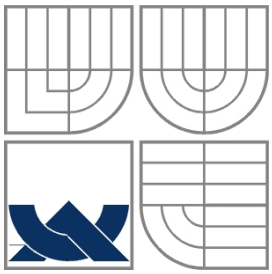
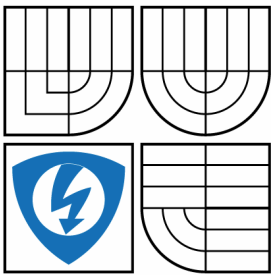


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

PRŮMYSLOVÝ STAVOVÝ INDIKÁTOR

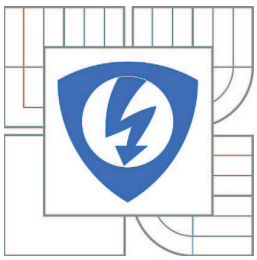
INDUSTRIAL STATE INDIKATOR

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. MARTIN ZAMRZLA
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. ZDENĚK BRADÁČ, Ph.D.
SUPERVISOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student:	Bc. Martin Zamrzla	***ID:***	98399
Ročník:	2	***Akademický rok:***	2010/2011

NÁZEV TÉMATU:

Průmyslový stavový indikátor

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte koncepci stavového průmyslového indikátoru pro zobrazování průmyslových dat vybaveného rozhraním RS232/RS485 a sadou digitálních vstupů a výstupů. Vybavte mikrokontrolérem řady I51. Navrhněte elektroniku indikátoru, navrhněte a realizujte příslušné DPS, osaďte je a oživte elektroniku modulu. Vybavte programovým vybavením a otestujte správnou funkci HW i programového vybavení.

DOPORUČENÁ LITERATURA:

Pavel Herout: Učebnice jazyka C, KOPP, 2004, IV. přepracované vydání, ISBN 80-7232-220-6
Dle pokynů vedoucího práce.

Termín zadání:	7.2.2011	***Termín odevzdání:***	23.5.2011

Vedoucí práce: doc. Ing. Zdeněk Bradáč, Ph.D.

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

UPOZORNĚNÍ:

Abstrakt

Tato práce se zabývá studiem materiálů potřebných pro návrh a realizaci průmyslového stavového indikátoru. V teoretické části práce jsou postupně popsány veškeré použité komponenty včetně potřebných výpočtů, dále jsou popsány také komunikační rozhraní použité na indikátoru. Po teoretické části následuje popis protokolu MODBUS. V praktické části je nejprve popsána samotná realizace spolu s oživením celého indikátoru, poté je podrobně popsané programové vybavení mikrokontroléru a na závěr je uveden popis testovací aplikace.

Klíčová slova

Mikrokontrolér, LCD, MODBUS protokol

Abstrakt

This thesis deals with study of materials needed for proposing and implementation of the industry state indicator. In the theoretical part there are describe all used components including calculations. Then there is a description of the communication interface used on the indicator. The theoretical part is then followed by a description of MODBUS protocol. The practical part is divided into a few sections. First one decribes the implementation itself together with the idicator activation. Second one describes the programme equipment of microcotroller. The conclusion is about description of the application testing.

Keywords

Microcontroller, LCD, MODBUS protocol

Bibliografická citace:

ZAMRZLA, M. *Průmyslový stavový indikátor*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2011. 111s. Vedoucí diplomové práce byl doc. Ing. Zdeněk Bradáč, Ph.D.

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma Průmyslový stavový indikátor 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 diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové 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 části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.“

V Brně dne: **23. května 2011** ……
podpis autora

Poděkování

Děkuji vedoucímu diplomové práce doc. Ing. Zdeňku Bradáčovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce.

V Brně dne: **23. května 2011** ……
podpis autora

1 OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK

ASCII – American Standard Code for Information Interchange
CRC – Vycliv Redundary Check
EEPROM – Electrically Erasable and Programmable Read Only Memory
FLASH – Flash EEPROM
GND – Ground
HW – Hardware
ISP – In-System Programming
LCD – Liquid Crystal Display
LED – Light Emitting Diode
LSB – Lest Significant Bit
MISO – Master Input Slave Output
MOSI – Master Output Slave Input
MSB – Most Significant Bit
PLC – Programmable Logic Controler
RAM – Random Access Memory
RTU – Remote Terminal Unit
RXD – Receive Data
SCLK – Serial Clock
SPI – Serial Peripheral Interface
SS – Slave Select
TTL – Transistor-Transistor Logic
TXD – Transmitt Data
UART – Universal Asynchronous Receiver-Transmitter

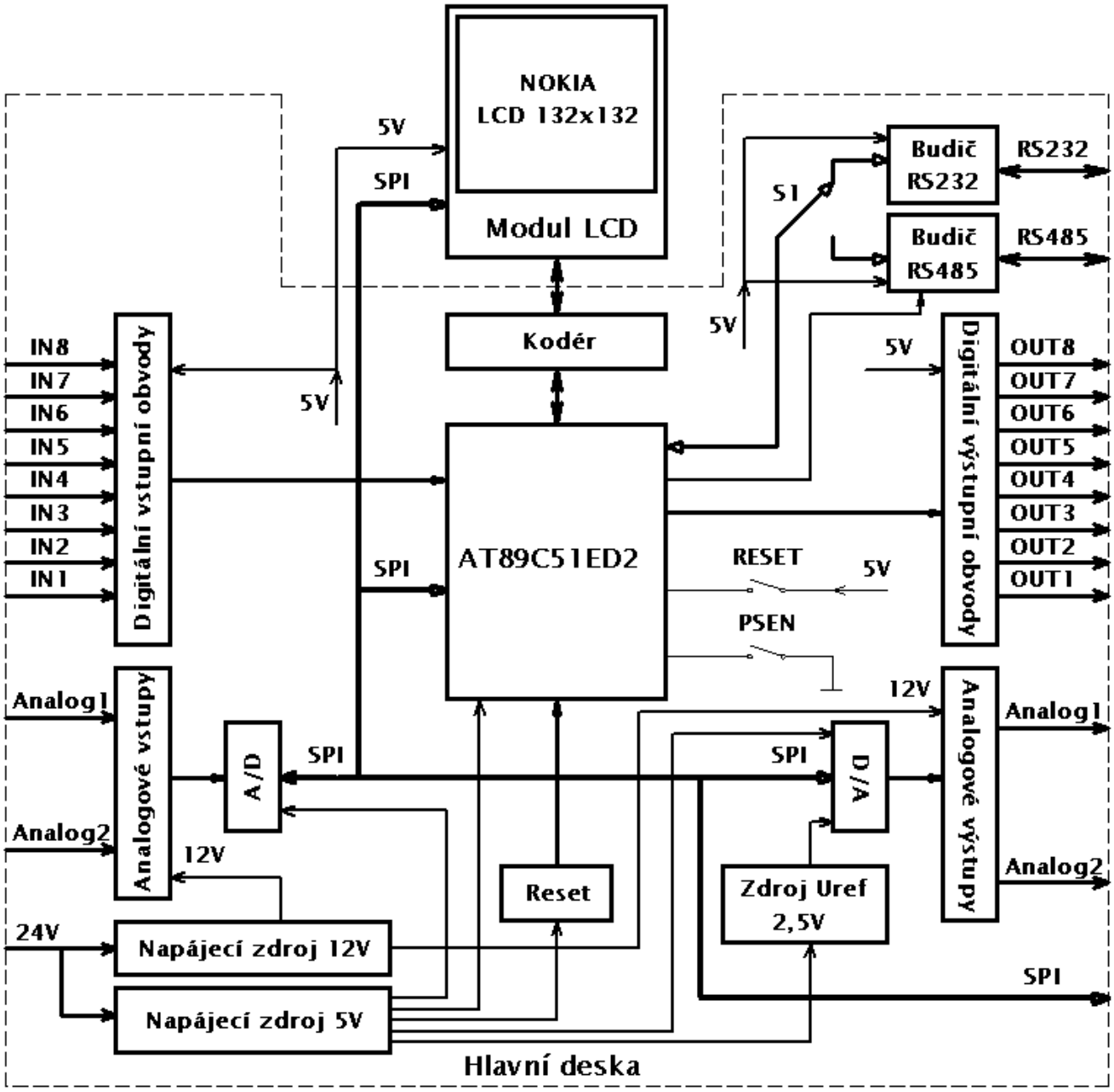
2 ÚVOD

Cílem této diplomové práce je návrh a realizace elektroniky průmyslového stavového indikátoru pro zobrazení průmyslových dat. Tento průmyslový indikátor je vybaven mikrokontrolérem ATMEL řady I51 a s uživatelem komunikuje prostřednictvím grafického barevného displeje, nebo prostřednictvím aplikace, která běží na operačním systému Windows a s mikrokontrolérem komunikuje pomocí protokolu MODBUS. V první části je popsáno blokové schéma indikátoru spolu s popisem jednotlivých komponent. Druhá část je věnovaná detailnějšímu popisu jednotlivých použitých součástek včetně odkazů na manuály. V této části je nejprve popsán samotný modul displeje s ovládacími prvky, poté je popsán návrh DC/DC měniče pro napájení podsvětlení LCD, dále je popsán samotný mikrokontrolér spolu s podpůrnými obvody a následuje popis vstupních obvodů, výstupních obvodů, analogových vstupů, analogových výstupů, komunikačních obvodů a napájecího zdroje. Ve třetí části je popsán protokol MODBUS. Ve čtvrté části je popsána realizace indikátoru spolu s popisem desky plošných spojů a na závěr je uveden postup osazení a oživení indikátoru. Pátá část se zabývá programovým vybavením mikrokontroléru. Je zde uveden popis všech modulů programu. Nejprve se popisuje hlavní modul, dále pak ovládání LCD displeje, protokol MODBUS a na závěr SPI komunikace. V šesté části je popis ovládací aplikace vytvořené pro indikátor spolu s nastavením komunikace a chybovým hlášením. Za tímto následuje v sedmé části závěr, kde je uveden celkový pohled na realizaci indikátoru a případné chyby, které se řešily při realizaci. V poslední osmé části je seznam použité literatury. V příloze je vývojový diagram protokolu MODBUS, vývojový diagram výpočtu kontrolního součtu CRC, schéma modulu LCD a hlavního modulu, dále pak navržené desky plošných spojů spolu s rozmístěním součástek. Na závěr je uveden seznam použitých součástek.

3 POPIS PRŮMYSLOVÉHO INDIKÁTORU

Tento průmyslový stavový indikátor je vybaven mikrokontrolérem firmy ATMEL AT89C51ED2 v patici PLCC44. Jako zobrazovací element je použitý grafický barevný displej NOKIA LCD-00569 (viz. *Obr. 4.1*). Pro ovládání indikátoru a pro ovládání menu displeje je indikátor vybaven okolo displeje šesti tlačítky. S okolními perifériemi (např. PC) tento indikátor komunikuje buďto prostřednictvím sériové linky RS-232, nebo RS-485. Volbu komunikace si uživatel volí pomocí přepínače umístěného za svorkovnicí pro linku RS-485. Přes zmiňovanou linku RS-232 je možné rovněž použitý mikrokontrolér programovat. V indikátoru je také implementovaný protokol MODBUS RTU, přes který jde jednak celý ovládat, ale také číst či zapisovat stavy jednotlivých digitálních vstupů, výstupů, analogových vstupů a analogových výstupů. Pro ovládání indikátoru je vytvořena aplikace, kde jsou v jednom okně umístěny veškeré ovládací prvky a také funkce diagnostiky protokolu MODBUS. Na indikátoru je také implementovaná sériová sběrnice SPI[6]. Přes tuto sběrnici komunikuje mikrokontrolér s A/D převodníkem, D/A převodníkem, na modulu LCD se samotným LCD displejem a elektronickým potenciometrem, který je zde použit pro regulaci intenzity jasu displeje. Tato linka je také vyvedena konektorem pro možnost komunikace indikátoru s dalším zařízením. Dále je na indikátoru možno využít osm digitálních vstupů, osm digitálních výstupů, dva analogové vstupy a dva analogové výstupy. Digitální vstupy a výstupy jsou galvanicky odděleny od elektroniky indikátoru, aby nedošlo nedopatřením k poškození elektroniky jak indikátoru tak vnějších obvodů. U digitálních vstupů je jedná o optickou vazbu tvořenou optočleny PC817B. U digitálních výstupů se jedná o elektromechanickou vazbu použitím relé. Průběžný stav jak vstupů tak výstupů je možno sledovat na indikačních led diodách. Na analogových vstupech indikátoru je možno měřit stejnosměrné napětí v rozsahu 0 – 10V. Analogové vstupy jsou dále ošetřeny transily proti případnému přepětí na vstupech, ale i proti přepólování. Na analogových výstupech je možné nastavovat napětí v rozsahu 0 – 10V. Celý indikátor je napájen za stejnosměrného zdroje 24V. Blok napájecího zdroje má dvě části a to část 12V pro napájení operačních zesilovačů u analogových vstupů/výstupů a část 5V pro napájení mikrokontroléru, převodníků A/D a D/A, prioritního kodéru, modulu LCD, budičů RS-232/RS-485 a jednotlivé relé požité ve výstupním obvodu. Napájecí část je také ošetřena jednak diodou proti případnému přepólování a také transilem proti přepětí. Celé blokové schéma navrhnutého indikátoru je ukázáno na *Obr. 3.1.*

3.1 Blokové schéma indikátoru



Obr. 3.1 Blokové schéma indikátoru

4 NÁVRH HARDWARU INDIKÁTORU

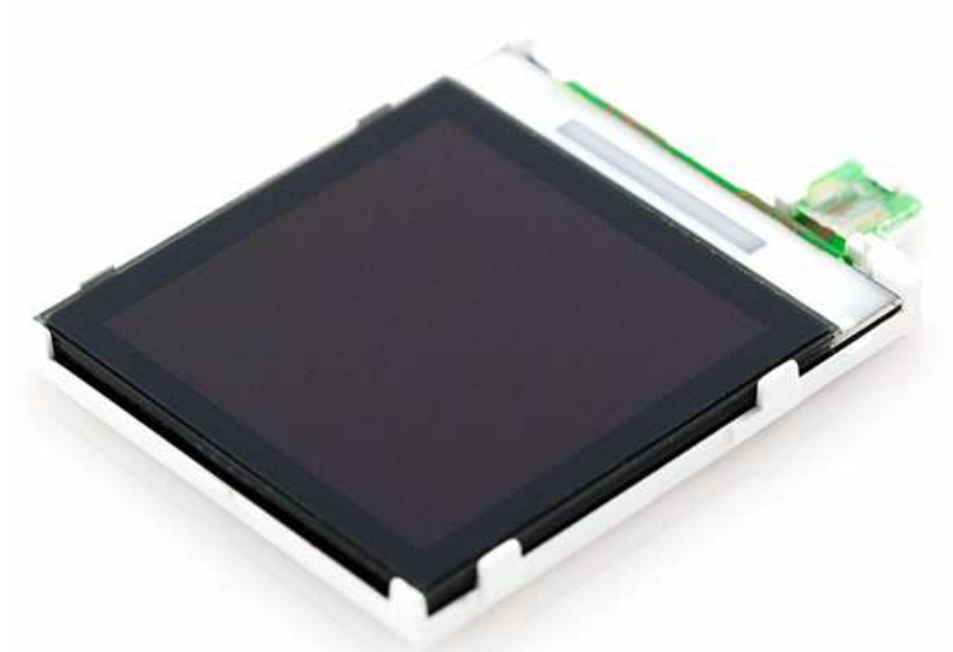
4.1 Grafický LCD displej

Jako zobrazovací element je použitý grafický barevný displej NOKIA LCD-00569 (viz. *Obr. 4.1*), který můžeme najít v různých mobilních telefonech NOKIA (6100,6610,7210,7250). Displej používá jeden ze dvou řadičů a to buďto EPSON S1D15G00[4], nebo PHILIPS PCF8833[3]. Použitý displej na indikátoru má řadič PHILLIPS PCF8833.

Grafický displej má 132x132 pixelů a připojuje se k aplikaci konektorem Hirose DF23[5]. Zapojení tohoto konektoru viz. *Tab. 4.2*. U displeje můžeme využít dvě orientace (řádky/sloupce). První způsob defaultní (viz. *Obr. 4.2*). Druhý způsob, který lze využít a je využit u tohoto indikátoru je na *Obr. 4.3*. Tato orientace je otočená o 180°. Použitá je z důvodu místa pro ovládací tlačítka, které jsou orientovány na pravé straně a pod LCD displejem. Displej komunikuje s okolím pomocí sériového 9. bitového rozhraní SPI. Napájecí napětí pro displej je 3,3V a pro podsvětlení 6-7V

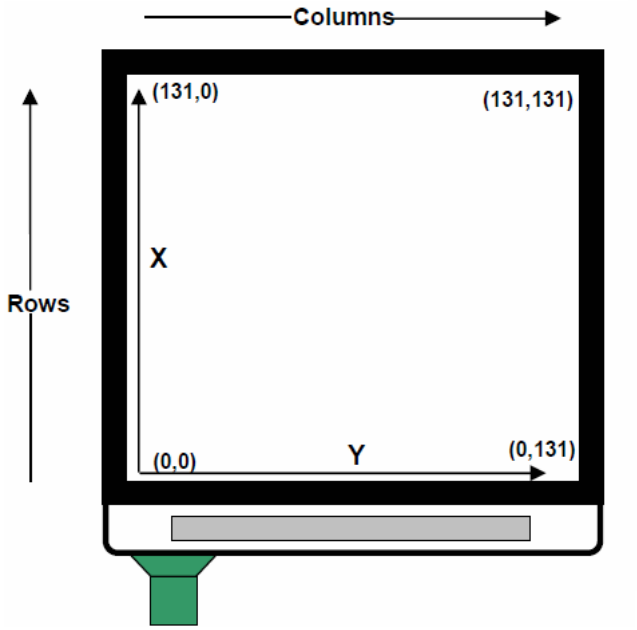
Základní parametry	
Velikost displeje	1,35x1,58”
Velikost aktivní plochy displeje	1,2x1,2”
Napájecí napětí	3,3V/2-3mA
Napájecí napětí LED podsvětlení	6-7V/40-50mA

Tab. 4.1 Základní parametry LCD displeje



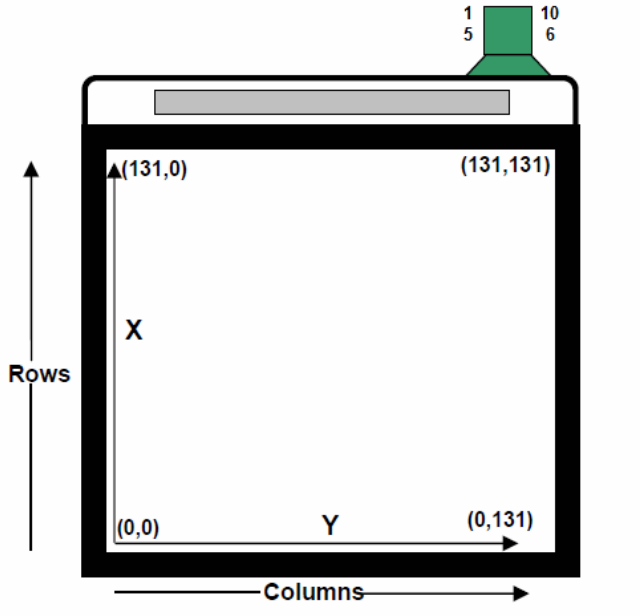
Obr. 4.1 Displej LCD-00569

Použito z [1]



Obr. 4.2 Defaultní orientace displeje

Použito z [2]



Obr. 4.3 Otočena orientace o 180°

Použito z [2]

Zapojení konektoru na LCD:

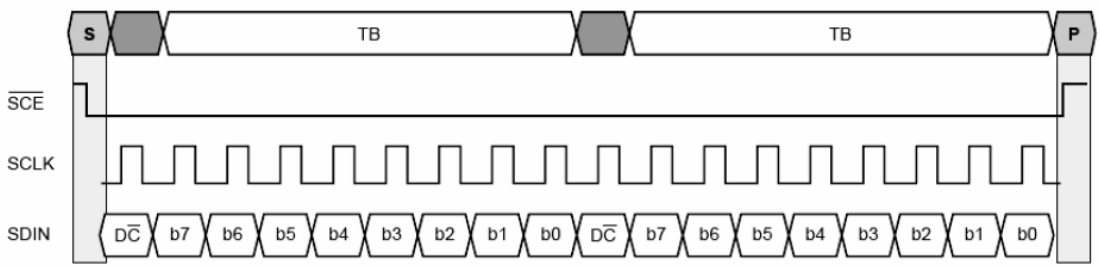
Pin	Funkce
1	*VCC* – Napájecí napětí 3,3V
2	$\overline{RESET}$ – Resetování displeje
3	*DATA* – Datový vstup
4	*CLK* – Hodinový vstup
5	$\overline{CS}$ – Výběr Chipu (slave)
6	*VCC* – Napájecí napětí 3,3V
7	*N.C.*
8	*GND* – zem(0V)
9	*VLED* – – zem(0V)
10	*VLED* + – napájecí napětí pro podsvětlovací led diody 6-7V

Tab. 4.2 Zapojení konektoru LCD displeje

4.1.1 Komunikace LCD displeje sběrnicí SPI

Displej komunikuje s mikrokontrolérem pomocí implementované sběrnice SPI. Mikrokontrolér vykonává funkci mastera, generuje hodinové a datové signály. Displej je zde výhradně jako slave zařízení. Časování signálů je vidět na *Obr. 4.4.* Jak je vidět jedná se o 9. bitovou komunikaci. 9. bit zde signalizuje zda-li se jedná o příkazový bajt $D/\overline{C} = \log.0$, nebo zda se jedná o datový bajt $D/\overline{C} = \log.1$. Poté následuje 8 datových bitů.

Komunikace začíná uvedením pinu $\overline{CS}$ do log. 0. Tím vybereme slave zařízení se kterým má master (mikrokontrolér) komunikovat. Data jsou platná s náběžnou hranou hodinového vývodu (CLK).



Obr. 4.4 Časování signálů komunikace SPI.

Použito z [2]

4.2 Popis zobrazovacího modulu s displejem

Zobrazovací modul je na samostatné desce plošného spoje. Tento modul se připojuje k hlavní desce pomocí dvou konektorů X1_LCD a X2_SWITCH, které jsou umístěny u horní a u spodní hrany desky plošného spoje. Konektor X1_LCD slouží jako napájecí konektor, komunikační konektor sběrnice SPI pro displej a elektronický potenciometr (signály $MOSI$, SCK, $\overline{LCD\_CS1}$, $\overline{POT\_CS2}$), resetovaní signál displeje ($\overline{LCD\_RESET}$), zapínání/vypínání podsvětlení(LCD_ON). Zmiňovaný elektronický potenciometr je v modulu použitý pro nastavení intenzity jasu podsvětlení. Popis zapojení jednotlivých pinů viz. *Tab. 3.3*

Pin	Název	Popis
1	$+5V$	Napájecí napětí modulu
2	LCD_ON	Zapínání/vypínání podsvětlení LCD
3	$\overline{LCD\_RESET}$	Reset LCD
4	$\overline{POT\_CS2}$	$\overline{CS}$ elektronického potenciometru
5	$\overline{LCD\_CS1}$	$\overline{CS}$ LCD displeje
6	GND	Signálová zem
7	$MOSI$	Datový signál SPI
8	GND	Signálová zem
9	SCK	Hodinový signál SPI
10	GND	Signálová zem

Tab. 4.3 Popis konektoru X1_LCD

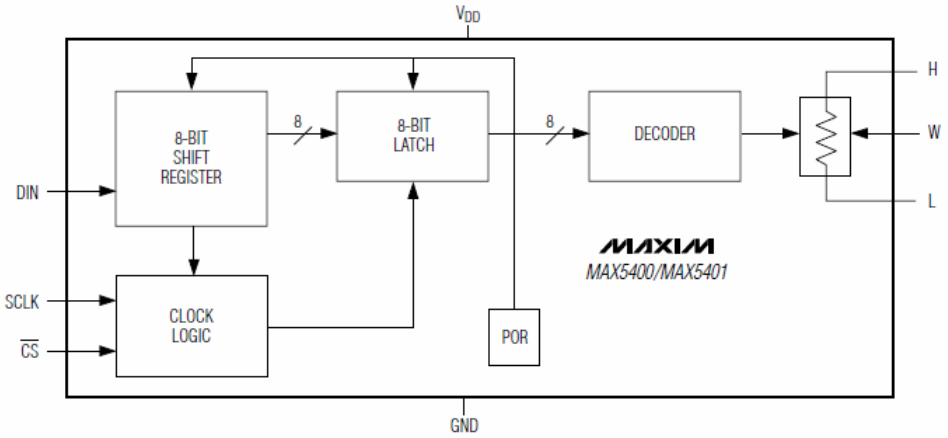
Druhý konektor X2_SWITCH slouží pro čtení stavu jednotlivých ovládacích tlačítek. Tyto tlačítka jsou aktivní v log. 0. Jednotlivé keramické kondenzátory u tlačítek slouží k zmírnění zákmitů způsobených stiskem tlačítka. Rezistory u tlačítek slouží jako pull-up tedy drží stav výstupu v log. 1. Jakmile dojde ke stisku tlačítka na výstupu se objeví log. 0. Výstupy z tlačítek nejsou připojeny přímo na vstupní port mikrokontroléru, ale na kodér. Toto řešení je použité z důvodu nedostatku vstupů mikrokontroléru. Jedná se o prioritní kodér 74HC148[16]. Při tomto řešení mikrokontrolér vyhodnocujeme místo šesti tlačítek pouze tři výstupní bity kodéru.

4.2.1 Digitální potenciometr MAX5400

Jedná se o elektronický potenciometr firmy MAXIM MAX5400. Tento potenciometr pracuje jako klasický mechanický potenciometr, který obsahuje pevný rezistor a digitálně řízený jezdec. Napájecí napětí pro potenciometr je od 2,7V do 5,5V. Tento potenciometr poskytuje 256 kroků pro nastavení výstupního rezistoru. Vyrábějí se ve dvou verzích a to 50kΩ a 100kΩ. V této aplikaci je použit 50kΩ. Základní vlastnosti potenciometru viz. *Tab. 4.4.*

Napájecí napětí	2,7 – 5,5V
Velmi nízký napájecí proud	0,1μA
Počet kroků nastavení rezistoru	256
Výstupní rezistor	50kΩ
Komunikace	SPI
Pouzdro	Miniaturní 8 pinové SOT23

Tab. 4.4 Vlastnosti potenciometru



Obr. 4.5 Blokové schéma potenciometru MAX5400

Použito z [10]

Pin	Název	
1	*L*	Rezistor
2	*GND*	Zem 0V
3	$\overline{CS}$	Výběr chipu (slave zařízení)
4	*DIN*	Vstup sériových dat
5	*SCLK*	Vstup hodinového signálu
6	V_{DD}	Napájecí napětí
7	*W*	Jezdec rezistoru
8	*H*	Rezistor

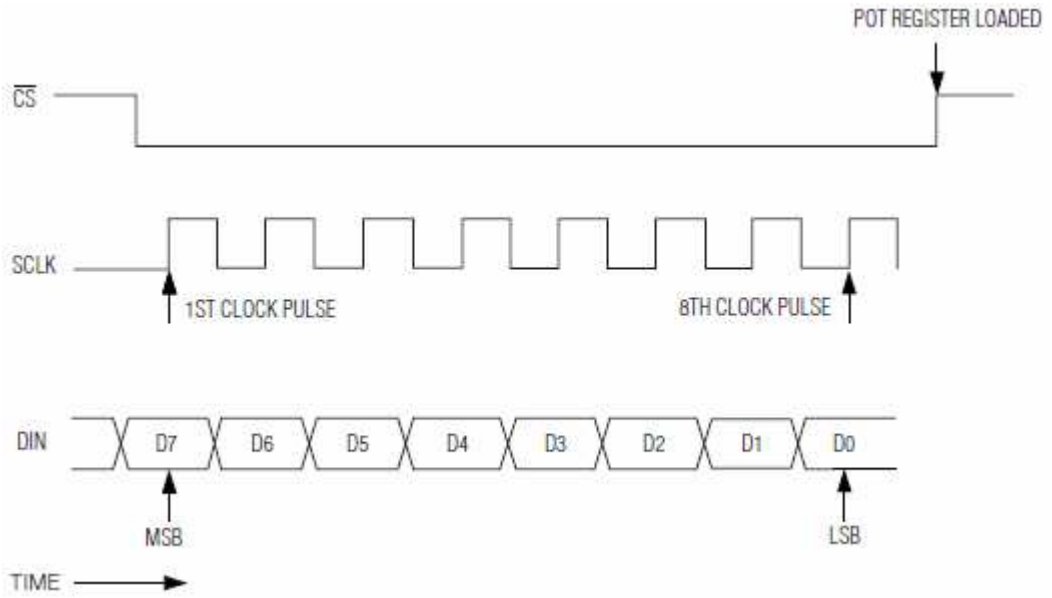
Tab. 4.5 Popis jednotlivých pinů potenciometru

4.2.1.1 Komunikace potenciometru prostřednictvím sběrnice SPI

Potenciometr používá sériové rozhraní SPI k tomu, aby ovládal pozici jezdce na rezistoru. Komunikace začíná, když přijde na pin $\overline{CS}$ log. 0. Poté jsou data(*DIN*) přicházející s náběžnou hranou hodin(*SCLK*) načítána do 8bitového posuvného registru tak jak je znázorněno na *Obr. 4.6*. Jak je vidět z diagramu musí být po celou dobu sériového přenosu dat signál $\overline{CS}$ v log. 0. Poté co přijde všech 8 bitů tak jsou dekódovány s náběžnou hranou $\overline{CS}$. Dekodér převede 8bitový údaj na odpovídající hodnotu rezistoru tuto hodnotu můžeme odečíst z grafu, který najdeme v [10], nebo jí můžeme jednoduše vypočítat pomocí vztahu(1).

$$\text{Výstupní odpor mezi piny W a L: } R_{OUT} = \frac{DIN}{256} 50.10^3 \qquad [\Omega] \qquad (1)$$

Digitální potenciometr dále obsahuje resetovaní obvod po připojení napájecího napětí (POR – power on reset), který nastaví po připojení napětí jezdec rezistoru potenciometru na poloviční hodnotu tedy v našem případě 25kΩ.



