V případě, že má lokomotiva zastavit svůj pohyb je do registru Virtual_Autodata načtena hodnota 0.

Skutečné a virtuální kolejiště se stýkají ve dvou pevně daných bodech. První bod tvoří traťové úseky, které jsou v tabulce traťových úseků na adresách 260 a 500. Druhým bodem jsou traťové úseky začínající na adrese 120 a 840. Traťový úsek na adrese 260 bude zkráceně označován úsekem 260 a podobně budou označovány i ostatní traťové úseky. Přechod mezi virtuální a reálnou částí je rozdělěn na přechod z reálné části do virtuální a na přechod z virtuální části do reálné. Přechod z reálné části do virtuální se může uskutečnit ve dvou případech. První případ nastane v

momentě, kdy lokomotiva vjede z úseku 270 do úseku 260. Druhý případ nastane, když lokomotiva vjede z úseku 110 do úseku 120. Pokud nastane první případ, zjistí se obsazenost odstavného nádraží "Editta". Pokud jsou všechny koleje volné, nastaví se výhybky tak, že lokomotiva přijede na třetí kolej. V případě, že třetí kolej není volná, přestaví se výhybky na první kolej. Pokud je obsazena i první kolej výhybky se přestaví na druhou kolej. V případě, že není volná kolej, lokomotiva se zastaví a bude stát na vstupu do nádraží. Po nastavení výhybek skutečná lokomotiva automaticky odjede na odstavné nádraží "Editta". Při přechodu z reálné části do virtuální je nutné předat řízení lokomotivy automatu, který provede přejezd z reálné části do virtuální. Druhý případ je stejný jako první, rozdíl je pouze v prioritě odstavných kolejí na nádraží "Editta"'. Jako první se testuje na obsazenost první koleje, potom třetí koleje a na konec koleje druhé. Signál k přechodu z virtuální části do reálné se objeví v momentě, kdy se stane aktivním traťový úsek 840 nebo 500. Při přechodu z virtuální části do reálné se nejprve zjistí, na které koleji lokomotiva stojí a zda výhybky nejsou uzamčeny. Pokud výhybky nejsou uzamčeny, přestaví se na kolej, na které stojí lokomotiva. Po přestavění výhybek se lokomotiva rozjede. Vjezdem do traťového úseku 120 resp. 260 se ukončí aktivita traťového úseku 840 resp. 500. Přechod z virtuální části do reálné musí provést automat lokomotivy.

5.2.2.2 Funkční blok FB2, FB4, FB6 a FB8 - Vlakové cesty

Funkční bloky FB2 a FB4 staví jednotlivé vlakové cesty mezi sousedními nádražími. Stavba vlakové cesty probíhá následujícím způsobem. Nejprve zkontroluje volnost a zamknutí traťových úseků mezi sousedními nádražími. Pokud je některý z traťových úseků obsazen, ukončí se stavba vlakové cesty. V opačném případě se provede se kontrola zamknutí výhybek. Pokud jsou výhybky volné, zjistí se, na následujícím nádraží, na který má přijet lokomotiva, je volná kolej. Pokud je na nádraží alespoň jedna volná nezamknutá kolej, provede se stavba vlakové cesty. Zamkne se jedna volná kolej na příjezdovém nádraží. Systém výběru volné koleje pro zamknutí je proveden tak, že se jako první vybere kolej, která se přímá. Jestliže přímá kolej není volná, vybere se nejbližší volná kolej. Po zamknutí volné koleje se provede přestavení výhybky na zamknutou kolej a zamkne se i výhybka. Dále se provede zamknutí celého úseku mezi nádražími. Následně na to, se zjistí výchozí kolej nádraží, ze kterého lokomotiva bude odjíždět a přestaví se výhybky na ni a zamknou se. Nakonec se nastaví světla na všech semaforech na dané vlakové cestě. Po nastavení semaforů může lokomotiva dostat povel k rozjezdu. Tento povel dává operátor. Funkční bloky FB2 a FB6 staví vlakové cesty pro lokomotivu BR221 a funkční bloky FB4 a FB8 staví vlakové cesty pro lokomotivu V180.

5.2.3 Funkce FC

5.2.3.1 Funkce FC100 - zpracování dat přijatých po RS232

Funkce FC100 provádí zpracování dat přijatých po sériovém kanálu v modulu WAGO 750-650. Z tohoto modulu jsou data načtena funkcí SFC14 a uložena do datového bloku DB1. Celková velikost dat je 36B. Tyto data obsahují informaci o stavech traťových úseku. Model tyto data posílá jako jeden celek zakončený značkou signalizující konec dat. Touto značkou jsou tři byte s hodnotou $0xFFh$. Přijatá data se skládají z trojice bytů obsahující adresu, data a kontrolní součet. Zpracování dat se provádí následujícím způsobem. Načtou se tři byty, provede XOR součet prvních dvou bytů, tj. adresy a dat a výsledek se porovná se třetím bytem. Pokud se sobe rovnají, přijatá data jsou správná a mohou být podle nich modifikována data v tabulce traťových úseků. V případě, že se nerovnají, data se zahodí a pokračuje se dále ve zpracovávání.

5.2.3.2 Funkce FC101 - Ovládání výhybek

Funkce FC101 provádí dvě hlavní činnosti. První hlavní činností funkce FC101 je aktualizace dat v datovém bloku DATARS232 podle změny stavů výhybek reálného kolejiště. Obsah datového bloku se potom posílá po RS232 mikrokontroléru v modelu. Aktualizace dat v datovém bloku se provádí pouze pokud dojde ke změně stavu. V opačném případě aktualizace neprobíhá a nežádá se o poslání i těchto dat po RS232.

Druhou hlavní činností funkce je aktualizace dat v tabulce úseků. Každý úsek v tabulce traťových úseků má ve svém záznamu adresu svého předcházejícího souseda a následujícího souseda. Pokud je úsek poslední v jednom či druhém směru, tak místo adresy v příslušném směru je koncová značka $0xFFFFh$. Tohoto systému značení navazování úseku se využívá při přestavování výhybek. Úsek, který není výhybkou spojen s výstupním úsekem nádraží má v příslušném směru značku $0xFFFFh$. Pokud bude tento úsek správně spojen s výstupním úsekem bude mít v příslušném směru adresu úseku, který obsahuje výhybky.

5.2.3.3 Funkce FC102 - Zamykání a odemykání výhybek

Funkce FC102 provádí zamykání a odemykání výhybek podle stavu 8b registrů příznaků, který má každá vyhybka. Stavy jednotlivých bitů v těchto registrech aktualizují jednotlivé funkce a funkční bloky programu. Pokud je alespoň jeden z bitů v registru ve stavu log. 1, potom funkce FC102 provede uzamknutí výhybky tj. nastaví příznak uzamčení v 8b registru 21 a 23 v datovém bloku DB3. Registr 21 a 23 je

pamět, kde jsou uloženy příznaky pro celé skupiny výhybek, které spolu na kolejišti sousedí. Následující tabulka ukazuje sdružení semaforů a výhybek do skupin.

5.2.3.4 Funkce FC106 - Inicilizace tabulky traťových úseků

Funkce FC106 provádí inicializaci modelu nebo při reinicializaci modelu nastavení tabulky traťových úseků do základního stavu.

5.2.3.5 Funkce FC107 - Ovládání semaforů

Funkce FC107 ovládá všechny semafory jak ve virtuálním a tak v reálným kolejišti. V případě, že byly změněny stavy semaforů, které jsou na skutečném kolejišti, funkce FC107 provede aktualizaci dat semaforů v datovém bloku DataRS232, jehož obsah se potom odešle po RS232. V modelu železnice se používá odjezdový semafor a vjezdový semafor.

