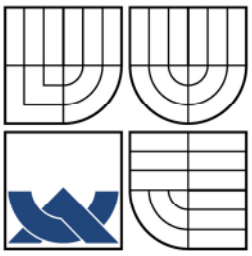
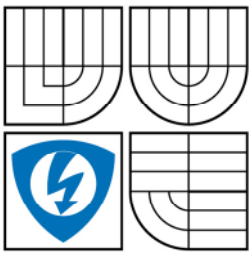


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A
KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘÍCÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

TESTER OSAZENÝCH DESEK PLOŠNÝCH SPOJŮ
PRINTED CIRCUIT BOARD TESTER

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE TOMÁŠ STACHERA
AUTHOR

VEDOUCÍ PRÁCE Ing.MILOSLAV ČEJKA , CSc.
SUPERVISOR

BRNO 2008

VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ
Fakulta elektrotechniky
a komunikačních technologií
Ústav automatizace a měřicí techniky

Bakalářská práce

bakalářský studijní obor
Automatizační a měřicí technika

Student: Stachera Tomáš *ID:* 77696
Ročník: 3 ***Akademický rok:*** 2007/08

NÁZEV TÉMATU:

Tester osazených desek plošných spojů

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte diagnostický systém pro testování osazených desek plošných spojů. Navrhněte SW i HW řešení multiplexeru stimulačních signálů, multiplexeru odezev i generování stimulů.
1/ Seznamte se s metodami testování osazených desek
2/ Navrhněte HW řešení multiplexeru pro tester
3/ Navrhněte SW pro řízení multiplexeru
4/ Ověřte systém a zhodnoťte jeho možnosti

DOPORUČENÁ LITERATURA:

Firemní literatura National Instruments,
Switching Handbook 5th edition. Firemní literatura KEITHLEY

Termín zadání: 1.2.2008 ***Termín odevzdání:*** 2.6.2008

Vedoucí projektu: Ing. Miloslav Čejka, CSc.

prof. Ing. Pavel Jura, CSc.
předseda oborové rady

UPOZORNĚNÍ:

Autor bakalářské práce nesmí při vytváření bakalářské práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Vysoké učení technické v Brně
Fakulta elektrotechniky a komunikačních technológii
Ústav automatizace a měříci techniky

Tester osazených desek plošních spojú
Bakalářská práce

Stúdijni obor : Automatizační a měříci technika
Student : Tomáš Stachera
Vedouci projektu : Ing. Miloslav Čejka, CSc.

Abstract:

Témov tejto bakalárskej práce bude navrhnuť tester osadených dosiek plošných spojov zo zameraním sa na HW a SW časť multiplexoru stimulačných signálov. V prvej časti sa budem zaoberať všeobecným popisom testovania dosiek plošných spojov a na konkkrétnom prípade z praxe ukážem zapojenie testovacieho systému zo zapojením multiplexoru. V druhej časti popíšem HW zapojenie celej dosky multiplexoru s konkrétnymi hodnotami súčiastok. V tretej časti budem popisovať SW pre mikrokontrolér multiplexoru a riadiaci počítač. V záverečnej časti zhodnotím možnosti multiplexoru a jeho parametre.

Kľúčove slová: multiplexor,tester, merací systém, doska plošných spojov,testovanie.

Brno University of Technology
The faculty of Electrical Engineering and Communication
Department of Control,Measurement and Instrumentation

Printed circuit board tester

Bachelor's thesis

Specialisation od study: Automation and Measurement
Student : Tomáš Stachera
Supervisor : Ing. Miloslav Čejka, CSc.

Abstract:
The themes of this work is make a projection of printed circuit tester with focus on HW ans SW part of signal multiplexor . In the first part i will do general description of printed circuit board testing and on concrete example i will show connection of test system with connection of multiplexor. In the second part i will describe HW part of multiplexor board with real values of componets. In the third part i will describe SW for microcontroller and computer. In the end part i will review possibility of multiplexor and his parameters.

Keywords: multiplexor, tester, measurement system,printed board,testing.

Bibliografická citace

Tomáš Stachera. *Tester osazených desek plošných spojú*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008. 65 strán.,1 příloha. Vedoucí práce : Ing. Miloslav Čejka, CSc.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Tester osazených desek plošných spojů jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářske práce dále prohlašuji, že v souvislosti s vytvořením této bakalářske práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne : 23.5. 2008 Podpis:

Poděkování

Děkuji tímto Ing. Miloslavovi Čejkovi, CSc. za cenné připomínky a rady při vypracování bakalářske práce.

V Brně dne : 23.5.2008 Podpis:

OBSAH

1. ÚVOD 9
2. UKÁŽKA KONKRÉTNYCH POŽIADAVIEK NA TESTOVANIE OSADENEJ DOSKY PLOŠNÝCH SPOJOV 11
3. BLOKOVÉ RIEŠENIE PRIPOJENIA MULTIPLEXOROVÝCH KARIET K TESTOVACIEMU SYSTÉMU 16
4.PREVODNÍK USB/RS485 18
5.REALIZÁCIA HW ČÁSTI MULTIPLEXOROVÉJ DOSKY 20
5.1 Blokové riešenie multiplexorovej karty 20
5.2 Podrobný popis zapojenia obvodu mikrokontroléru a prevodníku napäťových rozhraní RS 485/UART 22
5.3 Podrobný popis zapojenia obvodu spínacích tranzistorov a obvodu Relé. 25
5.4 Podrobný popis zapojenia Power down obvodu 28
6.NÁVRH KOMUNIKAČNÉHO PROTOKOLU 30
6.1. Všeobecný popis protokolu 30
6.2 Popis formátu správy 31
6.3.Popis dátovej časti 32
6.4 Popis príkazov 33
6.5 .Popis časti Checksum 35
6.6. Spracovávanie správ 37
7. PODROBNÝ POPIS SW PRE MIKROKONTROLER 40
7.1. Možnosti vývoja SW pre mikrokontroler 40
7.2. Naprogramovanie ovládania sériového rozhrania UART 41
7.3 Prijatie správy 42
7.4 Spracovanie správy 44
7.5 Funkcia IDN() 46
7.6 Funkcia OUT 47
7.7 Funkcia ADR 51
7.8 Funkcia STA 52
8. POPIS PROGRAMU PRE RIADIACI POČÍTAČ 54
8.1 Vývojový diagram pre vysielanie a spracovávanie správ mikrokontrolerom . 54

8.2. Komunikačný program pre riadiaci počítač 56
8.3 Popis programu 56
8.4 Popis jednotlivých funkcii programu 58
9. MOŽNOSTI MULTIPLEXOROVEJ KARTY 61
10 ZÁVER 64
11.ZOZNAM POUŽITEJ LITERATÚRI 65

ZOZNAM OBRÁZKOV

Obrázok 1 : Bloková schéma testovacieho systému.....8

Obrázok 2 : Bloková schéma diagnostického systému pre testovanie uvedenej dosky.....12

Obrázok 3 : Podrobná schéma zapojenia multiplexoru pri testu uvedenej dosky...13

Obrázok 4 : Bloková schéma pripojenia Multiplexorových kariet k PC.....16

Obrázok 5 : Schéma zapojenia prevodníku USB /RS485.....18

Obrázok 6: Blokové riešenie multiplexorovej karty.....19

Obrázok 7: Rozmiestnenie pinov mikrokontroléru ATMEGA 8515.....22

Obrázok 8: Schéma zapojenia obvodu mikrokontroleru a prevodníku napäťových rozhraní RS485/UART.....24

Obrázok 9: Schéma zapojenia obvodu spínacieho tranzistoru a relé (1 vetva).....25

Obrázok 10: Schéma zapojenia Power down modulu.....27

Obrázok 11: Prevádzka typu Master-Slave 29

Obrázok 12: Formát správy.....30

Obrázok 13: Dátova časť protokolu.....31

Obrázok 14: Formát správy s konkrétnymi hex hodnotami.....36

Obrázok 15: Spracovávanie správy mikrokontrolerom.....37

Obrázok 16: Výpis funkcie Prijatie správy.....42

Obrázok 17: Vypis funkcie SpracovanieSpravy().....44

Obrázok 18: Funkcia IDN.....46

Obrázok 19: Výpis funkcie OUT.....49

Obrázok 20: Funkcia ADR.....50

Obrázok 21: Spracovanie správy pri parametri 0x30.....52

Obrázok 22: Nulovanie prvku poľa stat.....52

Obrázok 23: Vývojový diagram spracovanie správ riadiacim počítačom.....54

Obrázok 24: Panel testovacieho programu pre ovládanie multiplexorovej karty.....56

Obrázok 25: Oneskorenie zopnutia kontaktu relé oproti výstupu mikrokontroléru.61

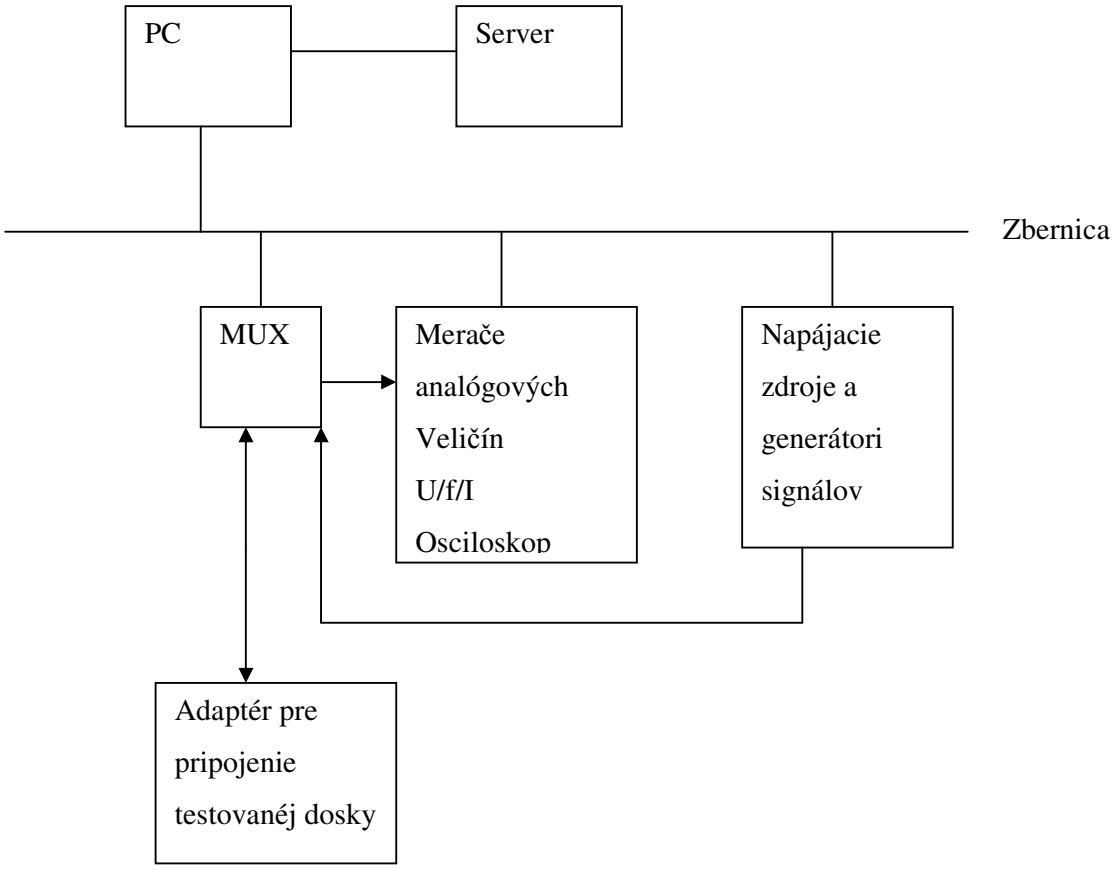
Obrázok 26: Oneskorenie vypnutia relé oproti výstupu z mikrokontroleru.....62

ZOZNAM TABULIEK

Tabuľka 1 : Testovacie požiadavky na testovanú dosku.....12

1. ÚVOD

V sučastnej dobe sa používa množstvo elektronických prístrojov ktoré používajú jednu prípadne viac dosiek plošných spojov .Tieto dosky obsahujú množstvo elektronických obvodov které je potrebné testovať podľa požiadaviek danných výrobcom.Pri sučastnej sériovej výrobe je nutné použíť automatizovaných testovacích systémov. Tieto testovacie systémy sú rôzne zložité podľa typu zariadenia ktoré je potrebné testovať. Základná koncepcia testovacích systémov je na obrázku 1.1.



Obr. 1. Bloková schéma testovacieho systému

Celý systém je riadený počítačom ktorý ovláda pomocou rôznych zbernic ostatné zariadenia.Dôležité je zaznamenávanie nameřených dát ktoré se ukladajú na server

pre dalšie spracovanie. Testovaná doska sa vetšinou nepripája priamo k jednotlivým zariadeniam testovacieho systému ale prostrednictvom multiplexorov.Dôvodom je že se odoberajú signály a ostatné merané veličiny z viac miest na testovanej doske a bolo by potrebné mať velké množstvo presných meračov signálov a dalších analogových věličín čo by bolo ekonomicky a technicky nevýhodné.Meřače analógových veličín môžu byť klasické multimetre, osciloskopy ktoré môž byť riadené z PC prostredníctvom RS232 nebo GPIB apod. Môžu to byť tiež meracie karty do rôznych slotov (PCI,PXI a pod).Napájecie zdroje môžu byť AC,DC,zdroje rôznych signálov podľa potrieb testovaných dosiek. Samozrejme musia mať možnosť byť riadené prostredníctvom nejakej zbernice ,prípadne niektoré sú riadené analógovými signálmi.

Na trhu je momentálne viacero výrobcov testovacích zariadení ako Schaffner,SAAB test systems a pod. Týto výrobcovia vetšinou ponúkajú špecializované systémy pre konkrétne aplikácie,ktorých cena je však pomerne vysoká.Medzi nejlepšie firmi v tomto obore patrí firma National instruments,ktorá ponúka široké množstvo modulárnych systémov,rôznych meracích kariet ,SW a iného zariadenia na testovanie. National instrument používa PXI slot ku ktorému su vyrábané rôzne karty ,analógove,digitálne,vstupne / výstupné apod. Asi jediným problémem PXI systému je stále pomerne vysoká cena . Dalšou možnosťou je zostaviť vlastní návrh testovacieho systému s použitím štandartných mericích prístrojov a napájacích zdrojov a vytvorením vlastného návrhu vstupne výstupných kariet.

2. UKÁŽKA KONKRÉTNYCH POŽIADAVIEK NA TESTOVANIE OSADENEJ DOSKY PLOŠNÝCH SPOJOV

Požiadavky na funkčné testovanie môžu byť rôzne v závislosti od požiadaviek výrobcu zákazníka ,funkcie apod. Uvediem príklad testovacej inštrukcie pre testovanie riadiacej dosky pre napájecie systémy telekomunikačných stanic. Jedná ae o riadiacu dosku ktorá má rozhranie CAN cez ktoré komunikuje z nadradenými a podriadenými modulmi. Doska obsahuje 4.analógove vstupy ,pre merenie prúdu,napätia systému ,napätia záložného batérioveho zdroja a vstup pre meranie teploty. Ďalej má 12 digitálnych vstupov pre signály z rôznych senzorov a 5 digitálnych výstupov.V požiadavkach pre testovanie je uvedené merané napätia na niektorých merných bodoch. Požiadavky na testovanie môžeme zadať v následúcej tabuľke.