Obr. 4.6 Časování signálů komunikace SPI.

Použito z [10].

4.2.2 Návrh napájecích obvodů pro zobrazovací modul

Celý zobrazovací modul je napájen 5V, které musíme dále upravit na hodnotu 3,3V pro napájení displeje. O toto se stará stabilizátor LF33. Jedná se o stabilizátor v základním katalogovém zapojení. Maximální vstupní napětí V_I = 16V, maximální výstupní proud 1A, což bohatě dostačuje pro použitý LCD displej. Na vstupní i výstupní straně stabilizátoru jsou zapojené blokovací kondenzátory o kapacitě 100nF.

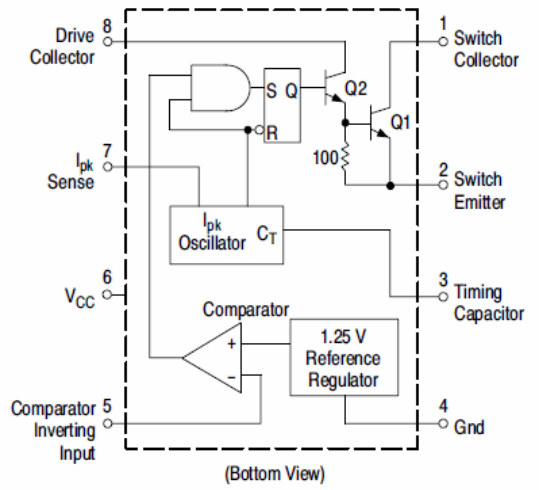
Pro napájení podsvětlení displeje je potřeba upravit napájecí napětí modulu z 5V na hodnotu 7,7V. O toto se stará DC/DC měnič MC34063A. Hodnota výstupního napětí 7,7V místo 7V je potřebná z toho důvodu, protože LCD displej spínáme PNP tranzistorem BC807 na kterém vzniká mezi emitorem a kolektorem úbytek napětí cca. 0,7V. Celkové schéma napájecích obvodů viz. *Obr. 4.9.*

4.2.2.1 Návrh obvodu pro podsvětlení displeje

Základem obvodu pro úpravu napětí z 5V na hodnotu 7,7V je použit DC-DC měnič MC34063A v doporučeném katalogovém zapojení. Tento obvod je možno použít jako zvyšující měnič(Step-Up), snižující měnič(Step-Down) nebo invertující měnič(Voltage-Inverting). V této aplikaci je použitý jako zvyšující měnič. Jmenovité maximální hodnoty měniče viz *Tab. 4.6.*

Vstupní napětí	3 – 40V
Výstupní napětí	1,25 – 40V
Výstupní proud	1,5A
Max. pracovní kmitočet	100kHz
Účinnost	7 – 89%

Tab. 4.6 Maximální hodnoty MC34063A



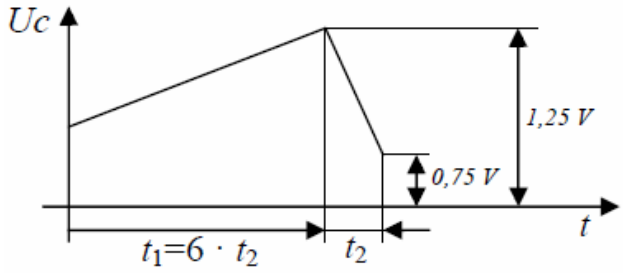
Obr. 4.7 Vnitřní zapojení MC34063A
Použito z [7]

MC34063 je monolitický řídicí obvod obsahující všechny aktivní funkce, požadované pro DC-DC konvertory. Tento integrovaný obvod obsahuje vnitřní teplotně kompenzovaný referenční zdroj, komparátor, oscilátor s šířkovou modulací a obvodem proudového omezení, budič a výstupní spínač pro vysoké proudy. Tato řada obvodů byla zkonstruována specielně pro zvyšující, snižující a invertující měniče. Všechny tyto funkce jsou obsaženy v pouzdru s osmi vývody, jak ukazuje *Obr. 4.7*.

4.2.2.2 Popis funkce MC34063A:

Oscilátor se skládá ze zdroje proudu a spínače, které nabíjí a vybíjí vnější časovací kondenzátor C_T mezi horním a dolním prahem. Typické nabíjecí a vybíjecí proudy jsou 35mA respektive 200mA, jejichž poměr je kolem 1:6. Doba nabíjení kondenzátoru je tedy 6-krát delší než doba vybíjení, jak je ukázáno na *Obr. 4.8*.

Horní práh je roven vnitřnímu referenčnímu napětí 1,25V a dolní práh je přibližně 0,75V. Oscilátor běží nepřetržitě s frekvencí odpovídající zvolené hodnotě C_T. Během části periody, kdy se kondenzátor nabíjí, je na vstupu A součinového hradla log. 1. Když výstupní napětí spínaného regulátoru je pod jmenovitou hodnotou, log. 1 je také na vstupu B hradla. Tato podmínka nastaví do jedničky klopný obvod (Latch) a způsobí, že jeho výstup Q sepne výstupní tranzistor (Q1). Když oscilátor dosáhne horní úrovně, C_T se začne vybíjet a na vstupu A hradla se objeví log. 0. Tato log. úroveň také vynuluje klopný obvod a výstupní tranzistor se zavře.



Obr. 4.8 Časové průběhy napětí na kondenzátoru C_T

Převzato z [24]

Proudového omezení je dosaženo monitorováním úbytku napětí na vnějším odporu, zapojeném v sérii s napětím VCC a výstupním spínačem. Toto napětí je sledováno pinem Ipk Sense. Když toto napětí překročí hodnotu 330mV, obvod pro omezení proudu vytvoří další cestu proudu pro nabíjení časovacího kondenzátoru C_T. To způsobí rychlého dosažení horního prahu oscilátoru, čímž se zkrátí doba sepnutí výstupního tranzistoru a omezí se množství energie uložené v cívce.

Činnost spínacího regulátoru při přetížení způsobí velmi krátkou, ale konečnou dobu sepnutí výstupu, po níž následuje buď normální, nebo prodloužený interval vypnutí způsobený oscilátorem. Rozšíření intervalu vypnutého výstupu je výsledkem nabíjení C_T nad horní práh působením proudového omezení.

4.2.2.3 Výpočet DC/DC měniče:

Pro výpočet je nutné znát hodnotu výstupního napětí, maximální výstupní proud, vstupní napětí a pracovní kmitočet. Jelikož je v obvodu zapojena dioda musíme brát v úvahu také úbytek napětí, který na ní vznikne v propustném směru. Jelikož se v obvodu pracuje s vysokým kmitočtem zvolíme rychlé spínací diody (Schottkyho diody). Dále musíme brát ohled na úbytek napětí na tranzistoru v sepnutém stavu. Hodnoty pro výpočet napájení podsvětlení displeje viz. *Tab. 4.7*

Vstupní napětí	$V_{IN} = 5V$
Minimální vstupní napětí	$V_{IN(\min)} = 4{,}5V$
Výstupní napětí	$V_{OUT} = 7{,}7V$
Maximální výstupní proud	$I_{OUT} = 60mA$
Pracovní kmitočet	$f = 40kHz$
Úbytek napětí na diodě	$V_F = 0{,}55V$
Úbytek napětí na tranzistoru	$V_{SAT} = 0{,}3V$
Zvlnění výstupního napětí	$V_{ripple(pp)} = 50mV$

Tab. 4.7 Hodnoty pro výpočet napájení podsvětlení

Výpočet poměru časů t_{on} a t_{off} [7]:

$$\frac{t_{on}}{t_{off}} = \frac{V_{OUT} + V_F - V_{IN(\min)}}{V_{IN(\min)} - V_{SAT}} = \frac{7{,}7 + 0{,}55 - 4{,}5}{4{,}5 - 0{,}3} = 0{,}8928 \qquad (2)$$

Doba periody T [7]:

$$T = \frac{1}{f} = \frac{1}{40000} = 2{,}5.10^{-5}\,s \qquad [\text{s}] \qquad (3)$$

Doba rozepnutí tranzistoru t_{off} [7]:

$$t_{off} = \frac{T}{\frac{t_{on}}{t_{off}} + 1} = \frac{2{,}5.10^{-5}}{0{,}8928 + 1} = 1{,}3208.10^{-5}\,s \qquad [\text{s}] \qquad (4)$$

Doba sepnutí tranzistoru t_{on} [7]:

$$t_{on} = T - t_{off} = 2{,}5.10^{-5} - 1{,}3208.10^{-5} = 1{,}1792.10^{-5}\,s \qquad [\text{s}] \qquad (5)$$

Hodnota časovacího kondenzátoru [7]:

$$C_T = 4.10^{-5}.t_{on} = 472\,pF \qquad [\text{pF}] \tag{6}$$

Použijeme katalogovou hodnotu $470\,pF$

Špičková hodnota kolektorového proudu tranzistoru [7]:

$$I_{PK(SWITCH)} = 2I_{OUT(\max)}\left(\frac{t_{on}}{t_{off}}+1\right) = 2.0{,}06.(1{,}8928) = 0{,}227A \qquad [\text{A}] \tag{7}$$

Hodnota indukčnosti tlumivky [7]:

$$L_{(\min)} = \left(\frac{V_{IN(\min)} - V_{SAT}}{I_{PK(SWITCH)}}\right)t_{on} = \left(\frac{4{,}5-0{,}3}{0{,}227}\right).1{,}1792.10^{-5} = 218uH \qquad [\mu\text{H}] \tag{8}$$

Použijeme katalogovou hodnotu $220uH$

Hodnota rezistoru $R64$ [7]:

$$R64 = \frac{0{,}3}{I_{PK(SWITCH)}} = \frac{0{,}3}{0{,}227} = 1{,}32\Omega \qquad [\Omega] \tag{9}$$

Použijeme katalogovou hodnotu $1{,}5\Omega$

Hodnota filtračního kondenzátoru $C26$ [7]:

$$C26 = 9\frac{I_{OUT}.t_{on}}{V_{ripple(pp)}} = 9\frac{0{,}06.1{,}1792.10^{-5}}{0{,}05} = 127\mu F \qquad [\mu\text{F}] \tag{10}$$

Použijeme katalogovou hodnotu $220uF$

Výstupní hodnota měniče je dána vztahem [7] $V_{OUT} = 1{,}25.\left(1+\frac{R66}{R67}\right).$ [V] (11)

$V_{OUT} = 1{,}25.\left(1+\frac{R66}{R67}\right) \Rightarrow$ požadované napětí je 7,7V tedy poměr rezistorů $\frac{R66}{R67} = 5{,}16$

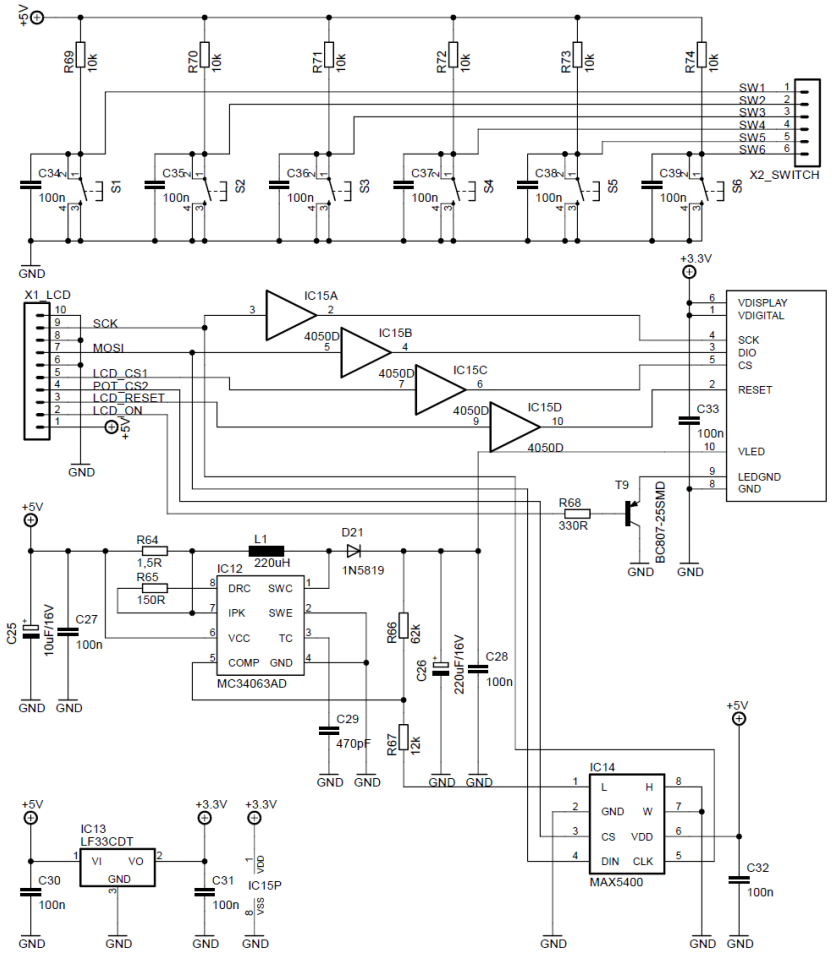
Jak již bylo uvedeno pro nastavení intenzity podsvětlení displeje je zde použit elektronický potenciometr, který má hodnotu výstupního rezistoru 50kΩ a tuto hodnotu zle rozdělit mezi 256 kroků. Tedy na jeden krok jezdce potenciometru vychází zhruba 195Ω. Zvolíme dostatečně velké hodnoty rezistorů R66 a R67, abychom měli dostatek kroků pro nastavení intenzity. Dá se říci že čím větší R66 a R67 tím více kroků pro nastavení

Zvolíme R66 = 62kΩ a z toho tedy poměrem R67= 12kΩ

Při těchto hodnotách rezistorů nám vychází od 6V do 7V zhruba 20 kroků nastavení intenzity podsvětlení.

Pro ovládání podsvětlení je použit tranzistor T9 BC807-25. Jedná se o PNP s maximálními hodnotami U_{EC} = 50V, I_E = 800mA. Zapnutí podsvětlení se provede přivedením log. 0 na pin LCD_ON konektoru X1_LCD. Zapínání a vypínání podsvětlení ovládá přímo mikrokontrolér. Podsvětlení se automaticky vypíná při nečinnosti delší jak cca. 1min. Zapíná se stiskem jakéhokoliv tlačítka na modulu LCD. Bylo by možné mít zapnuté podsvětlení trvale, ale z důvodu úspory energie je použito toto řešení.

Jako poslední je důležité také uvést, že napěťové výstupní úrovně mikrokontroléru ATMEL jsou 5V, ale napěťové úrovně LCD displeje jsou 3,3V. Proto je nutné komunikační signály *MOSI* , *SCK* , $\overline{LCD\_CS1}$ a $\overline{LCD\_RESET}$ převést z 5V logiky na 3,3V logiku. O toto se stará obvod 74HC4050. Jedná se o šest nezávislých neinvertujících oddělovačů. Ochranný obvod na vstupu umožňuje převod ze systému s vyšším napájecím napětím na nižší. Na následujícím obrázku je celkové schéma zapojení napájecích obvodů, elektronického potenciometru, zapínání/vypínání podsvětlení a neinvertujících oddělovačů.



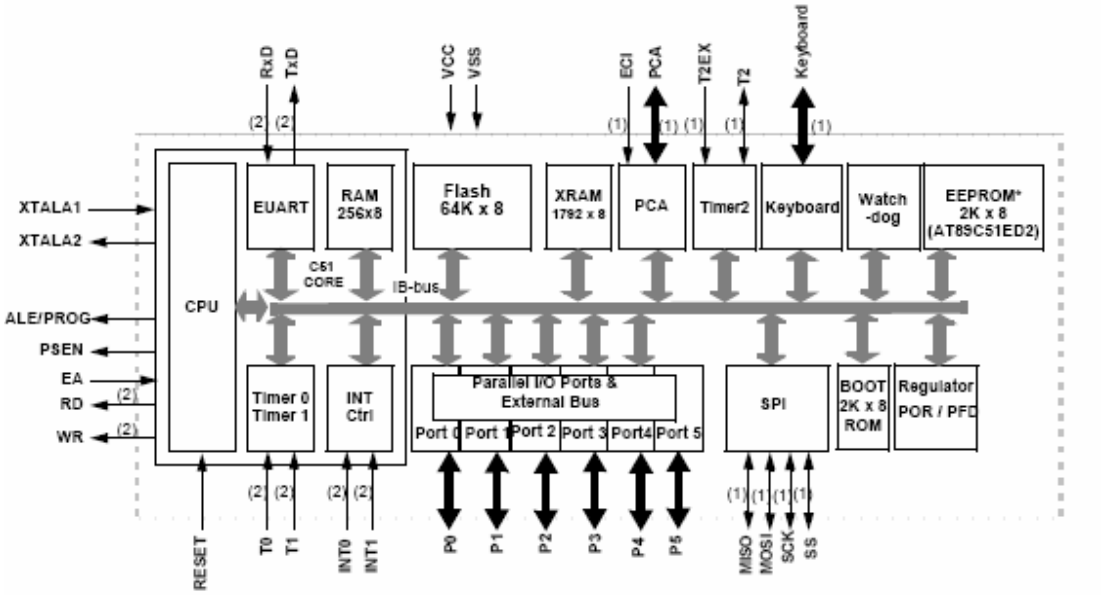
Obr. 4.9 Zapojení modulu spolu s napájecí částí pro podsvětlení a napájení LCD

4.3 ATMEL AT89C51ED2

Na indikátoru je použitý mikrokontrolér firmy ATMEL z rodiny procesorů 8051 a to konkrétně AT89C51ED2. Obvod je založen na jádře x51. Blokové schéma je znázorněno na *Obr. 4.10*. Použité pouzdro mikrokontroléru je PLCC44 viz *Obr. 4.11*. Jedná se o vysoce výkonný, nízko příkonový 8bitový mikrokontrolér s 64KB FLASH paměti pro kód, čtyřmi osmibitovými porty P0 až P3, třemi 16bitovými čítači/časovači, 256 bytů RAM paměti, 2048 bytů EEPROM paměti, ISP a sériovou linku UART.

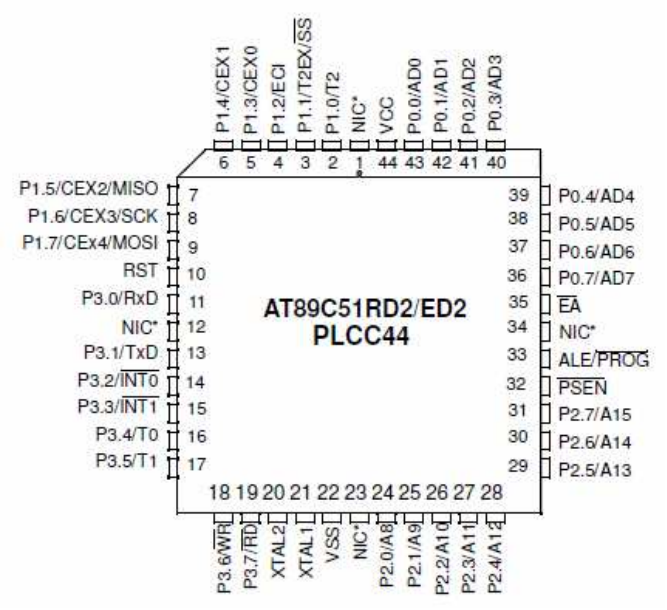
4.3.1 Základní vlastnosti AT89C51ED2

- napájecí napětí 4-5,5V
- Takt až 60MHz pro standardní mód
- 30MHz pro X2 mód (6 hodinových cyklů XTALu na strojový cyklus)
- 64kB Programové paměti
- 1792B externí RAM (XRAM)
- 256B interní datové RAM
- 32 vstupně/výstupních linek k libovolnému použití
- 9 zdrojů přerušení
- ISP – podpora sériového programování
- BOOT Leader
- 3 zabudované 16bitové čítač/časovače
- SPI sběrnice



Obr. 4.10 Blokové schéma AT89C51RD2 (U AT89C51ED2 nejsou P4,P5)

Použito z [6]



Obr. 4.11 Rozložení vývodů AT89C51ED2

Použito z [6]

4.3.2 Zapojení mikrokontroléru a podpůrných obvodů

Mikrokontrolér AT89C51ED2 je napájen ze zdroje 5V, který se připojuje na pin 44 (VCC). 5V je také nutné přivést na pin 35 $(\overline{EA})$, aby mikrokontrolér vykonával program z vnitřní paměti programu. Pro správnou funkci mikrokontroléru jsou dále zapotřebí podpůrné obvody.

Taktování je zajištěno připojením krystalového oscilátoru Q1 na pin 21 $(XTAL1)$. Hodnota tohoto oscilátoru je zvolená na 30MHz. Tato hodnota není úplně ideální co se týče přesného časování sériové linky, ale je použitá z důvodu co nejrychlejšího vykreslování LCD displeje. Mikrokontrolér je po resetu nastaven do X2 módu(6 hodinových cyklů oscilátoru na jeden strojový cyklus). Takto nastavený a zapojený mikrokontrolér dosahuje svého maxima co se týče rychlosti vykonávání instrukcí.

Resetovaní obvod je tvořen jednak RC obvodem tvořeným kondenzátorem C8 a rezistorem R6 (tento resistor by se mohl vynechat jelikož je obsažen v pouzdru mikrokontroléru) a jednak resetovacím obvodem MAX810. Tento resetovaní obvod hlídá velikost napájecího napětí a jeho úkolem je generovat resetovací impuls pro mikrokontrolér vždy, když hodnota napájecího napětí klesne pod 3,08V.

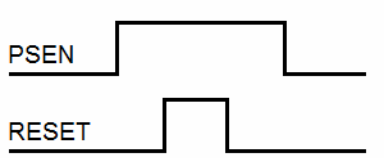
Pro celou aplikaci je ještě významná další vlastnost zvoleného mikrokontroléru. A to způsob nahrávání řídícího programu. Programování jednoho kusu v profesionálních programátorech je zdlouhavé a jejich cena je také dosti vysoká. Zvolený mikrokontrolér umožňuje programování hned třemi způsoby.

- První je klasický způsob umístění mikrokontroléru do programátoru.
- Druhý způsob je programování pomocí SPI rozhraní.
- Poslední způsob programování je specifický pro mikrokontrolér AT89C51ED2. Jedná se o funkci bootloaderu, která programuje procesor pomocí sběrnice UART.

4.3.3 BOOT LOADER

Funkce Bootloader umožňuje při určité kombinaci tlačítek nahrát do mikrokontroléru program pomocí sériové linky UART. Funkce bootloader může být aktivována dvěma způsoby a to hardwarově nebo softwarově.

Hardwarový způsob – Pomocí tlačítka PSEN se aktivuje vnitřní bootloader mikrokontroléru, kterým se po sériové lince RS232 programuje FLASH a EEPROM mikrokontroléru. Tlačítko ovládá pin 32 $\left(\overline{PSEN}\right)$, který je testován po resetu CPU a je-li na úrovni 0V spustí program LOADER na adrese FC00h a tím dojde k aktivaci ISP režimu programování. Pokud nebude po resetu tlačítko aktivní neaktivuje se ISP programování a spustí se vždy program od adresy 0000h. Na desce jsou umístěna dvě tlačítka a to tlačítko RESET pro reset procesoru a tlačítko PSEN pro ovládání zmiňovaného pinu $\overline{PSEN}$. Aktivace bootloaderu se provede následujícím způsobem. Nejdříve se stiskne tlačítko PSEN poté se k němu stiskne tlačítko RESET, dále se pustí tlačítko RESET a poté se pustí i tlačítko PSEN. Tím je aktivován bootloader procesoru. Popsaná sekvence kroků je znázorněna v grafu na *Obr. 4.12*. Graf neznázorňuje logické úrovně signálů ale pouze stisknutí a puštění tlačítek.



Obr. 4.12 Sekvence stisknutí tlačítek PSEN a RESET

Softwarový způsob – Provádí se vynulováním bitu BLJB v registru HSB(Hardware Security Byte) a byte BSB(Boot Status Byte) musí být nenulový.

Po naprogramování je potřeba softwarově aktivovaný bootloader deaktivovat, aby se po resetu začal vykonávat program. Deaktivace se provádí vynulováním bytu BSB. K tomu je určen příkaz , který se bootloaderu zasílá po sériové lince UART z pragramu FLIP.

4.4 Vstupní obvody

4.4.1 Digitální vstupy

Digitální vstupy indikátoru jsou realizovány pomocí osmi nezávislých optočlenů. Na tyto vstupy se mohou připojovat různá zařízení s různými výstupními potenciály stejnosměrného napětí v rozsahu 5-24V.

Galvanické oddělení je tvořeno pomocí optočlenů OK1 – OK8. Jedná se o optočlen v SMD provedení (PC817B-SMD). Napěťový izolační stav tohoto optočlenu je 7,5kV. Jednotlivé vstupy jsou složeny z rezistoru, ochranné diody proti přepólování (1N4007SMD), indikační diody LED zelené barvy (SMD velikost 1206).

Výstup z optočlenu čili z kolektoru tranzistoru je přímo připojen na vstup mikrokontroléru. Při sepnutí optočlenu bude na výstupu log. 0. V log. 1 nám výstup drží tzv. pull-up rezistor. Tyto rezistory jsou připojeny na port mikrokontroléru P0 proti napájecímu napětí 5V. Vstupní rezistor je nutné zvolit s ohledem na vstupní napětí. Při maximálním napětí čili 24V by měla být hodnota rezistoru zhruba 1,2kΩ viz. výpočet(12). K připojení jednotlivých vstupů je použita svorkovnice ARK103/2. Jedná se o šroubovací svorkovnici do DPS, kam se vodiče připojují pod úhlem 45°. Celkové schéma zapojení jednoho digitálního vstupního obvodu je na *Obr. 4.13*.