Světlo	Popis významu
Zelená	Volno
Červená	Stůj a zákaz posunu
Bíla	Posun dovolen
Žlutá	Maximální povolená rychlost 40km/h

Tabulka 5.7: Odjezdový semafor

Světlo	Popis významu
Horní žlutá	na následujícím návěstí může být změna stavu
Zelená	Volno
Červená	Stůj
Dolní žlutá	Maximální povolená rychlost 40km/h

Tabulka 5.8: Vjezdovy semafor

Odjezdové semafory (tab. 5.7) jsou sdružené pro všechny koleje na nádraží, ze kterých povolují odjezd z nádraží na širou trať. Základním stavem na všech semaforech je červená. Pokud na semaforu svítí pouze červené světlo, je zakázán odjezd vlaku z nádraží daným směrem a zakázán posun v dané části nádraží. Pokud svítí zelené světlo na semaforu, vlak může opustit nádraží. V případě, že se zeleným světlem svítí i dolní žluté světlo, vlak může je maximální rychlostí 40km/h. Rozsvícené bílé světlo dovoluje posun v daném úseku nádraží.

Vjezdový semafor (tab. 5.8) dovoluje vlaku, který se nachází na širé trati přijet na nádraží. Pokud na semoforu svítí pouze červená barva, musí vlak zastavit před

semaforem a čekat až mu bude dovolen vjezd na nádraží. Pokud svítí pouze zelené světlo, vlak má volno a může přijet na nádraží. V případě, že na semaforu svítí zelené světlo spolu se žlutou, vlak přijíždějící na nádraží může přijet na nádraží, ale maximální vjezdová rychlost je 40km/h. Tato světelná kombinace se používá u situací, kdy vlak bude uhýbat z přímé koleje na postraní a výhybky nedovolují přejezd maximální povolenou rychlostí v daném traťovém úseku. Rozsvícené horní žluté světlo značí, že na následujícím návěstí bude změna stavu.

5.2.4 Hlavní program

Hlavní program se skládá z několika jednoduchých částí, které zajišťují řízení celého modelu železnice. Ihned po spuštění se provede inicializace modulu sériového kanálu WAGO 785-650, potom se z inicializuje obsah paměti PLC a nakonec inicializace se provede prvotní přenos dat do modelu železnice, čímž se nastaví výhybky a semafory do základní polohy.

Po inicializaci se začne komunikovat prostřednictvím modulu Wago 750-650 s modelelem po RS232. Komunikace mezi modelem železnice a PLC probíhá tak, že každých cca 200ms PLC pošle paket s daty a s požadavkem na zpětné poslaní nových dat ze snímačů. Posílaná datová struktura paketu se nachazí v datovém bloku DB2 a je znázorněna tabulkou tab. 5.9.

První tři byte jsou požadavek pro poslání nových dat z snímačů. Dalších šest byte představuje dva příkazy pro lokomtivy BR221 a V180. Následujících 12 byte jsou data, které v sobě nesou informace o poloze výhybek v kolejišti. Posledních 36 byte jsou informace o stavu světel na semaforech. Většinou se neposílá celá datová struktura, ale jenom část. Velikost posílaného paketu je ovlivněna změnami dat, které se týkají výhybek a semaforů. Pokud došlo během scancyklu PLC ke změně polohy výhybky, pošlou se i všechna data určující polohu výhybek. V případě, že došlo ke změně stavu světel na semaforech je poslán paket o maximální velikosti tj. jsou poslána všechna data.

Hlavní program v PLC probíhá následovně. Nejprve se načtou data. Načtení dat trvá 8 až 9 scancyklů PLC podle toho jak jsou data načítána z modulu WAGO 750-650. Během načítaní dat z modulu WAGO 750-650 je celé řízení modelu železnice zastaveno tj. nevolají se žádné jiné funkce než SFC14 a SFC15. Funkce SFC14 načítá data z modulu WAGO 750-650 a funkce SFC15 do něj zapisuje. V případě, že je načteno 36B dat, otestuje se koncová značka přenosu dat. Koncovou značku dat jsou tři po sobě jdoucí byte s hodnotou $0xFFh$. V případě úspěšného testu se zpracují přijatá data pomocí funkce FC100. Potom se podle stavu vstupních proměnných zavolají již zmiňované funkční bloky a funkce. K ovládání lokomotivy BR221 se

Adresa zařízení	Data	CRC	Popis
0x10h	DDDDDDDDD	CCCCCCCC	rychlost lokomotivy BR221
0x11h	DDDDDDDDD	CCCCCCCC	rychlost lokomotivy V180
0x20h	0000DDDDD	CCCCCCCC	výhybky 1 − 4
0x21h	0000DDDDD	CCCCCCCC	výhybky 4 − 8
0x22h	0000DDDDD	CCCCCCCC	výhybky 9 − 12
0x23h	0000DDDDD	CCCCCCCC	výhybky 13 − 15
0x40h	0000DDDDD	CCCCCCCC	semafor č.6
0x41h	0000DDDDD	CCCCCCCC	semafor č.8
0x42h	0000DDDDD	CCCCCCCC	semafor č.7
0x43h	0000DDDDD	CCCCCCCC	semafor č.11
0x44h	0000DDDDD	CCCCCCCC	semafor č.3
0x45h	0000DDDDD	CCCCCCCC	semafor č.4
0x46h	0000DDDDD	CCCCCCCC	semafor č.12
0x47h	0000DDDDD	CCCCCCCC	semafor č.5
0x48h	0000DDDDD	CCCCCCCC	semafor č.1
0x49h	0000DDDDD	CCCCCCCC	semafor č.2
0x4Ah	0000DDDDD	CCCCCCCC	semafor č.10
0x4Bh	0000DDDDD	CCCCCCCC	semafor č.19

Tabulka 5.9: Data určená k přenosu do modelu

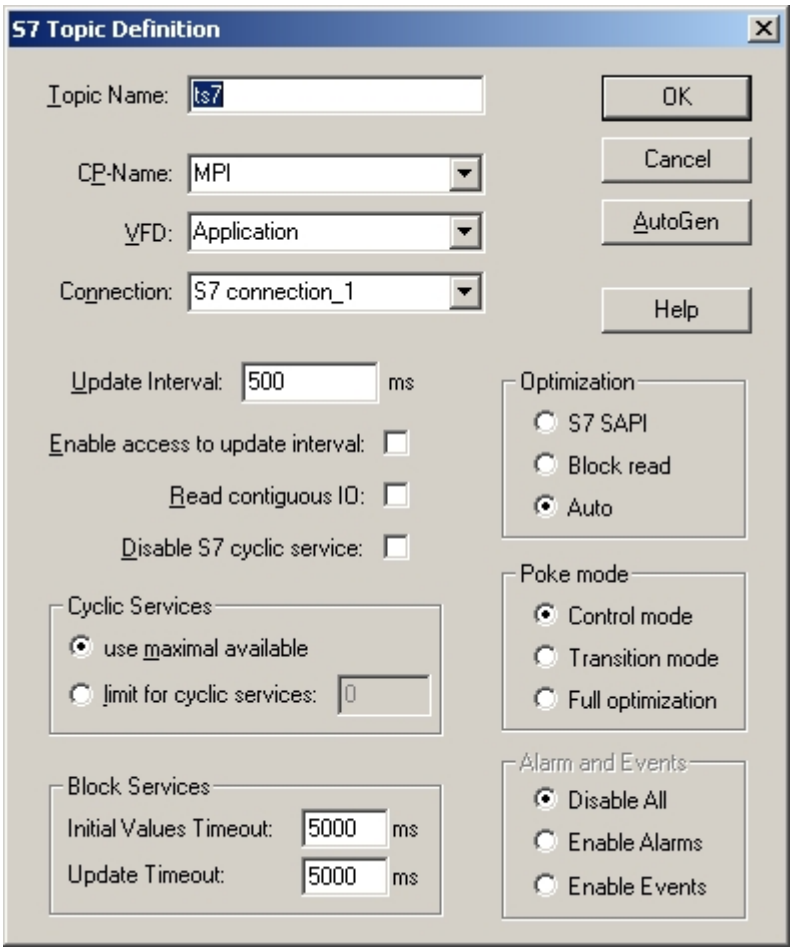
v hlavním programu volá funkční blok FB1 s datovým blokem DB5, k ovládání lokomotivy V180 se volá funkční blok FB1 s datový blokem DB10. Následně na to se provede aktualizace stavu výhybek a semaforů zavoláním funkcí FC101 a FC107. Když je lokomotiva BR221 řízena pomocí předprogramované vlakové cesty, zavolá se ještě před funkčním blokem FB1 funkční blok FB2 nebo FB6. U lokomotivy V180 je to podobně, jediný rozdíl je v tom, že se zavolají funkční blok FB4 nebo FB8. Posílání dat do modelu železnice po RS232 se začíná provádět vždy každý 30. scancyklus. Během přenosu se opět volají pouze funkce SFC14 a SFC15 pomocí kterých se odešlou data z datového bloku DB2. Doba odesílání všech dat záleží na velikosti posílaného paketu. Délka paketu je ovlivněna změnami poloh výhybek a změnami stavů světel semaforů. Maximální délka paketu je 66B.