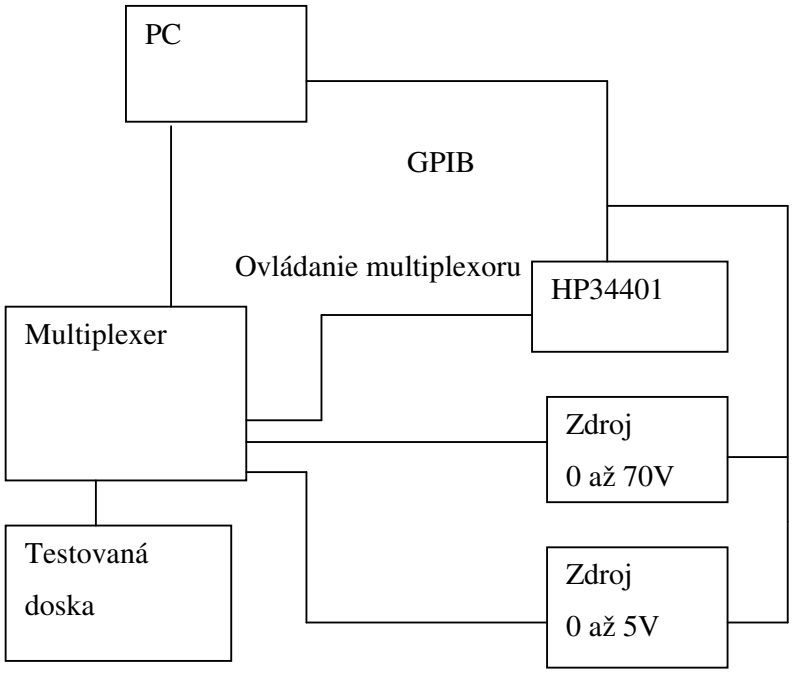
Meranie	**Rozsah**
Analogový vstup 1	0-60V, +/- 0,1%
Analogový vstup 2	0-60V, +/- 0,1%
Analogový vstup 3	0-200mV, +/- 0,1%
Analogový vstup 4	0-0,5V, +/- 0,1 %
Digitálny vstup 1	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 2	Log 0 < 15V ; Log 1 > 50V

Digitálny vstup 3	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 4	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 5	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 6	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 7	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 8	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 9	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 10	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 11	Log 0 < 15V ; Log 1 > 50V
Digitálny vstup 12	Log 0 < 15V ; Log 1 > 50V
Digitálny výstup 1	Výstupy relé
Digitálny výstup 2	Výstupy relé
Digitálny výstup 3	Výstupy relé
Digitálny výstup 4	Výstupy relé
Merací bod 1	5,1V až 5.5V
Merací bod 2	2,5V až 2,8 V
Merací bod 3	9,7 V až 10,3V
Merací bod 4	4,786 až 4,802 V

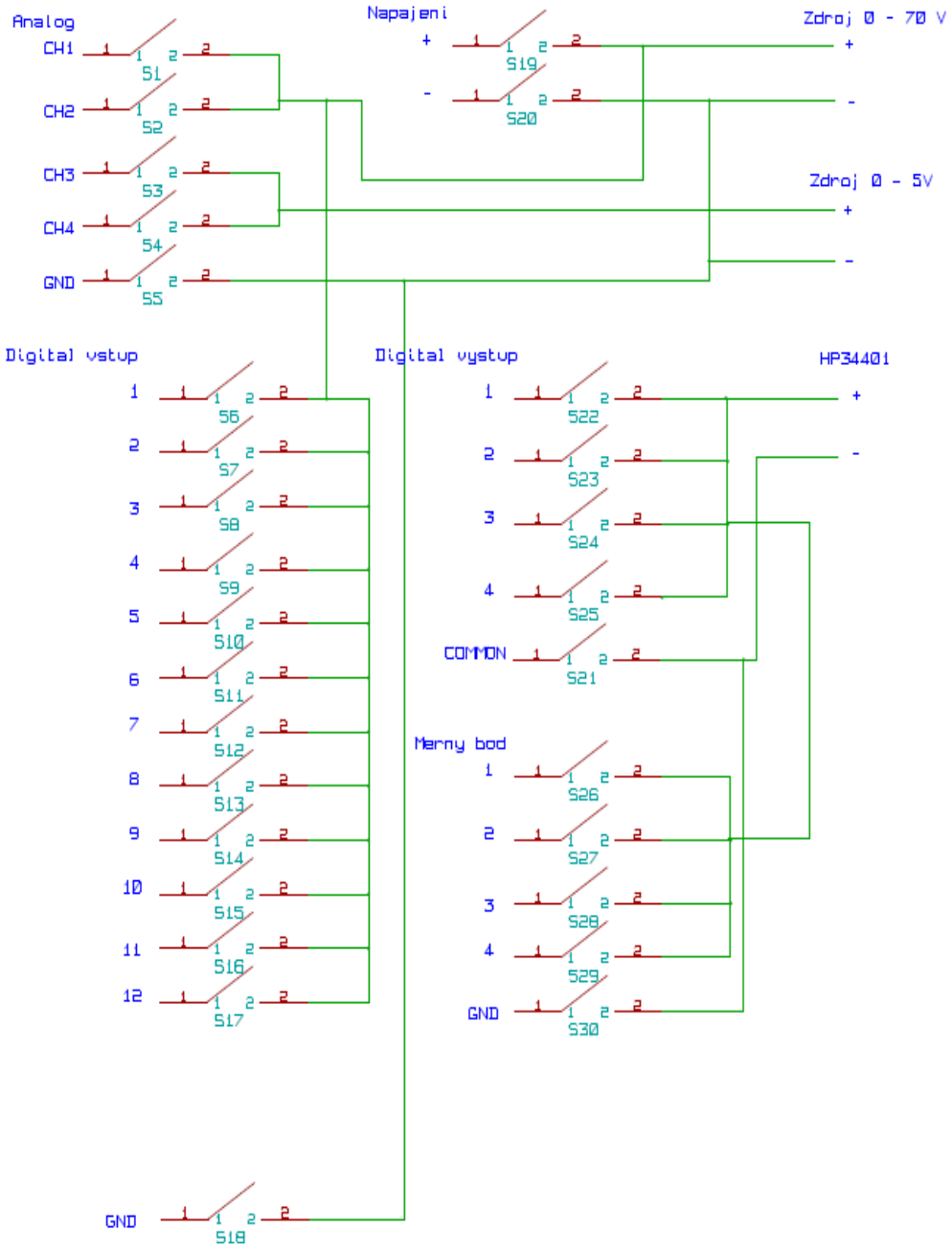
Tabuľka 1. Testovacie požiadavky na testovanú dosku

Všetky signály z analogových vstupov meria testovaná doska s presnosťou 0,1% a posiela ich cez CAN zbernicu do nadriadených modulov,počas testu do počítača. Táto presnosť se počas testu musí overiť v celom rozsahu s danou presnosťou.Treba overiť aj digitálne vstupy a výstupy podľa tabuľky. Túto dosku chceme testovať automaticky ,výsledky spracovávať v počítači a vyhodnocovať ,jednotlivé testy a test celéj dosky.

Na automatický test tejto dosky môžeme použíť jeden presný multimeter ,napríklad HP 34401 s GPIB rohraním, 1jednosmerný zdroj 0 až 70V s GPIB, 1.jednosmerný zdroj 0 až 5V s GPIB a kartu alebo karty multiplexoru aby sa dali prepínať jednotlivé signály.



Obr.2 .Bloková schéma diagnostického systému pre testovanie uvedenej dosky



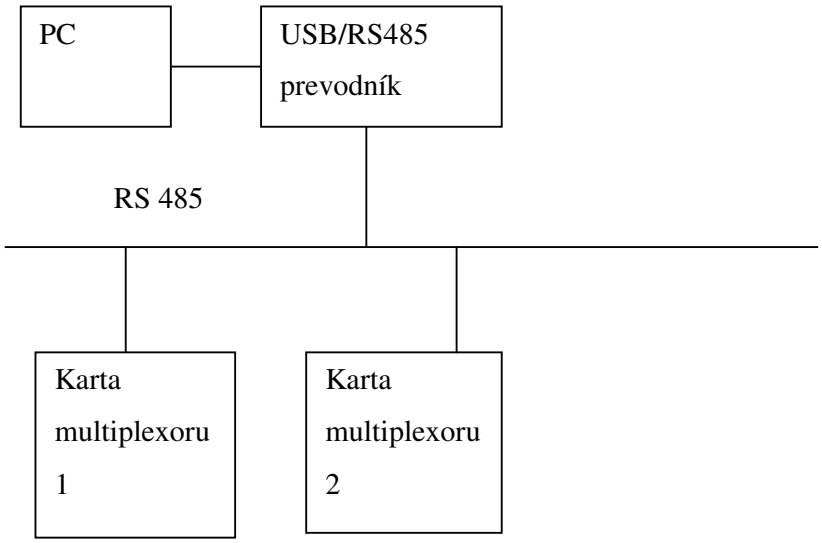
Obr.3. Podrobná schéma zapojenia multiplexoru pri testu uvedenej dosky

Jedno z riešení ako zapojiť multiplexor aby vyhovoval požiadavkám na testovanie uvedenej dosky ja na obrázku 3. Zo schémy vyplýva že pre naše potreby musíme mať kartu z aspoň 30 digitálnimi výstupmi. Digitálne výstupy je nejlepšie mať ako výstupy reléových kontaktov. Návrhom téjto karty sa budem ďalej zaoberať.

3. BLOKOVÉ RIEŠENIE PRIPOJENIA MULTIPLEXOROVÝCH KARIET K TESTOVACIEMU SYSTÉMU

Pri návrhu multiplexorovej karty musíme vychádzať z toho že bude sučasť nejakého testovacieho systému a musíme počítat s tým že systém bude používať viac takých kariet, prípadne testovací systém môže používať dalšich kariet na podobnom princípe, karty anologových vstupov a výstupov a podobne.Najskôr musíme navrhnuť systém ktorým budú karty komunikovať s riadicim systémom. Potom navrhneme konkrétne hardwarove riešenie . Z konštrukčného a praktického hľadiska je výhodne použiť kartu z 30 digitálnymi výstupmi. Ak bude potrebné viac digitálnych výstupov bude možné bez problémov použiť viac kariet. Pri riešení komunikácie medzi kartou a ríadiacim počítačom musíme zvoliť komunikačné rozhranie počítača. Pre riadenie testovacieho systému by malo být použitý štandartné PC. V súčastnosti sa na PC najviac používa rozhranie USB,ktoré by bolo vhodne tiež použiť pre riadenie našej karty. Vzhľadom na to že chceme používať viac kariet v jednom testovacom systéme nie je vhodné mať každú kartu na zvlášť USB porte ale by bolo výhodnejšie použíť prevodník na nejakú inú zbernicu ktorá by bola súčasťou jednotlivých kariet. Nejvýhodnejšie by bolo použíť nejakú paralelnú zbernicu aby se dalo bez problémov pripojiť viac kariet. Zo známych zbernic som sa rozhodol pre RS485. Táto zbernica umožňuje pripojiť 32 zariadení,čo je pre našu aplikáciu postačujúce. RS485 pracuje v polo duplexnom alebo plne duplexnom režime. Ja budem využívať polo duplexný režim. Pri tomto režime je pre komunikáciu potrebné mať 2 vodiče ,používajú sa krútené dvojlinky. Na tieto vodiče se pripájajú paralelne všetky zariadenia. Každá dvojlinka musí byť ukončená rezistorom asi 120 Ohmov. Podľa špecifikácie RS 485 môže být dĺžka vodičov až 1200 metrov. V tejto aplikácii nebudeme používať také dĺžky,používané dĺžky budú do 1 metra. Táto konfigurácia umožňuje prenosové rýchlosti do 10 Mb/s. Pre túto aplikáciu budeme využívať prenosové rýchlosti podstatne nižšie a budú závisieť od

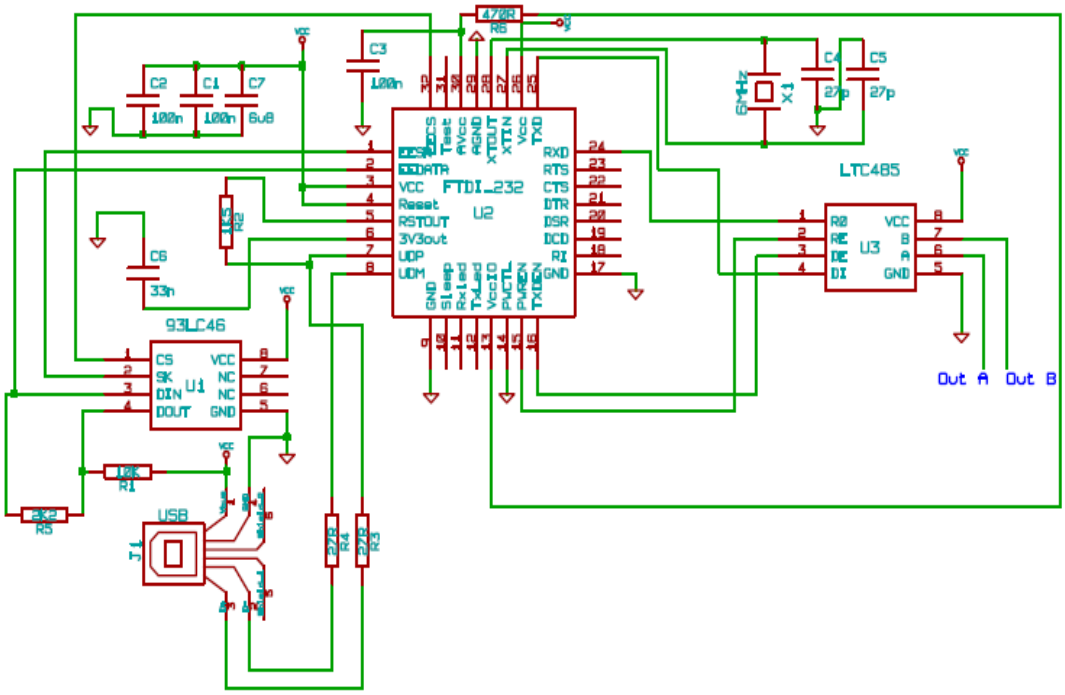
parametrov použitých sučástiek a potrebám danej aplikácie. Na karte budu použíté prevodníky napäťových rozhraní RS485 na UART ktorý sa dá pripojiť na vstup mikrokontroleru ktorý bude súčasťou karty.