Maximální hodnoty LED diody: $U_F = 2{,}4V$

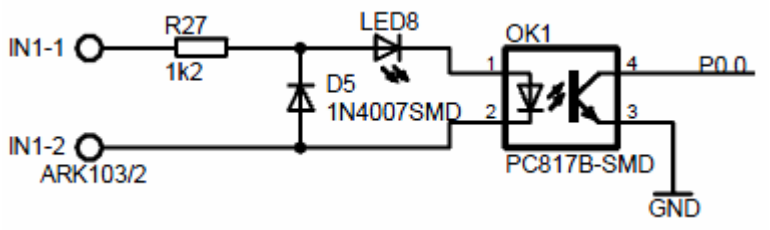
$I_F = 30mA$

Maximální hodnoty Optočlenu: LED dioda: $U_F = 1{,}4V$

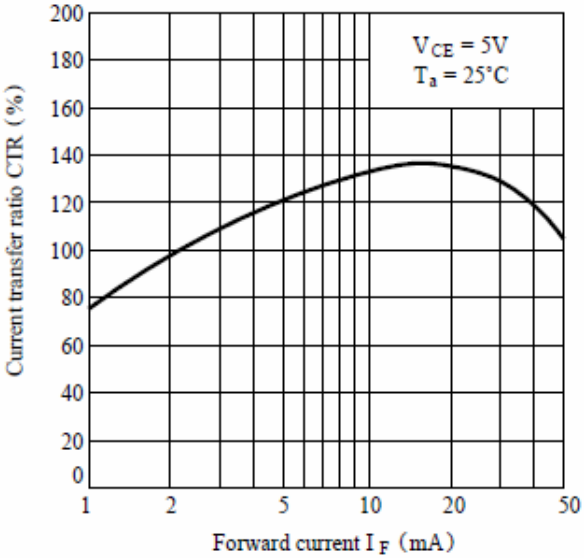
LED dioda: $I_F = 50mA$

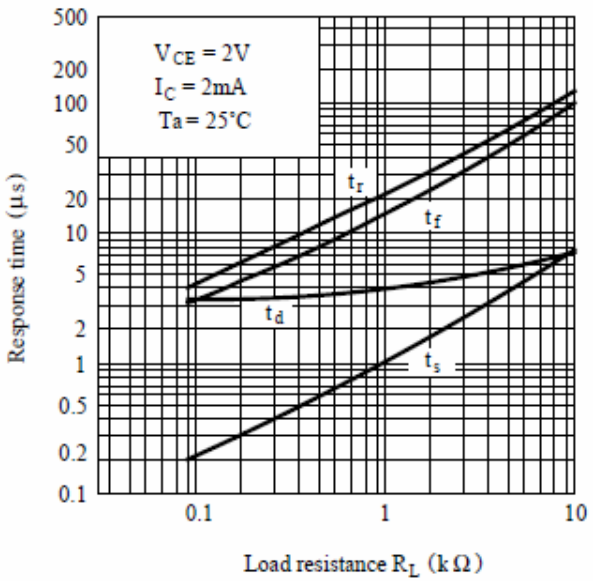
Napětí mezi kolektorem a emitorem: $U_{CE} = 35V$

Poud kolektoru: $I_C = 50mA$



Obr. 4.13 Zapojení digitálního vstupu





Obr. 4.14 Závislost proudu LED optočlenu na činiteli přenosu
Použito z [11]

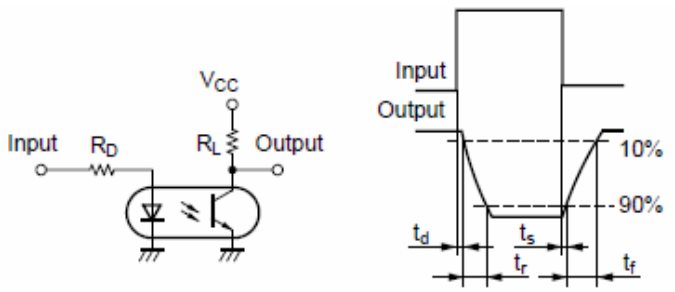
Obr. 4.15 Doba náběžné a sestupné hrany v závislosti na rezistoru R_L
Použito z [11]

Z grafu na *Obr. 4.14* je patrné, že činitel přenosu bude největší zhruba při proudu cca. 18mA. Z tohoto vypočítáme velikost rezistoru R27. Pro hodnotu vstupního napětí 24V bude následující hodnota rezistoru R27:

Výpočet rezistoru R27: $$R_{27} = \frac{U_{IN} - U_{LED8} - U_{OK1}}{I_{OK1}} = \frac{24 - 2{,}05 - 1{,}2}{0{,}018} \doteq 1200\Omega \qquad (12)$$

Pro vstupní napětí 5V bude při hodnotě R27 = 1k2 proud LED optočlenu I_{OK1} = 1,5mA to odpovídá činiteli přenosu CRT = 90%.

Na *Obr. 4.15* jsou do grafu vyneseny doby náběžné a sestupné hrany výstupu optočlenu v závislosti na použitém rezistoru. Pro náš případ(pull-up rezistor velikosti 4,7kΩ) bude doba náběžné hrany bude cca. 60µs a doba sestupné hrany bude cca. 50µs. Použité zapojení na měření náběžné a sestupné hrany je na *Obr. 4.16*. Na tomto obrázku jsou také ukázány jednotlivé časové konstanty.



Obr. 4.16 Typické zapojení optočlenu – vlevo, náběžná sestupná hrana - vpravo
Použito z [11]

4.4.2 Analogové vstupy

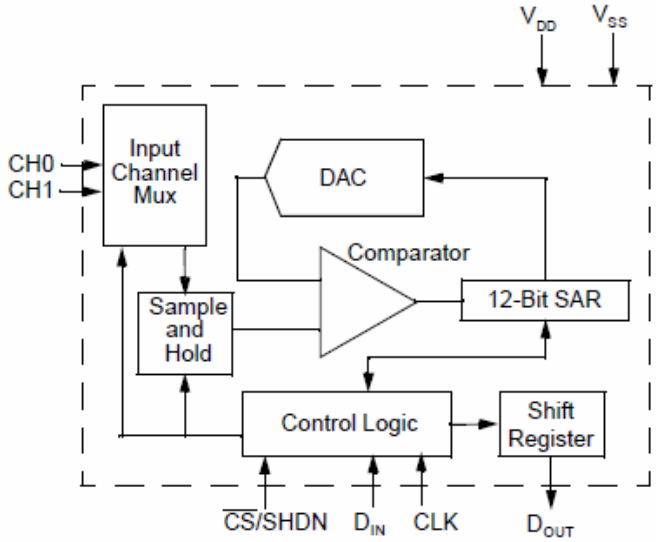
4.4.2.1 MCP3202 – Dvoukanálový A/D převodník

Pro měření analogového napětí je indikátor vybaven převodníkem od firmy MICHROCHIP MCP3202. Jedná se o 12-bitový A/D převodník vybaven sběrnicí SPI.

4.4.2.1.1. Vlastnosti převodníku MCP3202:

- 12-bitové rozlišení
- 2 kanály
- diferenciální a integrální nelinearita ± 1 LSB
- analogové vstupy použitelné jako SE(single-ended – měří proti GND) nebo jako pseudodiferenční pár
- zabudovaný obvod S/H(Sample & Hold, vzorkovač)
- SPI rozhraní usnadňující komunikaci s mikrokontroléry
- maximální vzorkovací rychlost 100kSPS (100 000 vzorků za sekundu) pro napájecí napětí 5V a 50kSPS pro napětí 2,7V
- napájecí napětí v rozsahu 2,7 až 5,5V
- použitá nízko příkonová CMOS technologie(klidový proud až 5nA, v aktivním režimu 550µA při napájení 5V)
- průmyslový rozsah pracovních teplot (-40°C až +85°C)

Blokové schéma obvodu je na *Obr. 4.17.* Vývody V_{DD}/V_{REF} a V_{SS} slouží přivedení napájecího napětí. Toto napětí je zároveň použito jako napěťová reference.



Obr. 4.17 Vnitřní zapojení převodníku MCP3202

Použito z [12]

Vývody $CH0$ a $CH1$ jsou vstupy jednotlivých kanálů. Tyto vstupy mohou být konfigurovány jako nezávislé kanály v režimu SE(Single-Ended, napětí se měří proti GND), nebo do pseudodiferenčního režimu(napětí se měří mezi vstupy, lze volit orientaci $CH0$ - $CH1$ nebo $CH1$ - $CH0$).

$\overline{CS}$ / $SHDN$ vybírá režim obvodu. Je-li $\overline{CS}$ / $SHDN$ = 1, je obvod vypnut (Shut-Down) a má pak sníženou spotřebu(cca 5nA), Je-li $\overline{CS}$ / $SHDN$ = 0, je obvod připraven na vyslání převedených dat (Chip Select). Vzorkování vstupního napětí před převodem probíhá při $\overline{CS}$ / $SHDN$ = 1.

CLK je používán pro zahájení převodu a také pro synchronizaci vysílaných resp. přijímaných bitů.

D_{IN} slouží pro příjem vstupních konfiguračních dat. Jedná se o bity stanovující aktivní kanál, jeho režim a způsob vysílání dat.

D_{OUT} slouží pro vysílání výsledku převodu. Bity tohoto výstupu se mění sestupnou hranou CLK .

4.4.2.1.2. Základní princip činnosti

A/D převodník MCP3202 pracuje jako sériový aproximační registr spojený s D/A převodníkem. V této architektuře je vzorek vstupního napětí uložen na kondenzátoru v zabudovaném obvodu S/H. Vzorkování probíhá po dobu 1,5 hodinového cyklu CLK po příjmu start-bitu. Následně se vstupní spínač rozpojí a produkuje se 12-bitový výstupní kód.

Analogové vstupy umožňují používat dva kanály, nebo jeden pseudodiferenční pár. Konfigurace se provádí před každým převodem vysláním konfiguračních bitů na vodiči D_{IN} . Ideální pro převod je, aby byly vstupy $CH0$ a $CH1$ buzeny u odporu blízkého nule. Jinak se výstupní odpor zdroje signálu přičte k pracovnímu odporu obvodu S/H. Důsledkem pak bude zvýšení nelinearity převodu.

Výstupní kód(13) produkovaný na výstupu D_{OUT} závisí na velikosti vstupního napětí a na velikosti referenčního napětí V_{REF} . Je-li vstupní napětí nižší než 0V, je vrácen výstupní kód 000h. Je-li vstupní napětí vyšší než V_{REF} - 1LSB, je vrácen výstupní kód FFFh. To platí i v diferenčním režimu(zde se ale napětí uvažuje mezi vstupy).

Výstupní digitální kód je dán vzorcem: Výstupní kód = $\frac{4096 * V_{IN}}{V_{DD}}$ (13)

kde V_{IN} - analogové vstupní napětí

V_{DD} - napájecí napětí převodníku

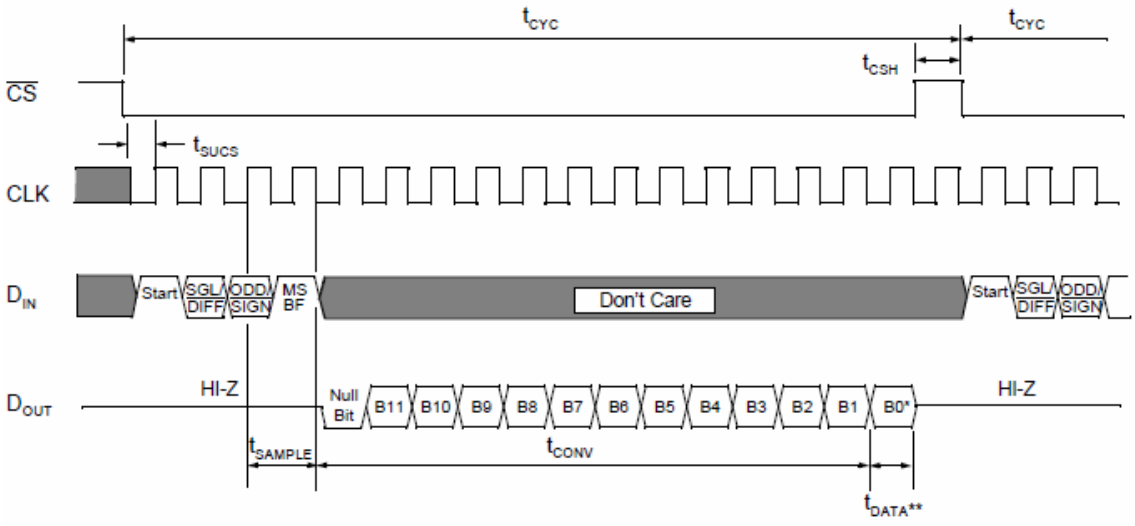
4.4.2.1.3. Sériová komunikace

Komunikace řídicího mikrokontroléru s převodníkem MCP3202 je provedena podle standardu SPI. Přenos začíná sestupnou hranou signálu $\overline{CS}$ / $SHDN$. První jedničkový bit vyslaný na vodiči D_{IN} synchronizovaný hodinovým impulzem CLK (CS/SHDN = 0) představuje start-bit. Následují bity SGL/DIFF (určuje režim vstupu: SE, nebo pseudodiferenční) a ODD/SING (vybírá kanál v SE režimu, nebo určuje polaritu v pseudodiferečním režimu) viz *Tab 4.8*.

	Konfigurační bity		Výběr kanálu	
	$SGL / \overline{DIFF}$	$ODD / \overline{SIGN}$	0	1
SE režim	1	0	+	
	1	1		+
Pseudodiferenční režim	0	0	IN+	IN-
	0	1	IN-	IN+

Tab. 4.8 Význam konfiguračních bitů

Poslední je bit MSBF, který určuje formát přenosu výstupních dat. Je-li MSBF = 1, jsou data vysílána počínaje nejvyšším bitem (MSB) a nadbytečné hodinové impulzy vyvolají na DOUT nuly. Je-li MSBF = 0, vyšlou se data nejdříve ve formátu MSB až LSB a potom ve formátu LSB až MSB. Sériová komunikace je znázorněna na *Obr. 4.18*.



Obr. 4.18 Sériová komunikace SPI převodníku MCP3202

Použito z [12]

4.4.2.2 OZ TLC272

Aby analogové vstupy byly schopné měřit napětí v rozsahu 0-10V je nutné tento rozsah napětí nejdříve převést na rozsah 0-5V. K tomuto slouží operační zesilovač TLC272 zapojený jako neinvertující zesilovač spolu s připojenými rezistory.

4.4.2.2.1. Výpočet hodnot rezistorů pro OZ TLC272

Max. vstupní napětí $U_1 = 10V$

Požadované výstupní napětí (vstupní napětí A/D převodníku) $U_2 = 5V$

Předpokládáme R11 = R15 a R13 = R17

Pro takto zapojený operační zesilovač platí: $K = \frac{U_2}{U_1} = \frac{R_{15}}{R_{17}} = \frac{R_{13}}{R_{11}} = 0{,}5$ (14)

Poměr rezistorů by měl být 0,5

Zvolíme R13 = R17 = 2kΩ $\Rightarrow$ R11 = R15 = 1kΩ

Pro kontrolu vypočítáme hodnotu U_2 z takto navržených rezistorů:

Napětí na rezistoru R_{13}: $U_{R13} = U_1 \frac{R11}{R13 + R11} = 10.\frac{1000}{2000 + 1000} = 3{,}333V$ [V] (15)

Napětí na rezistoru R_{15}: $U_{R15} = U_{R13}$ [V] (16)

Výstupní napětí: $U_2 = \frac{U_{R15}(R15 + R17)}{R15} = \frac{3{,}333.3000}{2000} \doteq 5V$ [V] (17)

Aby bylo možné výstupní napětí nastavit na přesných 5V jsou místo pevných rezistorů R17 a R13 použité trimry o dvojnásobné hodnotě odporu. S těmito trimry máme rozsah pro nastavení výstupního napětí od cca 1V až do cca 7V. Vypočtené hodnoty rezistorů platí i pro druhý analogový vstup tedy pro rezistory R12,R14,R16,R18. Oba analogové vstupy jsou dále ošetřeny transily proti přepětí a přepólování. Analogové vstupní napětí se připojuje na svorkovnice ARK103/2. Jedná se o šroubovací svorkovnice do DPS, kam se vodiče připojují pod úhlem 45°.

4.5 Výstupní obvody

4.5.1 Digitální výstupy

Výstupní obvody indikátoru jsou realizovány pomocí osmi nezávislých výstupů. Jedná se o reléové výstupy a tudíž jsou dokonale galvanicky odděleny od elektroniky indikátoru a umožňují spínat napětí až 250V/10A.

Sepnutí jakéhokoliv relé je indikováno červenou LED diodou. Výstupní zařízení se připojují na kontakty OUT1 – OUT8. Jednotlivé relé jsou napájeny napájecím napětím 5V. K aktivaci příslušného relé dojde v okamžiku sepnutí tranzistorů T1 – T8. Tento tranzistor je ovládán přímo z mikrokontroléru. K sepnutí tranzistoru dojde v okamžiku vyslání log. 0 na příslušném portu mikrokontroléru.

Při vypnutí napájení přiváděného na vinutí relé, dojde ke zpětné indukci a to několikrát vyššího napětí, než bylo původně připojeno. Dioda D13 zapojená v závěrném směru se v tomto okamžiku stává „zkratovacím obvodem“ a škodlivé napětí odstraní.

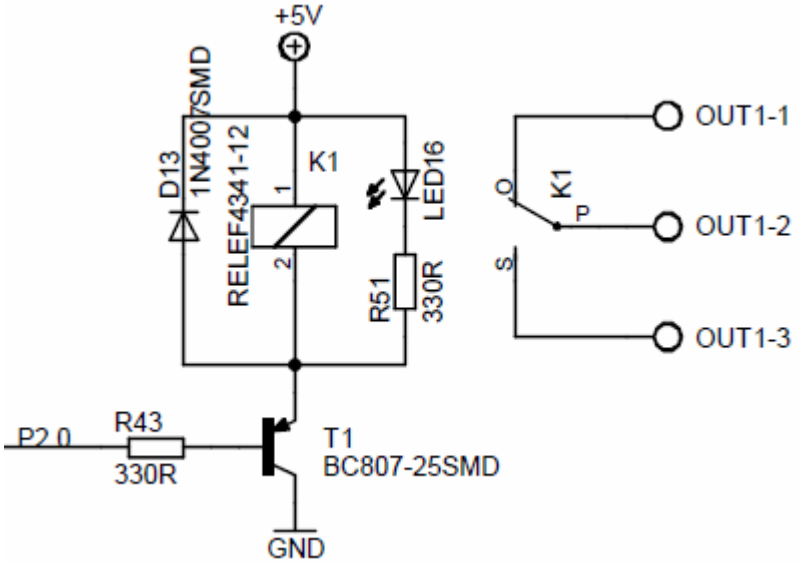
Použitý tranzistoru je typu PNP a je ve funkci spínače. Použití PNP tranzistoru je z toho důvodu, aby nedocházelo ke spínání jednotlivých relé při resetu mikrokontroléru, kdy na jednotlivých výstupních portech je stav log. 1.

Celkové schéma zapojení výstupního obvodu je na *Obr. 4.19.*

Rezistor indikační LED diody pro proud 10mA:

$$R_{51} = \frac{U - U_{LED} - U_{CE}}{I_{LED}} = \frac{5 - 2{,}1 - 0{,}6}{0{,}01} \doteq 240\Omega \qquad [\Omega] \qquad (18)$$

Z důvodu úspory proudového zatížení volíme rezistor o hodnotě 330Ω. Při této hodnotě vychází proud cca 7mA.



Obr. 4.19 Zapojení digitálního výstupu

4.5.2 Analogové výstupy

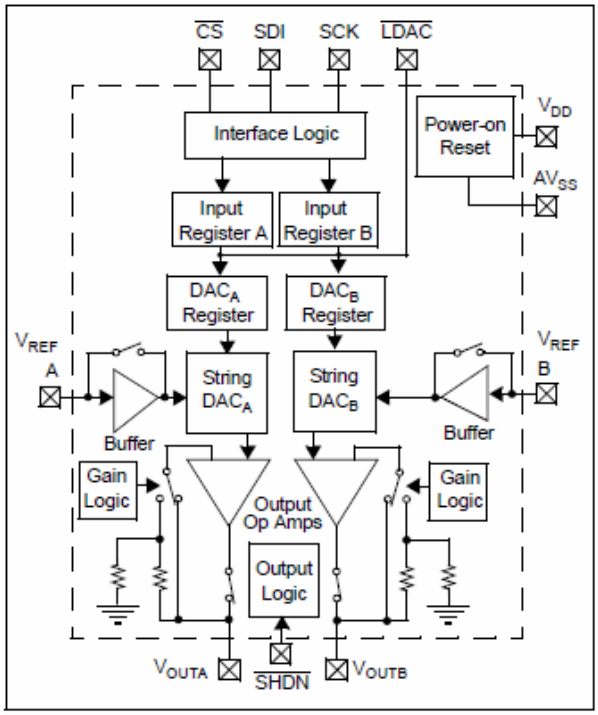
4.5.2.1 MCP4922 – Dvoukanálový D/A převodník

Obvod MCP4922[13] vyrábí firma Microchip. Jedná se o digitálně-analogový převodník se dvěma analogovými kanály a 12bitovým rozlišením, který komunikuje pomocí sběrnice SPI.

4.5.2.1.1. Vlastnosti převodníku MCP4922:

- rozlišení 12 bitů
- typická diferenciální nelinearita $\pm 0{,}2$ LSB
- typická integrální nelinearita $\pm$ 2 LSB
- maximální kmitočet SPI sběrnice až 20MHz
- doba ustálení maximálně 4,5 μs
- volitelné zesílení výstupu 1x nebo 2x
- napájecí napětí v rozsahu 2,7 až 5,5V

Vnitřní zapojení obvodu MCP4922 je uvedeno na *Obr. 4.20.*



Obr. 4.20 Vnitřní zapojení MCP4922

Použito z [13]

D/A převodník komunikuje s mikrokontrolérem pomocí sběrnice SPI a to signály *SDI* (vstupní data), *SCK* (hodiny přenosu) a $\overline{CS}$ (výběr obvodu).

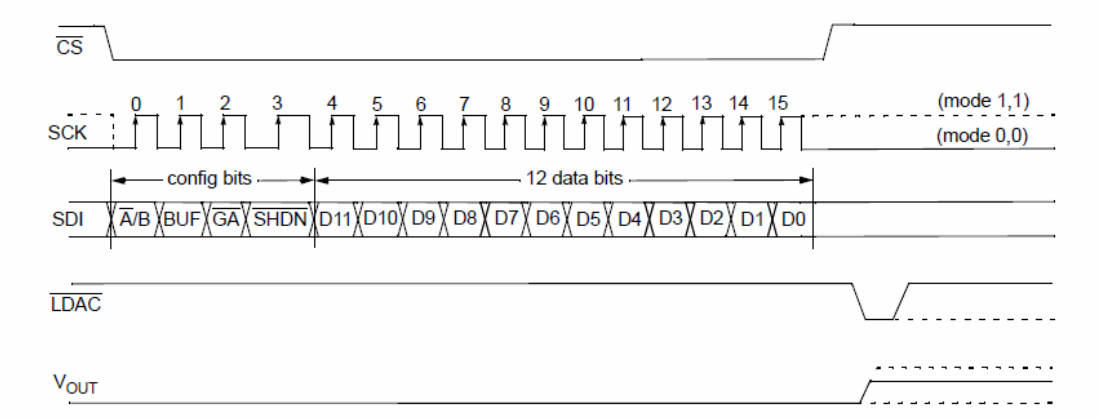
Data přijatá sběrnicí SPI jsou uložena do vstupního registru, signálem $\overline{LDAC}$ je lze přepsat do záchytného registru typu latch a přivést na vnitřní D/A převodník. Pokud synchronizaci tímto signálem nechceme používat, lze ponechat $\overline{LDAC}$ připojený trvale na log. 0. K přepisu dat směrem na vnitřní D/A převodník pak dojde náběžnou hranou signálu $\overline{CS}$.

Vývody V_{REFA}, V_{REFB} slouží pro přivedení referenčního napětí. Referenční napětí může být přivedeno na D/A převodník buď přímo, nebo přes oddělovací zesilovač (buffer). Referenční napětí se může pohybovat v rozmezí 0,040V až Ucc – 0,040V. U indikátoru je referenční napětí vytvořeno stabilizátorem TL431[19], které dodává referenční napětí 2,5V.

Vývod $\overline{SHDN}$ slouží pro hardwarové vypnutí převodníku. Při aktivaci $\overline{SHDN}$ (log. 0) se odpojí rozhraní SPI a sníží se spotřeba převodníku na cca. 1,5µA. Výstupní napětí je k dispozici na vývodech V_{OUTA}, V_{OUTB} . Zesílení koncového zesilovače lze volit 1x nebo 2x. Výstupní napětí je pak dáno vzorcem(19)(N je nastavené číslo, G je zisk):

$$U_{OUT} = U_{REF} \frac{N.G}{4096} \qquad [V] \qquad (19)$$

Data odeslaná mikrokontrolérem do převodníku MCP4922 mají délku 16bitů. Kromě 12 datových bitů se ještě odesílají úvodní bity, které konfigurují činnost převodníku viz. *Obr. 4.21.*



Obr. 4.21 Přenos dat D/A převodníku MCP4922

Použito z [13]

bit 15 $\overline{A}/B$ - Výběr převodníku pro který budou platit data. 1 = zápis na D/A_B 0 = zápis na D/A_A

bit 14 BUF - Vstupní buffer V_{REF}. 1 = zapnuto, 0 = vypnuto

bit 13 $\overline{GA}$ - volba zisku výstupního zesilovače. 1 = zisk 1x, 0 = zisk 2x

bit 12 $\overline{SHDN}$ - Softwarové vypnutí převodníku kvůli snížení spotřeby. Softwarové vypnutí má spotřebu cca. 5μA.

Bit 11-0 $D11:D0$: Datové bity. 12-bitové číslo určené k převodu na napětí podle rovnice (20). Velikost čísla je mezi 0 a 4095.

4.5.2.2 OZ TLC272

Aby byl analogový výstup schopen dodávat výstupní napětí v úrovni 0 – 10V, je zapotřebí upravit výstupní napětí z D/A převodníku. Výstupní napětí z D/A převodníku je dáno již zmiňovaným vztahem(10). Při zvoleném zisku 2 výstupního zesilovače D/A převodníku a při referenčním napětí $U_{REF} = 2{,}5V$ vychází výstupní napětí:

$$U_{OUT} = U_{REF}\frac{N.G}{4096} = 2{,}5\frac{4096.2}{4096} = 5V \qquad [V] \qquad (20)$$

Z výpočtu vychází že je zapotřebí výstupní napětí z D/A převodníku dále upravit a to zdvojnásobit. O toto se bude starat stejně jako u analogového vstupu operační zesilovač TLC272 zapojený jako neinvertující zesilovač spolu s připojenými rezistory.

4.5.2.3 Výpočet hodnot rezistorů pro OZ TLC272:

Vstupní napětí D/A převodníku $U_1 = 5V$

Požadované výstupní napětí $U_2 = 10V$

Předpokládáme R19 = R23 a R21 = R25

Pro takto zapojený operační zesilovač platí: $K = \frac{U_2}{U_1} = \frac{R_{21}}{R_{19}} = \frac{R_{25}}{R_{23}} = 2$ (21)

Poměr rezistorů by měl být 2

Zvolíme R21 = R25 = 2kΩ $\Rightarrow$ R19 = R23 = 1kΩ

Pro kontrolu vypočítáme hodnotu U_2 z takto navržených rezistorů:

Napětí na rezistoru R_{21}: $U_{R21} = U_1 \frac{R19}{R19 + R21} = 10.\frac{1000}{1000 + 2000} = 3{,}333V$ [V] (22)

Napětí na rezistoru R_{23}: $U_{R21} = U_{R23}$ [V] (23)

Výstupní napětí: $U_0 = \frac{U_{R23}(R25 + R23)}{R23} = \frac{3{,}333.3000}{1000} \doteq 10V$ [V] (24)

Aby bylo možné výstupní napětí nastavit na přesných 10V jsou místo pevných rezistorů R21 a R25 použité trimry o dvojnásobné hodnotě odporu. S těmito trimry máme rozsah pro nastavení výstupního napětí od cca. 3,5V až do cca. 10,5V. Vypočtené hodnoty rezistorů platí i pro druhý analogový výstup tedy pro rezistory R20,R22,R24,R26. Analogové výstupní napětí je přivedeno na svorkovnice ARK103/2. Jedná se o šroubovací svorkovnice do DPS, kam se vodiče připojují pod úhlem 45°.