Kapitola 6

Visualizace

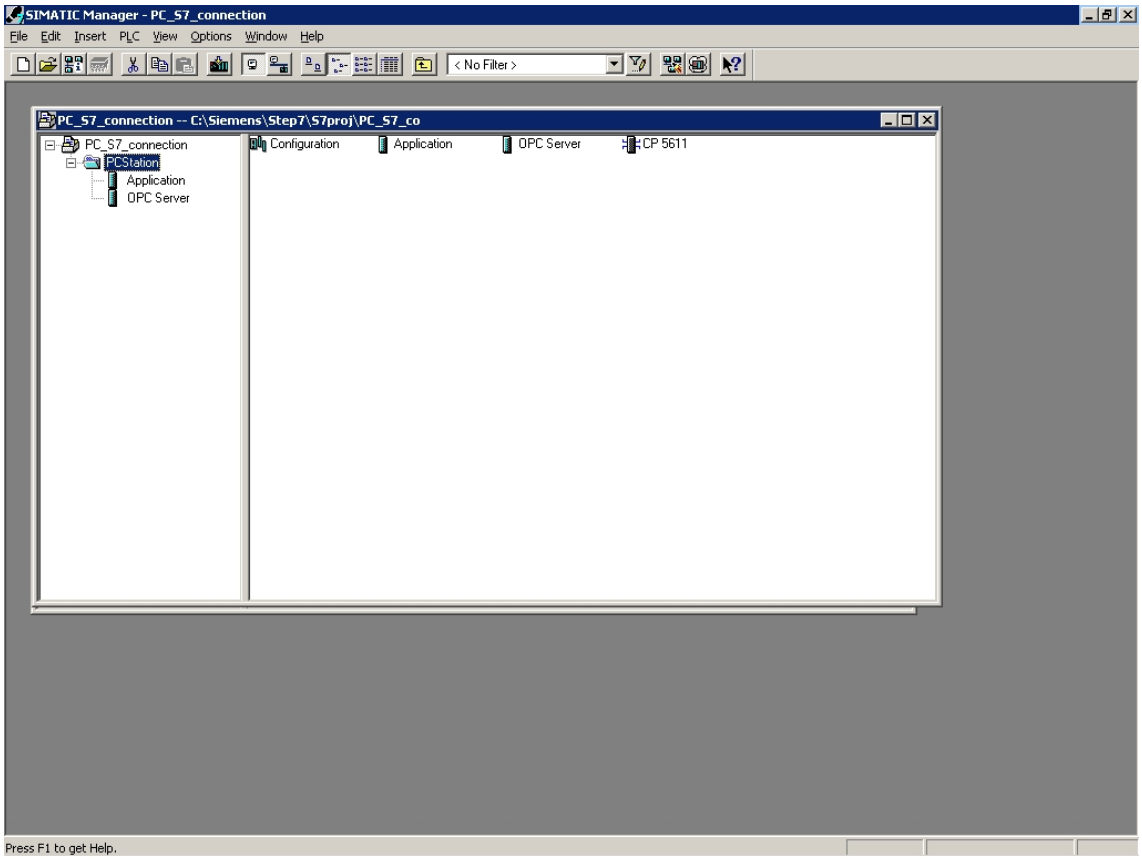
6.1 Konfigurace I/O serveru S7

Ke komunikaci mezi PLC automatem a aplikací visualizačního prostředí InTouch je třeba nakonfigurovat DDE server S7. Po spuštení DDE serveru se menu zvolí položka Configure / Topic definition / New.. . Po zvolení položky "New" se objeví následující okno, ktere se vyplní, přesně tak jak je uvedeno na obrázku.



Obrázek 6.1: Nastavení DDE serveru

Dále je nutné v project manageru Simaticu nahrát projekt PC_S7-connection do karty v počítači. Projekt už byl vytvořen v rámci výuky a je nahrán na CD.



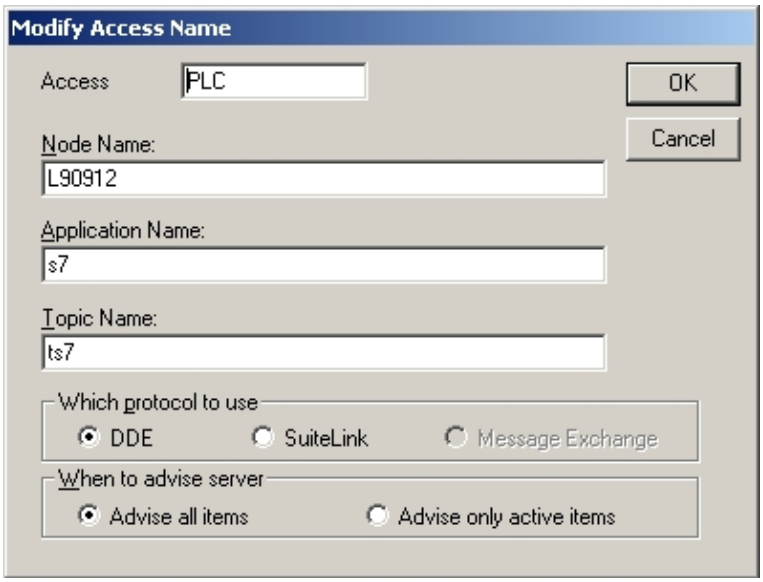
Obrázek 6.2: Nastavení karty v počítači

Pro další práci je nutné si zapamatovat si obsah políčka Topic Name, které bude potřeba ještě při konfiguraci visualizačního prostředí InTouch.

6.2 Visualizace v prostředí InTouch

6.2.1 Základní nastavení v prostředí InTouch

Při založení projektu ve visualizačním prostředí je nutné nakonfigurovat přístupový bod do DDE serveru. Konfigurace se provede podle obrázku obr. 6.3.