Obr.4. Bloková schéma pripojenia Multiplexorových kariet k PC

4.PREVODNÍK USB/RS485

Prevodník USB/RS485 zabezpeči pripojenie multiplexorových kariet k PC.Multiplexorové karty budú mať rozhranie RS 485 , ktoré nie je obvykle súčasťou počítačov. Nejbežnejším rozhraním v súčastných PC je USB.Prevodníkov USB/RS485 je trhu mnoho. Ja sa sústredím na návrh vlastného riešenia postaveného na doporučených zapojeních výrobcov. USB rozhranie je sériova zbernice kde se data prenášajú diferenčne. USB konektor obsahuje 4 vývody. +5V, Data+, Data- a GND [1]. Pripojenie prevodníku k USB sa dá řešit rôzne. Ja som si vybral riešenie popísane v [1]. Základom prevodníku bude obvod od firmi FDI Chip FT 232 BM. Jeho zapojenie su dobre popísané a sú k dispozícii tiež ovládače pre operačné systémi Windows,Linux MAC OS. To veľmi uľahčí vývoj aplikacie ktorá bude ovládať multipexorové karty.Použijem zapojenie uvedené v [1]. V tomto zapojení to bude prevodník napäťových úrovni USB na UART,ktorý bude mať signály TX a RX a tie budú budiť budiče RS 485. Ako budiče RS485 použijem obvody firmy Linear technology LTC 485.



Obr.5. Schéma zapojenia prevodníku USB /RS485

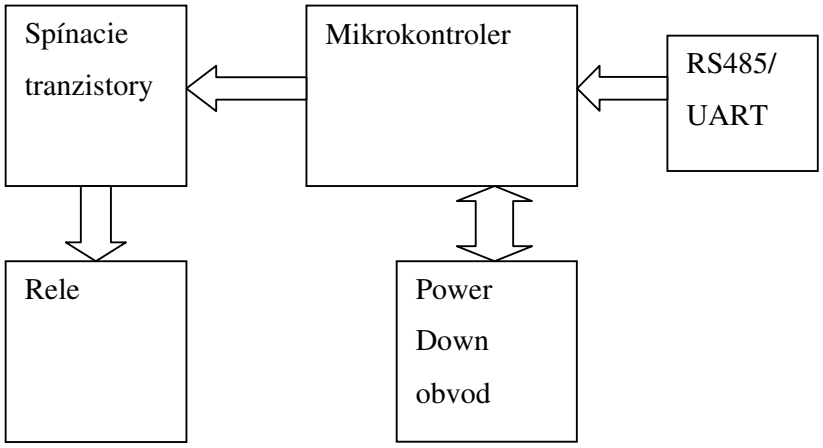
Zapojenie prevodníku je katalogové zapojenie obvodu FT 232 ,popísané v [2]. Celý prevodník bude napájaný z USB portu . V EEPROM U1 jsou zapísané idenifikačné data podľa ktorých je zariadenie jednoznačne identifikované operačným systémom. Prevodník U3 LTC 485 prevádza UART signály z U2 na rozhraní RS485 které má výstupy OutA a Out B,ktoré sa privádzajú na zbernicu po ktorej budú komunikovať s ostatnými zariadeniami. Pretože sa jedná o poloduplexný mód je potrebné prepínať vysielanie a príjem,ktoré zabězpečí obvod U2 na piny 2 a 3 obvodu U3.

5.REALIZÁCIA HW ČÁSTI MULTIPLEXOROVÉJ DOSKY

V nasledujúcich odstavcoch sa budem zaoberať podrobným popisom realizácie HW časti jednotlivých blokov multiplexorovej karty .

5.1 BLOKOVÉ RIEŠENIE MULTIPLEXOROVEJ KARTY

Multiplexorová karta má mať 30 digitálnich výstupov, musí mať rozhranie RS 485 pre komunikáciu s riadiacim počítačom. Digitálne výstupy budú riešené ako spínacie kontakty relé. Blokové riešenie karty je na následujúcom obrázku.



Obr.6.Blokové riešenie multiplexorovej karty

Jádrom multiplexorovej karty bude Mikrokontroler ktorý bude zpracovávať signály s riadiaceho počítača, dekódovať ich a posielať signály na spínacie tranzistory.Pretože Mikrokontroler nebude mať rozhranie RS485 ale len UART,musí být zaradený mezi rozhranie RS485 a Mikrokontroler ovod prevodníku napäťových rozhraní RS485 na UART.Výstupy z mikrokontroleru niesu dostatočne prúdove ani napeťovo dimenzované pri spínaní relé,musí sa na výstupne piny mikrokontroleru zaradiť spínacie prvky ktoré signál dostatočne zosília pre spoľahlivé zopnutie rel. Pretože multplexorová karta bude mať 30 digitálnych výstupov musíme počítať s 30 relé. Relé sú mechanické prvky ktoré majú omedzenú životnosť,ktorá závisi od počtu zopnutí a prudu ktorý kontaktami prechádza.Dôležitý je tiež prechodový odpor kontaktov ktorý nám zabezpečuje výrobca len pre určitý počet zopnutí. Zhoršenie prechodového odporu môže výrazne ovplyvniť úroveň signálu ktoré prechádzajú cez multiplexorovu kartu a tým aj skresľuje merenie. Z tohoto dôvodu je dobré mať na karte funkciu ktorá by počítala počet zopnutí každého relé a tieto výsledky by potom predávala do riadiaceho počítača ktorý,by potom operátora upozornil pri prekročení určitej nastavenej hodnoty zopnutí. Na základe tejto informácie by sa mohli jednotlivé relé na hranici garantovanej životnosti vymeniť. Tieto informácie musia byť uložené na konkrétnej multiplexorovej karte . Pretože je nutné mať tieto informácie uložené i počas vypnutia karty je potrebné aby boli uložené v EEPROM Mikrokontroleru. Z hľadiska životnosti EEPROM ktorá má určený maximálny počet zápisov nie je vhodné aby sa tam ukladali informácie pri každom zopnutí relé. Riešením je ukladať počet zopnutí počas behu karty do nejakej premennej a potom v určitý čas ich zapísať do EEPROM.Aby sa tieto data nestratili pri vypnutí karty musia byť počas vypnutia uložené do EEPROM. Toto zabezpečuje Power down obvod ktorý pri vypnutí pošle signál procesoru na zápis dát do EEPROM a oneskorý vypnutie procesoru o čas potrebný na zápis dat.

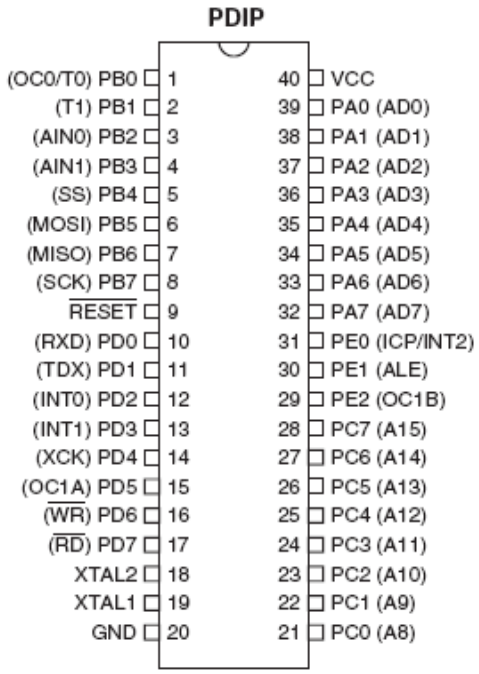
5.2 PODROBNÝ POPIS ZAPOJENIA OBVODU MIKROKONTROLÉRU A PREVODNÍKU NAPÄŤOVÝCH ROZHRANÍ RS 485/UART

Mikrokontroler bude jadrom multiplexorovej karty. Jeho úlohou bude spracovávať signály z riadiaceho počítača ich dekódovanie a budenie jednotlivých tranzistorov. Ďalej musí býť schopný uchovávať štatistické údaje o počte zopnutí jednotlivých relé. Na základe týchto požiadaviek definujeme očakávane parametry pre mikrokontroler. Pretože má karta mať 30 digitálnych výstupov musí mať mikrokontrolér aspoň 30 výstupných pinov ktoré budú ovládať jednotlivé budiace tranzistory.Ďalej musí mať UART vstup na ktorý sa pripojí prevodník napäťových rozhrní RS485/UART. Tento prevodník bude naviac pre svou činnost potrebovať 2 vstupno výstupne piny.Ďalej mikrokontrolér musí mať EEPROM kde se budú zapisovať jednotlivé štatistické data o počne zopnutí relé.

A ešte budeme potrebovať jeden vstupno / výstupný pin pre pover down obvod. Celkom budeme teda potrebovať 33 vstupno / výstupných pinov, vstupy UART,ďalej EEPROM s dostatočnou kapacitou na zápis štatistických údajov. Ďalej musí mať mikrokontrolér priaznivú cenu a musí byť k dispozícii vhodné vývojové prostredie v ktorom by bolo možné obvody naprogramovať. Je viac výrobcov mikrokontrolérov ktoré by vyhovovali naším požiadavkám napr. Microchip,Motorola Almel. Z hľadiska dostupnosti obvodov na trhu a dostupnosti vývojových prostredí sa javí ako nejvhodnejší Atmel. Pre mikrokontrolery Atmel je k dispozíici volne šíritelné vývojove prostredie s prekladačom jazyka C Win AVR pre OS Windows. Existuje aj varianta AVR GCC pro operační systém LINUX. Po kompilácii sa hex súbor naprogramuje priamo do mikrokontroleru bežnými programátormi.

Z mikrokontroléru firmy Atmel vyhovuje našim požiadavkám viac typov mikrokontrolerov ja som si vybral mikrokontroler Atmega 8515. Jeho popis je v datasheete od výrobcu [3] .Podle datasheetu má tento obvod 35 programovatelných vstupno výstupných pinov, ktorých súčasťov je aj UART vstup který využije 2 piny na UART RX a UART TX. Teda ostáva 33 vstupno /výstupných pinov čo vyhovuje našim požiadavkám.

Ďalej má tento obvod 512 Bytov EEPROM čo bude určite postačovať pre našú aplikáciu. Celkom má tento obvod 40 pinov . Vyrába sa v 3 druhoch púzdier PDIP, TQFP a PLCC.

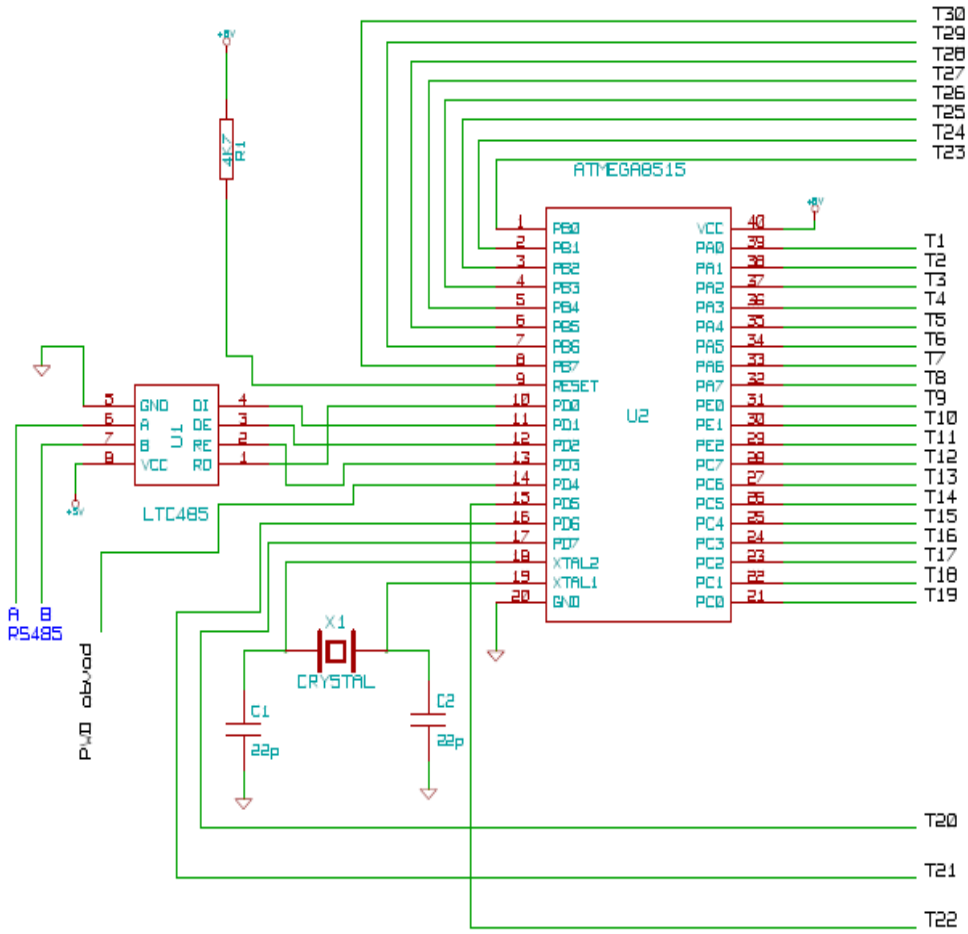


Obr.7. Rozmiestnenie pinov mikrokontroléru ATMEGA 8515 [3]

Dva piny VCC a GND sa používajú pre napájanie ,napájanie obvodu je 4,5 až 5,5 V. Ďalej má 2 piny pre pripojenie kryštalu XTAL1 a XTAL2, jeden pin pre RESET. A nakoniec 35 Vstupno výstupných pinov. PA0 až PA7, PB0 až PB7,PC0 až PC7, PD0 až PD7 a nakoniec PE0 až PE2. Piny PD0 a PD1 su vyhradené pre UART vstup PD0 pre RXD a PD1 pro TXD. Kryštál môže být v rozmedzí 0-16 MHz. Voľba vhodného kryštálu bude závisieť od použitej prenosovej rýchlosti pri komunikácii medzi mikrokontrolérom a riadiacim počítačom. EEPROM má povolené 100 000 zápisov a čítaní,z čoho vyplýva že nemôžeme do EEPROM zapisovat pri každom zopnutí Rele pretože by sa za krátky čas prečerpal povolený limit zápisov a

mikrokontroler (respektíve jeho EEPROM) by bol nefunkčný.Preto zápis do EEPROM bude vykonavaný iba pri vypnutí multiplexorovej karty prípadne po zasláni požiadavky z riadiaceho počítača. Vstupne /výstupné piny nakonfigurované ako výstupné môžu pri napájacom napätí 5 V dodávať prud maximálne 20mA , v prípade že by boli všetky v zopnutom stave tak maximálny odber zo všetkých pinov môže byť 200mA. V takom prípade môžeme počítať s maximálne 6mA na pin. Táto hodnota bude potrebná pre výpočet zosílenia spínacích tranzistorov. Odber prúdu celého mikrokontroléru je 200mA plus odber jednotlivých pinov. Teda maximálny odber celého mikrokontroléru nepresiahne 400mA. S touto hodnotou budeme počítať pri návrhu „Power down obvodu“.

Ako prevodník napäťových rozhraní RS485/UART je použitý obvod LTC 485 ktorý sa používa aj v prevodníku USB/RS485. Tento obvod má 2 vstupy rozhrania RS485 vstup A a B,potom má 2 napájacie piny , 2 piny pre UART výstup a dva prepínacie piny,ktoré se prepínajú podľe toho či sa vysiela alebo prijíma.Toto prepínanie zabezbečí mikrokontroler.V obvode mikrokontroléru sú ešte zapojené kryštál s dvomi kondenzátormi 22pF. Hodnota kryštálu je závisla od prenosovej rýchlosti ktorou bude mikrokontroler komunikovať s ríadiacim počítačom. Jako nejvýhodnejšia hodnota kryštálu sa javí hodnota 3,6864MHz od ktorej se dá odvodiť vätšina používaných prenosových rychlosti.Rezistor R1 4K7 slúži ako pripojenie RESET pinu na +5V. Výstupy T1 až T30 slúžia na pripojenie jednotlivých spínacích tranzistorov na spínanie výstupných relé.Výstup PWD obvod služi na pripojenie k Power down obvodu a bude aktívny počas odpojenia napájecieho napätia.Počas jeho aktivácie se uloží všetky potrebné data do EEPROM mikrokontroleru.