4.6 Komunikační porty

4.6.1 Komunikační port RS-232

Komunikaci procesoru s okolím zabezpečuje plně duplexní sériový kanál. Přes tento sériový kanál je možno jednak komunikovat s okolím, ale také je umožněno samotný mikrokontrolér programovat. Protože použitý mikrokontrolér podporuje sériovou komunikaci UART, je zapotřebí už jen převodník úrovní napětí. V této aplikaci využíváme pouze tři vodiče linky RS-232 a to vodič RxD, TxD a GND. Je možné využívat i ostatní vodiče (dva výstupní a čtyři vstupní). Význam jednotlivých vodičů je uveden v tabulce *Tab 4.9*.

Vývod	Význam
1	Vstup – DCD(Data Carrier Detect) – Detektor přijímaného signálu
2	Vstup – RxD(Receive Data) – Přijímaná data
3	Výstup – TxD(Transmit Data) – Vysílaná data
4	Výstup – DTR(Data Terminal Ready) – Pohotovost DTE
5	GND(Ground) – Signálová zem
6	Vstup – DSR(Data Set Ready) – Pohotovost DCE
7	Výstup – RTS(Request To Send) – Požadavek k vysílání
8	Vstup – CTS(Clear To Send) – Pohotovost k vysílání
9	Vstup – RI(Ring Indikator) – Indikátor volání

Tab. 4.9 Význam jednotlivých vodičů na lince RS232

RS-232 používá dvě napěťové úrovně. Logickou 1 a 0. Log. 1 je někdy označována jako *marking state* , nebo také klidový stav, Log. 0 se přezdívá *space state*. Log. 1 je indikována zápornou úrovní, zatímco logická 0 je přenášena kladnou úrovní výstupních vodičů. Povolené napěťové úrovně jsou uvedeny v *Tab. 4.10*.

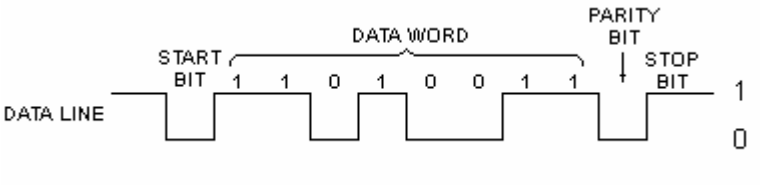
Úroveň	Vysílač	Přijímač
Log. 0	+5V až +15v	+3V až +25V
Log. 1	-5V až -15V	-3V až -25V
Nedefinovaný	-3V až +3V	

Tab. 4.10 Napěťové úrovně pro RS232

RS-232 používá asynchronní přenos informací. Každý přenesený byte konstantní rychlostí je proto třeba synchronizovat. K synchronizaci se používá sestupná hrana Start bitu. Za ní již následují posílaná data. Jak ukazuje *Obr. 4.22* začíná asynchronní přenos dat tzv. start-bitem, což je přechod signálu z log. 1 na log. 0(sestupná hrana). Následují datové bity (nesou informaci; zde délky 8 bitů s obsahem 11010011_2), první je nejméně významný bit. Za datovými bity může být paritní bit(na *Obr. 4.22* není) a rámec je ukončen stop-bitem(přechod do log. 1).

Paritní bit slouží pro zabezpečení přenosu. Obsahuje totiž informaci o stavu datových bitů. Při sudé paritě je paritní bit nastaven tak, aby celkový počet jedniček v datových bitech a paritní bit byl sudý. Při liché paritě je to naopak.

Při asynchronním přenosu není třeba přenášet hodinový signál, protože přijímač i vysílač musí být nastaveny na stejnou přenosovou rychlost. Synchronizace obou obvodů proběhne pomocí start-bitu. Stop-bit vytváří prodlevu nutnou pro zpracování dat přijímačem a vyhodnocení úspěšnosti přenosu.



Obr. 4.22 Asynchronní sériový přenos

Použito z [25]

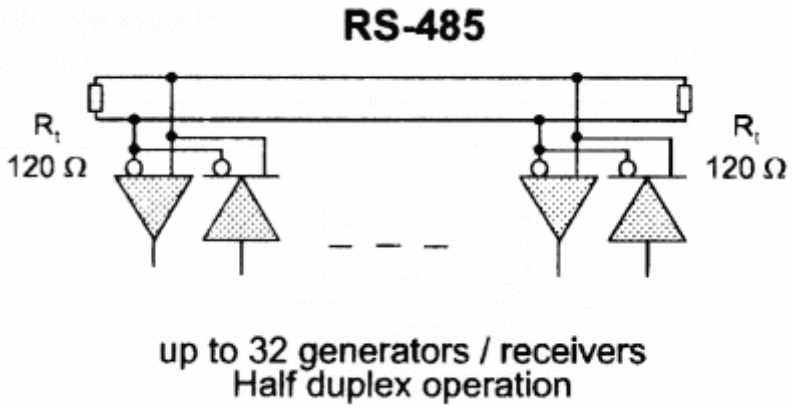
Logické úrovně TTL a RS-232 jsou mezi sebou převáděny obvodem firmy Maxim MAX232[8]. Ten obsahuje dvojici převodníků TTL/RS-232 a měnič napětí 5V/$\pm$12V. Zabudovaný měnič napětí na principu nábojové pumpy potřebuje ke své činnosti pouze čtyři externí kondenzátory C10,C11,C12 a C13. Hodnoty těchto kondenzátorů jsou zvoleny podle doporučení výrobce pro daný typ a to na hodnotu 100nF. Napájecí napětí pro tento převodník je 5V. Sériová linka je vyvedena na konektor CANON9 do desky plošných spojů.

4.6.2 Komunikační port RS-485

Pro přenos dat mezi zařízeními na delší vzdálenosti se často používá sériová komunikace RS-485. Lze ji doporučit pro nejrůznější přenosy dat v průmyslovém prostředí. Při správném provedení je spolehlivost přenosu v porovnání s linkou RS-232, nebo proudovou smyčkou vysoká. RS-485 je tedy průmyslová sběrnice, která slouží přenos informace na vzdálenost až 1600 m a pro malý objem dat (1 až 100 kbitů za sekundu).

Asynchronní komunikace minimalizuje počet vodičů potřebných k přenosu, čímž se zlevňuje komunikační vedení. Komunikace probíhá po 2 zkroucených vodičích (twistový pár), které jsou označované „A“ a „B“ a jsou vysílačem buzeny v protifázi. Přijímač vyhodnocuje jejich napěťový rozdíl. Tímto principem se odstraní součtové (aditivní) rušení.

U rozvětvených linek může být počet zařízení na lince maximálně 16, avšak existují přijímače s menší zátěží, takže jich může být až 128. Linka by měla být provedena jako linie s krátkými odbočkami a ne jako strom, nebo hvězda viz *Obr. 4.23.*



Obr. 4.23 Rozvětvená RS-485

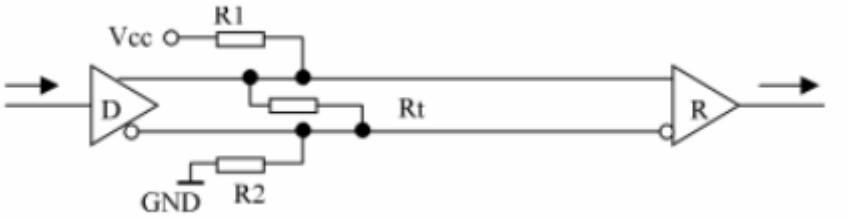
Použito z [26]

Napěťové úrovně linky RS-485 jsou menší v porovnání s RS-232, typický napěťový rozdíl mezi vodiči je 2 V. Aby přijímač mohl pracovat diferenciálně, nesmí být rozdíl mezi zemí vysílače a zemí přijímače větší než 7 V. V opačném případě se vstupy přijímače zahltí a dojde k přerušení komunikace.

Linka RS-485 používá jeden pár vodičů pro oba směry toku dat. Je tedy třeba směr komunikace přepínat a to může být problém zvláště v případech, kdy s touto možností software nepočítá. Přepínání směru komunikace jistě bude vyřešeno u zařízení, které obsahuje už standardně linku RS-485. Pokud však používáme zařízení s vyvedenou linkou RS-232 (například počítač PC) a následným převodníkem RS-232/RS-485, je třeba přepínání směru zajistit.

Impedanční zakončení linky RS-485 je věc dosti problematická. Samozřejmě je nutné na konce linky zapojit rezistor o shodné hodnotě s impedancí vedení a tím zabránit odrazům na vedení. Vhodné je volit zakončení okolo 120Ω.

Stejně jako je nutné impedanční zakončení tak je také definování klidového stavu linky. Protože při komunikaci po lince RS-485 se vysílače odpojují, dochází k dobám, kdy na linku žádné zařízení nevysílá. V této době není stav linky definován a linka je extrémně citlivá na indukovaná napětí (poruchy), které se jeví jako přicházející data. Proto je třeba definovat klidový stav linky připojením rezistorů podle *Obr. 4.24* (předpokládáme, že v klidu je vodič B zápornější než A.



Obr. 4.24 Zapojení impedančního zakončení RS485

Použito z [27]

Komunikaci mikrokontroléru s okolím pomocí linky RS-485 zabezpečuje u indikátoru převodník úrovní ze sběrnice RS485 na TTL úroveň LTC485[9]. Tento převodník je připojen jednak přímo na sběrnici RS-485 a jednak na mikrokontrolér (piny P3.0 – Rxd, P3.1 – Txd, P3.2).

Jelikož RS-485 používá pouze jeden pár vodičů pro oba směry toku dat, je třeba komunikaci přepínat. K tomu slouží pin P3.2. U převodníku se jedná o piny RE a DE. Při log. 0 na P3.2 se bude jednat o přijímač. Při log. 1 na P3.2 se bude jednat o vysílač. Použitý rezistor R4 – tzv. terminační rezistor o hodnotě 120R je zde připojen přes jumper JP2 tudíž je možno rezistor odpojit.

Maximální přenosová rychlost pro sběrnici RS-485 je 10Mb/s, ovšem dosažení tak vysoké rychlosti v průmyslových podmínkách bývá často nemožné a mnohdy nebývá ani potřebné, proto jsou v obvyklé praxi používané rychlosti mnohdy až o několik řádů nižší. Pro indikátor je to 9600 - 115200b/s.

Unipolární transily D3 a D4(SMBJ6.8A) (prvky přepěťové ochrany) 6.8V zajišťují ochranu vodičů A a B proti lokální (oddělené) zemi budiče a přijímače linky RS485. Rezistory R3 a R5 nám definují již zmiňovaný klidový stav linky.

Problém u použitého mikrokontroléru je ten, že má pouze jeden sériový kanál. V daný časový okamžik proto může být aktivní pouze jedno rozhraní. Pro přepínání rozhraní slouží přepínač S10 umístěný za konektorem pro RS-232 a svorkovnicí pro RS-485.

4.7 Napájecí zdroj

Celý indikátor je napájen ze zdroje stejnosměrného napětí 24V. Jelikož použité periferie potřebují různá napájecí napětí je nutné napájecí napětí upravit a to na hodnoty +12V pro operační zesilovač TLC272, +5V (pro zobrazovací modul, mikrokontrolér, vstupní obvody, A/D, D/A převodníky, elektronický potenciometr, kodér, komunikační obvody).

Pro úpravu napětí + 12V je použitý klasický lineární stabilizátor z řady 7812(7812 CD2T SMD[15]) v pouzdru D^2PAK.

Maximální hodnoty tohoto stabilizátoru:

Pro výstupní napětí 5 – 18V je max. vstupní 35V
Pro výstupní napětí 20,24V je max. vstupní 40V
Maximální výstupní proud 1,5A

Pro úpravu napětí na +5V již není použitý klasické stabilizátor, ale DC/DC měnič firmy TRACO a to typ TEN3-2411[14]. Jedná se o DC/DC měnič vysoké účinnosti, která dosahuje u tohoto typu 79% . Maximální proudové zatížení je 500mA což pro náš indikátor stačí.

V zapojení napájecí části je dále také použita dioda D2 jako ochrana proti přepólování. Ochrana proti přepětí je na vstupu transil D1 na napětí 30V. Kondenzátory C18 – C22 slouží jako blokovací pro integrované obvody(A/D, D/A, AT89C51ED2, MAX323CWE, LTC485). Chod každého dílčího zdroje nám signalizují diody LED1 a LED2.

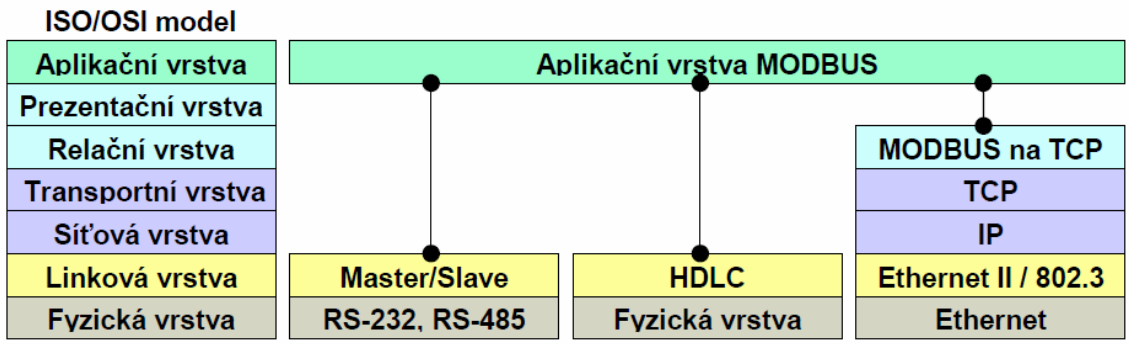
Výpočet hodnoty rezistoru pro LED1:

$$R_1 = \frac{U - U_{LED}}{I_{LED}} = \frac{5 - 2{,}1}{0{,}01} \doteq 1k\Omega \qquad [\Omega] \tag{25}$$

Pro napájecí větev +5V vychází hodnota rezistoru pro indikační LED2 zhruba 240Ω. Jak již bylo uvedeno u digitálních výstupů, z důvodu úspory proudové zátěže volíme hodnoty rezistorů 330 Ω pro všechny indikační LED diody (toto neplatí pro indikační LED u digitálních vstupů), které nejsou napájené z větve +5V.

5 PROTOKOL MODBUS

MODBUS je komunikační protokol na úrovni aplikační vrstvy ISO/OSI modelu, umožňující komunikaci typu klient-server mezi zařízeními na různých typech sítí a sběrnic. Vytvořen v roce 1979 firmou MODICON. V současné době je podporována celá řada komunikačních médií např. sériové linky typu RS-232, RS-422, RS-485, optické a rádiové sítě nebo Ethernet s využitím protokolu TCP/IP. Komunikace probíhá metodou požadavek-odpověď a požadovaná funkce je specifikována pomocí kódu funkce jež je součástí požadavku.

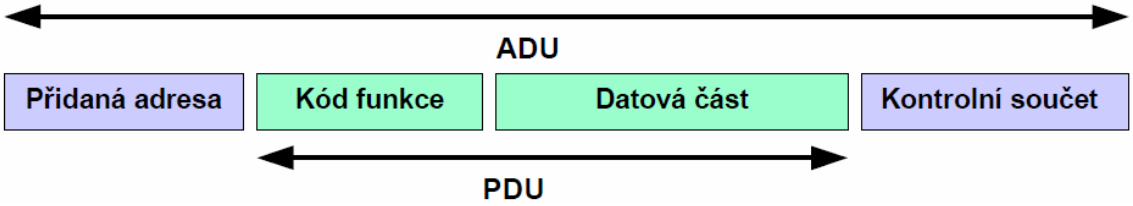


Obr. 5.1 Příklady implementace

Použito z [28]

5.1 Popis protokolu

Protokol MODBUS definuje strukturu zprávy na úrovni protokolu (PDU–Protokol Data Unit) nezávisle na typu komunikační vrstvy. V závislosti na typu sítě, na které je protokol použit, je PDU rozšířena o další části a tvoří tak zprávu na aplikační úrovni (ADU – Application Data Unit).



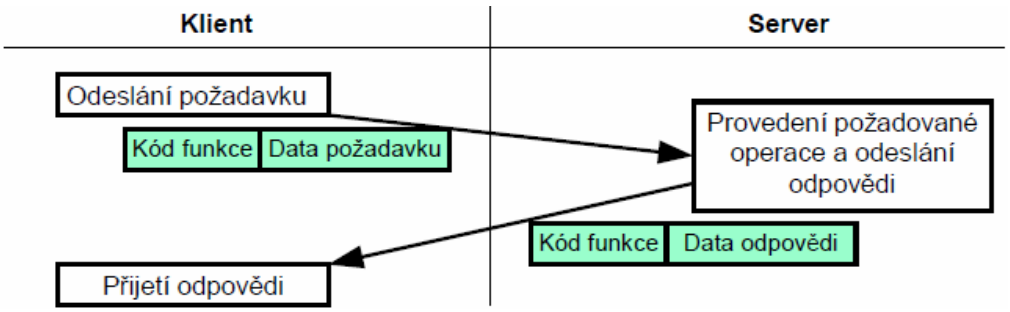
Obr. 5.2 Základní tvar MODBUS zprávy

Použito z [28]

Kód funkce udává serveru jaký druh operace má provést. Rozsah kódů je 1 až 255, přičemž kódy 128 až 255 jsou vyhrazeny pro oznámení záporné odpovědi(chyby). Některé kódy funkcí obsahují i kód podfunkce upřesňující blíže požadovanou operaci. Obsah datové části zprávy poslané klientem slouží serveru k uskutečnění operace určené kódem funkce. Obsahem může být například adresa

a počet vstupů, které má server přečíst, nebo hodnota registrů, které má server zapsat. U některých funkcí nejsou pro provedení operace zapotřebí další data a v tom případě může datová část ve zprávě úplně chybět.

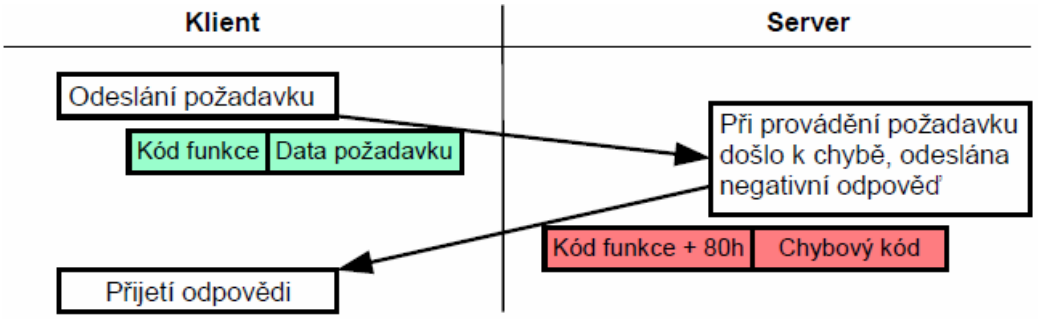
Pokud při provádění požadované operace nedojde k chybě(*Obr. 5.3*), odpoví server zprávou, která v poli Kód funkce obsahuje kód provedené(požadované) funkce jako indikaci úspěšného vykonání požadavku. V datové části odpovědi předá server klientovi požadovaná data(pokud nějaká jsou).



Obr. 5.3 MODBUS transakce s bezchybným provedením požadavku

Použito z [28]

Pokud při vykonávání požadované operace dojde k chybě (*Obr. 5.4*), je v poli Kód funkce vrácen kód požadované funkce s nastaveným nejvyšším bitem indikující neúspěch. V datové části je vrácen chybový kód upřesňující důvod neúspěchu.



Obr. 5.4 MODBUS transakce s chybou při provádění požadavku

Použito z [28]

Max. velikost PDU na sériové lince = 256 – adresa(1 byte) – kontrolní součet CRC (2 byty) = ***253bytů***
Velikost ***ADU na RS-485*** = 253bytů PDU + adresa(1 byte) + CRC(2 byty) = ***256bytů***

Protokol MODBUS definuje 3 základní typy zpráv(PDU):

- ***Požadavek(Request PDU)*** – 1 byte Kód funkce, n bytů Datová část požadavku (adresa, proměnné, počet proměnných…)
- ***Odpověď'(Response PDU)*** – 1 byte Kód funkce, m bytů Datová část odpovědi (přečtené vstupy, stav zařízení…)
- **Záporná odpověď'(Exception Response PDU)** – 1 byte Kód funkce + 80h (indikace neúspěchu), 1 byte Chybový kód(identifikace chyby)

5.2 Kategorie kódů funkcí

MODBUS protokol definuje tři skupiny kódů funkcí:

Veřejné kódy funkcí

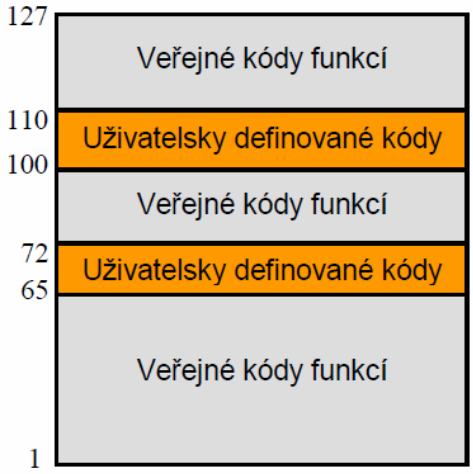
- jasně definované
- je garantovaná unikátnost
- veřejně zdokumentované
- schváleny společností MODBUS-IDA.org

Uživatelsky definované kódy funkcí

- dva rozsahy uživatelsky definovaných funkcí (65 – 72 a 100 – 110)
- umožňují uživateli implementovat funkci, která není definovaná touto specifikací
- není garantována unikátnost

Rezervované kódy funkcí

- kódy funkcí, které jsou v současnosti používány některými firmami a které nejsou dostupné pro veřejné použití



Obr. 5.5 Kategorie funkčních kódů

Použito z [28]

5.3 Definice funkčních kódů

				Kódy funkcí		
				Kód	Podfunkce	hex
Přístup k datům	Bitový přístup	Fyzické diskrétní vstupy	Čti diskrétní vstupy	02		02
		Interní bity nebo fyzické cívky	Čti cívky	01		01
			Zapiš jednu cívku	05		05
			Zapiš více cívek	15		0F
	16-bitový přístup	Fyzické vstupní registry	Čti vstupní registr	04		04
		Interní registry nebo fyzické výstupní registry	Čti uchovávající registry	03		03
			Zapiš jeden registr	06		06
			Zapiš více registrů	16		10
			Čti/zapiš více registrů	23		17
			Zapiš registr s maskováním	22		16
			Čti FIFO frontu	24		18
	Přístup k záznamům v souborech		Čti záznam ze souboru	20	6	14
			Zapiš záznam do souboru	21	6	15
Diagnostika			Čti stav	07		07
			Diagnostika	08	00-18,20	08
			Čti čítač kom.událostí	11		0B
			Čti záznam kom. událostí	12		0C
			Sděl identifikaci	17		11
			Čti identifikaci zařízení	43	14	2B
Ostatní			Zapouzdřený přenos CAN základní odkaz	43 43	13,14 13	2B 2B

Tab. 5.1 Definice funkčních kódů

Použito z [28]

5.4 Záporné odpovědi

Když klient posílá serveru požadavek, očekává na něj odpověď. Mohou nastat čtyři situace:

- Jestliže server přijme bezchybně požadavek a je schopen jej normálně zpracovat, vrátí klientovi normální odpověď.
- Jestliže server požadavek nepřijme z důvodu komunikační chyby, není vrácena žádná odpověď. Na straně klienta dojde k vypršení časového limitu pro příjem odpovědi.
- Jestliže server přijme požadavek, ale detekuje komunikační chybu(parita, CRC...), nevrací žádnou odpověď. Na straně klienta dojde k vypršení časového limitu pro příjem odpovědi.
- Jestliže server přijme bezchybně požadavek, ale není schopen jej normálně zpracovat, vrátí klientovi zápornou odpověď s udáním důvodu neúspěchu.

Normální a záporná odpověď se liší nejvyšším bitem kódu funkce. Je-li bit nulový,jedná se o normální odpověď, je-li bit nastavený, jedná se o zápornou odpověď. V případě záporné odpovědi je v datové části předán kód chyby. V následující *tab. 5.2* je seznam možných chybových kódů.

MODBUS chybové kódy		
Kód	Název	Význam
01	Ilegální funkce	Požadovaná funkce není serverem podporovaná
02	Ilegální adresa dat	Zadaná adresa je mimo serverem podporovaný rozsah
03	Ilegální hodnota dat	Předávaná data jsou neplatná
04	Selhání zařízení	Při provádění požadavku došlo k neodstranitelné chybě
05	Potvrzení	Kód určený k použití při programování. Server hlásí přijetí platného požadavku, ale jeho vykonání bude trvat delší dobu
06	Zařízení je zaneprázdněné	Kód určený k použití při programování. Server je zaneprázdněn vykonáváním dlouho trvajícího příkazu.
08	Chyba parity paměti	Kód určený k použití při práci se soubory. Server při pokusu přečíst soubor zjistil chybu parity.
0A	Brána – přenosová cesta nedostupná	Kód určený k práci s bránou(gateway). Brána není schopná vyhradit interní přenosovou cestu od vstupního portu k výstupnímu. Pravděpodobně je přetížená, nebo nesprávně nastavená.
0B	Brána – cílové zařízení neodpovídá	Kód určený k práci s bránou(gateway). Cílové zařízení neodpovídá, pravděpodobně není přítomno.

Tab. 5.2 MODBUS chybové kódy

Použito z [28]

5.5 MODBUS na sériové lince

MODBUS Seriál Line protokol je protokol typu Master-Slave a je definován na úrovni 2 ISO/OSI modelu. Na fyzické úrovni 0 ISO/OSI modelu můžou být použita různá sériová rozhraní, například RS-232, nebo RS-485 a jejich varianty.

5.5.1 Princip protokolu

Jedná se o Master/Slave protokol. V jeden okamžik může být na sběrnici pouze jeden master a 1 až 247 slave jednotek. Komunikaci vždy zahajuje master, slave nesmí nikdy vysílat data bez pověření mastera.
Master posílá požadavky slave jednotkám ve dvou režimech:
unicast režim – master adresuje jedné konkrétní slave jednotce a ta pošle odpověď.
broadcast režim – master posílá požadavek všem jednotkám, žádná jednotka neodpoví.

5.5.1.1 Adresovací pravidla

Adresní prostor zahrnuje 256 různých adres.

0	1 až 247	248 až 255
Broadcast adresa	Individuální adresa slave jednotky	Rezervováno

Tab. 5.3 Adresní prostor

Master nemá žádnou specifickou adresu, pouze slave jednotky musejí mít adresu a ta musí být v celé MODBUS síti jedinečná.

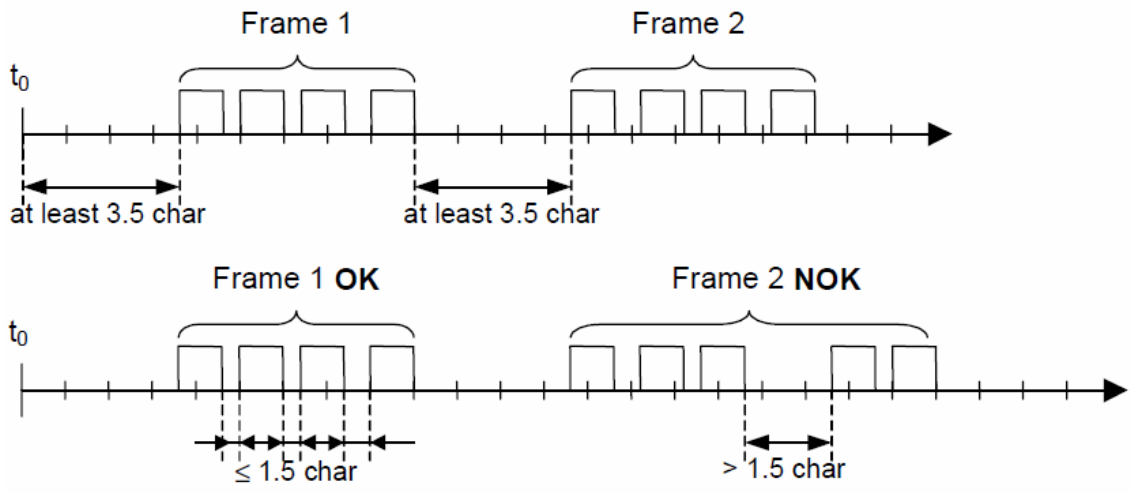
Na *Obr. 5.2* je znázorněn základní formát MODBUS aplikační zprávy na sériové lince. Zpráva kromě standardní MODBUS PDU obsahuje pole ***Adresa jednotky***. Toto pole obsahuje adresu slave jednotky. Pole ***Kontrolní součet*** slouží k detekci chyb a obsahuje ***CRC*** kód v závislosti na vysílacím režimu.