Obrázek 6.3: Konfigurace přístupového bodu

6.2.2 Visualizace

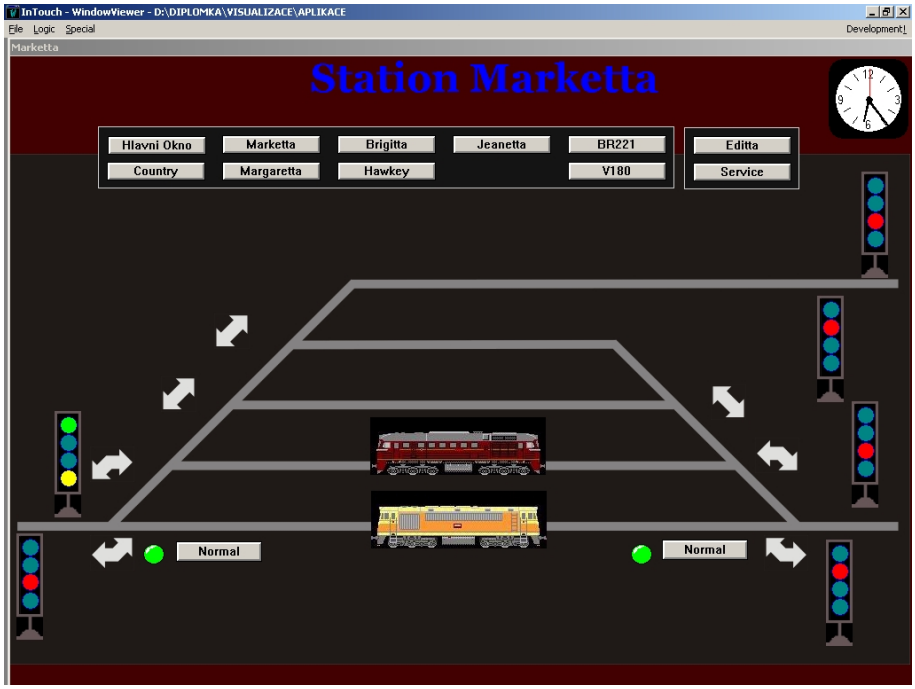
Visualizace slouží jako komunikační rozhraní mezi operátorem a řídicím systémem. Pro visualizaci modelového kolejiště byla zvolena jako podkladová barva černá, stejně jako je použita i ve skutečnosti např. na Hlavním nádraží v Praze. Pokud je kolej volná, kolej je znázorněna šedivou barvou a v případě, že kolej je obsazena je tato barva světle šedá. Dále při zobrazování obsazenosti koleje na nádraží je ještě zobrazen symbol lokomotivy, která je na dané koleji. Zobrazení semaforu je stejné jako ve skutečnosti.

Po spuštění se objeví hlavní okna visualizace, které funguje jako rozcestník. Odtud se operátor dostane dostane na všechny okna ve visualizaci. Systém obrazovek je vytvořen tak, že všechny okna nádraží jsou mezi sebou propojeny. Výhodou tohoto systému je fakt, že na přepnutí z jednoho nádraží na druhé postačí pouze jedno kliknutí myši. Navíc z každého okna se nechá otevřít okno s ovládáním jak lokomotivy BR221 tak i lokomotivy V180. Dále je možné z každého nádraží dostat na okno, kde je zobrazen souhrnný pohled na celé kolejiště.

6.2.2.1 Visualizace nádraží

Všechna nádraží jsou zobrazována stejné stylu. Pod názvem stanice se nachází panel s tlačítky, pomocí kterého se operátor lehce přepne na požadovanou obrazovku. Vedle tohoto panelu je ješte jeden malý panel, který přepíná na dvě servisní obrazovky. První servisní obrazovka zobrazuje odstavné nádraží "Editta", kam se odstavují lokomotivy při vjezdu lokomotivy do virtuální části kolejiště. Druhá obrazovka slouží k zobrazení stavu pouze skutečného modelu železnice. Ovládaní semaforů se provádí

kliknutím na světlo semaforu, které se má rozsvítit. Změnou stavu jednoho světla se mohou změnit i stavy ostatních semaforů. Výhybky se ovládají podobně jako semafory. Změna polohy výhybky se provede kliknutím na příslušnou výhybku. Pokud je výhybka zamknuta, nezobrazí se obdélník okolo ovládacího prvku výhybky. V případě, že operátor chce přesunout výhybku přestože je zamknuta, musí výhybku nejprve odemknout pomocí tlačítka "Emergency"' a potom může provést změnu. V momentě, kdy do vjezdového traťového úseku vjede lokomotiva, zesvětlá daný traťový úsek. Po vjezdu lokomotivy na jednu z kolejí nádraží, příslušná kolej se zesvětlá a uprostřed koleje se objeví obrázek lokomotivy, která vjela do daného úseku. Po opuštení koleje obrázek zmizí.



Obrázek 6.4: Visualizace nádraží Marketta

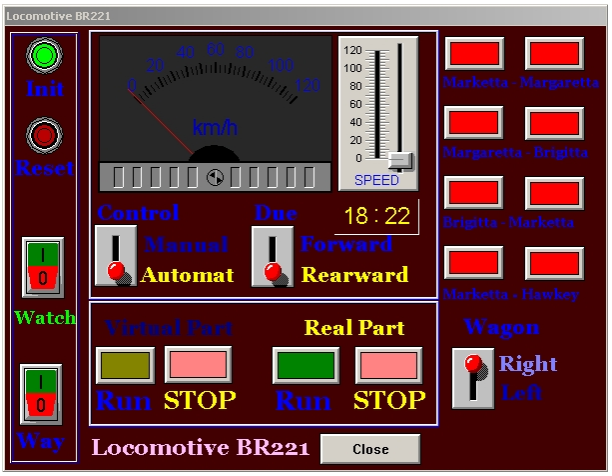
6.2.2.2 Visualizace celého modelu kolejiště

Zobrazení celého modelu kolejiště se nese ve stejném stylu jako jsou zobrazeny nádraží. Koleje jsou naznačeny tmavě šedou čárou. Obsazený traťový úsek je zobrazen světle šedou čárou. U každého nádraží je tlačítko, po jehož stiknutí se operátor dostane na obrazovku daného nádraží. Dále se na obrazovce nachází stejný panel jako u obrazovek nádraží, který slouží k rychlému přepínání mezi jednotlivými obrazovkami.

Obrázek 6.5: Visualizace celého modelu

6.2.2.3 Ovládaní lokomotivy BR221 a V180

Každá lokomotivy má svoje ovládací okno, které je rozděleno na dvě části. První část se nachází v levé polovině okna a slouží k řízení pohybu lokomotivy. Druhá část,která se nachází v pravé polovině je určená ke stavbě předprogramovaných vlakových cest.



Obrázek 6.6: Ovládání lokomotivy BR221

Vzhledem k tomu, že lokomotiva se pohybuje jak reálné části kolejiště tak i ve virtuální části, tak bylo nutné použít dvě tlačítka STOP. Jedno tlačítko je určeno pro lokomotivu ve virtuální části, druhé tlačítko STOP je určeno pro lokomotivu na skutečném kolejišti. Ostatní ovládací prvky tj. tachometr, slider, přepínání směru jízdy se při přejezdu z virtuální části do reálné a opačně přepínají na lokomotivu,

která je zrovna aktivní tj. pokud je lokomotiva ve virtuální části tak se ovládací prvky vztahuji k virtuální lokomotivě a pokud je lokomotiva ve reálné části kolejiště pak se ovládací prvky vztahují k skutečné lokomotivě.

Kapitola 7

Závěr

Řešení diplomové práce bylo rozděleno na tří na sebe navazující části. První část řešení se zabývala stavbou modelu kolejiště. Druhá část se věnovala řídicímu algoritmu v PLC Simatic S7-315 2DP a třetí část řešení diplomové práce se zabývala vytvořením visualizace jak reálného modelu kolejiště tak i virtuálního.

Stavba modelu železnice probíhala po ucelených etapách. Nejprve bylo postaveno kolejište na desce stolu o šířce 1,2 m a délce 2,4 m. Ke stavbě kolejište bylo použito "modelové kolejivo" velikosti TT tj. model železnice je zmenšen oproti skutečnosti v poměru 1:120. K přestavování výhybek byly použity přestavníky využívající k přesunu výhybky posuvný mostek posunovaný motorkem . Odběr jednoho motorku při normální provozu je průměrně 300 mA. Při přestavování všech 15 výhybek se průměrný celkový odběr proudu rovná hodnotě 4,5 A. Dále bylo v modelu použito 12 čtyřsvětelných semaforů. Každé nádraží má na svém vstupu jeden vjezdový semafor a jeden odjezdový semafor. Kvůli sledování pohybu lokomotivy po kolejišti je celé kolejiště rozřezáno na traťové úseky, které jsou napájeny z DCC zesilovače vždy přes proudový snímač. K řízení pohybu lokomotiv byl použit systém Digital Command Control, který lokomotivám prostřednictvím kolejí posílá pakety s příkazy. V každé lokomotivě je dekodér těchto paketů, který zajistí splnění přijatých příkazů. Vzhledem k tomu, že PLC bylo příliš pomalé na generování těchto příkazů, byl mezi model a PLC vložen elektronický systém. Tento elektronický systém funguje jako řídicí elektronika modelu a komunikační rozhraní mezi modelem a nadřízeným systémem. Komunikace mezi nadřízeným systémem a modelem probíhá po sériovém kanálu RS232 rychlostí 9600 $Baud/s$. Řídicí elektronika modelu je schopna od nadřízeného systému přijímat příkazy pro obě lokomotivy, všechny semafory a výhybky. Nadřazenému systému je naopak schopna posílat informace o poloze výhybek a stavech jednotlivých traťových úseků. V případě, že komunikace mezi nadřazeným systém neexistuje tj. nepříjde během 1s žádný požadavek na poslání dat, řídicí elektronika odpojí od koleje napájecí napětí. V případě, že komunikace