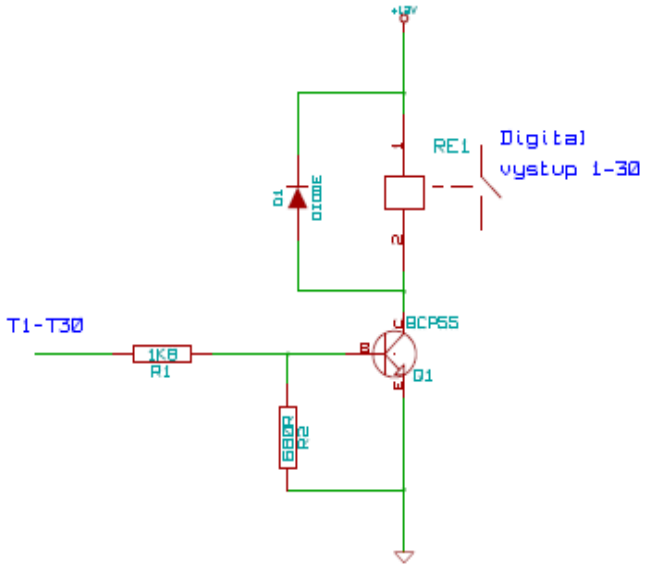
Obr.8 .Schéma zapojenia obvodu mikrokontroleru a prevodníku napäťových rozhraní RS485/UART

5.3 PODROBNÝ POPIS ZAPOJENIA OBVODU SPÍNACÍCH TRANZISTOROV A OBVODU RELÉ.

Obvod spínacich tranzistorov bude slúžiť na zopnutie jednotlivých relé podľa signálu z mikrokontroleru. Ako spínacie relé som vybral typ N4100 s napájacím napätím 12V. Odpor cievky relé je 720 Ohmov. Z Ohmovho zákona som vypočítal prúd cievkou ktorý je 17mA. Z katalogu výrobcu som určil maximálny počet zopnutí pri ktorom sú garantované všetky parametre. Táto hodnota je 100 000 zopnutí. S touto hodnotou budem počítať v štatistike ktorá bude informovať o vypršaní životnosti

jednotlivých relé. Je si potrebné uvedomiť aké maximálne prúdy a napätia ktoré môžu protékať cez kontakty v zopnutom stave. Rovnako si treba uvedomiť rozdiel mezi prudmi ktoré pretekajú už zopnutými kontaktmi a prudmi ktoré sú na kontaktoch v okamžiku spínania relé.Tieto prúdy a napätia bývajú podstatne nižšie ako prúdy a napätíia ktoré môžu prechádzať už zopnutými kontaktami. Pretože sú zdroje rovnako riadené riadiacim počítačom nejlepšie riešenie je spínať kontakty relé pri vypnutých zdrojoch prípadne pri malých napätiach.

Ako spínaci tranzistor som vybral typ BCP55. Podľa datasheetu výrobcu sa jedná o tranzistor s povoleným napätím kolektor emitor 60V a s povoleným kolektorovým prúdom 1A. Pri našej aplikácii bude napätie kolektor emitor 12V a kolektorový prud 17mA,takže pre našu aplikáciu plne vyhovuje. V zapojení tranzistora ako spínača sa nepoužíva maximálny prúdovy zosilovací činitel tranzistoru ale se „vnutí“tranzistoru požadovaný prúdový zosilovací činitel. Táto hodnota býva od 10 do 50 ja použijem hodnotu 10.Pri tejto hodnote je kratšia doba zopnutia tranzistoru ale je dlhšia vypínacia doba tá sa dá však zkrátiť zaradením odporu medzi bázu a emitor tranzistoru.



Obr.9. Schéma zapojenia obvodu spínacieho tranzistoru a relé (1 vetva).

Prúd do báze tranzistoru I_B vypočítame podľa vzťahu 1.1

$$I_B = \frac{I_C}{h_{21E}} \qquad (1.1)$$

Prúd I_C je známi je to prúd ktorý prechádza relé ,teda 17mA a tiež h_{21E} sme určili hodnotu 10. Výsledný prúd do báze teda bude: $I_B = \frac{17mA}{10} = 1{,}7mA$. Ďalej zvolíme prúd rezistorom R_2 ktorý by mal byť porovnateľný s prúdom I_B teda volíme hodnotu 1mA. Hodnotu rezistoru R_2 určíme zo vzťahu 1.2

$$R_2 = \frac{U_{BE}}{I_{R2}} \qquad (1.2)$$

Hodnota napätia U_{BE} je 0,6V,teda môžeme určiť hodnotu rezistoru R_2

$R_2 = \frac{0{,}6}{0{,}001} = 600\Omega$, Volíme rezistor $680\,\Omega$. ďalej pre určenie hodnoty rezistoru R_1 musíme vypočítať hodnotu prúdu týmto rezistorom zo vztahu 1.3

$$I_{R1} = I_{R2} + I_B \qquad (1.3)$$

Po dasadení $I_{R1} = 1mA + 1{,}7mA = 2{,}7mA$.Ďalej určíme hodnotu napätia na rezistore R_1 podľa vzťahu 1.4

$$U_{R1} = U_{vst} - U_{BE} \qquad (1.4)$$

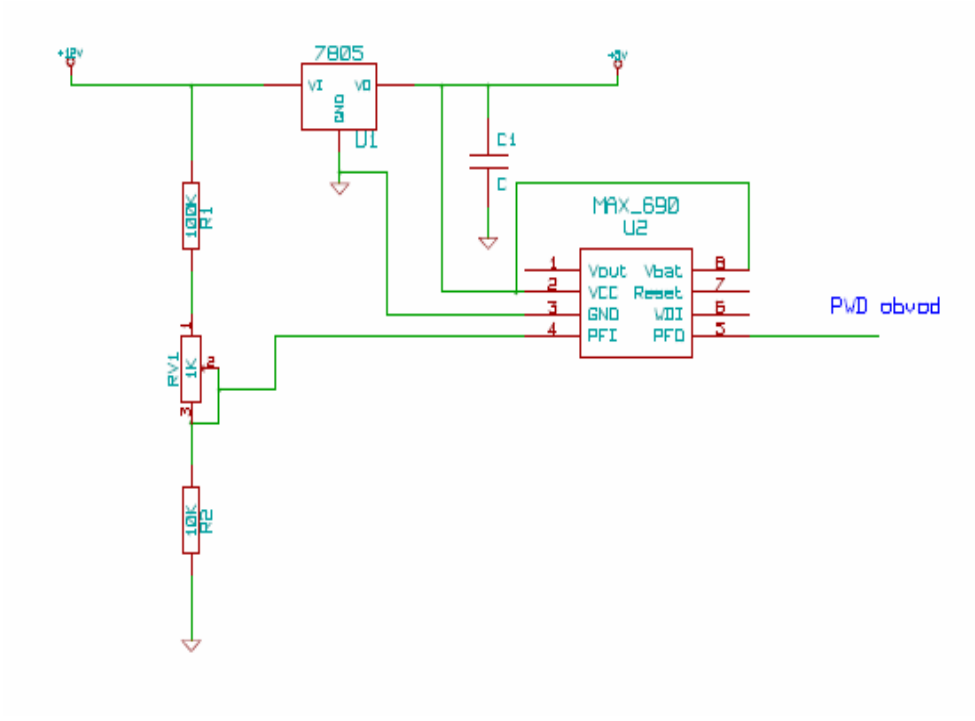
Hodnota napätia U_{vst} je výstupné napätie z mikrokontroleru ktoré je 5V.Po dosadení dostaneme $U_{R1} = 5 - 0{,}6 = 4{,}4V$ .A z Ohmovho zákonu vypočítame hodnotu

rezistoru R_1, $R_1 = \frac{U_{R1}}{I_{R1}} = \frac{4{,}4}{0{,}0027} = 1629\Omega$. Volíme hodnotu rezistoru R_1 1K8.

Dioda D_1 zapojená paralelne k relé v závernom smere slúži na ochranu tranzistoru proti napätiu vznikajúcom na cievke relé pri vypnutí relé. Napájanie celého obvodu je 12V . Multiplexorová doska obsahuje celkom 30 vetví tohoto obvodu pripojené na výstupy z obvodu mikrokontroleru T1 až T30. Spínacie kontakty jednotlivých relé tvorí 30 nezávislych digitálnych výstupov multiplexorovej karty.

5.4 PODROBNÝ POPIS ZAPOJENIA POWER DOWN OBVODU

Power down obvod zabezpečuje napájanie jednotlivých blokov multiplexorovej karty a dáva signál mikrokontroleru o vypnutí napájacieho napätia.Po tomto signále musí zabezpečit napájanie mikrokontroleru po dobu potrebnú na zápis štatistických údajov do EEPROM mikrokontroleru.



Obr.10 Schéma zapojenia Power down modulu

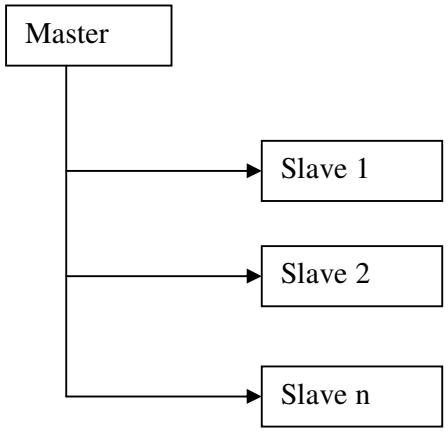
Napájanie celej multiplexorovej dosky je 12V. Toto napätie je privedené do Power down modulu na stabilizátor U1 ktorý z toto napätie upraví a stabilizuje na 5V. Toto napätie je potrebné pre napájane mikrokontroleru.Ďalej je v tomto obvodu zapojený integrovaný obvod U2 MAX 690,ktorý v tomto zapojení doporučenom výrobcom [4] plní funkciu Power fail obvodu. Na pin 4 PFI tohoto obvodu je privedené cez delič R1,RV1,R2 napájacie napätie 12V. Po vydelení tohoto napätia bude na pine 4 1,3V. Trimer RV1 je potrebné nastaviť tak aby pri napájacom napätí 11,5V bolo na pine 4 U2 napätie 1,3V. Pokiaľ toto napätie klesne pod 1,3V na pine 5 PFD je Log 0.Táto hodnota je privedená na mikrokontroler a signalizuje odpojenie napájania. Mikrokontroler potrebuje na zápis do EEPROM podľa datasheetu 8ms. Aby boli data spoľahlivo zapísané potrebujeme aby po zaslaní signálu Power down bolo napájacie napätie aspoň 4,4V po dobu 20ms. Toto nám zabezpečí kondenzátor C1. Pretože maximálny možný odber mikrokontroléru je 400mA pri napätí 5V. Podľa Ohmovho zákona bude teda v tomto prípade odpor obvodu 12,5 Ohmů. Kondenzátor sa bude exponenciálne vybíjať s časovou konštantou $\tau = R.C$. V časovom okamžiku $\tau/8$ bude napätie $U = 5.e^{-\frac{\tau}{8}} = 5.e^{-0,125} = 4,41\text{V}$. Pokiaľ potrebujeme mať toto napätie po dobu 20mS musí byť časová konštanta $\tau = 8.20ms = 160ms$. Hodnotu kondenzátoru C tedy určíme zo vzťahu $C = \frac{\tau}{R} = \frac{0,16}{12,5} = 12,8mF$. Podľa vyrábaných typov môžeme voliť kapacitu 0,1F.

6.NÁVRH KOMUNIKAČNÉHO PROTOKOLU

V tejto kapitole sa budem zaoberať návrhom komunikačného protokolu podľa ktorého budú multiplexorové karty komunikovať z riadiacim počítačom.

6.1. VŠEOBECNÝ POPIS PROTOKOLU

Komunikačný protokol je súbor pravidiel podľa ktorých komunikujú medzi sebou zariadenia na spoločnej zbernici. Zbernica ktorú budem pri tomto návrhu používať je RS485. Táto zbernica môže pracovať buď v plne duplexnom móde alebo polo duplexnom móde. Ja budem používať polo duplexný mód. Pri tomto móde sa používajú 2 vodiče na ktoré sa paralelne pripájajú jednotlivé zariadenia. Táto zbernica umožňuje pripojiť až 32 zariadení. V tomto návrhu budem počítať s pripojením maximálne 16 zariadení. Jedno zariadenie bude riadiaci počítač a ostatných 15 zariadení budú multiplexorové prípadne iné karty. Ja budem používať prevádzku typu Master- Slave .Pri jej popise budem čerpať z literatúry [6].



Obr.11 Prevádzka typu Master-Slave [6].

Při prevádzke Master-Slave, je jedna stanica nadradená typu Master a riadi činnosť ostatných staníc typu Slave.Master môže vysielať kedykoľvek a komunikovať s kterou koľvek stanicou typu Slave. Stanice typu Slave môžu komunikovať iba zo stanicou typu Master,a vysielajú iba v prípade povolenia od stanice typu Master. V tomto prípade bude stanica typu Master riadiaci počítač a stanice Slave budú jednotlivé multiplexorové karty. Každá multiplexorová karta bude mať Slave adresu podľa ktorej bude jednoznačne určené s ktorou kartou počítač komunikuje. Všetky karty budú trvalo na prijíme a vysielať budú iba v prípade požiadavky od riadiaceho počítača.

6.2 POPIS FORMÁTU SPRÁVY

Každá správa bude mať presne stanovený formát ktorý bude obsahovať všetky potrebné informácie na správnu činnosť zariadenia.Formát správy je na obrázku 12.

Zdroj	Cieľ	Počet	Data	Checksum

Obr.12 Formát správy

Každá zpráva sa bude vysielať jako skupina znakov v Hex formáte. Jako prvý znak bude adresa zdroja zprávy. Táto adresa bude obsahovať iba jeden znak.Adresa riadiaceho počítača bude 0x0F , adresy multiplexorových kariet budú 0x00 až 0x0E.

Ku každej bude pripočítaná hodnota 0x30,takže adresa 0x0F sa bude prenášať ako 0x3F. Adresa cieľa bude obsahovať adresu zariadenia pre ktoré je určená.Táto adresa bude obsahovať rovnako iba jeden znak ,hodnoty adries a pripočítavanie čísla 0x30 je rovnaké jako u adresy cieľa . Tretí znak formátu správy je počet a informuje o počte znakov v políčku Data. Políčko počet bude obsahovať tiež iba jeden znak. K výslednemu počtu znakov v políčku Data sa pripočíta hodnota 0x30.Takže pokiaľ bude v políčku Data počet znakov 7 (hex hodnota 0x07) ,výsledný znak ktorý bude prenášaný je 0x37. V políčku Data sa budú prenášať jednotlivé príkazy na ovládanie . Toto políčko bude vysvetlené v ďalšom odstavci. V políčku Checksum sa bude prenášať informácia o kontrólnom súčte znakov v správe aby sa potvrdilo či bola zpráva prijatá korektne. Toto políčko bude popísané v ďaľších odstavcoch.