5.5.1.2 Vysílací režimy

MODBUS protokol definuje dva sériové vysílací režimy, MODBUS RTU a MODBUS ASCII. Režim určuje v jakém formátu jsou data vysílána a jak dekódována. Každá jednotka musí podporovat režim RTU, režim ASCII je nepovinný. Všechny jednotky na jedné sběrnici musejí pracovat ve stejném vysílacím režimu. Dále bude popsán pouze režim RTU jelikož z důvodu zjednodušení je v indikátoru implementován pouze tento režim.

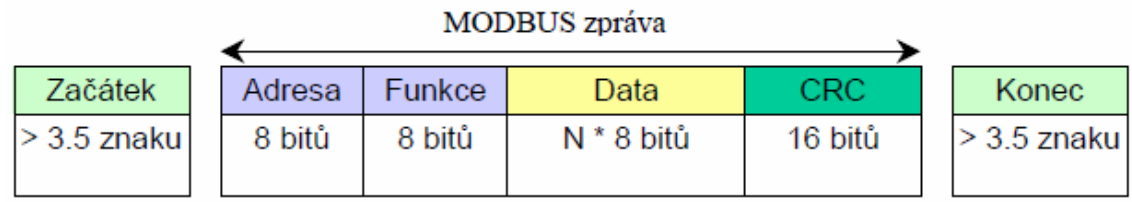
5.5.1.3 MODBUS RTU

Vysílání zprávy musí být souvislé, mezery mezi znaky nesmějí být delší než 1,5 znaku. Začátek a konec zprávy je identifikován podle pomlky na sběrnici delší než 3,5 znaku viz. *Obr. 5.6*. Formát RTU rámce je znázorněn na *Obr. 5.7*.



Obr. 5.6 Časové intervaly 1,5 a 3,5 char mezi znaky

Použito z [28]



Obr. 5.7 RTU rámec zprávy

Použito z [28]

K detekci chyb slouží 16-bitové CRC pole s generujícím polynomem $x^{16} + x^{15} + x^2 + 1$

Formát znaku(11 bitů):
- 1 start bit
- 8 datových bitů
- 1 bit parita
- 1 stop bit

Každá jednotka musí podporovat sudou paritu. Pokud není použita parita, je nahrazena druhým stop-bitem.

6 REALIZACE INDIKÁTORU

6.1 Deska plošných spojů

Celý indikátor je realizován na dvou deskách plošných spojů. První z nich obsahuje svorkovnici pro připojení napájecího napětí +24V, spínaný zdroj TRACO POWER, stabilizátor 7812CD2T, mikrokontrolér AT89C51ED2, 2 kanály analogových vstupů, 2 kanály analogových výstupů, 8 digitálních vstupů, 8 digitálních výstupů, indikační LED diody jak pro digitální vstupy tak pro digitální výstupy, kodér, krystalový oscilátor, komunikační konektor CANON9Z pro linku RS-232, svorkovnici pro linku RS-485, dvojici tlačítek (RESET, PSEN), vyvedenou sběrnici SPI konektorem MLW06, přepínač linek RS-232/RS485, svorkovnice pro (digitální vstupy, digitální výstupy, analogové vstupy, analogové výstupy), konektory BL10G pro připojení modulu s LCD displejem a pasivní součástky pro správnou funkci všech dílčích částí. Jedná se o oboustrannou desku s prokovenými otvory o rozměrech 125x154mm.

Druhá deska plošných spojů obsahuje samotný LCD displej, který se připojuje pomocí konektoru Hirose DF23, dvě kolíkové lišty(10-pinovou a 6-pinovou) pro komunikaci modulu a pro čtení stavu tlačítek, sadu šesti tlačítek pro ovládání, na spodní straně desky dále spínaný zdroj pro podsvětlení LCD, stabilizátor LF33 pro napájení LCD, inventor 74HC4050 a elektronický potenciometr. Jedná se o oboustrannou desku s prokovenými otvory o rozměrech 48x73mm. Celý modul s LCD displejem drží na spodní desce pomocí distančních sloupků vysokých 12mm.

6.1.1 Osazení a oživení desek plošných spojů

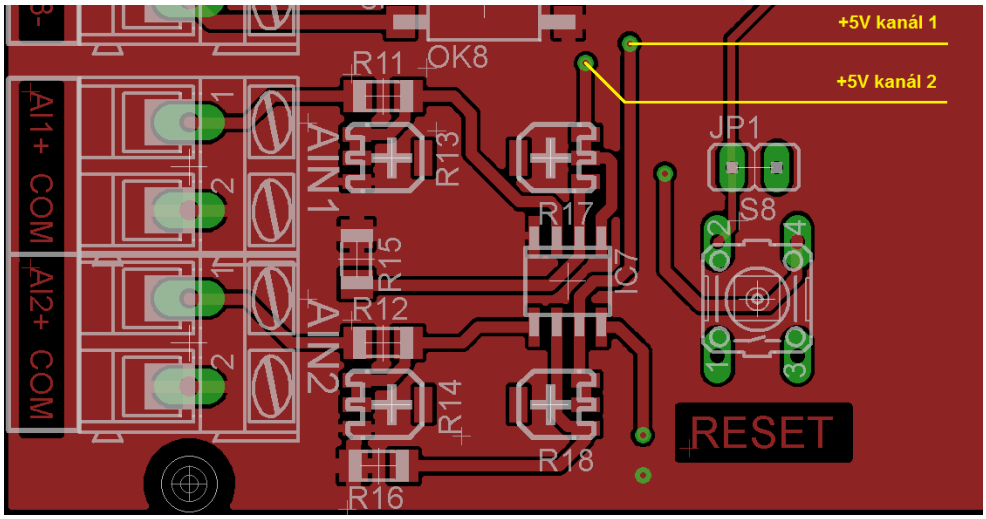
Před začátkem osazování desek se ještě jednou přesvědčíme, že jsou desky v pořádku, a že na nich není nějaký problém, který mohl vzniknout při výrobě či při návrhu. Při osazování začínáme u pasivních součástek(rezistory, kondenzátory). Poté pokračujeme v osazování ostatních součástek(LED, diody, tranzistory, trimry, konektory, svorkovnice…) podle velikosti, ale neosazujeme plošný spoj integrovanými obvody. Jestliže ale integrovaný obvod má patici tak osadíme plošný spoj všemi paticemi. Poté osadíme spínaný zdroj a lineární stabilizátor. U desky modulu s LCD displejem postupujeme stejně jako u hlavní desky a dáváme obzvlášť pozor při pájení konektoru pro LCD displej. Když máme všechno(doposud zmíněné) osazené můžeme hlavní desku připojit +24V(desku modulu pro LCD zatím nepřipojujeme). Po připojení napětí by se měly hnedka rozsvítit dvě červené indikační LED diody signalizující +12V a +5V. Jestliže je všechno bez problémů a rozsvítily se nám obě LED diody můžeme přistoupit ke kontrole napájení všech integrovaných obvodů a konektoru pro modul LCD. Voltmetrem ještě pro kontrolu změříme, zda máme na výstupu spínaného zdroje +5V a na výstupu stabilizátoru +12V. Poté přistoupíme k samotným integrovaným obvodům. Měříme napětí +5V na pinech mikrokontroléru AT89C51ED2(pin 44 a pin

35), kodéru 74HC148(pin 16), A/D MCP3202(pin 8), D/A MCP4922(pin 1 a 9), budiče MAX232ECWE(pin 16), budiče LTC485(pin 8) a konektoru pro připojení modulu (pin 1). Poté ještě zkontrolujeme napětí +12V pro operační zesilovače TLC272(pin 8). Zda-li je všechno v pořádku můžeme osadit jednotlivé integrované obvody, ale zatím bez A/D převodníku a D/A převodníku. Když máme vše osazené připojíme napájecí napětí +24V a poté připojíme také napětí +10V na analogové vstupy. Nastavíme trimry R13,R17,R14,R18 vstupní napětí pro A/D převodník na +5V. Měříme na bodech viz. *Obr. 6.1*. Když máme obě nastavené odpojíme obě napětí a můžeme osadit A/D převodník. Poté připojíme zas obě napětí a doladíme vstupní napětí +5V pro oba kanály. U D/A převodníku zkontrolujeme napětí +2,5V(pin 11 a 13) jestliže souhlasí můžeme také D/A převodník osadit(samozřejmě bez napětí). Výstupní napětí D/A převodníku je potom možné doladit trimry R23, R24, R25, R26.

U desky modulu si musíme pomoc z externího zdroje +5V, které přivedeme na pin 1 a 0V na pin 10 konektoru X1_LCD. U modulu zkontrolujeme napětí +5V na pinech měniče MC34063A(pin 6), elektronického potenciometru(pin 6), vstupu stabilizátoru LF33. Dále zkontrolujeme napětí +3,3V na pinech inventoru 74HC4050(pin 1), LCD(pin 1 a 6). Je-li vše v pořádku můžeme osadit všechny integrované obvody.

Jestliže máme všechny napětí zkontrolovány tak ještě zkontrolujeme případný zkrat mezi piny 2,3,5 u konektoru pro sériový kanál. Když je i to v pořádku můžeme přistoupit k nahrání programu do mikrokontroléru. Po nahrání programu vypneme napájecí napětí a připojíme modul, ale zatím bez LCD. Připojíme napětí a po chvilce by jsme měli naměřit cca +7V na pinu 10 u konektoru pro LCD. Zda-li je to v pořádku můžeme vypnout napájecí zdroj a osadit samotný LCD displej.

Jestliže jsme při osazování zejména konektoru pro LCD a elektronického potenciometru postupovali pečlivě tak zapojení funguje napoprvé.



Obr. 6.1 Měřící body pro A/D převodník

7 PROGRAMOVÉ VYBAVENÍ MIKROKONTROLÉRU

Celý program je napsán ve vývojovém prostředí µVision V4.02 od firmy Keil. Je vytvořen celkem ze sedmi modulů psaných v jazyce C. Ke každému souboru(.c) je k dispozici příslušný hlavičkový soubor(.h). Výjimku tvoří soubor projekt.c, ve kterém je uvedena jako jediná funkce *main()*. Seznam jednotlivých modulů a k nim vložených hlavičkových souborů viz *tab. 7.1*.

Název modulu	Vložené hlavičkové soubory
projekt.c	reg51xd2.h, spi.h, lcd.h, modbus.h,screen_button.h
screen_button.c	reg51xd2.h, studio.h, stdlib.h, screen_button.h, spi.h, lcd.h modbus.h
lcd.c	reg51xd2.h, lcd.h, spi.h, font.h
spi.c	reg51xd2.h, spi.h
modbus.c	reg51xd2.h, modbus.h, spi.h, lcd.h, crc16.h, screen_button.h
crc16.c	reg51xd2.h, crc16.h
font.c	font.h

Tab. 7.1 Seznam modulů a k nim vložených hlavičkových souborů

7.1 Modul projekt.c

Jedná se o hlavní modul celého programu. Kromě vložení potřebných hlavičkových souborů modul obsahuje hlavní funkci *main()* a dále nekonečnou smyčku. V hlavní funkci se nejprve nastaví dvojnásobná rychlost vykonávání instrukcí(1/6 hodinového kmitočtu) v registru CKCON0, dále se nastaví velikost XRAM na 1792B v registru AUXR. Poté přicházejí na řadu inicializační funkce, kde se nejprve inicializuje komunikační sběrnice SPI, dále se inicializuje a vymaže samotný LCD displej(vykreslí bílá obrazovka), dále se inicializují použité proměnné pro komunikaci MODBUS, dále se inicializuje komunikace MODBUS a jako poslední se provede inicializace digitálních vstupů, výstupů a nastavení indikátoru. Po inicializaci přichází na řadu vykreslení úvodní obrazovky indikátoru. Po vykreslení se pošle příkaz do LCD(zapnutí LCD) a zapne se LED podsvětlení.

Po těchto krocích vstoupí program do nekonečné smyčky, kde se neustále dokola testuje stisknutí jakéhokoliv tlačítka, přítomnost příchozí zprávy protokolu MODBUS a cyklicky čte stav digitálních vstupů a zapisuje na digitální výstupy.

7.2 Modul screen_button.c

Jedná se o nejrozsáhlejší modul celého programu. Nejrozsáhlejší je z toho důvodu, protože v tomto modulu je obsaženo kompletní ovládání indikátoru, vykreslování obrazovek na LCD a cyklické čtení vstupů a výstupů. Implementované funkce můžeme rozdělit do čtyř logických celků a to 1. funkce pro jednotlivé tlačítka, 2. funkce pro vykreslení jednotlivých obrazovek, 3. inicializační funkci pro vstupy, výstupy a nastavení indikátoru, 4. cyklické čtení vstupů a zápis výstupů.

7.2.1 Funkce tlačítek

V cyklicky čtené funkci *buton()* z hlavní smyčky se vykonávají dvě úlohy. První z nich je reset časovače 1 při každém stisknutí jakéhokoliv tlačítka(toto neplatí při ovládání přes MODBUS). Při uplynutí 1 minuty se vypíná podsvětlení LCD displeje. Zapnutí podsvětlení se provede stiskem jakéhokoliv tlačítka. Druhá úloha spočívá v testování horních 3 bitů portu P3(provádí se bitový součin portu P3 a čísla 0xE0) a podle stavu jednotlivých bitů vybírá funkci, která se má volat. Při ovládání indikátoru přes protokol MODBUS je situace stejná. Stavy jednotlivých tlačítek jsou kódované stejně jako hardwarové tlačítka. Kombinace jednotlivých bitů je znázorněna v *tab. 7.2*.

P3.7	P3.6	P3.5	hex	Funkce
1	1	0	0xC0	function_button_up()
1	0	1	0xA0	function_button_setting()
1	0	0	0x80	function_button_down()
0	1	1	0x60	function_button_select()
0	1	0	0x40	function_button_menu()
0	0	1	0x20	function_button_back()

Tab. 7.2 Kombinace jednotlivých tlačítek

Předtím než se ale volají jednotlivé funkce se testuje zda-li není zapnuté cyklické čtení vstupů a výstupů přes protokol MODBUS. Při zapnutí cyklického čtení nejsou aktivní ani hardwarová ani softwarová tlačítka. Toto řešení je použito z toho důvodu, protože mikrokontrolér není schopen obsluhovat jednak přerušení od sériové linky(každých 200ms) a jednak obsluhovat LCD displej. Interval 200ms je zvolený proto, aby se daly odečítat hodnoty jak digitálních tak analogových vstupů při rychlejších dynamických procesech.

Funkce *function_button_up*

Tato funkce tlačítka je aktivní v případek, když potřebujeme pohybovat kurzor v hlavním menu, digitálních vstupech, digitálních výstupech, analogových výstupech, nastavení, nastavení LCD. Dále je aktivní při nastavování hodnot u analogových výstupů, MODBUS adresy, komunikační rychlosti, parity, LCD kontrastu a LCD podsvětlení.

Funkce *function_button_setting*

Tato funkce tlačítka je aktivní v případek, když potřebujeme označit a potvrdit nastavovanou veličinu(D/A převodník, MODBUS adresa, komunikační rychlost, parita). U digitálních vstupů toto tlačítko slouží pro pevné nastavení vstupu bez ohledu na aktuální stav vstupu(force). U digitálních výstupů tímto tlačítkem můžeme nastavovat jednotlivé výstupy. V nastavení LCD je změna v tom, že pohyb kurzoru je realizován tímto tlačítkem a ne tlačítkem UP nebo DOWN.

Funkce *function_button_down*

Tato funkce tlačítka je aktivní v případek, když potřebujeme pohybovat kurzor v hlavním menu, digitálních vstupech, digitálních výstupech, analogových výstupech, nastavení, nastavení LCD. Dále je aktivní při nastavování hodnot u analogových výstupů, MODBUS adresy, komunikační rychlosti, parity, LCD kontrastu a LCD podsvětlení.

Funkce *function_button_select*

Tato funkce tlačítka je aktivní v případech, když potřebujeme vstoupit do jednotlivých položek hlavního menu, nebo vstoupit do položky nastavení LCD. Na LCD se nad tímto tlačítkem v modré spodní liště v pravém dolním rohu zobrazí ZVOLIT když je tlačítko aktivní.

Funkce *function_button_menu*

Tato funkce tlačítka je aktivní v případech, když potřebujeme vstoupit hlavního menu. Na LCD se nad tímto tlačítkem v modré spodní liště uprostřed zobrazí MENU když je tlačítko aktivní.

Funkce *function_button_back*

Tato funkce tlačítka je aktivní v případech, když potřebujeme udělat krok zpět z jakéhokoliv submenu či hlavního menu. Na LCD se nad tímto tlačítkem v modré spodní liště v levém dolním rohu zobrazí ZPET když je tlačítko aktivní.

7.2.2 Funkce pro vykreslování obrazovek

Funkce *main_page()*

Tato funkce slouží pro vykreslení hlavní obrazovky indikátoru. Nejprve se vykreslí horní a spodní modrá lišta, dále ve spodní lište uprostřed MENU, dále se vykreslí samotný název zařízení tedy PRUMYSLOVY STAVOVY INDIKATOR a jako poslední se vykreslí v pravém dolním rohu nad modrou spodní lištou VUT BRNO. Funkce pro vykreslování řetězce znaků a modrých obdélníků bude popsána později.

Po stisknutí tlačítka(MENU) pro vstup do menu se nejdříve vykreslí MENU orámované a uskočené o jeden sloupec doleva a o jeden řádek dolů. Poté se smaže nápis PRUMYSLOVY STAVOVY INDIKATOR, VUT BRNO, MENU v modré spodní liště a zavolá funkce *main_menu()*. Tento celý proces je ve funkci *function_button_menu.*. Celá vykreslená obrazovka je na *Obr. 7.1*.

Funkce *main_menu()*

Tato funkce slouží pro vykreslení obrazovky s nabídkou hlavního menu. Nejprve se vykreslí ZPET ve spodní liště vlevo nad tlačítkem BACK, ZVOLIT ve spodní liště vpravo nad tlačítkem SELECT, MENU v horní liště uprostřed. Poté se vykreslí již samotné položky menu (DIGITALNI VSTUPY, DIGITALNI VYSTUPY, ANALOGOVE VSTUPY, ANALOGOVE VYSTUPY, NASTAVENI). Jako poslední se vykreslí značka kurzoru u dané položky menu. Pohyb tohoto kurzoru zle ovládat již zmiňovanými tlačítky UP a DOWN. Celá vykreslená obrazovka je na *Obr. 7.2.*

Po stisknutí tlačítka SELECT pro vstup do libovolné položky menu se nejdříve vykreslí ZVOLIT orámované a uskočené o jeden sloupec doleva a o jeden řádek dolů. Stejné orámování a uskočení je i u zvolené položky. Poté se smažou jednotlivé položky menu, MENU v horní liště, orámované ZVOLIT ve spodní liště a zavolá se funkce podle vybrané položky menu(*digital_input(), digital_output(), analog_input(), analog_output(), nastaveni()*). Toto vše je obsaženo ve funkci *function_button_select().*

Po stisknutí tlačítka(BACK) se nejdříve vykreslí orámované a uskočené ZPET, smaže se nápis ZVOLIT, jednotlivé položky menu, MENU v horní liště a zavolá se funkce *main_page().*Toto vše je obsaženo ve funkci *function_button_back().*

Obr. 7.1Obrazovka hlavní stránky

Obr. 7.2 Obrazovka hlavního menu

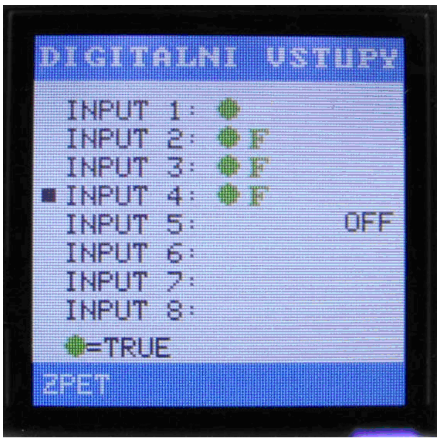
Funkce *digital_input()*

Tato funkce slouží pro vykreslení obrazovky s jednotlivými digitálními vstupy. Nejprve se vykreslí DIGITALNI VSTUPY v horní modré liště, poté jednotlivé vstupy(INPUT 1: až INPUT 8:), NO/OFF u tlačítka SETTING podle aktuálního stavu vstupu, informace o signalizaci pro log. 1. Jako poslední se vykreslí značka kurzoru u daného vstupu. Zelené F u zelené signalizace log. 1 znamená že je daný vstup manuálně nastavený do log. 1(FORCE). Celá vykreslená obrazovka je na *Obr. 7.3.*

Jednotlivé vstupy můžeme nastavovat do log. 1 tlačítkem SETTING. Digitální vstupy jsou cyklicky čtené, takže aktuální stavy jednotlivých vstupů se nám přímo zobrazují na LCD. Jestliže je jakýkoliv vstup v log. 1 bez manuálního nastavení zobrazuje se zelená signalizace bez F viz první vstup na *Obr. 7.4.*

Po stisknutí tlačítka(BACK) se nejdříve vykreslí orámované a uskočené ZPET, smažou se všechny digitální vstupy(i jejich stavy), ON/OFF u tlačítka SETTING, DIGITALNI VSTUPY v horní liště a jako poslední se smaže orámované ZPET. Poté se volá funkce *main_menu()*. Toto vše je obsaženo ve funkci *function_button_back().*

Obr. 7.3 Obrazovka digitálních vstupů



Obr. 7.4 Obrazovka nastavených digitálních vstupů

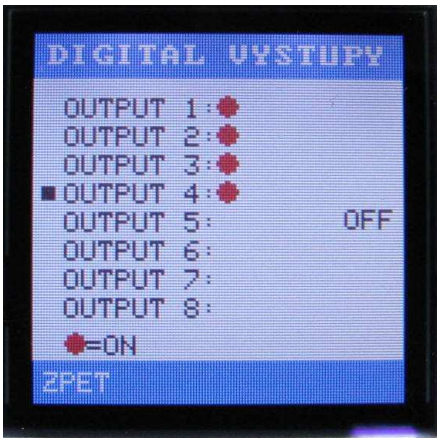
Funkce *digital_output()*

Tato funkce slouží pro vykreslení obrazovky s jednotlivými digitálními výstupy. Nejprve se vykreslí DIGITALNI VYSTUPY v horní modré liště, poté jednotlivé výstupy(OUTPUT 1: až OUTPUT 8:), NO/OFF u tlačítka SETTING podle aktuálního stavu výstupu, informace o signalizaci pro log. 1(sepnutí relé). Jako poslední se vykreslí značka kurzoru u daného výstupu. Celá vykreslená obrazovka je na *Obr. 7.5.*

Jednotlivé výstupy můžeme nastavovat do log. 1 tlačítkem SETTING. Digitální výstupy jsou cyklicky čtené, takže aktuální stavy jednotlivých vstupů se nám přímo zobrazují na LCD. Sepnutí prvních čtyř relé je na *Obr. 7.6.*

Po stisknutí tlačítka(BACK) se nejdříve vykreslí orámované a uskočené ZPET, smažou se všechny digitální výstupy(i stavy výstupů), ON/OFF u tlačítka SETTING, DIGITALNI VYSTUPY v horní liště a jako poslední se smaže orámované ZPET. Poté se volá funkce *main_menu()*. Toto vše je obsaženo ve funkci *function_button_back()*.

Obr. 7.5 Obrazovka digitálních výstupů



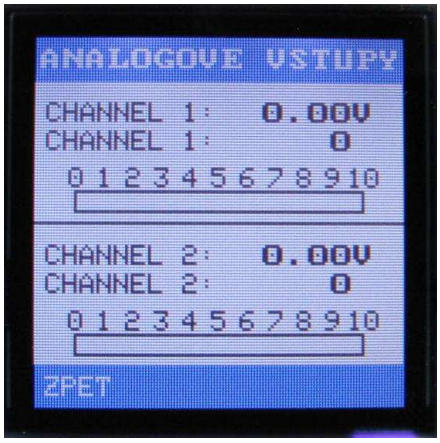
Obr. 7.6 Obrazovka nastavených digitálních výstupů

Funkce *analog_input()*

Tato funkce slouží pro vykreslení obrazovky s jednotlivými analogovými vstupy. Nejprve se vykreslí ANALOGOVE VSTUPY v horní modré liště, poté pro 1. kanál CHANNEL 1: pro float hodnotu napětí, CHANNEL 1: pro dekadickou hodnotu, čísla 1 až 10 nad baragrafem a rám baragrafu. To samé platí i pro druhý kanál pod dělící čárou. Takže CHANNEL 2:, CHANNEL 2:, 1 až 10 nad baragrafem a rám baragrafu. Celá vykreslená obrazovka je na *Obr. 7.7.*

Oba analogové vstupy jsou cyklicky čtené takže jednotlivé hodnoty napětí u obou kanálů se zobrazují přímo na LCD viz *Obr. 7.8.*

Po stisknutí tlačítka(BACK) se nejdříve vykreslí orámované a uskočené ZPET, smažou se oba kanály i jejich aktuální hodnoty, dělící čára, ANALOGOVE VSTUPY v horní modré liště a jako poslední se smaže orámované ZPET. Poté se volá funkce *main_menu()*. Toto vše je obsaženo ve funkci *function_button_back()*.



Obr. 7.7 Obrazovka analogových vstupů

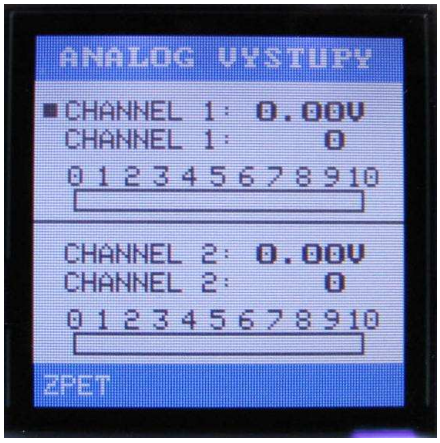
Obr. 7.8 Aktuální hodnoty analogových vstupů

Funkce *analog_output()*

Tato funkce slouží pro vykreslení obrazovky s jednotlivými analogovými výstupy. Nejprve se vykreslí ANALOGOVE VYSTUPY v horní modré liště, poté pro 1. kanál CHANNEL 1: pro float hodnotu napětí, CHANNEL 1: pro dekadickou hodnotu, čísla 1 až 10 nad baragrafem a rám baragrafu. To samé platí i pro druhý kanál pod dělící čárou. Takže CHANNEL 2:, CHANNEL 2:, 1 až 10 nad baragrafem a rám baragrafu. Jako poslední se vykreslí značka kurzoru u daného výstupu. Celá vykreslená obrazovka je na *Obr. 7.9.*

Oba analogové výstupy jsou cyklicky zapisované do D/A převodníku v případě, že daný kanál není v režimu nastavení. Režim nastavení se provede tlačítkem SETTING. Po zmáčknutí tlačítka SETTING se u zvoleného kanálu označí dekadická hodnota a je možné nastavovat výstupní napětí D/A převodníku(tlačítky UP a DOWN). Rozsah nastavení analogového napětí je 0 – 10V. Jeden stisk tlačítka(UP/DOWN) zvýší/sníží výstupní napětí o 0,01V. Potvrzení nastavené hodnoty se provede také tlačítkem SETTING. Po potvrzení hodnoty se aktuální nastavená hodnota vykreslí na baragrafu zvoleného kanálu. Režim nastavení pro oba kanály je obsažen ve funkci *function_button_setting()*. Nastavení kanálu je znázorněno na *Obr. 7.10.*

Po stisknutí tlačítka(BACK) se nejdříve vykreslí orámované a uskočené ZPET, smažou se oba kanály i jejich aktuální hodnoty, dělící čára, ANALOGOVE VYSTUPY v horní modré liště a jako poslední se smaže orámované ZPET. Poté se volá funkce *main_menu()*. Toto vše je obsaženo ve funkci *function_button_back()*.