mezi modelem a nadřízeným systémem je příliš intenzivní, mikrokontrolér přestane zvládat generování DCC signálu podle normy NMRA 9.2.1 a dojde k zastavení pohybu lokomotiv. Ke komunikaci mezi modelem a okolním světem se používá přesně definovaný protokol. Řídicí elektronika je složena z centrální řídicí jednotky a periferií k ní připojených. Jádrem řídicí jednotky je mikrokontrolér PIC16F877 od firmy Microchip. Mezi periferie patří ovládaní výhybek a semaforů, proudové snímače a zesilovač DCC signálu. Výše zmiňované odpojení kolejí od napájení provádí DCC zesilovač na pokyn mikrokontroléru PIC16F84. Mikrokontrolér PIC16F84 sleduje vstup zesilovače a v případě, že není na vstupu pulsní signál, vydá povel k odpojení. Druhou etapou řešení diplomové práce bylo vytvoření ukázkového řídicího algoritmu. Počáteční snaha o vytvoření systému podobnému jako funguje na skutečné železnici musela být korigována, protože postavený model nebyl dostatečně rozlehlý. Řídicí program umožňuje lokomotivy řídit buď pomocí automatu nebo ručním řízením. Obě dvě lokomotivy mají svoji vlastní logiku, která se sleduje a hlídá pohyb lokomotivy po kolejišti a v případě hrozby kolize je schopna lokomotivu zastavit. Vnitřní logika lokomotivy dále pozná místo, kde lokomotiva přejíždí z reálné do virtuální části kolejiště a doveze skutečnou lokomotivu na odstavné nádraží, na kterém lokomotiva čeká do doby než se lokomotiva ve vizualizaci vrátí zpět do skutečného kolejiště. Dále řídicí program ovládá zhlaví nádraží.

Visualizace je vytvořena ve visualizačním prostředí InTouch. Visualizace se skládá z hlavní obrazovky, která funguje jako rozscetsník do dalších oken. Ve visualizaci je zobrazen celkový pohled na celé kolejiště. Dále jsou zobrazena všechna nádraží a řídicí panely lokomotiv BR221 a V180. Mezi všemi okny jednotlivých nádraží se nechá přepínat.

V budoucnu lze zapojit snímače koncové polohy přestavníků polohy výhybek na paralelní sběrnici a zajistit tak lepší zpětnou vazbu modelu. Mikrokontrolér PIC16F877 již v sobě obsahuje program, který je schopen přečíst data o poloze přestavníků a poslat je nadřízenému systému. Dále je možné model inovovat instalací ovladačů rozpojných kolejí, které byly při stavbě modelu zahrnuty do modelu železnice. Ovladače se nechají připojit na port D mikrokontroléru PIC16F877.

Model železnice bude v budoucnu sloužit jako model ve cvičeních předmětu Řídicí technika. Při řízení modelu si studenti budou moci důkladně procvičit programování PLC ve STEP7.

Literatura

[1] National Model Railroad association, DCC Standard S-9.1 [online]. Poslední revize 2002-09-05. http://www.dcc.info/standards_rps/s91.html.

[2] National Model Railroad association, DCC Standard S-9.2 [online]. Poslední revize 2002-09-05. http://www.dcc.info/standards_rps/s92.html.

[3] National Model Railroad association, DCC - Introduction [online]. Poslední revize 2001-02-14. http://www.nmra.org/beginner/dccbasic.html.

[4] Klub modulové železnice ZABABOV, Přestavník Fulgurex, [online]. http://www.zababov.cz/vs/Moduly/Stavby/Fulgurex.htm

[5] PIC16F87X Data Sheet , Philips Semiconductors, leden 2001. (PDF)

[6] PIC16F84 Data Sheet , Philips Semiconductors, leden 2001. (PDF)

[7] Součástky pro elektrotechniku. Praha: GM electronic, 2002.

Dodatek A

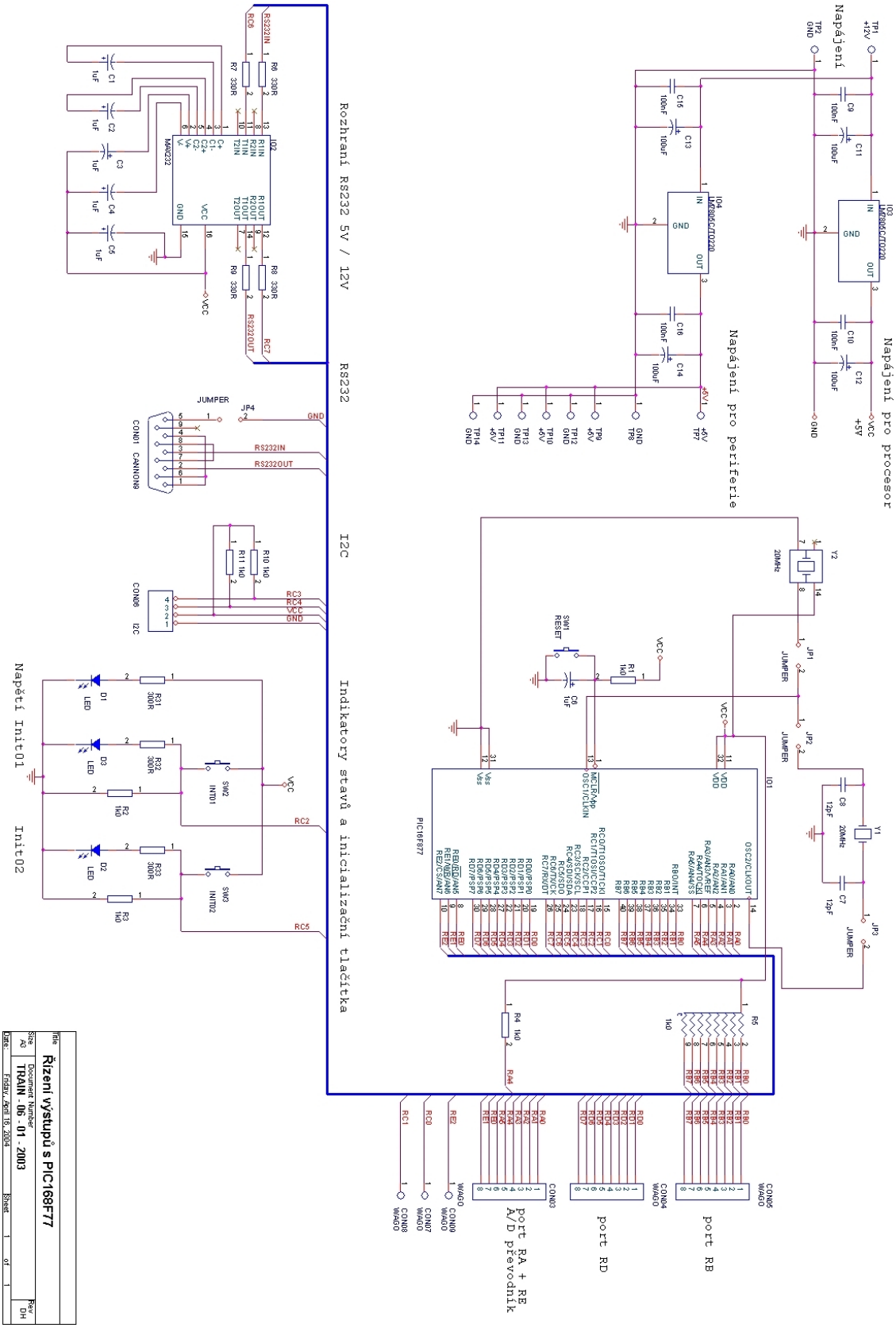
Použité zkratky

DCC - Digital Command Control
NMRA - National Model Railroad Association
I^2C - Inter Integrated Circuit
USART - Universal Synchronous Asynchronous Receiver Transmitter
EEPROM - Electrical Erase Programmable Read Only Memory
RAM - Random Access Memory
TTL - Tranzistor Tranzistor Logic
RISC - Reduced Instruction Set Computer
SPI - Serial Peripheral Interface