6.3.POPIS DÁTOVEJ ČASTI

Dátova časť protokolu obsahuje príkazy a parametre na základe ktorých sa ovláda multiplexorová karta. Základna štruktúra dátovej časti je na obrázku 13.

1 Byte	0 až 20 Byte
Príkaz	Parametre

Obr.13 Dátova časť protokolu

Dátova časť protokolu sa skladá z 2 častí. Prvou je Príkaz o veľkosti 1 Byte. Druhou časťou je parameter ktorý môže byť prázdny alebo môže mať až 20 Bytov.

6.4 POPIS PRÍKAZOV

Jednotlivé príkazy budú ovládať všetky funkcie multiplexorovej karty, prípadne iných druhov kariet. Ďalej uvediem popis každého príkazu:

- IDN (hex hodnota 0x30) – Príkaz IDN : Príkaz na identifikáciu jednotlivého zariadenia. Príkaz z hex parametrom 30 (hex hodnota 0x30) môže vysielať iba Master a je to požiadavka na zaslanie sériového čísla zariadenia. Každá vyrobená karta má jedinečné sériové číslo na základe ktorého je ju možné identifikovať. Odpoveď Slave zariadenia na tento príkaz je IDN a za ním ako parameter nasleduje sériove číslo. Sériove číslo je zapisované priamo v hex formáte,hex hodnoty budú odpovedať hodnotám z ASCII tabuľky. Príklad , sériove číslo M0001 bude zapísane v hex formáte ako: 0x39 0x1C 0x1C 0x1C 0x1D. Sériové číslo každej karty je uložené v EEPROM mikrokontroleru a ukladá sa tam pri výrobe.
- OUT (hex hodnota 0x31) – Príkaz OUT: Tento príkaz slúži na nastavenie digitálnych výstupov. Pokiaľ tento príkaz vysiela Master je nasledovaný 8 Bytami, ktoré určujú stav digitálnych výstupov. Prvý Byte obsahuje výstupy T1 až T4, druhý byte T5 až T8 , posledný Byte bude na ovládanie T29 a T30. Digitálne výstupy sa kóduju následovne. Stavy jednotlivých výstupov ovládajú 4.bity .výstup T1 má MSB ,T4 LSB a pod. Tieto bity sa vyjadria v hex hodnote ku ktorej je pripočítana hodnota 0x30 Príklad: Chceme nastaviť do zopnutého stavu výstup T2,T7,T12 a T27,ostatné budú vo vypnutom stave. 1. Byte bude: 0100 (hex 0x04 + 0x30 = 0x34) ,2.Byte 0010 (hex 0x02 + 0x30 = 0x32), 3. Byte bude 0001 (hex 0x01 + 0x30 = 0x31), 4.Byte 0000 (hex 0x00 + 0x30 = 0x30), 5.Byte 0000 (hex 0x00 + 0x30 = 0x30), 6.Byte 0000 (hex 0x00 + 0x30 = 0x30), 7.Byte 0010 (hex 0x02 + 0x30

= 0x32),8.Byte 0000 (hex 0x00 + 0x30 = 0x30). Odpoveď Slave zariadenia po úspešnom prijatí správy bude príkaz OUT bez parametru.

- ADR (hex hodnota 0x32) – Príkaz ADR, slúži na nastavenie SLAVE adresy. Príkaz má len 1 parameter veľkosti 1 Byte .Tento príkaz s parametrom môže vysielať iba Master a nastavuje ním novú Slave adresu jednotlivej karty.Parameter určuje Slave adresu. Ku tejto hodnotesa pripočíta hodnota 0x30. Príklad: Pokiaľ chceme nastaviť Slave adresu na hodnotu 09 ,parameter bude 0x09 + 0x30 = 0x39. Odpoveď Slave zariadenia po úspešnom vykonaní príkazu je ADR bez parametru. Hodnota Slave adresy je uložená v EEPROM mikrokontroleru.
- STA (hex hodnota 0x33) –Príkaz STA,slúži na načítanie štatistických údajov o zopnutí jednotlivých relé. Po tomto príkaze nasleduje parameter ktorý má 1 Byte môže mať dve hodnoty 0x30 a 0x31. Hodnota 0x30 znamená požiadavku o prečítanie údaju o počte zopnutí daného výstupu, hodnota 0x31 znamená vynulovanie štatistiky pre konkrétny výstup. Za týmto parametrom nasleduje číslo daného digitálneho výstupu. Každá číslica je prenášaná ako 1 Byte a jej hex hodnota zodpovedá dannému číslu v ASCII tabuľke. Príklad: Chceme zistiť počet zopnutí digitálneho výstupu T21 : 1.parameter 0x30 potom nasleduje hex hodnota čísla 2 ,t.j 0x1E, potom hex hodnota čísla 1, t.j 0x1D. Odpoveď Slave je následovná: Slave pošle príkaz STA ,potom nasleduje číslo digitálneho výstupu kódovaného ako pri príkaze z Mastra. A potom následuje hodnota o počte zupnutí,kde je každá číslica kódovaná podľa hex hodnoty v ASCII tabuľke. Príklad : Pokiaľ je hodnota výstupu T21 12344,odpoveď bude vyzerať následovne: STA 0x33 ,číslo digitálneho výstupu 2 ,0x1E, 1 0x1D. číslo 1 0x1D, číslo 2 0x1E,číslo 3 0x1F,číslo 4 0x20 , číslo 4 0x20.

6.5 .POPIS ČASTI CHECKSUM

Táto časť formátu správy slúži na overenie či prijatá správa bola prijatá korektne. Pri prenose správy môžu následkom rôznych chýb prenosu sa poškodiť jednotlivé byty správy čo by pri dekódovaní správy spôsobovalo problémy. Preto sa ku správam pripájajú kontrolne reťazce podľa ktorých prijímacie zariadenie môže kontrolovať prijímanú správu. Metódy na kontrolovanie reťazcov sú rôzne. Medzi najznámejšie patria CRC16, BCC a podobne. Ja som si vybral metódu používanú ako alternatívnu pri kontrole TCP protokolu. Táto metóda je popísaná v [7].

Princíp tejto metódy spočíva v sčítavaní hex hodnôt jednotlivých Bytov správy. Výsledok súčtu sú 2 byty, Byte A a byte B. Pokiaľ je D,je pole hex hodnôt správy, i je poradie Bytu v správe. Hodnoty A a B vypočítavame podľa vzťahu 1.5

$$A = A + D[I]$$

$$B = B+A \qquad (1.5)$$

Počiatočné nastavenia A = 0, B=0. Ak A > 0xFF , A = A – 0xFF , Ak B > 0xFF , B = B – 0xFF

Príklad výpočtu:

Zdroj správy :0x3F

Cieľ správy : 0x30

Počet: 0x32

Príkaz: IDN 0x30

Parameter 0x30

1.) A = 0 ; B=0

2.) A=0 + 0x3F = 0x3F

B = 0 + 0x3F = 0x3F

3.) A= 0x3F + 0x30 = 0x6F

B=0x3F+0x6F= 0xAE

4.) A= 0x6F + 0x32 = 0xA1

B =0xAE+0xA1 = 0x14F – 0xFF = 0x50

5.) A = 0xA1+0x30= 0xD1

B =0x50+0xD1=0x121-0xFF = 0x22

6.) A=0xD1+0x30=0x101-0xFF = 0x02

B= 0x02+0x22= 0x24

S výpočtov nám vyšli hodnoty A= 0x02 a B= 0x24

Tieto hodnoty sa budú prenášať v 4 Bytoch následovne. Každá číslica sa prenáša samostatne s tým že sa ku každej číslici pripočíta hodnota 0x30. Prvé dva byty budú

hodnota A a ďalšie dva byty budú B. Takže Checksum tejto správy bude : 0x30 ,0x32 , 0x32, 0x34.

Výsledná správa vrátane Checksum bude v hex formáte :

0x3F 0x30 0x32 0x30 0x30 0x30 0x32 0x32 0x34

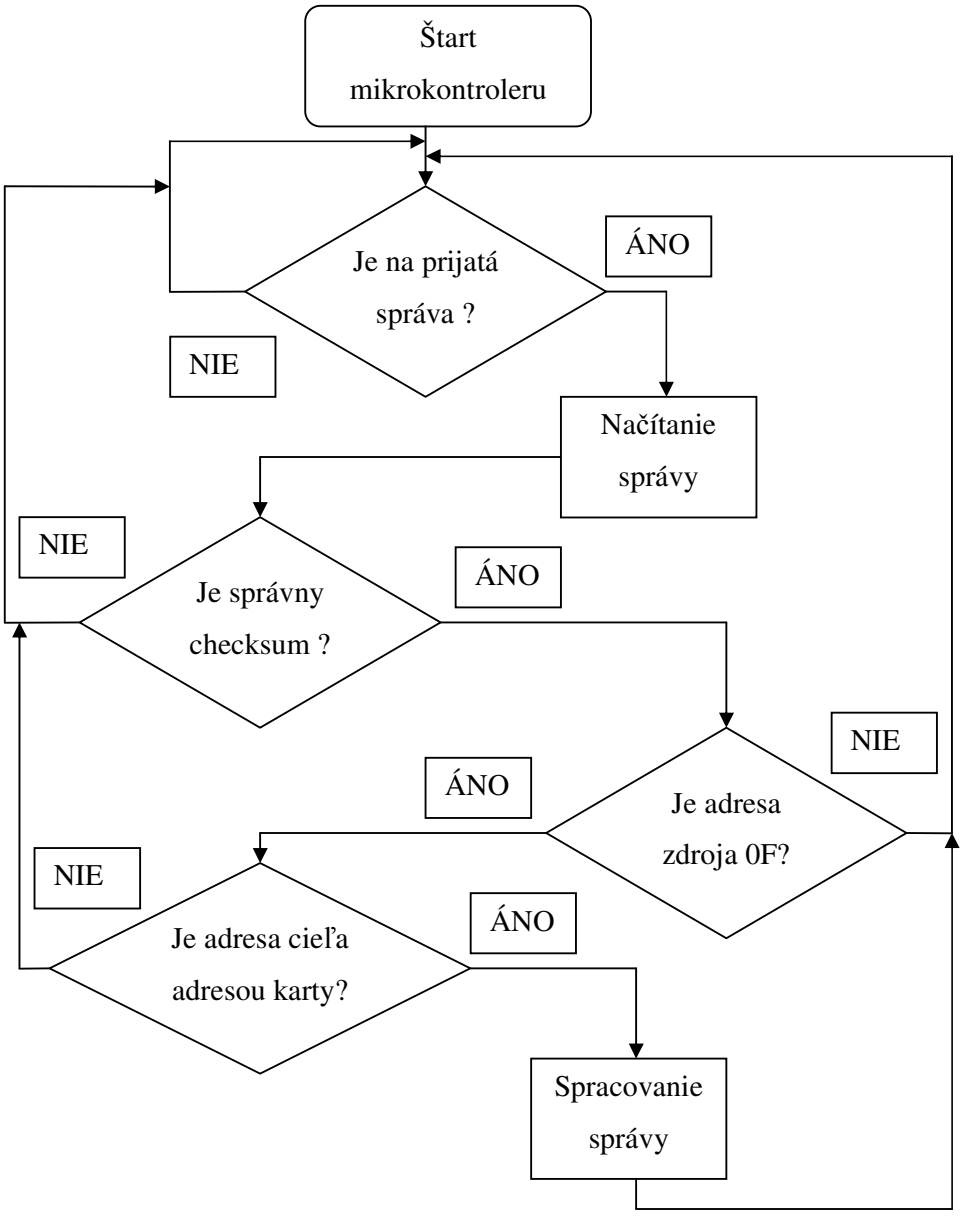
Zdroj	Cieľ	Počet	Data	Checksum
0x3F	0x30	0x32	0x30 0x30	0x30 0x32 0x32 0x34

Obr.14 Formát správy s konkrétnymi hex hodnotami

Na obrázku 14 je formát správy s konkrétnymi hex hodnotami prenášaných bytov správy z príkladu . Správa je vyslaná Mastrom k Slave s adresou 00 a je to požiadavka na sériove číslo multiplexorovej karty.

6.6. SPRACOVÁVANIE SPRÁV

V tejto časti sa budem venovať postupu ako bude mikrokontrolér a riadiaci počítač spracovávať správy. Mikrokontroler ja stále na prijíme a reaguje na prijímané správy. Spracovávanie správ mikrokontrolerom je na obrázku 15.



Obr.15 .Spracovávanie správy mikrokontrolerom

Mikrokontroler po štarte monitoruje prijímane správy. Pokiaľ je zachytená správa kontroluje sa jej kontrólny súčet (checksum). Pokiaľ je kontrola správna správa sa ďalej spracuváva pokiaľ nie tak správa sa ignoruje. Po kontrole súčtu sa kontroluje adresa zdroja ,pokiaľ je adresa 0F ,ynamená to adresu iadiaceho počítača a správa sa

ďalej spracuváva,pokiaľ je adresa iná správa sa ignoruje.Po kontrole adresy zdroja sa kontroluje adresa cieľa . Ak sa adresa zhoduje z adresou karty,správa sa vyhodotí a spracuje ,pokiaľ nie je správa ignorovaná. Po spracovaní správy je vyslaná odpoveď riadiacemu počítaču a mikrokontroler pokračuje v monitorovaní prijímu.

7. PODROBNÝ POPIS SW PRE MIKROKONTROLER

V následujúcej kapitole sa budem zaoberať návrhom a popisom SW pre mikrokontroler na multiplexorovej karte. V SW bude zakomponovaný aj vyššie uvedený protokol.

7.1. MOŽNOSTI VÝVOJA SW PRE MIKROKONTROLER

Pre správnu činnosť mikrokontroleru musíme napísať SW, v ktorom bude zakomponovaný komunikačný protokol ,musí byť schopný riadiť činnosť celej multiplexorovej karty podľa našich požiadavkov. SW môže byť napísaný pod rôznymi vývojovými prostrediami a rôznymi programovacími jazykmi. Kompilátor daného vývojového prostredia musí generovať hex kód ktorý potom programátorom naprogramujeme do mikrokontroleru. Jedným z najpoužívanejších programovacích jazykov ktoré sa používajú pre programovanie mikrokontrolerov je C. Tento jazyk poskytuje omnoho väčší programátorsky komfort ako assembler . V jazyku C budem aj ja písať SW pre multiplexorovú kartu. Ďalšou vecou s ktorou musíme počítať je dostupnosť vývojového prostredia a prekladača pre tento jazyk. Pre mikrokontrolery Atmel je k dispozícii vývojové prostredie s prekladačom WIN AVR, ktoré je popísané v [8]. Tento prekladač je k dispozícii zadarmo a nemá ani žiadne obmedzenia na veľkosť kódu. Kompilátor podporuje aj obvod ATMEGA 8515 ktorý pužívam na multiplexorovej karte. Tento prekladač je k dispozícii aj pre operačný systém Linux ako AVR GCC. K tomuto prekladaču sú k dispozícii aj rôzne knižnice ktoré sa môžu používať bez obmedzenia. Tieto knižnice mi v mnohom uľahčia vývoj SW. Ja budem využívať knižnicu pre ovládanie sériového rozhrania UART.