Obr. 7.9 Obrazovka analogových výstupů

Obr. 7.10 Aktuální hodnoty analogových výstupů(kanál 2 v režimu nastavení)

Funkce *nastaveni()*

Tato funkce slouží pro vykreslení obrazovky s nastavením komunikace a LCD indikátoru. Nejprve se vykreslí ZPET v levém dolním rohu v modré liště, poté NASTAVENI v horní modré liště. Dále KOMUNIKACE A LCD, MODBUS ADRESA: a aktuální nastavená adresa, BITU/SEKUNDA: a aktuální nastavená komunikační rychlost, PARITA: a aktuální nastavená parita, NASTAVENI LCD. Jako poslední se vykreslí značka kurzoru u dané položky. Celá vykreslená obrazovka je na *Obr. 7.11*.

Jednotlivé položky komunikace můžeme nastavovat tlačítkem SETTING. Po zmáčknutí tlačítka SETTING se zvolená položka označí a je jí možné tlačítky UP a DOWN nastavovat viz *Obr. 7.12*. Pro potvrzení je nutné zmáčknout tlačítko SETTING. Po potvrzení se daný nastavovaný parametr komunikace se zapíše do příslušné proměnné. Celá komunikace se inicializuje s novými nastavenými parametry při kroku zpět do hlavního menu. Při posunu kurzoru až na položku NASTAVENI LCD se v levém dolním rohu v modré liště zobrazí ZVOLIT a můžeme tlačítkem SELECT vstoupit do submenu nastavení NASTAVENI LCD(volat funkci *nastaveni_lcd()*). Bude popsáno později. Režim nastavení pro jednotlivé parametry je obsažen ve funkci *function_button_setting()*.

Po stisknutí tlačítka(BACK) se nejdříve vykreslí orámované a uskočené ZPET, smažou se všechny parametry i jejich aktuální hodnoty, KOMUNIKACE A LCD, NASTAVENI v horní modré liště a jako poslední se smaže orámované ZPET. Poté se volá funkce *main_menu()*. Toto vše je obsaženo ve funkci *function_button_back()*.

Obr. 7.11 Obrazovka nastavení

Obr. 7.12 MODBUS ADRESA (v režimu nastavení)

Funkce *nastaveni_lcd()*

Tato funkce slouží pro vykreslení obrazovky s nastavením parametrů LCD indikátoru. Nejprve se vykreslí ZPET v levém dolním rohu v modré liště, poté NASTAVENI LCD v horní modré liště. Dále LCD KONTRAST, minimální, aktuální, maximální hodnotu kontrastu, baragraf s aktuální hodnotou, LCD PODSVETLENI, minimální, aktuální, maximální hodnotu intenzity podsvětlení, baragraf s aktuální hodnotou. Jako poslední se vykreslí značka kurzoru u dané položky. Celá vykreslená obrazovka je na *Obr. 7.13* a *Obr. 7.14.*

U této funkce je změna v posuvu kurzoru na druhou položku. Už se neprování tlačítky UP a DOWN, ale tlačítkem SETTING. Tlačítka UP a DOWN slouží už přímo pro nastavení daného parametru bez potvrzování.

Po stisknutí tlačítka(BACK) se nejdříve vykreslí orámované a uskočené ZPET, smažou se oba parametry i jejich aktuální hodnoty a baragrafy, NASTAVENI LCD v horní modré liště a jako poslední se smaže orámované ZPET. Poté se volá funkce *nastaveni()*. Toto vše je obsaženo ve funkci *function_button_back().*

Obr. 7.13Obrazovka nastavení kontrastu

Obr. 7.14 Obrazovka nastavení podsvětlení

7.2.3 Inicializační funkce

V této funkci(*init_output_input_setting()*) se inicializují všechny globální proměnné použité jednak pro vykreslování jednotlivých obrazovek a ovládání indikátoru, ale dále proměnné pro čtení vstupů, zápis výstupů(jak digitálních tak analogových) i proměnných pro protokol MODBUS, nastavení elektronického potenciometru, nastavení defaultních parametrů komunikace a nastavení časovače 1.

7.2.4 Cyklické čtení vstupů a zápis výstupů

Tato funkce je volána přímo z hlavní smyčky celého programu. Obsahuje cyklické čtení digitálních vstupů a zápis výstupů pro protokol MODBUS. Dále cyklické čtení digitálních vstupů jen když jsme v menu pro digitální vstupy, cyklický zápis digitálních výstupů jen v menu pro digitální výstupy, cyklické čtení analogových vstupů jen v menu čtení analogových výstupů, cyklický zápis analogových vstupů jen v menu pro analogové výstupy a daný kanál není v režimu nastavení. Pokud nejsme v žádném menu ze zmiňovaných, tak se cyklicky čtou pouze digitální vstupy a digitální výstupy. Dané řešení je použito z důvodu zrychlení odezvy a ovládání celého indikátoru, ale také není nutné neustále číst A/D převodník a zapisovat na D/A převodník, když zrovna tyto hodnoty nevidíme zobrazené na LCD. Pokud ale přeci jenom potřebujeme cyklicky číst A/D převodník, nebo zapisovat na D/A převodník použijeme funkci z protokolu MODBUS, která bude popsána později.

V menu pro digitální vstupy se cyklicky tyto vstupy čtou a podle jejich stavu se vykresluje signalizace stavu viz. *Obr. 7.3*, nebo *7.4* . Je-li daný vstup v log. 1(bit portu P0, nebo manuálně nastaven) vykreslí se zelená signalizace u příslušného vstupu. Při stavu log. 0 se signalizace vymazávají(vykreslení bílého čtverce na místě zelené signalizace).

V menu pro digitální výstupy je situace obdobná jako u digitálních vstupů. Také se cyklicky čte proměnná pro zápis výstupů přes protokol MODBUS a proměnné pro manuální nastavení výstupů a podle jejich stavu se vykresluje signalizace jako u digitálních vstupů s tím rozdílem , že signalizace je červená viz. *Obr. 7.5*, nebo *7.6*.

V menu pro analogové vstupy se nejprve přečte hodnota z 1. kanálu A/D převodníku následně se vypočítá hodnota souřadnice y baragrafu, poté se baragraf vykreslí, převede se hodnota z 1. kanálu na float hodnotu a daná hodnota se vykreslí a nakonec se vykreslí i hodnota v dekadické podobě. Pro 2. kanál platí stejný postup jako pro 1. kanál viz. *Obr. 7.7*, nebo *7.8*.

V menu pro analogové výstupy se nejprve testuje zda-li není zvolený kanál v režimu nastavení. Jestliže není tak se ze zadané hodnoty vypočítá souřadnice y pro baragraf a baragraf vykreslí, zadaná hodnota se převede na float hodnotu a vykreslí spolu s dekadickou hodnotou. Poté se již zadaná hodnota pošle do D/A převodníku. Stejný postup platí pro oba kanály viz. *Obr. 7.9*, nebo *7.10*.

7.3 Modul lcd.c

V tomto modulu je obsažena kompletní inicializace a ovládání LCD displeje. Jedná se o funkce pro samotné zaslání příkazu, nebo dat do LCD, inicializaci LCD, smazání celého displeje, vykreslení jednoho pixelu, vykreslení linky, vykreslení obdélníku, vykreslení znaku a vykreslení řetězce. Před popisem jednotlivých funkcí si ještě popíšeme adresování pixelů, definování barvy a auto-inkrementační funkci LCD.

7.3.1 Adresace RAM paměti LCD

Použitý řadič Philips PCF8833 má velikost paměti 17424 slov(132x132), kde každé slovo má velikost 12 bitů(4 bity pro červenou, zelenou a modrou barvu). Adresace pixelu je zajištěna pomocí dvou příkazů. První z nich je příkaz pro adresu řádku(Page Address Set(PASET)) a druhý z nich je příkaz pro adresu sloupce(Column Address Set(CASET)). Každý příkaz má dále datovou část 2 byty, kde se nastavuje v 1. bytu počáteční adresa a v 2. bytu koncová adresa.

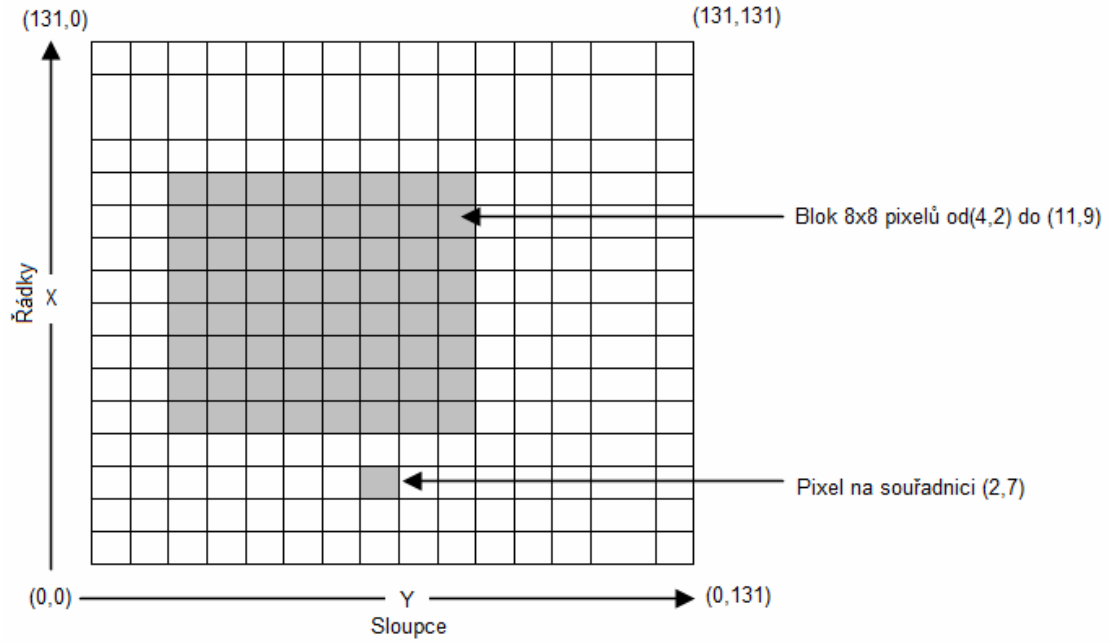
Pro adresování jednoho pixelu je specifické, že počáteční i koncová adresa(řádku a sloupce) je stejná např. podle *Obr. 7.15* (2,7) pak slouží následující sekvence kódu:

SendLCD(LCDCommand,PASET); - příkaz pro adresaci řádku
SendLCD(LCDData,2); - data - počáteční adresa řádku
SendLCD(LCDData,2); - data - koncová adresa řádku
SendLCD(LCDCommand,CASET); - příkaz pro adresaci sloupce
SendLCD(LCDData,7); - data - počáteční adresa sloupce
SendLCD(LCDData,7); - data - koncová adresa sloupce

Pro adresování celé oblasti například velikost 8x8pixelů od adresy (4,2) do adresy (11,9) viz. *Obr. 7.15* pak slouží následující sekvence kódu:

SendLCD(LCDCommand,PASET); - příkaz pro adresaci řádku
SendLCD(LCDData,4); - data – počáteční adresa řádku
SendLCD(LCDData,11); - data – koncová adresa řádku
SendLCD(LCDCommand,CASET); - příkaz pro adresaci sloupce
SendLCD(LCDData,3); - data – počáteční adresa sloupce
SendLCD(LCDData,9); - data – koncová adresa sloupce

Popis funkce *SendLCD(parametr1, parametr2)* bude uvedeno později. Zatím si vystačíme s tím, že *LCDCommand* – příkazový byte, *LCDData* – datový byte.



Obr. 7.15 Adresace pole pixelů a samostatného pixelu

7.3.2 Definování barvy

Řadič Philips PCF8833 má tři různé způsoby(rozlišení) pro definování barvy pixelu. Výběr se provede zasláním příkazu *SendLCD(LCDCommand,COLMOD)* do LCD a za příkazem následuje datová část(1 byte). Datová část určuje, která z možností se použije viz. následující *tab. 7.3.* Definování barvy se provede již v inicializaci LCD.

D2	D1	D0	Formát
0	0	0	neaktivní
0	0	1	neaktivní
0	1	0	**8-bit/pixel**
0	1	1	**12-bit/pixel**
1	0	0	neaktivní
1	0	1	**16-bit/pixel**
1	1	0	neaktivní
1	1	1	neaktivní

Tab. 7.3 Výběr rozlišení pro definování barvy pixelu

1. 12 bitů na pixel

Výběr této možnosti se provede zasláním již zmíněného příkazu(*COLMOD*) a za ním následující datová část, která obsahuje kód $0x03_h$. Pro zápis do paměti RAM LCD slouží příkaz(*RAMWR*). Za tímto příkazem následuje datová část, která již obsahuje informaci o barvě pro daný pixel viz. následující *Obr. 7.16*. Pro jeden pixel je potřeba zaslat 1,5 datového bytu tedy 12 bitů. Zbylé 4 bity ve druhém bytu nám nijak neovlivní druhý pixel jelikož u daného řadiče displeje(PCF8833) je zapotřebí pro zobrazení daného pixelu zaslat všech 12 bitů a až poté se daný pixel vykreslí. Pro jistotu můžeme za sekvencí zaslání druhého bitu poslat příkaz(*NOP*) do LCD viz. následující sekvence kódu. Dané kódování je použité pro kreslení linek a jednotlivých pixelů. Dále bude popsán ještě druhý způsob kódování barev.

0	0	1	0	1	1	0	0
R	R	R	R	G	G	G	G
B	B	B	B	0	0	0	0

příkaz RAMWR (zápis do paměti RAM)
Data 1. byte (4 bity pro červenou a zelenou)
Data 2. byte (4 bity pro modrou)

Obr. 7.16 Kódování 12 bitů/pixel

Sekvence kódu:

Před touto části bude nejdříve již zmíněná adresace pixelu, poté již data pro daný pixel

SendLCD(LCDCommand,RAMWR); - Zápis do paměti RAM
Protože je kód barvy 12-ti bitový tak nejdříve zašleme do RAM horních 8 bitů
SendLCD(LCDData,((color >> 4) & 0x00FF));
Poté spodní 4 bity doplněné nulami
SendLCD(LCDData,(((color & 0xF) << 4) | 0x00));
Pro jistotu zašleme prázdný příkaz *NOP*
SendLCD(LCDCommand,NOP);

Jak již víme z popisu tak je vidět i z *Obr. 7.16* pro jeden pixel je možno kódovat barvu celkem za tří složek(červená, modrá, zelená) a každou složku barvy můžeme kódovat do šestnácti odstínů. Z toho vyplívá již zmíněné, že pro každý pixel vychází 4096 kombinací barvy.

Druhou možností, kterou máme je ta, že za příkazem pro zápis do paměti RAM pošleme 3 datové byty, které budou obsahovat naráz barvu pro dva pixely viz. následující *Obr. 7.17*. Toto řešení je použito pro vykreslování jak obdélníků, nebo čtverců tak i ve vykreslování znaků na LCD. Výhoda spočívá v tom že se vykreslí naráz dva pixely.

0	0	1	0	1	1	0	0	příkaz RAMWR (zápis do paměti RAM)
R	R	R	R	G	G	G	G	Data 1. byte
B	B	B	B	R	R	R	R	Data 2. byte
G	G	G	G	B	B	B	B	Data 3. byte

Obr. 7.17 Kódování 12bitů/2 pixely

Data 1. byte – 4 bity červená, 4 bity zelená pro 1. pixel
Data 2. byte – 4 bity modrá pro 1. pixel, 4 bity červená pro 2. pixel
Data 3. byte – 4 bity zelená pro 2. pixel, 4 bity modrá pro 2. pixel

Sekvence kódu:

horních 8 bitů barvy
SendLCD(LCDData,((color >> 4) & 0x00FF));
prostředních 8 bitů z 12-ti bitů barvy
SendLCD(LCDData,(((color & 0x0F) << 4) | ((color >> 8) & 0xF)));
Spodních 8 bitů
SendLCD(LCDData,(color & 0x00FF));

2. **8 bitů na pixel**

Jelikož daná metoda není u indikátoru použita tak je zmíněné pouze jak vypadá datový byte pro daný pixel. Více informací o implementaci této metody v [2], nebo v [3].

0	0	1	0	1	1	0	0	příkaz RAMWR (zápis do paměti RAM)
R	R	R	G	G	G	B	B	Data 1. byte(3 b. červená,zelená, 2 b. modrá)

Obr. 7.18 Kódování 8bitů/1 pixel

3. **16 bitů na pixel**

Jelikož daná metoda není u indikátoru použita tak je zmíněné pouze jak vypadají datové byty pro daný pixel. Více informací o implementaci této metody v [2], nebo v [3].

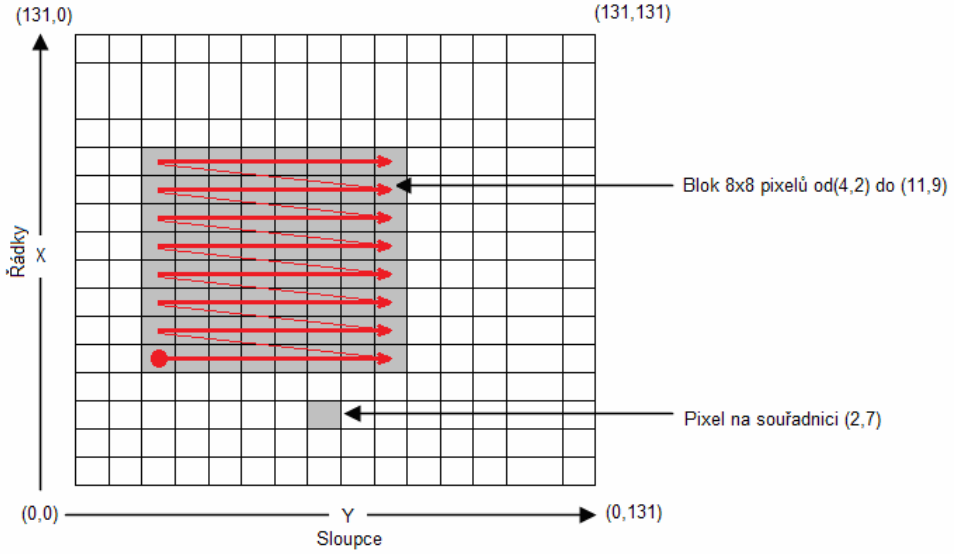
0	0	1	0	1	1	0	0	příkaz RAMWR (zápis do paměti RAM)
R	R	R	R	R	G	G	G	Data 1. byte
G	G	G	B	B	B	B	B	Data 2. byte

Obr. 7.19 Kódování 16bitů/1 pixel

Data 1. byte – 5 bitů červená, 3 bity zelená pro 1. pixel
Data 2. byte – 3 bity zelená pro 1. pixel, 5 bitů modrá pro 1. pixel

7.3.3 Auto-inkrementační funkce LCD

Tato funkce mám dovoluje použitím jednoduchého cyklu vykreslovat pole pixelů(znaky, obdélníky, čtverce). Funkce spočívá v tom, že při cyklickém vykreslování pixelů se inkrementuje adresa sloupce a když dosáhne zadané koncové adresy, tak se automaticky vrátí na počáteční adresu a inkrementuje adresu řádku. Tato funkce je znázorněná na *Obr. 7.20* při vykreslení obrazce např. 8x8 nám stačí cyklus o 64 krokách. Jestliže ale použijeme již zmíněné vykreslování po dvou pixelech tak nám stačí polovina kroků v cyklu pro vykreslení celého obrazce. Viz. následující sekvence kódu. Jak je vidět z kódu tak je adresování sloupce a řádku zasláno pouze jednou a příkaz pro zápis do RAM také jednou poté již data pro jednotlivé pixely. Z toho vyplívá, že při použití této funkce k vykreslení obrazce 8x8 zašleme do LCD 106bytů. Kdybychom ale adresovali každý pixel zvlášť tak pro stejný obrazec zašleme do LCD 576bytů takže při tomto přístupu vykreslování je to zhruba 5x pomalejší.



Obr. 7.20 Auto-inkrementační funkce

Sekvence kódu pro vykreslení pole pixelů 8x8:
Zde samozřejmě popsaná adresace pixelů ((PASET)(4,11),(CASET)(2,9))
Poté již zaslání příkazu pro zápis do RAM paměti
SendLCD(LCDCommand,RAMWR);
Cyklus přes zadané pole pixelů(/2 – vykreslování 2 pixelů)

```
for(i=0;i<((8*8)/2);i++){
        Již popsané dekódování 12-ti bitové barvy na 3 byty
        SendLCD(LCDData,((color>>4)&0x00FF));
        SendLCD(LCDData,(((color&0x0F)<<4)|(color>>8)));
        SendLCD(LCDData,(color&0x00FF));
 }
```

7.3.4 Funkce pro ovládání LCD

Funkce *SendLCD(int type,unsigned char dat)*

Tato funkce slouží pro zasílání jak příkazů tak dat do LCD. Zda-li se jedná o data, nebo příkaz je obsaženo v parametrech funkce.

Parametry funkce:

int type – tento parametr udává zda-li se jedná o příkazový či datový byte
(*LCDCommand(0x0000)* – příkaz, 9. bit je log. 0
LCDData(0x0100) – data, 9. bit je log. 1)
unsigned char dat – v tomto parametru je buďto kód příkazu, nebo samotná data

Jak již bylo popsáno v kapitole 4.1.1. LCD komunikuje s mikrokontrolérem pomocí sériové 9. bitové komunikace. První záměr byl v použití implementované sběrnice SPI v mikrokontroléru a devátý bit nastavovat softwarově, ale z neznámého důvodu toto řešení nefungovalo(nejspíše dochází při spouštění SPI sběrnice k zákmitům, které LCD vyhodnocuje jako přicházející bity). Tak jsem použil kompletně softwarové řešení. Výhoda tohoto řešení je v tom, že můžeme krátkými zpožďovacími smyčkami přesně definovat jak délku log. 1 tak log. 0, ale i mezeru mezi byty. Celý cyklus začíná vytvořením 9-ti bitového příkazu pro LCD, poté výběrem slave zařízení(LCD) se kterým master(mikrokontrolér) bude komunikovat. V samotném cyklu se generují hodinové impulsy na pinu P1.6(SCK) a bitová rotace (příkazu/dat) generuje jednotlivé logické úrovně na pinu P1.7(MOSI). Po odeslání všech devíti bitů se ještě deaktivuje slave zařízení(LCD).

Funkce *init_LCD()*

Jedná se o inicializační funkci LCD displeje. V ní se nejprve provede reset displeje a to sekvencí zasláním log. 0 na pin P3.4($\overline{LCD\_RESET}$), krátkým zpožděním (100ms), zasláním log. 1 na pin P3.4, krátkým zpožděním. Poté zašleme příkaz o inverzi LCD displeje(*INVON*), dále příkaz ve kterém definujeme barevný mód LCD(*COLMOD*) viz. kapitola 6.3.2, dále příkaz(*COLMOD*) spolu s datovou částí, ve které je zakódované otočení souřadnic x a y. Je to z důvodu použití LCD otočeného o 180°. Jako poslední část inicializace ještě nastavíme kontrast LCD a to příkazem(*SETCON*), za příkazem následuje datová část s hodnotou kontrastu. Poté můžeme již zaslat příkaz pro zapnutí LCD(*DISPON*), ale u indikátoru toto není použito z důvodu toho, že za inicializační částí probíhá mazání celého displeje. Takže po zapnutí indikátoru by bylo vidět jak se nejprve celá obrazovka vymaže a až poté se vykreslí úvodní obrazovka. Příkaz o zapnutí LCD se zašle až po vymazání celé obrazovky.

Funkce *LCDClearScreen(int color)*

Tato funkce slouží pro již zmíněné vymazání celé obrazovky. V parametrech funkce je pouze jeden a to kód barvy, kterou se má celý displej vymazat.

Parametry funkce:

int color – kód barvy(12-ti bitová hodnota rrrr gggg bbbb)

Doposud jsme používali termín vymazání displeje co si pod tím představit? Není to nic jiného než to, že se na všechny pixely LCD zapíše kód barvy, která je zakódovaná v parametru funkce. Vymazání displeje se provede v této funkci přes celý zobrazovaný prostor tedy přes 132x132 pixelů. U této funkce je použita již popisovaná auto-inkrementační metoda.

Funkce *LCDSetPixel(int x,int y,int color)*

Tato funkce slouží pro zobrazení jednoho pixelu na souřadnicích x a y.

Parametry funkce:

int x – souřadnice řádku pro daný pixel(0…131)

int y – souřadnice sloupce pro daný pixel(0…131)

int color – kód barvy jakým se má daný pixel zobrazit

(12-ti bitová hodnota rrrr gggg bbbb)

Funkce *LCDSetLine(int x0,int y0,int x1,int y1, int color)*

Tato funkce slouží pro vykreslení linky od souřadnice(x0,y0) do souřadnice(x1,y1). V této funkci je aplikován Bresenhamův algoritmus pro generování úsečky. Popis tohoto algoritmu je uveden v [31].

Parametry funkce:

int x0 – počáteční adresa řádku pro linku(0…131)

int y0 – počáteční adresa sloupce pro linku(0…131)

int x1 – koncová adresa řádku pro linku(0…131)

int y1 – koncová adresa sloupce pro linku(0…131)

int color – kód barvy jakým se má linka vykreslit

(12-ti bitová hodnota rrrr gggg bbbb)

Funkce *LDCSetRect(int x0,int y0,int x1, int y1,unsigned char fill, int color)*

Tato funkce slouží jednak pro vykreslení plného obdélníku, nebo čtverce od souřadnice(x0,y0) do souřadnice(x1,y1), ale také pro vykreslení jen rámu obdélníku či čtverce. Funkce je dále použita pro již popisované mazání určité části obrazovky(popisy, stavy vstupů/výstupů atd.) Jak jsem zmínil u funkce *LCDClearScreen* mazání není nic jiného než překreslení pixelů barvou danou v parametru funkce(většinou se jedná o stejnou barvu jakou má pozadí textu). Jestliže v parametr funkce *fill* obsahuje klíčové slovo FILL vykreslí se plný obdélník/čtverec o daných souřadnicích.

Jestliže ale v parametru funkce *fill* je klíčové slovo NOFILL tak se vykreslí pouze rám o daných souřadnicích. Vykreslení rámu ale již obstarává funkce pro vykreslení úsečky *LCDSetLine* s danými souřadnicemi a barvou. U této funkce je také použita metoda auto-inkrementace.

Parametry funkce:

int x0 – počáteční adresa řádku pro obdélník/čtverec(0…131)

int y0 – počáteční adresa sloupce pro obdélník/čtverec(0…131)

int x1 – koncová adresa řádku pro obdélník/čtverec(0…131)

int y1 – koncová adresa sloupce pro obdélník/čtverec(0…131)

unsigned char fill – booleovská hodnota zda-li se jedná o plný obrazec, nebo pouze rám(FILL – plný obrazec(log. 1) NOFILL – rám obrazce(log. 0))

int color – kód barvy jakým se má obdélník/čtverec, nebo rám vykreslit (12-ti bitová hodnota rrrr gggg bbbb)

Funkce *LCDPutChar(char c,int x,int y,unsigned char size,int fColor,int bColor)*

Tato funkce slouží pro vykreslení znaku na souřadnici *x, y* s barvou *fColor* a barvou pozadí *bColor*.