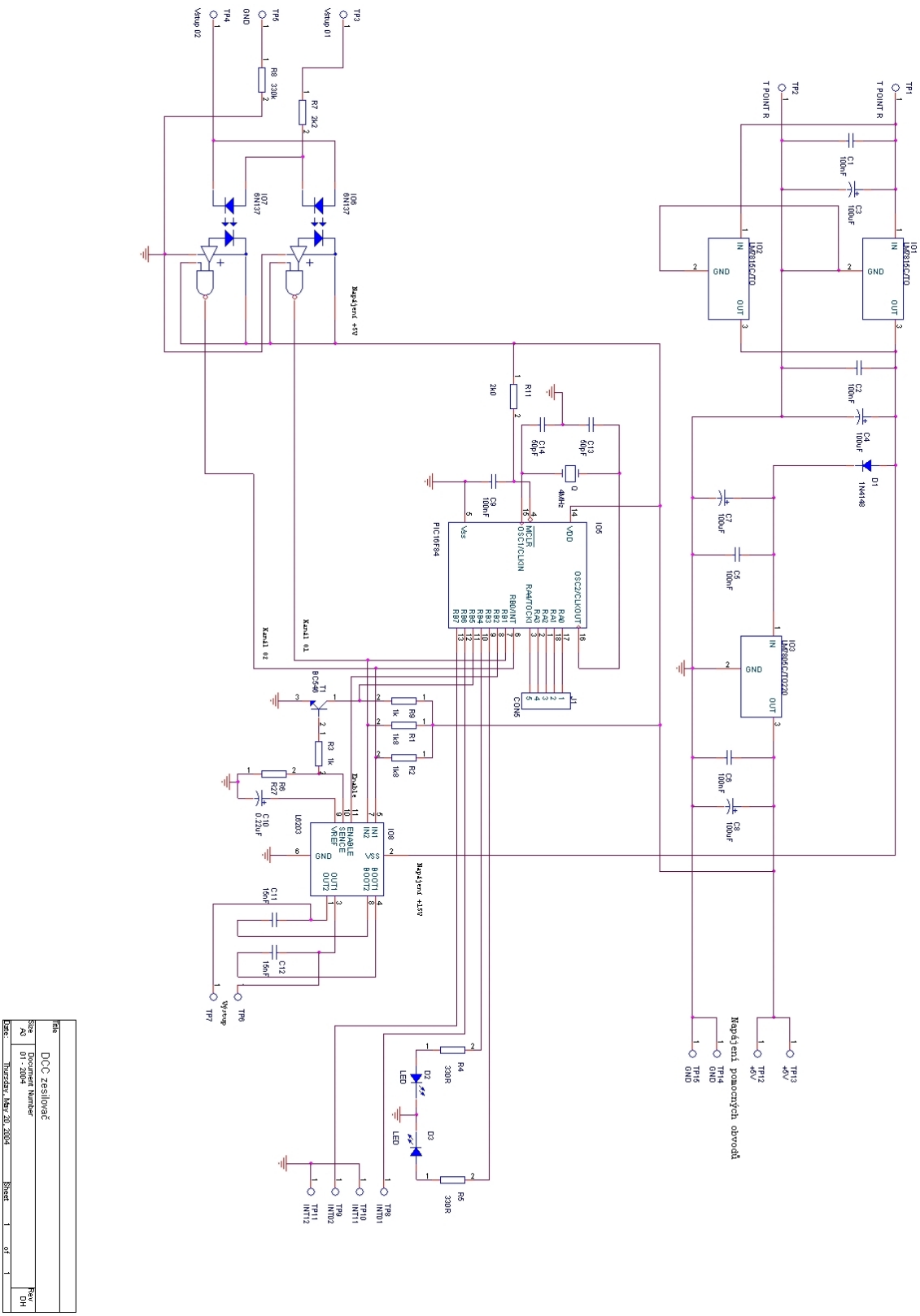
Dodatek B

Schémata

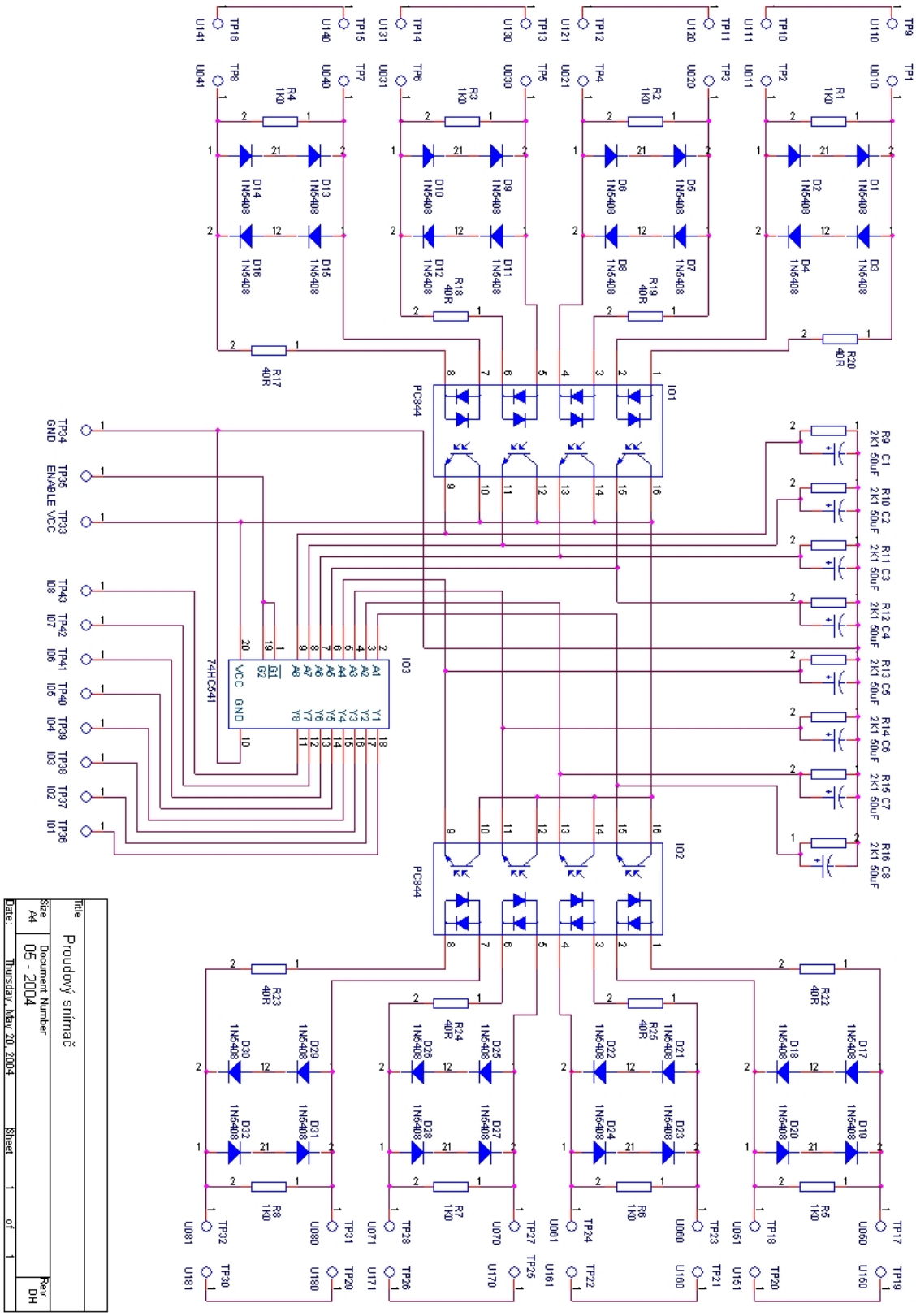
B.1 Schéma centrální řídicí jednotky



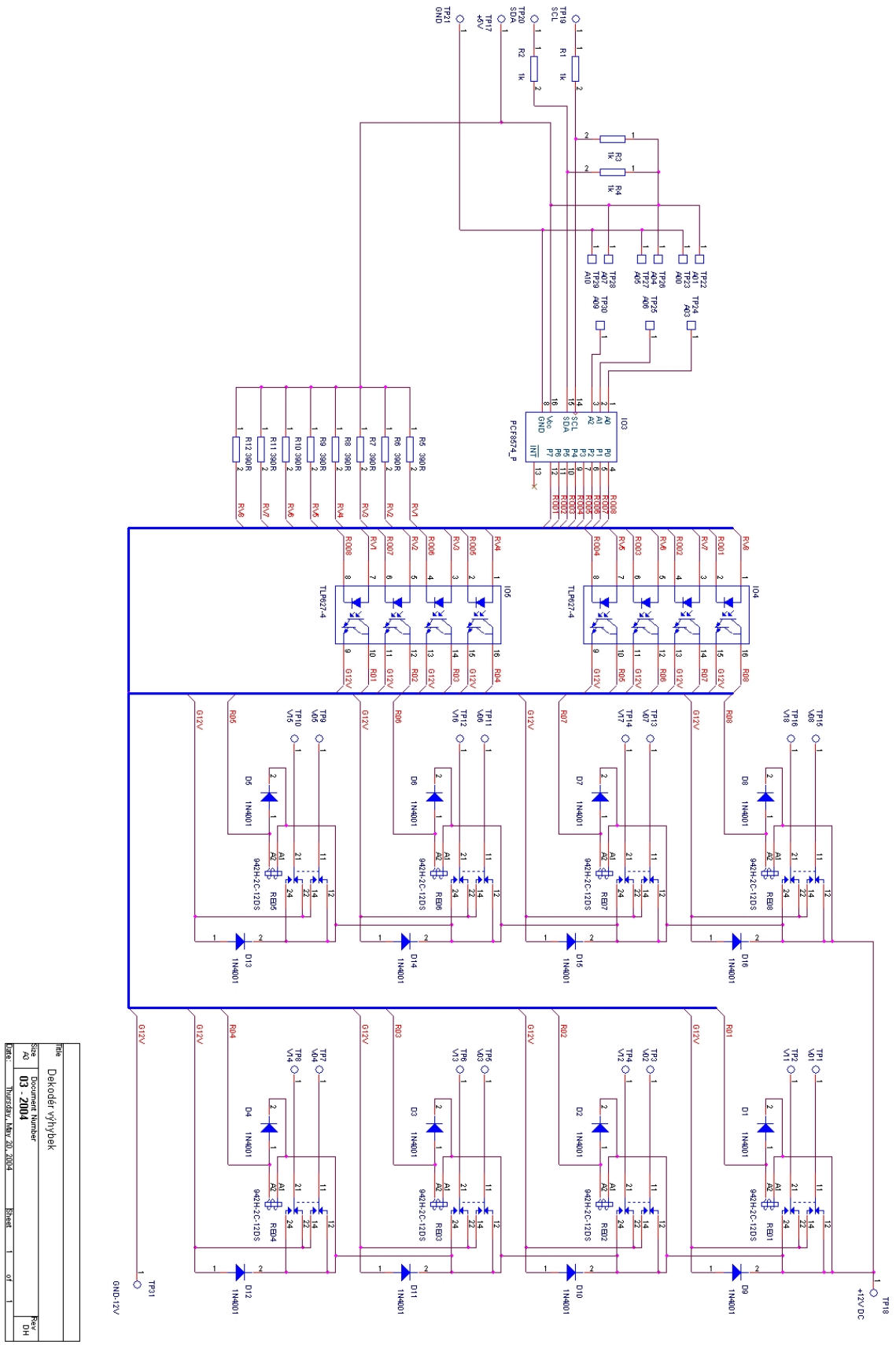
B.2 Schéma DCC zesilovače



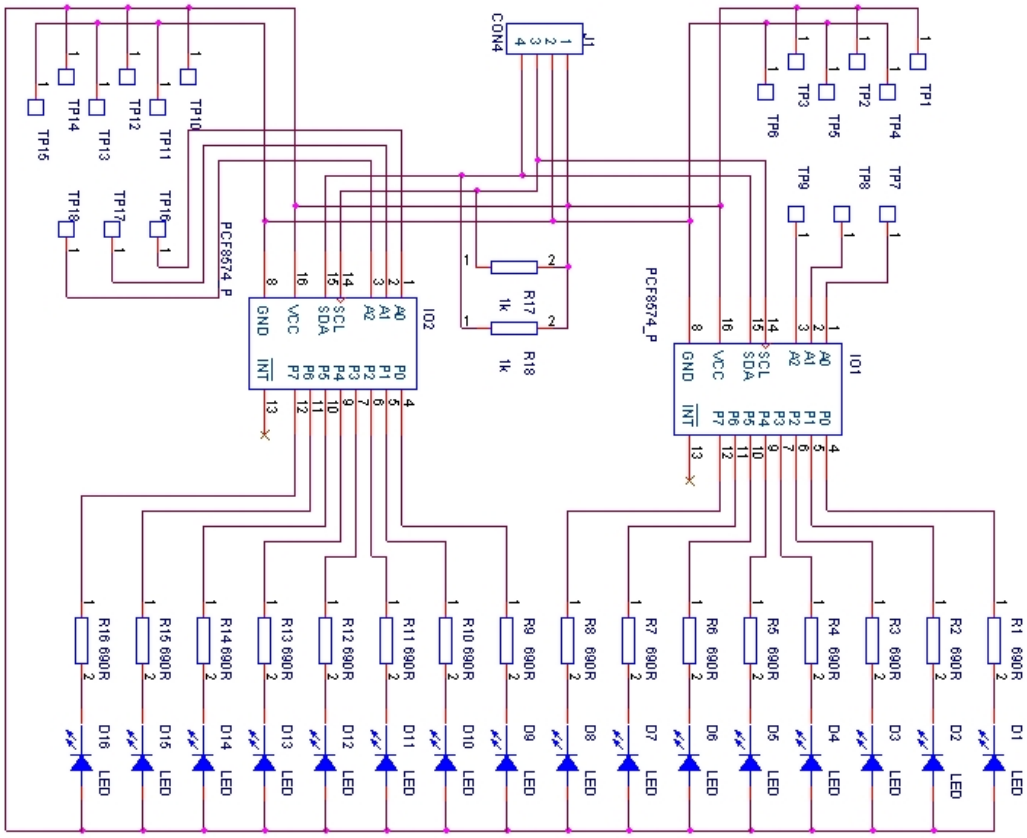
B.3 Schéma proudových snímačů