7.2. NAPROGRAMOVANIE OVLÁDANIA SÉRIOVÉHO ROZHRANIA UART

Ako som už spomínal prekladač WIN AVR má pre komunikáciu cez sériove rozhranie vytvorené knižnice ,ktoré su dostupné po nainštalovaní programu WIN AVR.Práca s týmito knižnicami je popísaná v [8]. V mojom projekte je názov knižníc uart.c a hlavičkový súbor uart.h. S použitím týchto knižníc je možné používať funkcie pre čítanie zo sérioveho portu fgets() a zápis na sériový port funkciu fprintf(). Pred ich použitím je potrebné ich nastaviť aby sa mohli používať ako standard input a standard output. Toto nastavenie urobíme funkciou uart_str :

FILE uart_str = FDEV_SETUP_STREAM(uart_putchar, uart_getchar, _FDEV_SETUP_RW);

Táto funkcia je v definíciach na začiatku súboru main.c. Všetky funkcie ktoré budem popisovať sú súčasťou súboru main.c

Vo funkcii int main() je nastavení pinov naprogramovaný cyklus for ktorý vlastne pracuje ako nekonečný cyklus a skončí sa až pri odpojení napájania alebo vyvolaním prerušenia. Funkcia fgets() pracuje tak že čaká na reťazec na pine RX,tento reťazec musí byť ukončený znakom return (0xOD v hex kóde). Po prijatí tohto znaku vráti prijatý reťazec ktorý sa spracuváva podľa hore uvedeného protokolu.Po spracovani správy sa vyšle odpoveď na pin TX pomocou funkcie fprintf() ,ktorá ma ako parameter výstupný reťazec. Keďže používam poloduplexnú prevádzku RS 485 je pred vysielaním potrebné prepnúť buďič linky RS 485 LTC 485 na vysielanie a po ukončení vysielania znovu prepnúť na príjem. Toto sa ovláda výstupnym pinom PD2.

7.3 PRIJATIE SPRÁVY

Po načítani vstupného reťazca je potrebné zabezpečiť jeho dekódovanie podľa protokolu a určenie či sa jedná o správu pre danú jednotku. Toto prijatie správy ma na starosti funkcia prijatie správy.

```
int PrijatieSpravy(char *ret)
{
int pocet,i,a,b,va,vb;
int back=0;

/* Vynulovanie poľa znakov správa[]  */
pocet=strlen(ret)-1;
for(i=0;i<59;i++)
 sprava[i]=0;

/* Skopýrovanie znakov zo vstupného reťazca do poľa správa[] a ich pretypovanie na
INT */
for(i=0;i<pocet;i++)
  sprava[i]=(int)ret[i];

  a=0;
  b=0;

/* Výpočet kontrólneho súčtu  */
for(i=0;i<(pocet-4);i++)
{
 a=a+sprava[i];
 if(a>255)
  a=a-255;

  b=b+a;
```

```
 if(b>255)
 b=b-255;
}
va=0;
 vb=0;

/* Načítanie kontrólneho súčtu  z prijatej správy */
va=(sprava[pocet-4]- 48)*16 + (sprava[pocet-3]-48);
vb=(sprava[pocet-2]- 48)*16 + (sprava[pocet-1]-48);

 if((va==a)&&(vb==b))
  back=1;

/* Kontrola adresy zdroja */
 if(back==1)
 {
  if(sprava[0]==0x3F)
  back=2;
  }

/* Kontrola adresy cieľa */
  if(back==2)
 {
  if((sprava[1]- 0x30)==adresa)
    back=3;
  }

return back;
}
```

Obr.16 Výpis funkcie Prijatie správy

Táto funkcia je typu Integer a má jeden vstupný parameter ktorým je vstupý reťazec ktorý bol prijatý . Jej úlohou je určiť či je správa korektne prijatá ,to znamená kontrola kontrólneho súčtu a ďalej je určenie či správa pochádza od mastra (adresa 0x3F) a či adresa cieľa je totožná z adresou dannej karty. Vstupná sprava sa najskôr uloží do poľa znakov správa []. Toto poľe je deklarované ako globálna premenná o 60 znakov typu Integer. Najskôr sa v prvom cykle for nastavia všetky prvky poľa na nulu a v druhom cykle for sa do tohto poľa priradia jednotlivé znaky vstupného reťazca,ktory je ale typu char tak sa všetky znaky musia najskôr pretypovať na INT. Dôvod je ten že s typom int sa mi bude lepšie pracovať pri spracovávani správy. Tretí cyklus for vypočíta kontrólny súčet správy podľa vzorca (1.5). Výsledky zapíše do premenných a ,b. Potom sa načítajú do premenných va,ab kotrólne súčty ktoré boli súčasťou správy. Pokiaľ sa rovnaju hodnoty a , va a hodnoty b ,vb ,je kontrólny súčet v poriadku a funkcia môže pokračovať ďalej,ak nie vracia hodnotu nula a ukončí sa. Pokiaľ je kontrólny súčet v poriadku ,kontroluje sa adresa zdroja ,je to prvý prvok poľa správa, a táto adresa musí byť 0x3F ,čo je adresa mastra,keďže podľa protokolu môže karta prijímať správy iba od mastra. Ak nie je to adresa mastra funkcia vráti hodnotu 1 a končí , ak áno pokračuje ďalej a spracuváva 2.prvok poľa ktorý je adresa cieľa ,pokiaľ adresa cieľa nie je totožná s adresou karty , ktorá je v globálnej premennej adresa,funkcia vracia 2 a končí pokiaľ sa adresy zhodujú funkcia vracia hodnotu 3 a končí. Poiaľ funkcia PrijatieSpravy() ,vráti inú hodnotu ako 3,znamená to že správa je buď nekorektná alebo nie je určená pre dannú karty a ďalej sa nespracuváva a čaka sa na ďalšiu správu. Pokiaľ funkcia PrijatieSpravy() vráti hodnotu 3 ,znamená to že správa patrí dannej karte a ďalej sa spracuváva.

7.4 SPRACOVANIE SPRÁVY

Funkcia SpracovanieSpravy() je volaná pokiaľ predchádzajúca funkcia vráti hodnotu 3. Jej úlohou je určiť o aký druh správy sa jedná (IDN,OUT,ADR,STA) . Telo funkcie je veľmi jednoduché a je na obrázku 17.

```
int SpracovanieSpravy(void)
{
int i;

/* Vynulovanie poľa znakov pre odpoveď */
for(i=0;i<30;i++)
 odpoved[i]=0;

/* Korola znaku určujúceho typ správy */
 switch(sprava[3])
 {
   case 0x30 :
      return 1;
          break;
  case 0x31:
        return 2;
         break;
   case 0x32:
         return 3;
       break;
   case 0x33:
          return 4;
      break;
 }
return 0;
}
```

Obr.17 . Vypis funkcie SpracovanieSpravy()

Táto funkcia je typu int a nemá parametre,vracia hodnoty podľa typu správy.

Prvý cyklus for vynuluje poľe znakov odpoved[]. V tomto poli sa budú ukladať znaky pre odpoveď karty Mastrovi , na základe prijatej správy. Funkcia využíva to že je 4 prvok poľa sprava určuje typ príkazu a podľa typu spríkazu vracia hodnotu. Ak je vrátena hodnota 1 je to tpríkaz IDN,ak je hodnota 2 ,ide o príkaz OUT, ak je vrátena hodnota 3 ide o príkaz ADR,ak je hodnota 4 ide o príkaz STA. Ďaľším krokom je spracovanie jednotlivých príkazov.

7.5 FUNKCIA IDN()

Táto funkcia musí vrátiť sériove číslo karty. Kód tejto funkcie je na obrázku 18.

```
int IDN(void)
{
 int i=0;

/* Korola parametru 0x30 */
 if(sprava[4]==0x30)
   {
        odpoved[0]=adresa+0x30;
        odpoved[1]=0x3F;
   odpoved[2]=0x36;
        odpoved[3]=0x30;

        for(i=0;i<5;i++)
         odpoved[4+i]=(int)serial[i];

   OutChecksum(9);

        return 1;
        }
return 0;
```

```
}
```

Obr.18 .Funkcia IDN

Táto funkcia je typu Integer a nemá žiadne parametre. Skontroluje či 5.znak má hodnotu 0x30,ak nie funkcia končí a vracia hodnotu 0. Ak je znak v poriadku nastaví do poľa odpoved jednotlivé prvky. Poľe odpoveď je typu Integer a zapisujú sa do neho všetky znaky ktoré sa vyšľú na TX pin mikrokontroleru. Prvou hodnotou je podľa protokolu adresa zdroja,v tomto prípade adresa karty + 0x30. Druhá hodnota je adresa cieľa ,čo je adresa mastra ktorá je 0x3F. Tretia hodnota je počet znakov,ktorý u tohto príkazu bude 6 ,teda zapísana hodnota bude 0x36. Štvrtý znak bude hodnota príkazu IDN ,teda 0x30. Potom nasledujú jednotlivé znaky sériového čísla v ASCII kóde. Sériove číslo sa bude skladať z piatich znakov.Prvý bude písmeno M a ďaľšie č4 naku sú čísla 0001 až 9999, ktoré bude mať každá karta iné. Toto sériove číslo bude uložené v globálnej premennej serial. Na záver je zavolaná funkcia OutChecksum(). Táto funkcia má jeden parameter ktorým je číslo pozície na ktoré sa v poli odpoved zapíšu hodnoty vypočítaneho kontrólneho súčtu.

7.6 FUNKCIA OUT

Táto funkcia po spracovaní príkazu OUT nastaví jednotlivé výstupné piny na základe spracovaného príkazu. Telo funkcie je na obrázku 19.

```
int OUT(void)
{
 int i,j,p;

/* Zapísanie hodnoty výstupných pinov do poľa obraz */
 for(i=0;i<30;i++)
  obraz[i]=t[i];
```

```
/* Vynulovanie poľa výstupných pinov */
 for(i=0;i<30;i++)
 {
   t[i]=0;
 }

/* Nastavovanie výstupných pinov podľa príkazov */
 for(i=0;i<7;i++)
 {
 p=0;
 p=sprava[i+4]-0x30;

  if(p>7)
      {
     for(j=0;j<4;j++)
      {
      t[4*i+(3-j)]=p % 2;
             p=p/2;
             }
            }
 if((p<8)&&(p>3))
  {
 for(j=0;j<3;j++)
      {
      t[4*i+(3-j)]=p % 2;
             p=p/2;
             }
           t[4*i]=0;
  }
    if((p<4)&&(p>1))
  {
```

```
 for(j=0;j<2;j++)
    {
    t[4*i+(3-j)]=p % 2;
          p=p/2;
          }
         t[4*i]=0;
         t[4*i +1]=0;
  }
}

/* Nastavenie pinov T29 a T30 */
 if(sprava[11]-0x30 == 3)
 {
  t[28]=1;
  t[29]=1;
 }

 if(sprava[11]-0x30 == 2)
 {
  t[28]=1;
  t[29]=0;
 }

  if(sprava[11]-0x30 == 1)
 {
  t[28]=0;
  t[29]=1;
 }

   if(sprava[11]-0x30 == 0)
 {
```

```
  t[28]=0;
  t[29]=0;
 }

 NastaveniePortov();

/* Nastavenie odpoveďe po spracovaní príkazu */
 odpoved[0]=adresa+0x30;
          odpoved[1]=0x3F;
    odpoved[2]=0x31;
          odpoved[3]=0x31;

 OutChecksum(4);

return 1;
}
```

Obr.19. Výpis funkcie OUT

Hodnoty jednotlivých pinov T1 až T30 sú v poli t[]. Pokiaľ je pin zopnutý tak hodnota prvku je 1 ak nie tak je hodnota prvku 0. V prvom cykle for sa presunie obsah ttohto poľa do poľa obraz. Je to preto lebo pole obraz sa bude porovnávať s poľom t po spracovaní príkazu OUT a pokiaľ sa budú prvky líšiť znamena to že sa zmenol počet zopnutí jednotlivého relé a zmení sa hodnota štatistiky jednotlivého výstupu a tým aj relé. V ďaľšom cykle for sa spracuvávaju jedntotlivé argumenty príkazov OUT. Spracuváva sa 8 argumentov. Prvý nastavuje výstupy T1 až T4 ,druhý T5 až T8 a tak ďalej. Keďže každý znak je prenášaný ako cele číslo plus hodnota 0X30 , toto číslo sa najskôr odpočíta a potom sa delí ako pri prevode do celého čísla do dvojkovej sústavy. Ostane nám tak 4 bitové číslo v dvojkovom kóde kde 1 zodpovedá zopnutému výstupu a 0 vypnutému výstupu. Tito hodnoty sa zapíšu do poľa výtupov t. Potom sa volá funkcia NastaveniePortov(), ktorá na zaklade poľa

t nastaví výstupné piny. Na záver funkcie sa nastaví odpoveď na spracovaný príkaz ako v predchadzajícivh prípadoch.

7.7 FUNKCIA ADR

Táto funkcia slúži na zmenu Slave adresy dannej karty. Telo funkcie je veľmi jednoduché a je na obrázku 20

```
int ADR(void)
{
int oldadr;

oldadr=adresa;
 adresa = sprava[4]-0x30;

 odpoved[0]=adresa+0x30;
 odpoved[1]=0x3F;
 odpoved[2]=0x31;
 odpoved[3]=0x32;

  OutChecksum(4);
return 1;
}
```

Obr.20 Funkcia ADR.

Táto funkcia iba zmení premennu adresa na adresu ktorú dostane ako parameter. Po tom pripravý odpoveď ,kde v odpovedi používa ako adresu zdroja ešte starú adresu. Po tom sa však už musí komunikovať s kartou cez novú adresu.