Parametry funkce:

char c – znak, který se má vykreslit

int x – adresa řádku, kde se má znak vykreslit(0…131)

int y – adresa sloupce, kde se má znak vykreslit(0…131)

unsigned char size – požadovaná velikost znaku(SMALL,MEDIUM,LARGE)

int fColor – požadovaná barva znaku

(12-ti bitová hodnota rrrr gggg bbbb)

int bColor – požadovaná barva pozadí znaku

(12-ti bitová hodnota rrrr gggg bbbb)

Znaky můžeme vykreslit ve třech velikostech a to SMALL(6x8 pixelů), MEDIUM(8x8 pixelů) a LARGE(8x16 pixelů). Velikosti znaků SMALL, MEDIUM a jejich kódování jsou pro ukázku znázorněny na *Obr. 7.21* a *7.22*. Jednotlivé kódy znaků jsou umístěny v paměti XRAM, ale některé nejsou použité z důvodu šetření paměti a ani by nebylo možné všechny znaky do paměti XRAM uložit. Velikost LARGE není použita vůbec. U velikostí SMALL a MEDIUM jsou použité kompletní velké písmena a čísla a znaky jako(: ; = . atd.). Malé písmena také nejsou použita. Kódy jednotlivých znaků jsou uloženy v souboru font.c(nepoužité znaky jsou ve formě komentáře). Jsou rozděleny do tří polí. Pole pro SMALL znaky(FONT6x8) má velikost 376 bytů([47][8]), pole pro MEDIUM znaky(FONT8x8) má velikost 384 bytů([48][8]). Pro zajímavost při použití kompletní sady znaků ve všech třech velikostech bychom potřebovali paměť o velikosti 3104 bytů.

0	0	1	0	0	0	0x20
0	1	0	1	0	0	0x50
1	0	0	0	1	0	0x88
1	0	0	0	1	0	0x88
1	1	1	1	1	0	0xF8
1	0	0	0	1	0	0x88
1	0	0	0	1	0	0x88
0	0	0	0	0	0	0x00

Obr. 7.21 Kódování písmene A velikost SMALL

0	0	0	1	1	0	0	0	0x18
0	0	1	1	1	1	0	0	0x3C
0	1	1	0	0	1	1	0	0x66
0	1	1	0	0	1	1	0	0x66
0	1	1	1	1	1	1	0	0x7E
0	1	1	0	0	1	1	0	0x66
0	1	1	0	0	1	1	0	0x66
0	0	0	0	0	0	0	0	0x00

Obr. 7.22 Kódování písmene A velikost MEDIUM

V prvním bytu každého pole(FONT6x8, FONT8x8, FONR8x16) je informace o počtu sloupců, ve druhém je informace o počtu řádků a ve třetím je informace o počtu bytů pro vykreslení zadaného znaku. Ve funkci se nejprve získá ukazatel na zvolené pole znaků, poté se do proměnných(*nCols,nRows,nByte*) načtou zmiňované parametry znaku. Dále se pomocí těchto parametrů a adresy znaku v ASCII vypočte adresa posledního bytu potřebného k vykreslení požadovaného znaku. Tato adresa je uložená v ukazateli *pChar*. Za tímto výpočtem následuje adresace RAM paměti LCD podle souřadnice x a y. Po adresaci zašleme do LCD příkaz pro zápis do RAM paměti. Poté následují dva cykly ve kterých se přistupuje k jednotlivým bytům znaku a vykreslují se na LCD. V prvním cyklu, který prochází přes všechny řádky zvoleného znaku se načte byte do proměnné *PixelRow*, který je uložen na adrese *pChar*. Následuje dekrementace adresy pro výběr dalšího bytu zvoleného znaku. Byte který je načtený v proměnné *PixelRow* se následně zpracovává z druhém cyklu, který je vnořený do prvního cyklu. Ve druhém cyklu, který prochází přes všechny sloupce znaku se byte uložený v *PixelRow* porovnává(bitový součin) s maskou($0x80_h$). Cyklus prochází po dvou sloupcích z důvodu toho, protože vykreslujeme dva pixely najednou. Porovnáváním s maskou se zjistí, na jaké pozici v bytu je log. 1 a log. 0. Log. 1 v bytu nese informaci o tom že na dané pozici je část znaku a log. 0, že na dané pozici je pozadí znaku. Provede se rotace masky o jedno místo doprava pro test druhého bitu. Při zjištění log. 1 u prvního pixelu se do proměnné *Word0* zapíše barva, která náleží pro znak a při log. 0 se zapíše barva, která náleží pro barvu pozadí. Totéž se provede i pro druhý pixel, akorát se nezapisuje barva do *Word0*, ale do *Word1*. Poté se již dané dva pixely vykreslí na LCD.

Celé volání funkce pro vykreslení znaku, který bude mít barvu červenou, pozadí bílé, velikost MEDIUM a bude na souřadnici 20,20 má pak tvar(stejný znak jako na *Obr. 7.22*):

LCDPutChar('A',20,20,MEDIUM,RED,WHITE);

Funkce *LCDPutStr(char *pString,int x,int y,int Size,int fColor,int bColor)*

Tato funkce slouží pro vykreslení celého řetězce znaků na souřadnici x a y s barvou *fColor* a barvou pozadí řetězce *bColor*

Parametry funkce:

*char *pString* – adresa prvního znaku řetězce

int x – adresa řádku, kde se má znak vykreslit(0…131)

int y – adresa sloupce, kde se má znak vykreslit(0…131)

unsigned char size – požadovaná velikost řetězce(SMALL,MEDIUM,LARGE)

int fColor – požadovaná barva řetězce

(12-ti bitová hodnota rrrr gggg bbbb)

int bColor – požadovaná barva pozadí řetězce

(12-ti bitová hodnota rrrr gggg bbbb)

V této funkci se nedělá nic jiného něž to, že se postupně načítací jednotlivé znaky z řetězce a volá se funkce pro vykreslení znaku. Po vykreslení znaku se posune souřadnice sloupce a hodnotu danou velikostí znaku. Jako poslední se testuje překročení adresy sloupce LCD(131). Při zjištění překročení se vykreslování zastaví.

Celé volání funkce pro vykreslení např. řetězce „AHOJ“ viz. *Obr. 7.23*, který bude mít barvu červenou, pozadí bílé, velikost SMALL a bude na souřadnici 20,20 má pak tvar:

LCDPutStr(“AHOJ”,20,20,SMALL,RED,WHITE);

0	0	1	0	0	0	1	0	0	0	1	0	0	1	1	1	0	0	0	0	1	1	1	0
0	1	0	1	0	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0
1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0
1	0	0	0	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	0	0	1	0	0
1	1	1	1	1	0	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0	1	0	0
1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	0	0	1	0	0
1	0	0	0	1	0	1	0	0	0	1	0	0	1	1	1	0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Obr. 7.23 Kódování řetězce „AHOJ“ velikost SMALL

7.4 Modul modbus.c

V tomto modulu je obsažena inicializace sériového portu mikrokontroléru, časovačů mikrokontroléru, proměnných použitých v protokolu MODBUS a funkce pro samotný protokol MODBUS. Pro zjednodušení je implementován pouze binární režim komunikace MODBUS RTU. K tomuto modulu je také přidružen modul kontrolního součtu CRC16.c.

7.4.1 Komunikace pomocí sériové linky

Sériovou komunikaci můžeme nastavovat v menu LCD. Přenosovou rychlost na 9600b/s, 19200b/s, 38400b/s, 57600b/s 115200b/s. Paritu na žádnou, sudou, lichou. Pro vysílání a příjem jsou vytvořeny globální pole o velikosti 256bytů, což je maximální velikost packetu pro MODBUS. Dále jsou vytvořené proměnné udávající pozici v bufferech a pro vysílací buffer je zde proměnná *TX_DATA_Len* udávající délku dat v bufferu.

Příjem dat probíhá v přerušení od sériové linky. V tomto přerušení se nejprve vynuluje proměnná(*timeout*), která udává počet přetečení časovače 0. Dále se testuje zda-li počet přijatých bytů větší než 256. Tuto hodnotu nám udává proměnná *RXpos*, která se inkrementuje s každým přijatým bytem(na počátku je tato proměnná vynulována). Je-li počet přijatých bytů větší jak 256 tak se navýší čítač počtu ztracených znaků(*CPT8*) a je indikována chyba ve zprávě(*error_frame*). Po tomto testu se kontroluje zda-li nepřekročil čas 1,5char mezi znaky. Jestliže ano, je nastavená indikace chyby ve zprávě. Jestliže ne, tak se přijatý byte v *SBUF* zapíše na příslušnou pozici v bufferu *RX_DATA*. Informaci o pozici je v již zmíněné proměnné *RXpos*. V přijatém bytu se dále testuje parita. Je-li chyba v paritě je nastavena indikace chyby ve zprávě a zvýší se čítač komunikačních chyb(*CTP2*). Není-li chyba v paritě tak se jako poslední nastaví opět předvolbou časovač 0, vynuluje se příznak přetečení časovače 0 poté se spustí časovač 0 a zvýší se čítač počtu přijatých bytů(*count*).

Vysílání probíhá také v přerušení. Odesílaná data se zapíší do bufferu *TX_DATA*. Do proměnné *TX_DATA_Len* se zapíše počet odesílaných bytů a vysílání se zahájí ručním nastavením příznaku *TI*, který ihned vyvolá přerušení. V přerušení se nejprve nastaví linka RS485 na vysílání, poté se postupně jednotlivé byty z bufferu *TX_DATA* zapisují do *SBUF* do té doby dokud proměnná *TXpos* nemá stejnou hodnotu jako proměnná *TX_DATA_Len*. Po odeslání všech bytů z bufferu *TX_DATA* se přepne zpátky linka RS485 na příjem, vynuluje se příznak od odeslaných dat *TI*, vynuluje se příznak od časovače 0 a časovač se spustí. Po uplynutí doby 3,5char od odeslání posledního znaku se časovač o zastaví, vynulují se příjmový a vysílací buffer, vynulují se čítače přetečení časovače 0 a povolí se příjem od sériové linky.

7.4.2 Funkce protokolu MODBUS

Funkce *init_modbus()*

V této funkci se provede kompletní inicializace sériové linky mikrokontroléru, časovače 0, časovače 1 a časovače 2.

Sériová linka mikrokontroléru je nastavena do módu 3. Tedy do 9bitového asynchronního přenosu dat(devátý bit je obsažen v TB8/RB8 bitu registru SCON a má význam paritního bitu). Bity se vysílají na TxD(P3.1) a přijímají na RxD(P3.0). Přenos začíná start-bitem(log. 0), následuje 9 datových bitů a poslední je stop-bit(log. 1). Přenosová rychlost je dána přetečením časovače 2. Více o módech sériové linky mikrokontroléru je možné najít v [21].

Časovač 0 je zde použit pro odměřování dvou časů použitých v protokolu MODBUS RTU a to 1,5char pro max. mezeru mezi znaky a 3,5char pro mezeru mezi zprávami. Časovač 0 pracuje v režimu 1 jako 16bitový časovač, který obsahuje 8bitů čítače TH0 a 8 bitů čítače TL0. Při přeplnění čítače, tj. při přechodu z obsahu $0xFFFF_h$ na $0x0000_h$ se nastavuje příznakový překlápěcí obvod TF0 v registru TCON a je generováno přerušení od časovače 0. Nastavení režimu časovače 0 se provede v registru TMOD. Při každém generování přerušení od časovače 0 se inkrementuje proměnná *timeout*. Dosáhne-li hodnota proměnné *timeout* čísla 11 signalizuje to dosažení času 1,5char. Dosáhne-li hodnota proměnné *timeout* čísla 26 signalizuje to dosažení času 3,5char. Pro výpočet registrů TH0 a TL0 při použití dvojnásobné rychlosti vykonávání instrukcí platí vztah(26):

$$TH0, TL0 = 65536 - \frac{f_{osc}}{4.BAUD\_RATE} \tag{26}$$

kde

$f_{osc} - frekvence\ \ kryslalového\ oscilátoru\ [Hz]$

$BAUD\_RATE - požadovaná\ \ přřenosov\ rychlost\ \ [b/s]$

Časovač 1 nemá s protokolem MODBUS nic společného je v indikátoru použit pro odměřování času cca. 1min pro vypnutí podsvětlení LCD displeje. Tento časovač pracuje úplně stejně jako časovač 0.

Časovač 2 zde pracuje jako generátor přenosové rychlosti pro sériový kanál. Generátor přenosové rychlosti zajistí velmi jemnou přeladitelnost kmitočtu sériového přenosu, která se navíc velmi snadno nastavuje. Generátor přenosové rychlosti se aktivuje nastavením bitů TCLK(pro vysílač), nebo RCLK(pro přijímač) z registru T2CON. K nastavení přenosové rychlosti slouží registry RCAP2L a RCAP2H. U čítače/časovače 2 je režim pro generátor přenosové rychlosti odlišný od ostatních

režimů. Nečítá se totiž 1/12 hodinového kmitočtu, ale 1/2 hodinového kmitočtu. To zajistí mnohem jemnější přeladění. Pro výpočet registrů RCAP2L a RCAP2H při použití dvojnásobné rychlosti vykonávání instrukcí(*CKCON0*) platí vztah(27):

$$RCAP2H, RCAP2L = 65536 - \frac{f_{osc}}{16.BAUD\_RATE} \tag{27}$$

kde

f_{osc} – *frekvence kryslalového oscilátoru* [*Hz*]

$BAUD\_RATE$ – *požadovaná přřenosov rychlost* [b/s]

Více o možnostech časovače 2 v [21].

Funkce *init_prom_modbus()*

Tato funkce slouží pro inicializaci proměnných použitých v komunikaci MODBUS. Vynulují ke všechny čítače jak komunikačních chyb tak čítače přetečení časovačů, příjmový a vysílací buffer a jako poslední se nastaví defaultní modbus adresa indikátoru na 2, přenosová rychlost na 115200b/s a parita sudá.

7.4.3 Funkce *modbus()*

Jak již bylo zmíněno je použit pouze binární režim komunikace MODBUS RTU. Celý protokol s výjimkou kontrolního součtu je pro svou jednoduchost realizován pouze v jedné funkci *modbus()*. Tato funkce je cyklicky volána z hlavní smyčky celého programu.

Ve funkci se nejprve testuje, zda-li nepřišla zpráva o restartu komunikace. Jestliže ano, jsou volány již popsané funkce *init_prom_modbus()* a *init_modbus()*. Poté se již testuje překročení času 1,5char. Při překročení času 1,5char(*timeout >=11*) se nastaví signalizace *time_t15* do log. 1. Následuje další test překročení času tentokrát 3,5char(*timeout >=26*), který signalizuje konec příchozí zprávy. Při překročení tohoto času následuje zakázání příjmu od sériové linky, zastavení časovače 0 a vynulování čítače přetečení časovače 0. Poté následují testy o platnosti zprávy. Jako první se testuje zda-li není ve zprávě chyba(test *error_frame*). Je-li ve zprávě chyba inkrementuje se čítač komunikačních chyb(*CPT2*) a zpracovávání se ukončí. V druhém kroku se testuje přijetí více než 3bytů dat. Při nesplnění této podmínky se opět inkrementuje čítač komunikačních chyb a zpracovávání se ukončí. Ve třetím kroku přichází na řadu načtení kontrolního součtu z přijaté zprávy a porovnání s vypočteným kontrolním součtem v mikrokontroléru. Pokud se kontrolní součty nerovnají inkrementuje se čítač komunikačních chyb a zpracovávání se ukončí. Jestliže přijatý kontrolní součet je stejný s vypočítaným inkrementuje se čítač počtu přijatých zpráv(*CPT1*) a přichází testování zda-li na prvním místě v příjmovém bufferu je adresa slave zařízení(indikátoru), nebo broadcast adresa(0). Jestliže přijatá adresa není slave zařízení a ani adresa brodcast

vynuluje se příjmový i vysílací buffer, vynulují se proměnné nesoucí informaci o pozici v bufferech a povolí se příjem od sériové linky. Jestliže ale adresa odpovídá adrese slave zařízení nebo broadcast adrese, inkrementuje se čítač počtu zpracovaných zpráv(*CPT4*) a přichází na řadu test zda-li se nejedná o broadcast adresu. Jestliže ano, zvýší se čítač nezodpovězených zpráv, vynulují se proměnné jako v předchozím kroku a povolí se příjem do sériové linky. Jestliže přijatá adresa odpovídá adrese slave zařízení přichází na řadu samotné zpracování přijaté zprávy. Podle kódu přijatého na druhém místě v přijímacím bufferu se vykoná příslušná funkce protokolu MODBUS.

Z veřejných funkcí jsou implementovány funkce čtení cívek(0x01), čtení diskrétních vstupů(0x02), diagnostika(0x08), čítač komunikačních událostí(0x0B), zápis více cívek(0x0F) a identifikace zařízení(0x2B). Dále jsou implementovány čtyři uživatelsky definované funkce a to zápis stavu tlačítek(0x66), cyklické čtení I/O, A/D(0x67), čtení A/D převodníku(0x68) a zápis D/A převodníku(0x69). Vývojový diagram, podle kterého byla funkce protokolu MODBUS programována je v uveden příloze 1. V příloze 2 je uveden diagram podle kterého byla naprogramovaná funkce kontrolního součtu CRC16.

7.4.3.1 Funkce 0x01 – Čtení cívek

Tato funkce slouží ke čtení stavu digitálních výstupů. V požadavku je specifikována adresa prvního výstupu a počet výstupů. V odpovědi je v jednom bytu přenášen stav všech osmi digitálních výstupů. Formát požadavku a odpovědi je uveden v *tab. 7.4* a *7.5*

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x01
Počáteční adresa	2 byty	0x00A0
Počet cívek	2 byty	0x0008

Tab. 7.4 Požadavek funkce 0x01

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x01
Počet bytů	1 byte	0x01
Stavy cívek	1 byte	0x00-0xFF

Tab. 7.5 Odpověď funkce 0x01

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů, počáteční adresa a počet digitálních výstupů. Jestliže není některá z podmínek splněná tak se posílá zpět kód obsahující chybovou funkci. Chybový kód se vytvoří přičtením hodnoty $0x80_h$ ke kódu funkce. V chybovém kódu se zasílá zpět modbus adresa, kód funkce(chybový), kód chyby viz následující *tab. 7.6.*

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x81
Kód chyby	1 byte	0x01, 0x02, 0x03 nebo 0x04

Tab. 7.6 Chyba funkce 0x01

7.4.3.2 Funkce 0x02 – Čti diskrétní vstupy

Tato funkce slouží ke čtení stavu digitálních vstupů. V požadavku je specifikována adresa prvního vstupu a počet vstupů. V odpovědi je v jednom bytu přenášen stav osmi digitálních vstupů. Formát požadavku a odpovědi je uveden v *tab. 7.7* a *7.8.*

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x02
Počáteční adresa	2 byty	0x00B0
Počet vstupů	2 byty	0x0008

Tab. 7.7 Požadavek funkce 0x02

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x02
Počet bytů	1 byte	0x01
Stavy vstupů	1 byte	0x00-0xFF

Tab. 7.8 Odpověď funkce 0x02

Chybový kód je vytvořen stejně jako u digitálních výstupů. Formát chyby je uveden v tab.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x82
Kód chyby	1 byte	0x01, 0x02, 0x03 nebo 0x04

Tab. 7.9 Chyba funkce 0x02

7.4.3.3 Funkce 0x08 – Diagnostika

Tato funkce slouží k provedení série testů pro zkontrolování komunikace mezi klientem(Master) a serverem(Slave), nebo ke kontrole různých interních chybových stavů serveru. Funkce používá dvoubajtový kód podfunkce, který specifikuje požadovaný typ testu. Normální odpověď obsahuje kopii požadavku případně další data, pokud jsou výsledkem testu. Formát požadavku a odpovědi je uveden v *tab. 7.10* a *7.11*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x08
Podfunkce	2 byty	Viz *tab. 7.12*

Tab. 7.10 Požadavek funkce 0x08

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x08
Podfunkce	2 byty	Viz *tab. 7.12*
Data	N*2 byty	

Tab. 7.11 Odpověď funkce 0x08

Kód podfunkce	Název
0x01	Restartuj komunikaci
0x0A	Vynuluj čítače
0x0B	Vrať počet zpráv(CPT1)
0x0C	Vrať počet komunikačních chyb(CPT2)
0x0D	Vrať počet negativních odpovědí(CPT3)
0x0E	Vrať počet zpracovaných zpráv(CPT4)
0x0F	Vrať počet nezodpovězených zpráv(CPT5)
0x12	Vrať počet ztracených znaků(zpráv)(CPT8)

Tab. 7.12 Kódy podfunkcí funkce 0x08

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů a zda-li je kód podfunkce podporovaný. Jestliže není některá z podmínek splněná tak se posílá zpět kód obsahující chybovou funkci. Formát chyby je uveden v *tab. 7.13.*

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x88
Kód chyby	1 byte	0x01, 0x03 nebo 0x04

Tab. 7.13 Chyba funkce 0x08

7.4.3.4 Funkce 0x0B – Čti čítač komunikačních událostí

Tato funkce slouží k získání stavového slova a hodnoty čítače komunikačních událostí(*ModbusEvent*). Čítač událostí je inkrementován po každém úspěšném dokončení požadavku. Normální odpověď obsahuje dvoubajtové stavové slovo a dvoubajtový počet událostí. Stavové slovo udává, zda je zařízení připraveno (0x0000), nebo zaneprázdněno vykonáváním předešlé funkce (0xFFFF). Odpověď je ale vždy 0x0000, protože všechny z podporovaných funkcí jsou vykonány ihned. Formát požadavku a odpovědi je uveden v *tab. 7.14* a *7.15*. Formát chyby je uveden v *tab. 7.16.*

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x0B

Tab. 7.14 Požadavek funkce 0x0B

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x0B
Status	2 byty	0x0000 nebo 0xFFFF
Počet událostí	2 byty	0x0000 až 0xFFFF

Tab. 7.15 Odpověď funkce 0x0B

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x8B
Kód chyby	1 byte	0x01 nebo 0x04

Tab. 7.16 Chyba

7.4.3.5 Funkce 0x0F – Zapiš více cívek

Tato funkce slouží k nastavení osmi digitálních výstupů do stavu ON nebo OFF. V požadavku je specifikována adresa prvního výstupu, který se má nastavit a hodnoty, na které se mají výstupy nastavit. Normální odpověď obsahuje počáteční adresu a počet nastavených výstupů. Formát požadavku a odpovědi je uveden *v tab. 7.17* a *7.18*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x0F
Počáteční adresa	2 byty	0x00B0
Počet výstupů	2 byty	0x0008
Počet bytů	1 byte	0x01
Hodnota výstupů	1 byte	0x00 až 0xFF

Tab. 7.17 Požadavek funkce 0x0F

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x08
Počáteční adresa	2 byty	0x00B0
Počet výstupů	2 byty	0x0008

Tab. 7.18 Odpověď funkce 0x0F

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů, počáteční adresa a počet digitálních výstupů. Jestliže není některá z podmínek splněná tak se posílá zpět kód obsahující chybovou funkci. Formát chyby je uveden v *tab. 7.19.*

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x8F
Kód chyby	1 byte	0x01,0x02,0x03 nebo 0x04

Tab. 7.19 Chyba funkce 0x0F

7.4.3.6 Funkce 0x2B – Čti identifikaci zařízení

Tato funkce slouží ke čtení identifikace a dalších údajů týkajících se popisu zařízení. Formát požadavku a odpovědi je uveden v *tab. 7.20* a *7.21*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x2B
Typ MEI	1 byte	0x0E
ID kód	1 byte	01/02/03/04
ID objektu	1 byte	0x00 až 0xFF

Tab. 7.20 Požadavek funkce 0x2B

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x2B
Typ MEI	1 byte	0x0E
ID kód	1 byte	0x01
Úroveň shody	1 byte	0x01 – základní identifikace
Pokračování	1 byte	0x00 – žádné není
ID objektu	1 byte	0x00
Počet objektů	1 byte	0x03
Objekt ID	1 byte	0x00 – Název výrobce
Délka objektu	1 byte	0x08 (VUT BRNO)
Objekt ID	1 byte	0x01 – kód produktu
Délka objektu	1 byte	0x1C (PRUMYSLOVY STAVOVY INDIKATOR
Objekt ID	1 byte	0x02 – verze produktu
Délka objektu	1 byte	0x04 (V1.0)

Tab. 7.21 Odpověď funkce 0x2B

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů, MEI a zda-li se jedná o základní identifikaci. Jestliže není některá z podmínek splněná tak se posílá zpět kód obsahující chybovou funkci. Formát chyby je uveden v *tab. 7.22*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0xAB
Typ MEI	1 byte	0x0E
Kód chyby	1 byte	0x01,0x02,0x03 nebo 0x04

Tab. 7.22 Chyba funkce 0x2B

7.4.3.7 Uživatelská funkce 0x66 – Zápis stavu tlačítek

Tato funkce slouží k ovládání indikátoru pomocí tlačítek vytvořených ve Windows aplikaci přes protokol MODBUS. V požadavku je specifikován počet bytů a stav tlačítek. V odpovědi je vrácen stav tlačítek. Formát požadavku a odpovědi je uveden v *tab. 7.23* a *7.24*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x66
Počet bytů	1 byte	0x01
Stav tlačítek	1 byte	0x20/0x40/0x60/0x80/0xA0/0xC0

Tab. 7.23 Požadavek funkce 0x66

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x66
Počet bytů	1 byte	0x01
Stav tlačítek	1 byte	0x20/0x40/0x60/0x80/0xA0/0xC0

Tab. 7.24 Odpověď funkce 0x66

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů. Jestliže není podmínka splněná tak se posílá zpět kód obsahující chybovou funkci. Formát chyby je uveden v *tab. 7.25*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x66
Kód chyby	1 byte	0x01,0x02,0x03 nebo 0x04

Tab. 7.25 Chyba funkce 0x66

7.4.3.8 Uživatelská funkce 0x67 – Cyklické čtení I/O, A/D převodníku

Tato funkce slouží pro cyklické čtení digitálních vstupů, digitálních výstupů a analogových vstupů. Jak již bylo zmíněné dříve při aktivaci této funkce nelze indikátor ovládat tlačítky a cyklické čtení je opakováno s intervalem 200ms. V požadavku je specifikován pouze kód funkce, počet bytů a signalizace cyklického

čtení. V odpovědi je vrácen počet bytů, hodnota digitálních vstupů, digitálních výstupů, analogových vstupů a analogových výstupů. Formát požadavku a odpovědi je uveden v *tab. 7.26* a *7.27*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x67
Počet bytů	1 byte	0x01
Signalizace cyklického čtení	1 byte	0x01

Tab. 7.26 Požadavek funkce 0x67

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x67
Počet bytů	1 byte	0x06
Stav digitálních vstupů	1 byte	0x00 až 0xFF
Stav digitálních výstupů	1 byte	0x00 až 0xFF
Stav analogových vstupů	2 byty	0x0000 až 0x0FFF

Tab. 7.27 Odpověď funkce 0x67

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů. Jestliže není podmínka splněná tak se posílá zpět kód obsahující chybovou funkci. Formát chyby je uveden v *tab. 7.28*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x67
Kód chyby	1 byte	0x01,0x02,0x03 nebo 0x04

Tab. 7.28 Chyba funkce 0x67

7.4.3.9 Uživatelská funkce 0x68 – Čtení A/D převodníku

Tato funkce slouží pro načtení analogové hodnoty z A/D převodníku. V požadavku je specifikován pouze kód funkce a počet bytů. V odpovědi je vrácen počet bytů a hodnota analogových vstupů. Formát požadavku a odpovědi je uveden v *tab. 7.29* a *7.30*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x68
Počet bytů	1 byte	0x04

Tab. 7.29 Požadavek funkce 0x68

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x68
Počet bytů	1 byte	0x04
Stav analogových vstupů	2 byty	0x0000 až 0x0FFF

Tab. 7.30 Odpověď funkce 0x68

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů. Jestliže není podmínka splněná tak se posílá zpět kód obsahující chybovou funkci. Formát chyby je uveden v *tab. 7.31*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x68
Kód chyby	1 byte	0x01,0x02,0x03 nebo 0x04

Tab. 7.31 Chyba funkce 0x68

7.4.3.10 Uživatelská funkce 0x69 – Zápis D/A převodníku

Tato funkce slouží pro zapsání analogové hodnoty do D/A převodníku. V požadavku je specifikován pouze kód funkce a počet bytů. V odpovědi je vrácen počet bytů a zapsaná hodnota analogových výstupů. Formát požadavku a odpovědi je uveden v *tab. 7.32* a *7.33*.