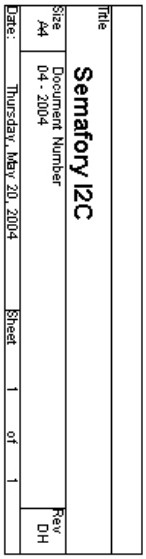


B.4 Schéma ovládání výhybek

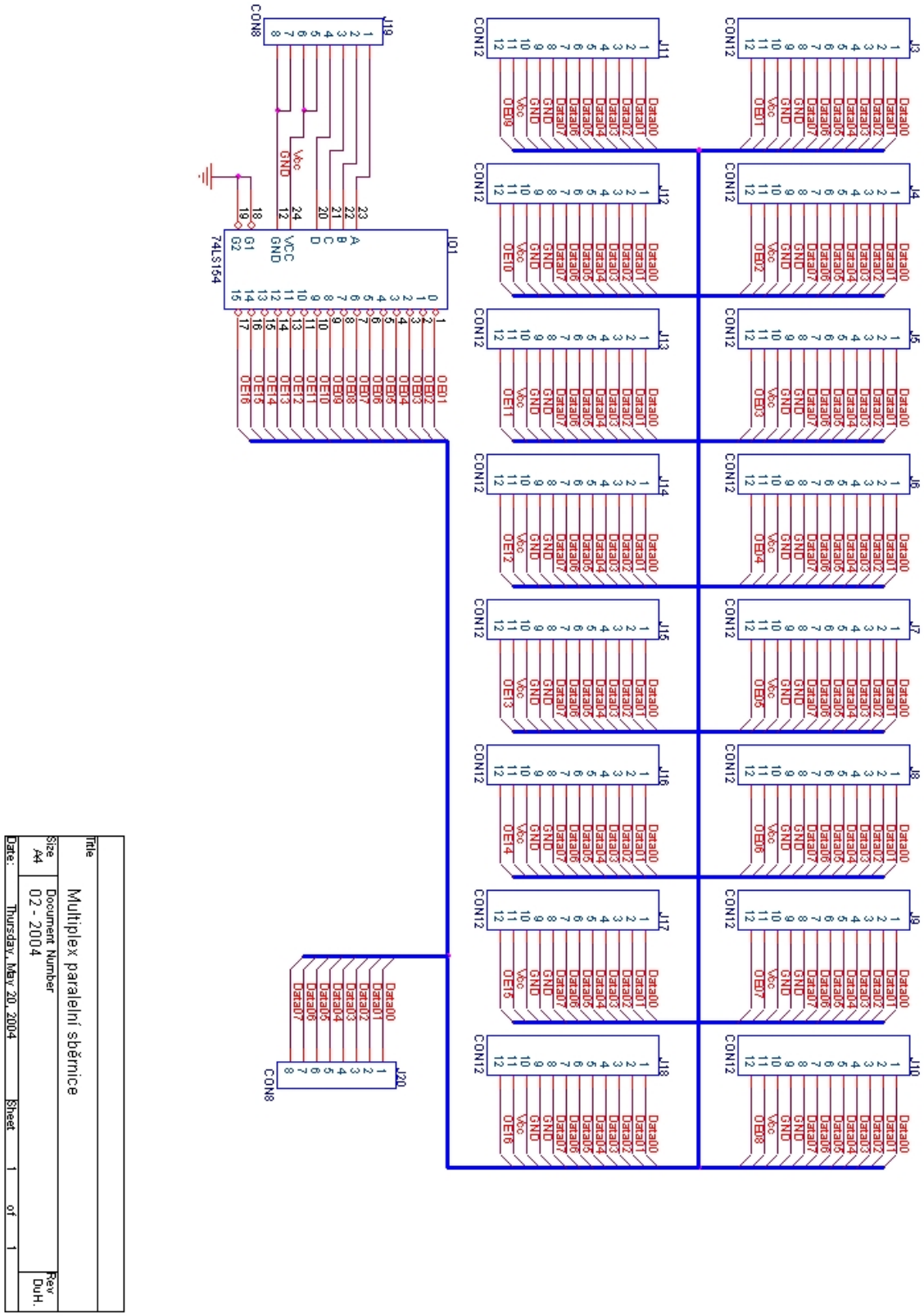


B.5 Schéma ovládání semaforů





B.6 Schéma multiplexoru paralelní sběrnice



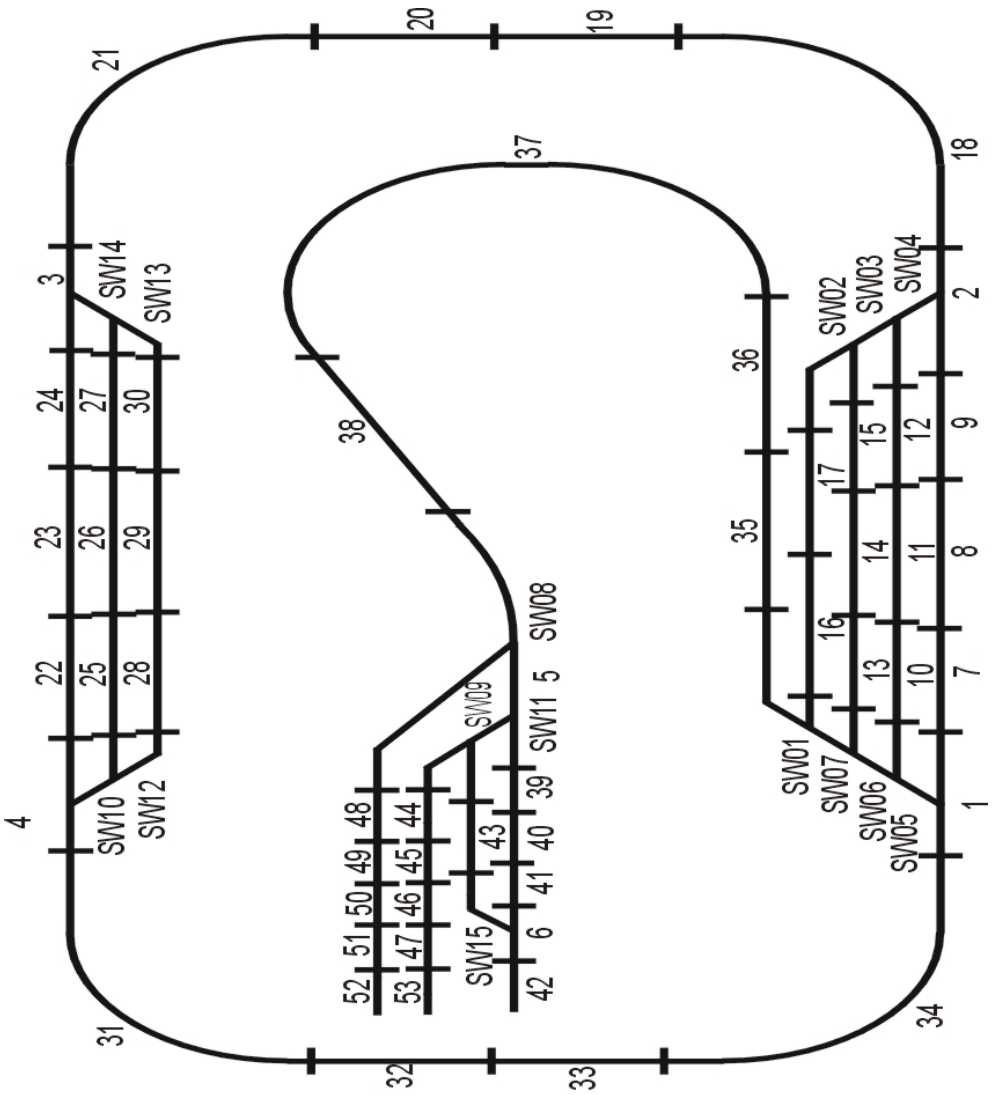
Dodatek C

Obsah CD

Přiložené CD obsahuje:
- katalogové listy některých použitých součástek
- vývojové prostředí MPLAB pro PIC16F877
- vývojové prostředí UIP pro PIC16F84
- program do PLC
- aplikace do visualizačního prostředí InTouch
- programy pro PIC16F877 a PIC16F84
- schémata a tištěné spoje řídicí elektroniky

Dodatek D

Nákres kolejiště



Obrázek D.1: Nákres kolejiště