7.8 FUNKCIA STA

Táto funkcia ma za úlohu získavať štatistické údaje o počte zopnutí jednotlivých výstupov a nulovať štatistické údaje jednotlivých vstupov. Tento príkaz má 2 parametre a to 0x30 ,pri ktorom sa načítavajú údaje o počte zopnutí vstupu ktorý je uvedený ako ďalšie dva parametre príkazu. Parameter 0x31 vynuluje výstup určený Ďalšími parametrami. Štatistické údaje sú uložené v poli znakov stat. Každý výstup má 2 prvky integer ktoré sú 16 bitové. Toto pole sa napĺňa pri spustení mikrokontroleru s EEPROM a aktualizuje sa pri každom príkaze OUT. Pri vypnutí mikrokontroleru sa toto pole ukladá do EEPROM. Spracovanie správy pri parametri 0x30 je na obrázku 21.

```
cislo=(sprava[5]-48)*10 + (sprava[6]-48);

 pocet=stat[2*cislo - 2];
 if(stat[2*cislo-1] != 0)
  pocet=pocet*stat[2*cislo -1];

 odpoved[0]=adresa+0x30;
 odpoved[1]=0x3F;
 odpoved[2]=0x39;
 odpoved[3]=0x33;
 odpoved[4]=sprava[5];
 odpoved[5]=sprava[6];

 ltoa(pocet,pom2,10);

 if(pocet<10)
 {
 odpoved[6]=0x30;
 odpoved[7]=0x30;
 odpoved[8]=0x30;
```

```
 odpoved[9]=0x30;
 odpoved[10]=0x30;
 odpoved[11]=(int)pom2[0];
 }
```

Obr.21 Spracovanie správy pri parametri 0x30

V prom riadku sa získava číslo výstupneho pinu z prijatej správy a to tak že prvý argument čísla znaku sa vynásobi 10 , keďže sa číslo každého výstupu prenáša po jednotlivých čísliciach v ASCII znakoch a prvý znak sú desiatky.K nemu sa pripočíta druhý argument čísla znaku a dostaneme tak číslo digitálneho výstupu. Keďže v poli stat má štatistika 2 prvky a to zapísane tak že prvý prvok má čísla od 0 do 10000 ak je hodnota 10000,prvok sa vynuluje a druhému sa pripočíta jednotka. ˇudaj o celkovom počte zopnutí sa uloží v premennej pocet. Potom nasleduje funkcia ltoa() ktorá prevedie hodnotu počet na desiatkové číslo a uloží ho do reťazca pom2. Túto hodnotu potom zapíšeme do poľa odpoved kde sa ostatné prvky tohto poľa nastavujuú ako v ostatných funkciach. Pokiaľ je parameter príkazu OUT 0x31,je potrebné vynulovať prvok poľa stat prisluchajúci dannému výstupu. Fragment kódu pre nulovenie je na obrázku 22.

```
cislo=(sprava[5]-48)*10 + (sprava[6]-48);

stat[2*cislo - 2]=0;
stat[2*cislo - 1]=0;
```

Obr.22 Nulovanie prvku poľa stat.

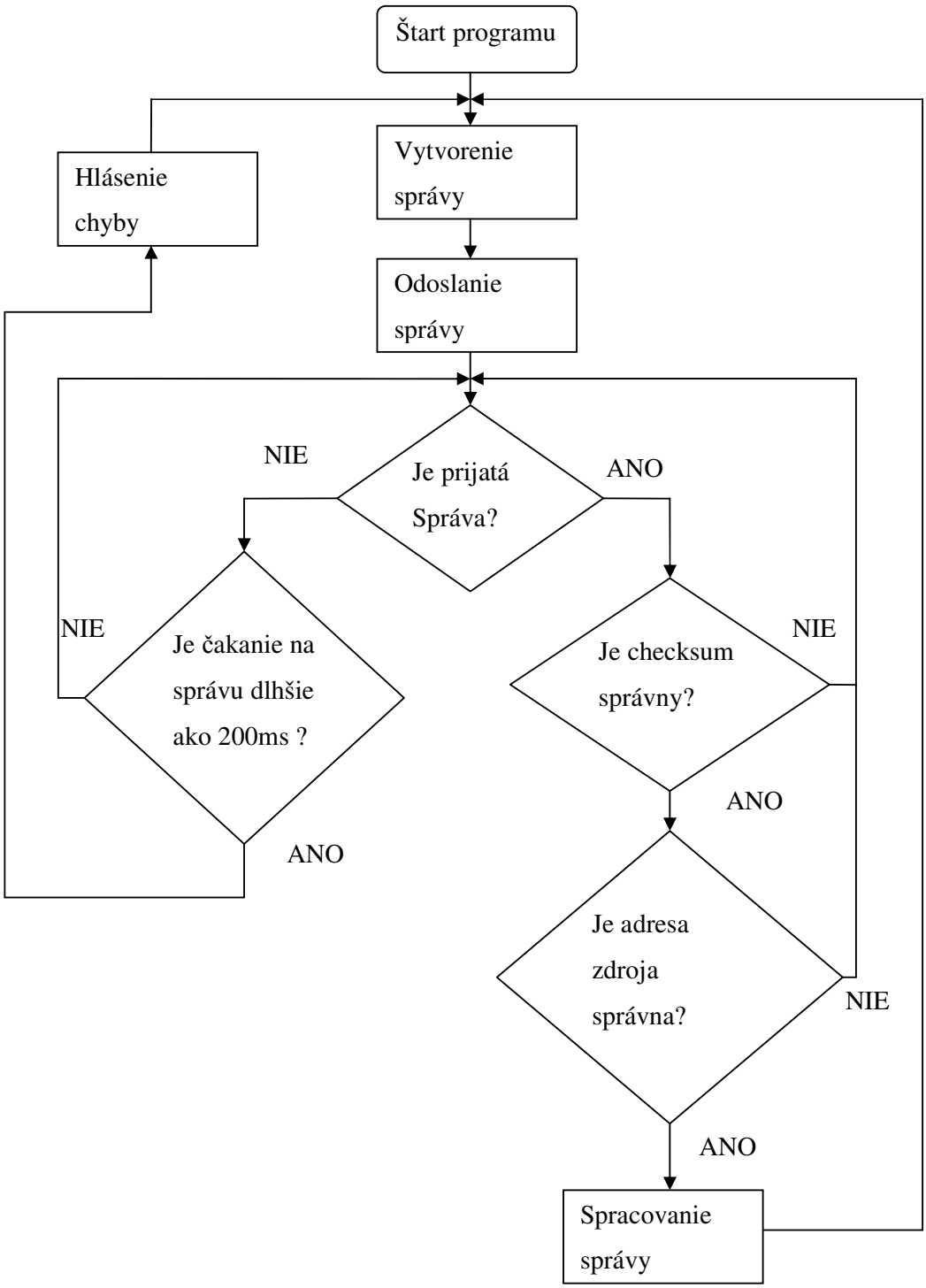
V tejto časti som sa snažil vysvetliť dôležité častí kódu pre mikrokontroler. Celý zdrojový text je v programe main.c v adresáry SW/AVR na priloženom CD.

8. POPIS PROGRAMU PRE RIADIACI POČÍTAČ

V tejto časti sa budem zaoberať popisom programu pre riadiaci počítač .Tento program bude riadiť činnosť jednotlivých kariet podľa protokolu uvedeného v časti 6.

8.1 VÝVOJOVÝ DIAGRAM PRE VYSIELANIE A SPRACOVÁVANIE SPRÁV MIKROKONTROLEROM

Na obrázku 23 je vývojový diagram spracovania správy riadiacim počítačom. Po vytvorení správy podľa protokolu sa správa odošle na sériový port a potom čaká na odpoveď od mikrokontroléru. Odpoveď musí prísť 200ms od poslania správy,ak nepríde tak riadiaci počítač vyhodnotí komunikačnú chybu . Pokiaľ je prijatá správa skontroluje sa kontrólny súčet správy ak nie je v poriadku čaká sa na ďaľšiu správu.Ak je kontrólny súčet v poriadku ,skontroluje sa adresa zdroja správy ,ktorá sa musí rovnať adrese cieľa správy ktorá bola vyslaná riadiacim počítačom. Ak sa adresy nerovnajú čaká sa na ďaľšiu správu ,ak sa adresy rovnaju prijatá správa sa môže spracovať.



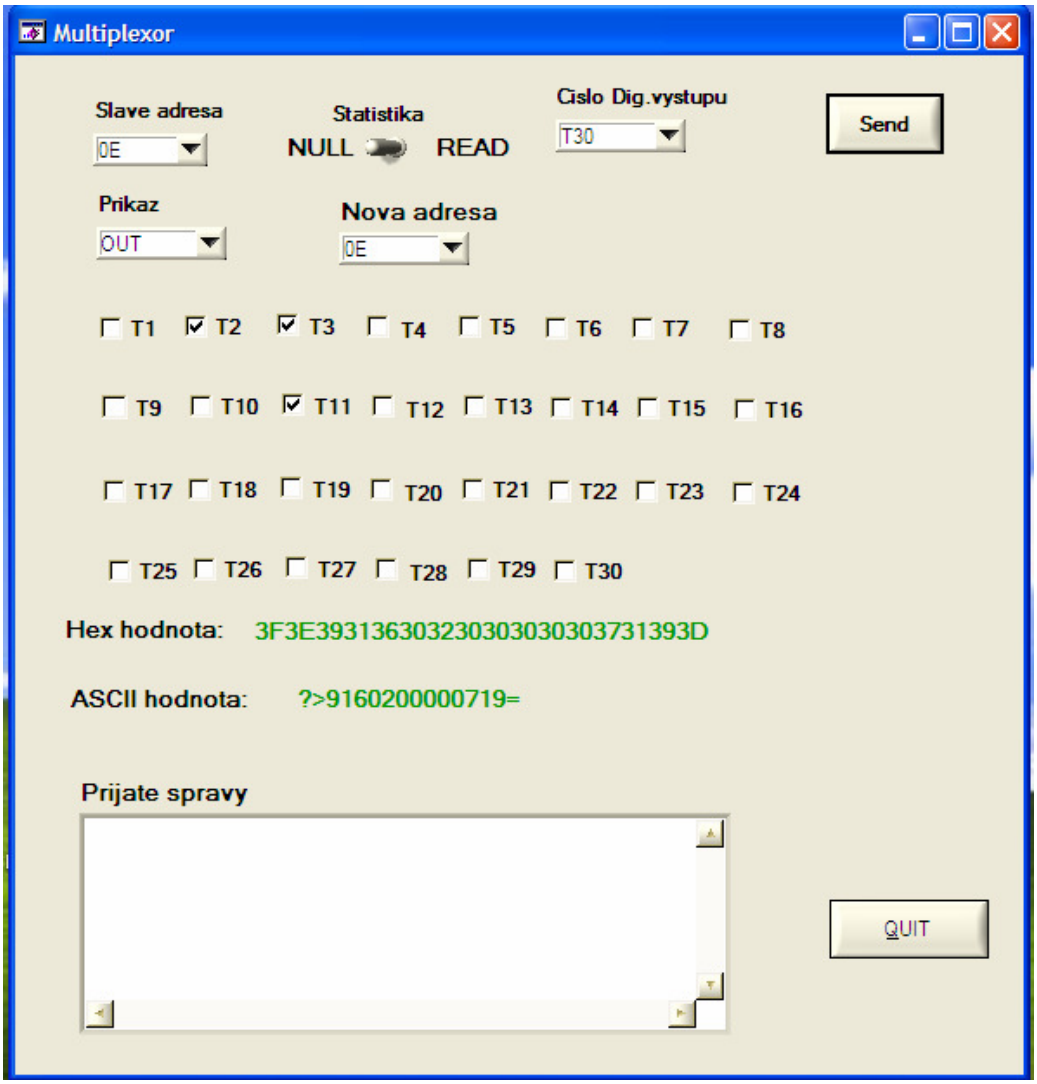
Obr.23. Vývojový diagram spracovanie správ riadiacim počítačom.

8.2. KOMUNIKAČNÝ PROGRAM PRE RIADIACI POČÍTAČ

Riadiaci počítač sa pripojuje k zbernici RS 485 prostredníctvom USB/RS485 prevodníku. V tomto prevodníku je použitý integrovaný obvod FT232. Tento obvod má 2 typy driverov ktoré sa nainštalujú do počítača. Jeden sa nainštaluje ako virtuálny sériovy port,kde sa naň pristupuje ako na klasický sériovy port,druhý typ driveru sa naištaluje ako osobitné USB zariadenie a potom sa naň pristupuje pomocou knznice dodanej výrobcom. V ukážkovom príklade budem používať virtuálny sériovy port. Ako vývojove prostredie použijem komerčný prekladač firmy National instruments Lab Window CVI. Demo verziu si je možné stiahnuť z [9]. Toto vývojove prostredie je určené pre vytváranie SW pre rôzne meracie systémy.Má množstvo knižníc na spoluprácu s rôznymi prístrojmi cez rôzne rozhrania. Veľmi jednoduchá je aj práca zo sériovými portami. Pre otvorenie sérioveho portu je jednoduchá funkcia OpenComConfig (), ktorá ma parametre číslo portu,prenosovú rýchlosť ,paritu,stop bit. Pre zatvorenie sérioveho portu sa používa funkcia CloseCom (),ktorá ma ako parameter číslo sérioveho portu. Funkcie na čítanie a zápis na sériovy port su taktiež veľmi jednoduché. Pre zápis na sériovy port sa používa funkcia ComWrt () ktorá ma parametre číslo portu ,reťazec znakov ktoré treba na port zapísať a dĺžku zapisovaného reťazca. Na čítanie zo sérioveho portu sa používa funkcia ComRd () ktorá má parametre číslo portu ,pole znakov kde sa zapíše prijímany reťazec a počet prečítanych znakov.

8.3 POPIS PROGRAMU

Program pre riadiaci počítač je napísany v jazyku c,celý projekt sa skladá zo súborov main.c,main.h,multiplexor.c, multiplexor.h.,main.uir. Súbor main.uir je uložený hlavný panel programu zo všetkými ovládacimi prvkami. Ich spracovanie je napísane vo funkcii main.c. kde sa volajú ostatné funkcia ktoré su hlavne umiestnené v súbore maultiplexor.c



Obr.24 Panel testovacieho programu pre ovládanie multiplexorovej karty

Na obrázku 24 je hlavný panel ovládacieho programu. V menu Slave adresa sa nastaví adresa karty s ktorou chceme komunikovať. Táto adresa môže mať rozsah 00 až 0E. V menu Prikaz si vyberieme príkaz ktorý chceme karte poslať ,teda IDN,OUT,ADR,STA. Pokiaľ si vyberieme príkaz IDN nenastavuje sa už žiadny parameter a môžeme poslať správu kliknutím na tlačidlo Send. V políčku Hex hodnota sa zobrazí hex hodnota celej správy a v políčku ASCII hodnota sa zobrazí ASCII hodnota celej správy. V políčku Prijate spravy sa zobrazia správy prijaté od

multiplexorových kariet. Ak zvolíme príkaz OUT,musíme nastavť checkboxi T1 až T30.Pokiaľ je odškrtnutý príslušný výstup sa zopne ak nie je výstup sa vypne. Pokiaľ vyberieme príkaz ADR,musíme nastaviť v políčku Nova adresa,novú Slave adresu multiplexorovej karty. Ak si zvolíme príkaz STA musíme v menu Cislo Dig.vystupu nastaviť číslo digitálneho výstupu z ktorého chceme čítať alebo nulovať štatistiku a musíme prepínačom Statistika nastaviť či chceme čítať alebo nulovať štatistiku danného digitálneho výstupu.