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x69
Počet bytů	1 byte	0x04

Tab. 7.32 Požadavek funkce 0x69

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x69
Počet bytů	1 byte	0x04
Stav analogových výstupů	2 byty	0x0000 až 0x0FFF

Tab. 7.33 Odpověď funkce 0x69

Předtím než se zašle odpověď se ve funkci testuje počet přijatých bytů. Jestliže není podmínka splněná tak se posílá zpět kód obsahující chybovou funkci. Formát chyby je uveden v *tab. 7.34.*

Modbus adresa	1 byte	0x02
Kód funkce	1 byte	0x69
Kód chyby	1 byte	0x01,0x02,0x03 nebo 0x04

Tab. 7.34 Chyba funkce 0x69

7.5 Modul SPI.c

V tomto modulu je obsažena kompletní inicializace a funkce pro ovládání sběrnice SPI. Jedná se o funkce čtení dat z A/D převodníku, zápis dat do D/A převodníku a zápis do elektronického potenciometru.

Funkce *init_SPI()*

V inicializaci SPI mikrokontroléru nastavíme master mód(bit MSTR v registru SPCON mikrokontrolér je ve funkci mastera), přenosovou rychlost na 1/4 frekvence krystalového oscilátoru(bity SPR2,SPR1,SPR0), CPOL a CPHA do log. 1, vynulujeme příznak o odeslání dat(SPIF v registru SPSTA), ale zatím sběrnici nepouštíme. Více informací o možnostech SPI v mikrokontroléru v [6].

Funkce *SPI_Write(unsigned char value)*

Tato funkce slouží k odeslání dat sběrnicí SPI. Data se vloží do registru SPDAT a čeká se na příznak o odeslání dat). Když jsou data odeslány příznak vynulujeme.
Parametry funkce:

unsigned char value – data(byte), který se má odeslat pomocí SPI

Funkce unsigned int *SPI_Read(unsigned char value)*

Tato funkce slouží k přijetí dat sběrnicí SPI. Data se vloží do registru SPDAT(například adresa A/D převodníku) a čeká se na příznak o odeslání dat. Když jsou data odeslány příznak vynulujeme. Přijatá data(například hodnota z A/D převodníku) poté máme uložené v registru SPDAT, které jsou vráceny.
Parametry funkce:

unsigned char value – data(byte např. adresa), který se má odeslat pomocí SPI

Funkce *unsigned int ADC(unsigned char channel)*

Tato funkce slouží k přečtení analogové hodnoty ze vstupů A/D převodníku. Ve funkci se nejprve aktivuje slave zařízení(A/D převodník) se kterým má master(mikrokontrolér) komunikovat. Poté se spustí sběrnice SPI(aktivace bitu SPEN v registru SPCON). Dále zapíšeme do A/D převodníku číslo kanálu, konfigurační bity a načteme hodnotu z A/D. Přečtená hodnota je ve formě 2 bytů. Tyto byty sloučíme do int hodnoty. Deaktivujeme slave zařízení, zastavíme SPI sběrnici a načtenou hodnotu z A/D převodníku vrátíme.
Parametry funkce:

unsigned char channel – číslo kanálu ze kterého se má číst analogová hodnota

Funkce *DAC(unsigned char channel, unsigned int value)*

Tato funkce slouží k zápisu dat do D/A převodníku. Funkce je podobná jako u A/D převodníku. Takže zkráceně popis - deaktivujeme slave zařízení, spustíme sběrnici SPI, upravíme hodnotu z int na 2 byty, kde horní 4 bity z prvního bytu jsou konfigurační. Byty odešleme do D/A převodníku. Deaktivujeme slave zařízení a zastavíme sběrnici SPI.

Parametry funkce:

unsigned char channel – číslo kanálu do kterého se má zapisovat

unsigned int value – hodnota, která se má zapsat do převodníku(0-4095)

Funkce *Potenciometr(unsigned char value)*

Jedná se o nejjednodušší funkci ve které je použita sběrnice SPI. U elektronického potenciometru neupravujeme žádné data a ani potenciometr nijak nekonfigurujeme. Ve funkci se nejprve aktivuje slave zařízení, spustí sběrnice SPI, do registru SPDAT zapíšeme data, která se mají zaslat do potenciometru. Po odeslání dat deaktivujeme slave a zastavíme sběrnici SPI.

Parametry funkce:

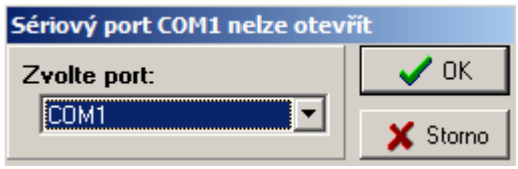
unsigned char value – hodnota, která se má zapsat do potenciometru(0-255)

8 WINDOWS APLIKACE PRO INDIKÁTOR

Tuto aplikaci jsem vytvořil pro ukázku, aby bylo možné celý indikátor ovládat z prostředí Windows, ale také pro testování komunikace protokolem MODBUS přes linku RS-232, nebo linku RS-485. Aplikace je napsána ve vývojovém prostředí Borland C++ Builder a umožňuje jednak celý indikátor ovládat tlačítky, ale také číst digitální vstupy/výstupy, zapisovat digitální výstupy, číst analogové vstupy, zapisovat na analogové výstupy, nastavovat parametry komunikace, číst diagnostické funkce, identifikaci zařízení a umožňuje komunikaci restartovat.

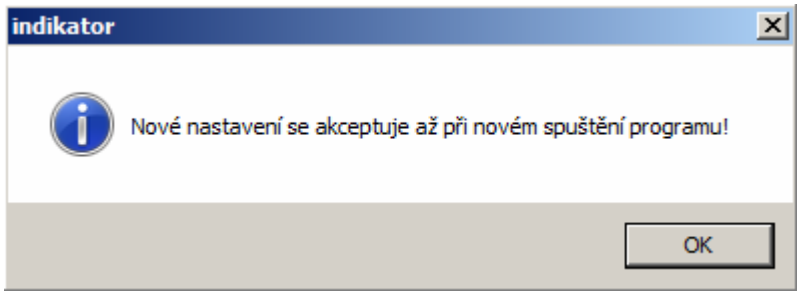
8.1 Nastavení parametrů komunikace

Veškeré parametry komunikace jsou uloženy v souboru indikátor.ini. Port je defaultně nastavený na COM1, komunikační rychlost na 115200b/s, 8 datových bitů, sudá parita a 1 stop-bit. Jestliže po spuštění souboru indikator.exe není dostupný komunikační port COM1 otevře se okno s možností vybrat jiný komunikačního port. Viz následující *obr. 8.1*. V okně jsou zobrazeny dostupné komunikační porty, ze kterých máme možnost vybírat.



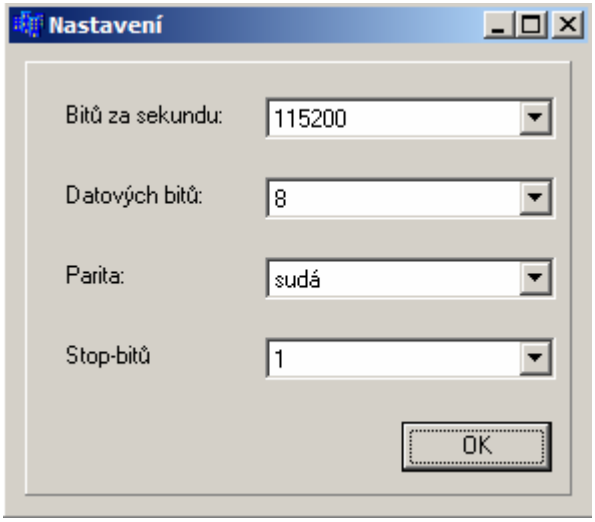
Obr. 8.1 Výběr komunikačního portu

Po zvolení nového komunikačního portu se aplikace uzavře a při novém spuštění je komunikační port již zapsán do souboru indikátor.ini a aplikace pracuje s nově zvoleným dostupným komunikačním portem.



Obr. 8.2 Zobrazení upozornění

Po spuštění aplikace je možné parametry komunikace nastavovat v hlavním menu v položce nastavení viz *obr. 8.3*. V tomto okně je možné nastavovat komunikační rychlost na(9600b/s, 19200b/s, 38400b/s, 57600b/s, 115200b/s), počet datových bitů je pevně nastaven na 8, paritu můžeme nastavovat na(žádnou, sudou, lichou) a počet stop-bitů na jeden nebo dva podle parity. Po stisknutí tlačítka OK se opět všechny parametry uloží do souboru indikator.ini a při dalším spuštění aplikace zůstávají parametry nastavené.



Obr. 8.3 Nastavení parametrů

8.2 Popis aplikace

Okno celé aplikace je vidět na *obr. 8.4*. Veškeré ovládací prvky pro indikátor jsou umístěny v jednou okně a jsou pro odlišení umístěné na panelech. Pro ovládání indikátoru slouží tlačítka *UP, SETTING, DOWN, SELECT, MENU, BACK* umístěny kolem panelu parametrů komunikace. Tyto tlačítka mají stejnou funkci jako hardwarová tlačítka umístěny na indikátoru. Tlačítko *Cyklické čtení* slouží pro zapnutí cyklického čtení vstupů/výstupů. Po aktivaci tohoto tlačítka je možnost pouze stejným tlačítkem čtení vypnout. Ostatní funkce nejsou aktivní a ovládací prvky jsou schovány. Poslední co není umístěné na panelu je vypisování prvních 11 bytů jak příjmového bufferu RX_DATA tak vysílacího TX_DATA, které se aktualizují po každém příjmu či vysílání dat.

Panel Digitální vstupy – Na tomto panelu je umístěno tlačítko pro načtení digitálních vstupů a políčka pro signalizaci stavu jednotlivých vstupů. Pro log. 1 platí označené políčko. Pro log. 0 je políčko prázdné.

Panel Digitální výstupy – Na tomto panelu jsou umístěny dvě ovládací tlačítka a to pro načtení a zápis digitálních výstupů. Pro načtení platí stejná funkce jako u digitálních vstupů. Pro zápis je nutné nejdříve označit digitální výstupy a poté zaslat

do indikátoru tlačítkem *Zapiš*. Platí stejná pravidla jako u vstupů log. 1 = označené, log. 0 = neonačené.

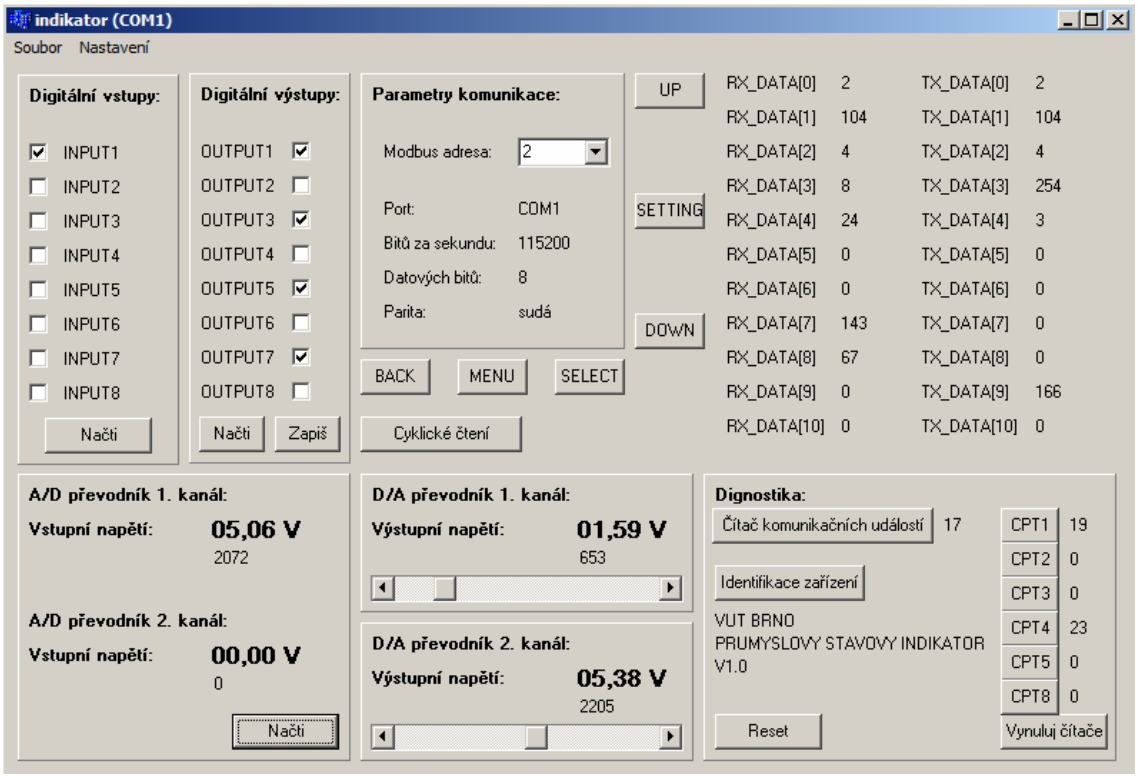
Panel A/D převodník – Opět je na panelu umístěno tlačítko pro načtení analogové hodnoty. Po stisknutí se zobrazí analogové hodnoty pro oba kanály jak ve float hodnotě napětí tak v dekadické podobě.

Panel D/A převodník – Na tomto panelu jsou umístěny dva posuvníky pro nastavení analogové hodnoty. Nastavená analogová hodnota je zobrazena jak ve formě float hodnoty napětí tak v dekadické podobě pro oba kanály. Po nastavení hodnoty napětí se nejdříve spustí časovač 100ms a až poté je hodnota poslána do indikátoru. Je to ošetření proti rychlým změnám nastavení napětí.

Panel Diagnostika – Na tomto panelu jsou umístěny kompletní ovládací prvky diagnostiky. Po zmáčknutí tlačítka se hodnota daného čítače zobrazí u tlačítka. Tlačítko reset slouží pro restart komunikace v indikátoru. Pod tlačítkem pro získání identifikace zařízení se zobrazí identifikace.

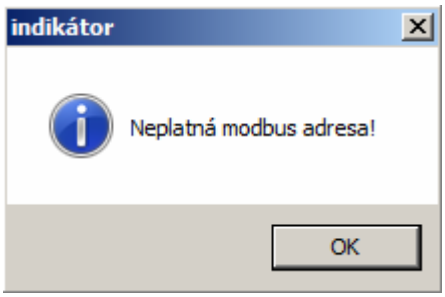
Panel Parametry komunikace – Na tomto panelu jednak nastavujeme adresu slave zařízení, ale také jsou zobrazeny nastavené parametry komunikace.

Jako hlavní stavební kámen celé aplikace je použita třída TSerial uvedená a popsaná v [22].



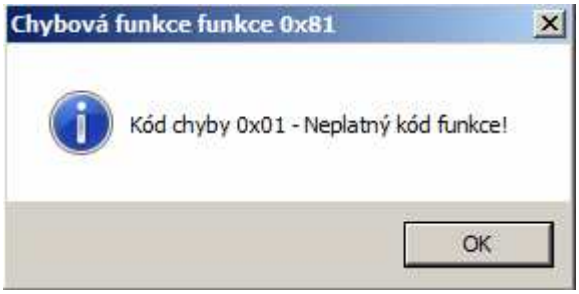
Obr. 8.4 Hlavní okno testovací aplikace

Aplikace také obsahuje výpis jednotlivých chybových hlášení. Při nezadání modbus adresy se zobrazí hlášení viz *obr. 8.5*.

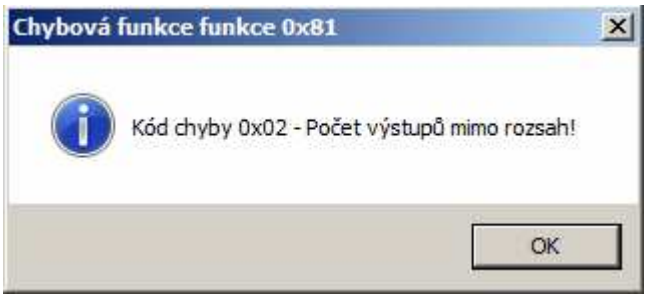


Obr. 8.5 Nezadaná modbus adresa

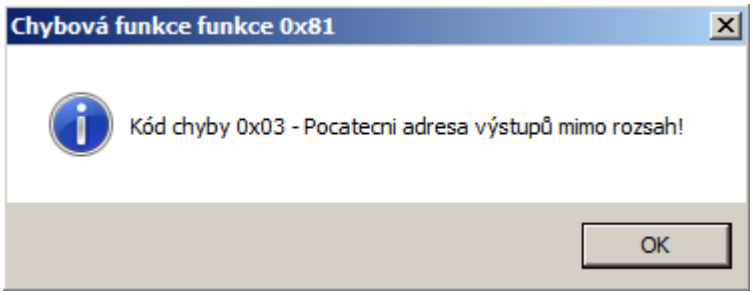
Přijaté byty, které obsahují chybovou funkci jsou zobrazeny v hlášení s přijatým kódem chybové funkce a kódem chyby. Například u čtení digitálních výstupů můžeme přijmout tyto chybové hlášení viz. *obr. 8.6*, *obr. 8.7*, *obr. 8.8*. Při zjištění chyby v kontrolním součtu pak hlášení viz *obr. 8.9.*



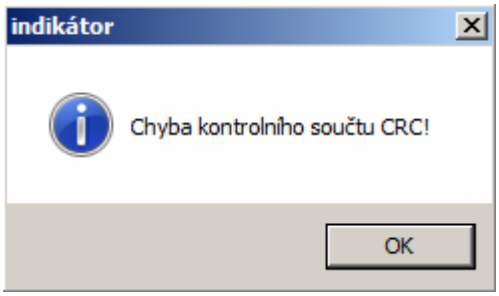
Obr. 8.6 Neplatný kód funkce



Obr. 8.7 Počet výstupů mimo rozsah



Obr. 8.8 Počáteční adresa výstupů mimo rozsah



Obr. 8.9 Chyba kontrolního součtu

9 ZÁVĚR

Cílem této diplomové práce byl návrh koncepce průmyslového stavového indikátoru pro zobrazování průmyslových dat vybaveného rozhraním RS-232/RS-485 a sadou digitálních vstupů a výstupů. Jednalo se o návrh elektroniky indikátoru spolu s programovým vybavením. Tato práce je pokračováním semestrální práce 1 a semestrální práce 2, kde jsem navrhoval hardware indikátoru a desky plošných spojů. Do hardwaru indikátoru byly přidány dva analogové vstupy a dva analogové výstupy.

Při návrhu hardwaru se vyskytla pouze jedna chyba a to u převodu z 5V logiky mikrokontroléru na 3,3V logiku pro LCD displej. Původně jsem realizoval tento převod pouze rezistorovým děličem napětí, ale jak se později ukázalo tak to nebylo nejlepší řešení, protože rezistory velice rušily komunikaci mezi mikrokontrolérem a LCD(vykreslování jiných barev, při cyklickém vykreslování rozostření číslic/písmen, nebo úplná ztráta dat). Pro odstranění tohoto rušení jsem dále upravil plošný spoj tak, aby mezi datovým a hodinovým signálem byla vedena zem(GND) a dané vodiče nebyly příliš blízko sebe a rezistorový dělič jsem nahradil TTL logikou použitím neinvertujícího oddělovače 74HC4050. Po této úpravě byla již komunikace bez problémů i při cyklickém překreslování LCD displeje. Další úprava spočívala v nahrazení NPN tranzistorů PNP tranzistory. Jak již bylo zmíněno náhrada byla z důvodu spínání výstupních relé při resetu mikrokontroléru. Jinak indikátor po osazení fungoval na první zapojení. Jen při ručním osazování konektoru LCD displeje a elektronického potenciometru je nutné mít dostatek trpělivosti.

O proti semestrálnímu projektu je změna také u krystalového oscilátoru. Původně jsem zamýšlel použít takový krystal, který by byl vhodný k přesnému nastavení sériové linky. Krystal o hodnotě kmitočtu 11,059Mhz je sice velice výhodný pro nastavení sériové linky, ale pro vykreslování LCD displeje je příliš pomalý a použití samotného krystalu o větší frekvenci jak 20MHz také není příliš vhodné. Nakonec jsem zvolil krystalový oscilátor o frekvenci 30Mhz. Toto sice přináší problémy s nastavením časování sériové linky, ale na druhou stranu vykreslování LCD displeje je dosti rychlé.

U programového vybavení jsem se delší dobu zabýval tím, proč nekomunikuje mikrokontrolér s LCD pomocí vestavěné sběrnice SPI. Problém je v tom, že mikrokontrolér má pouze 8bitový registr, ale LCD displej potřebuje ke komunikaci 9bitový příkaz. Nejdříve jsem to řešil způsobem, že devátý bit jsem nastavoval softwarově a zbytek hardwarovým SPI, ale nevedlo to k vyřešení problému. Tak jsem vytvořil softwarové SPI a LCD ihned začalo komunikovat s mikrokontrolérem. Sice nedosáhneme takové komunikační rychlosti, ale dané řešení má i své již popsané výhody. Řešením by bylo i použití mikrokontroléru, u kterého máme možnost velikost registru SPI nastavovat to bohužel u použitého nelze. V mikrokontroléru je navíc implementovaný protokol MODBUS. Je použit pouze binární režim RTU a podporuje 6 veřejných funkcí a 4 uživatelské.

Pro testování celého zařízení jsem vytvořil v prostředí C++ Builder aplikaci, ve které jsou obsaženy veškeré funkce, které indikátor poskytuje včetně ovládacích tlačítek. Celá aplikace komunikuje s indikátorem přes zmiňovanou linku RS-232, nebo RS-485 pomocí protokolu MODBUS.

Kdyby se měl indikátor používat v průmyslovém prostředí bylo by nutné především doplnit galvanické oddělení u komunikačních linek. Dále při menší modifikaci by bylo možné daný indikátor používat i jako vstupně/výstupní komunikační kartu pro PLC, nebo PC, které by s ní komunikovali prostřednictvím protokolu MODBUS. Modifikace by znamenala odstranit LCD a s ním spojené funkce z důvodu zrychlení odezvy a vyměnit krystalový oscilátor za oscilátor, pomocí kterého by jsme mohli nastavit přesné komunikační rychlosti např. s frekvencí 11,059MHz.

10 POUŽITÁ LITERATURA

[1] SPARKFUN. Color LCD 128x128 Nokia Knock-Off [online]. [cit. 2010-12-15]. Dostupný na WWW: http://www.sparkfun.com/products/569

[2] SPARKFUN. NOKIA 6100 LCD Display Driver [online]. [cit. 2010-12-15]. Revision 1. Dostupný na WWW: http://www.sparkfun.com/tutorial/Nokia%206100%20LCD%20Display%20Driver.pdf

[3] PHILLIPS. Datasheet PCF8833 [online]. [cit. 2003-02-14]. Dostupný na WWW: http://www.nxp.com/acrobat_download2/datasheets/PCF8833_1.pdf

[4] EPSON. Datasheet Epson S1D15G00 [online]. [cit. 2010-12-15]. Revision 1.0a. Dostupný na WWW: http://www.sparkfun.com/commerce/product_info.php?products_id=569

[5] HIROSE. Datasheet DF23 hirose [online]. [cit. 2004-04]. Dostupný na WWW: http://www.techdesign.be/shop/datasheets/df23%20hirose.pdf

[6] ATMEL. DatasheetAT89C51ED2 [online]. [cit. 2008-05]. Revision K. Dostupný na WWW: http://www.atmel.com/dyn/resources/prod_documents/doc4235.pdf

[7] STMicroelectronic. Datasheet MC34063 [online]. [cit. 2001-03]. Dostupný na WWW: http://www.datasheetcatalog.org/datasheets/2300/501448_DS.pdf

[8] MAXIM. Datasheet MAX232 [online]. [cit. 2010-07]. Revision 16. Dostupný na WWW: http://datasheets.maxim-ic.com/en/ds/MAX220-MAX249.pdf

[9] LINEAR. Datasheet LTC485 [online]. [cit. 2011-04]. Revision I. Dostupný na WWW: http://cds.linear.com/docs/Datasheet/485fi.pdf

[10] MAXIM. Datasheet MAX5400 [online]. [cit. 2000-10]. Revision 0. Dostupný na WWW: http://pdfserv.maxim-ic.com/en/ds/MAX5400-MAX5401.pdf

[11] SHARP. Datasheet PC817 [online]. [cit. 2010-12-15]. Dostupný na WWW: http://www.classiccmp.org/rtellason/chipdata/pc817.pdf

[12] MICROCHIP. Datasheet MCP3202 [online]. [cit. 1999-11-15]. Dostupný na WWW: http://cubloc.com/download/etc/MCP3202.pdf

[13] MICROCHIP. Datasheet MCP4922 [online]. [cit. 2004-05-28]. Dostupný na WWW: http://ww1.microchip.com/downloads/en/devicedoc/21897a.pdf

[14] TRACO. Datasheet TEN3-2411 [online]. [cit. 2009-12]. Dostupný na WWW: http://www.tracopower.com/fileadmin/medien/dokumente/pdf/datasheets/ten3.pdf

[15] ON Semiconductor. Datasheet 7812 CD2T [online]. [cit. 2009-09]. Revision 20. Dostupný na WWW: http://www.gme.cz/_dokumentace/dokumenty/934/934-070/dsh.934-070.1.pdf

[16] SGS-THOMSON. Datasheet 74HC148 [online]. [cit. 1992-10]. D Dostupný na WWW: http://www.datasheetcatalog.org/datasheets/270/491538_DS.pdf

[17] PHILLIPS. Datasheet 74HC4050 [online]. [cit. 1990-12]. Dostupný na WWW: http://ics.nxp.com/products/hc/datasheet/74hc4050.pdf

[18] TEXAS INSTRUMENTS. Datasheet TLC272 [online]. [cit. 1987-10]. [Revision 1994-09]. Dostupný na WWW: http://www.hep.upenn.edu/SNO/daq/parts/tlc272.pdf

[19] FAIRCHILD. Datasheet TL431 [online]. [cit. 2011-02-22]. Revision 1.0.4 Dostupný na WWW: http://www.fairchildsemi.com/ds/TL/TL431A.pdf

[20] DOC. ING. HEROUT, Pavel. Učebnice jazyka C, 1. díl a 2. díl. KOPP, České Budějovice, 2007.

[21] ING. MATOUŠEK, David. C pro mikrokontroléry ATMEL AT89S52, 6. díl. BEN – technická literatura Praha, 2007

[22] ING. MATOUŠEK, David. C++ Builder, 1. díl, 2. díl, 3. díl. BEN – technická literatura Praha, 2001.

[23] JEDLIČKA, Petr. Přehled obvodů řady CMOS 4000, 1. díl a 2. díl. BEN - technická literatura Praha, 2000.

[24] KREJČIŘÍK, Alexander, Napájecí zdroje, 1 vydání. BEN - technická literatura Praha, 1996.

[25] OLMR, Vít. Sériová linka RS-232 [online]. [cit. 2005-12-12]. Dostupný na WWW: http://hw.cz/rs-232

[26] STANĚK, Jan. RS 485 & 422 [online]. [cit. 1998-01-15]. Dostupný na WWW: http://hw.cz/docs/rs485/rs485.html

[27] ING. POUCHA, Pavel. Přenos dat po linkách RS485 a RS422 [online]. [cit. 2007-09-18]. Dostupný na WWW: http://www.rs485.cz/papouch1.htm

[28] ING. RONEŠOVÁ, Andrea. Přehled protokolu MODBUS [online]. [cit.2005-05]. Dostupný na WWW: http://home.zcu.cz/~ronesova/bastl/files/modbus.pdf

[29] MODBUS. MODBUS APPLICATION PROTOCOL SPECIFICATION V 1.1b. [online]. [cit. 2006-12-28]. Dostupný na WWW: http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b.pdf

[30] MODBUS. MODBUS over Serial Line Specification and Implementation Guide V1.02 [online]. [cit. 2006-12-20]. Dostupný na WWW: http://www.modbus.org/docs/Modbus_over_serial_line_V1_02.pdf

[31] WIKIPEDIA. Bresenham’s line algorithm [online]. [cit. 2011-04-24]. Dostupný na WWW: http://en.wikipedia.org/wiki/Bresenham%27s_line_algorithm