8.4 POPIS JEDNOTLIVÝCH FUNKCII PROGRAMU

V tejto časti popíšem jednotlivé funkcie ktoré su v súbore multiplexor.c . Týmito funkciami sa ovláda celá multiplexorová karta, komunikácia, tvorba príkazov a iné.

int OpenRS485(int portX) . Táto funkcia má jeden parameter typu integer a to je číslo virtuálneho sérioveho portu v počítači. Funkcia otvorí virtuálny sériovy port s požadovanými parametrami. Táto funkcia sa musí zavolať predtým ako chceme na tento port zapisovať ,prípadne s neho čítať. Táto funkcia vracia 1 pokiaľ otvorenie virtuálneho portu prebehlo úspešne, pokiaľ sa vyskytne nejaká chyba vracia 0.

int CloseRS485() . Táto funkcia ukončí komunikáciu a zatvorí virtuálny sériovy port. Nemá žiadne parametre.Predtým ako sa volá táto funkcia musí byť otvorený virtuálny sériovy port. Pokiaľ zatváranie bolo úspešné vracia 1 ,pokiaľ port nebol otvorený vracia 0.

char *bitTohex(char *vstup,int b1,int b2,int b3,int b4). Táto funkcia slúži na vytvorenie hex hodnoty ako znakového reťazca zo 4 hodnôt bitov typu integer. Tieto hodnoty môžu byť 1 alebo 0. Táto funkcia má 5 parametrov. 1.parameter je adresa reťazec návratovej hodnoty. Druhý až piaty parameter je typu integer a nastavuje sa s nímjednotlivé digitálne výstupy podľa protokolu pre príkaz OUT. Pokiaľ je hodnota 0 nastaví sa požadovaný výstup na 0,pokiaľ sa hodnota nerovná nule nastaví sa výstup na 1. Príklad: Ak máme parametre b1=0, b2=1, b3=0, b4=1, funkcia vráti reťazec char “35“. To je hodnota 5 plus 0x30 hex. Táto funkcia sa volá 8 krát na nastavenie všetkých 30 digitálnych výstupov.

char *prikaz(char *vstup,char *adr,char *prik,char* param) Táto funkcia vytvorí zo vstupných parametrov celú správu ako hex hodnotu vo formáte znakového reťazca podľa protokolu a vráti ju ako návratovu hodnotu. Prvy parameter je návratový reťazec, Druhý je adresa karty kam chceme správu posielať v hex formáte uloženom ako reťazec znakov s pripočítanou hodnotou 0x30. Napríklad adresa 03 bude vo formáte "33". Tretí parameter je typ príkazu ,teda OUT,ADR,IDN,STA. Môžeme ich priamo za zapísať vo formáte OUT,ADR a pod. V súbore multiplexor.h je makro ktoré ich prevedie do správneho formátu. Štvrtý parameter je parameter príkazu ktorý sa zadáva ako reťazec znakov. Napríklad ak je parameter 0x30 bude vo formáte "30".

char *hexToAscii(char* vstup) . Táto funkcia má za úlohu na základe vstupného parametra ktorým je reťazec znakov v hex formáte vrátiť reťazec znakov ASCII . Tento reťazec ASCII znakov môžeme poslať na virtalny sériovy port. Príklad : Vstupná hodnota je "30" . Funkcia vráti hodnotu "0".

int SendMes(char *retazec) . Funkcia má jeden vstupný reťazec, čo je správa v ASCII formáte , ktorá sa zapisuje na virtuálny sériový port. Táto funkcia ešte pripojí na koniec reťazca znak ENTER (0x0D v hex formáte).

char *ReadMes(char* vstup) . Táto funkcia vracia reťazec ktorý bol prijatý na virtuálny sériový port. Parameter vstup je adresa návratovej hodnoty.

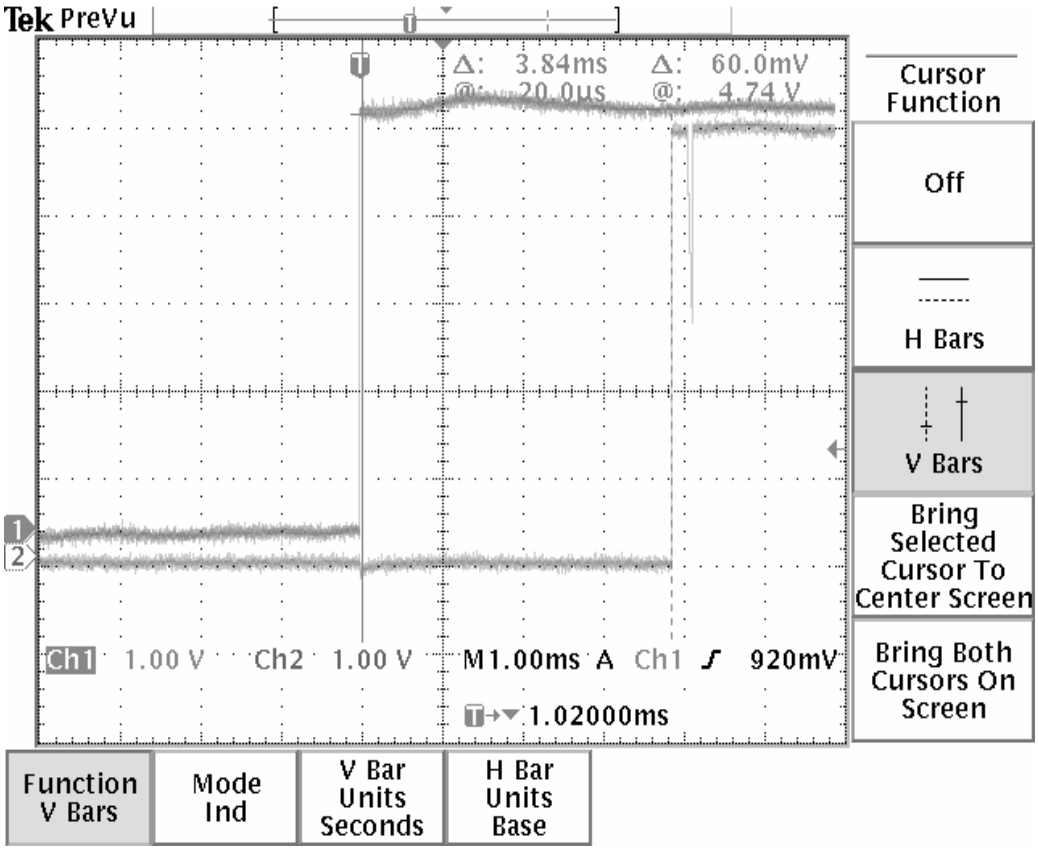
char *dekod(char* adr,char *vstup). Funkcia má dva vstupné parametre ,parameter adr je adresa karty od ktorej čakáme odpoveď. Parameter vstup je hodnota vstupného reťazca ktorý bol prijatý na virtuálny sériovy port. Funkcia správu dekóduje.Pokiaľ sa zhoduje kontrólny súčet, adresa zdroja a adresa cieľa , tak sa správa spracuje a vráti odpoveď , čiže dátove pole. Pokiaľ sa nezhoduje kontrólny súčet, adresa zdroja alebo cieľa tak funkcia vráti prázdny reťazec.

V tejto časti som chcel stručne popísať všetky funkcie ktoré používa program pre riadiaci počítač . Všetky tieto funkcie su v súbore multiplexor.c a ich volanie je možné cez hlavičkový súbor multiplexor.h . V súbore main.c sú teito funkcie volané v závislosti na ovládacich prvkov panelu. Súbory multiplexor.c a multiplexor.h môžeme použiť pri vývoji SW pre testovací systém ich zakomponovaním do projektu a ich volaním podľa pravidiel popísaných v tejto časti. Pokiaľ by sa

nepoužilo vývojové prostredie Lab Windows CVI je potrebné zmenť telá funkcii OpenRS485(int portX), CloseRS485(void), SendMes(char *retazec), ReadMes(char* vstup), v závisloti od práce z virtuálnym sériovým portom jednotlivých vývojových prostredí.

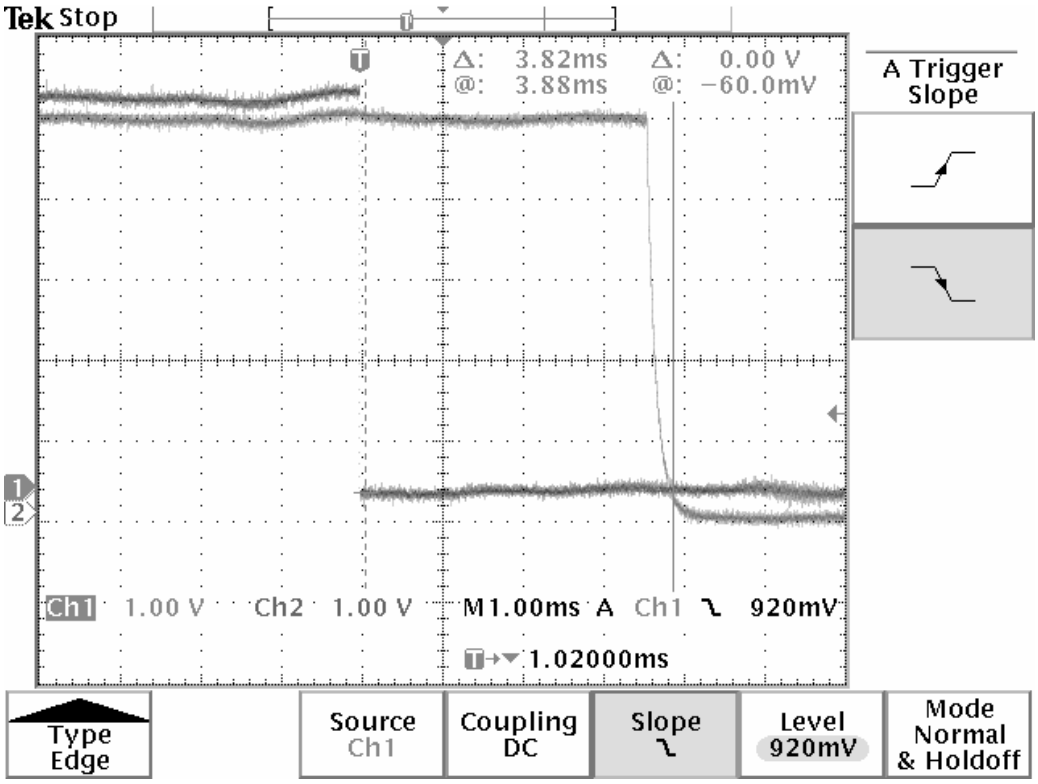
9. MOŽNOSTI MULTIPLEXOROVEJ KARTY

Táto multiplexorová karta sa dá použiť v rôznych typoch testovacích systémov.Je možné pripojiť na jednu zbernicu až 15 multiplexorových kariet, čím môžeme ovládať 450 digitálnych výstupov. Uvedený počet je postačujúci aj pre veľmi náročné a zložité funkčné testy osadených dosiek plošných spojov. Pri použití knižníc uvedených v časti 8 je možné zakomponovať ju do rôznych vývojových prostredí. Pri dodržaní uvedeného protokolu sa dá napísať SW aj pre iné programovacie jazyky. Je možné používať aj iné operačné systémy. Obvod FT 232 má drivere pre rôzne operačné systémy, Windows, Linux,Unix, MAC OS. Pre činnosť karty je dôležitá doba zopnutia a vypnutia jednotlivých relé. Táto doba je závisla na dobe prenosu zprávy ,ktorá závisi od prenosovej rýchlosti, doba zpracovania správy mikrokontrolerom , doba zopnutia tranzistora a doba zopnutia relé. Na obrázku 25 je nameraná doba zopnutia tranzistora a relé. Bolo to merané osciloskopom Tektronix TDS 3054B. Kanál 1 bol privedený z výstupu mikrokontroleru a kanál 2 bol pripojený na spínaci kontakt relé. Druhý pol spínacieho kontaktu bol pripojený na +5V.



Obr.25 Oneskorenie zopnutia kontaktu relé oproti výstupu mikrokontroléru

Doba oneskorenia zopnutia spínacieho kontaktu relé oproti výstupu z mikrokontroleru bola 3,84 ms. Pri zohľadnení prenosovej rýchlosti a doby spracovania správy v mikrokontrolere je treba počítať s oneskorením asi 5ms. Pri návrhu SW pre testovací systém treba s týmto oneskorením počítať. Ďalším dôležitým parametrom multiplexorovej karty je doba vypnutia relé. Túto dobu vypnutia som meral pri rovnakom zapojení ako dobu zopnutia v predchádzajúcom prípade, teda 1.kanál zapojený na výstupe mikrokontroleru a 2. kanál na spínacom kontakte relé. Výsledok ja na obrázku 26.



Obr.26 Oneskorenie vypnutia relé oproti výstupu z mikrokontroleru

Z obrázku 26 vidno že doba vypnutia relé meraná oproti výstupu mikrokontroleru bola 3,82 ms čo bola takmer rovnaká doba ako doba zopnutia. Pri návrhu zdroja pre multiplexorovú kartu treba počítať s maximálnym možným odberom,ktorý pri zopnutí všetkých relé. Tento odber pri napájacom napätí 12V bol nameraný 750mA. Teda napájaci zdroj pre jednu kartu musí byť 12 V a minimálne 750mA. Pri použití viacerých multiplexorových kariet sa prúdovy odber násobi počtom zapojených multiplexorových kariet.

10 ZÁVER

Cieľom tejto bakalárskej práce bolo navrhnúť diagnostický systém pre testovanie osadených dosiek plošných spojov zo zameraním sa na HW a SW časť multiplexoru stimulačných signálov. V prvej časti som sa zameral na ukážku zostavenia automatizovaného testovacieho systému pre testovanie konkrétnej dosky z praxe. Z toho bolo vidno že dôležitou časťou testovacieho systému je multiplexor. Je možné použiť meraciu ústredňu prípadne viackanálovu analógovu kartu ale v tom prípade je potrebné taktiež používať viackanálový napájaci zdroj v prípade nutnosti pripájať napätia na rôzne miesta, čo je pri zložitejších testoch náročne na počet prístrojov a ich ich zložitosť. V prípade použitia multiplexorových kariet je možné použiť len pár prístrojov a výstupy prípadne vstupy prepínať pomocou multiplexoru. Uvedená karta má releové výstupy , čo sú mechanické kontakty takže treba dodržiavať maximálne prúdy a napätia udávané výrobcom. Najlepšie je digitálne výstupy spínať a vypínať bez napätia čo je možné ošetriť v SW pre merací systém keďže aj zdoje napätia sú riadenie SW. Ďalšou vecou čo je potrebné dodržiavať je maximálny počet zopnutí udanný výrobcom. Uvedná karta má počítanie zopnutia jednotlivých výstupov,takže pri určitom počte zopnutí je možné nastaviť upozornenie pre operátora ktorý môže požadované relé vymeniť. SW časť návrhu umožňuje bezproblémove zakomponovanie ovládačov a hlavičkových súborov do rôznych typov vývojových prostredí na báze C jazyka. V prípade použitia iných jazykov je možné podľa popísaneho protokolu vytvoriť vlastné knižnice poďľa potreby vývojara.

11.ZOZNAM POUŽITEJ LITERATÚRI

[1]. David Matoušek : USB prakticky 1.díl

[2] .FTDI Chip Datsheet FTDI 232 http://www.ftdichip.com/FTProducts.htm

[3]. ATMEL Datasheet ATMEGA 8515

http://www.atmel.com/dyn/products/datasheets.asp?family_id=607

[4]. MAXIM Datasheet MAX 690

http://www.maxim-ic.com/quick_view2.cfm/qv_pk/1334

[5].Doc.Dr.Ing.Miroslav Patočka, František Burian : Sbírka řešených příkladu z řídici elektroniky (2004)

[6].Doc.Ing.František Zezulka,CsC :Prostředky prúmyslové automatizace

[7]. RFC 1145 http://www.faqs.org/rfcs/rfc1145.html

[8]. Win AVR http://winavr.sourceforge.net/

[9]. National insruments http://www.ni.com/