TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Sběr a vizualizace dat v průmyslových prostorách

Bakalářská práce

Studijní program: B2612 – Elektrotechnika a informatika
Studijní obor: 2612R011 – Elektronické informační a řídicí systémy

Autor práce: **Martin Šetina**
Vedoucí práce: Ing. Zbyněk Mader, Ph.D.

Liberec 2018

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

Data acquisition and visualization in industrial premises

Bachelor thesis

Study programme: B2612 – Electrical Engineering and Informatics
Study branch: 2612R011 – Electronic Information and Control Systems

Author: **Martin Šetina**
Supervisor: Ing. Zbyněk Mader, Ph.D.

Liberec 2018

Technická univerzita v Liberci
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2017/2018**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Martin Šetina**
Osobní číslo:	**M15000240**
Studijní program:	**B2612 Elektrotechnika a informatika**
Studijní obor:	**Elektronické informační a řídicí systémy**
Název tématu:	**Sběr a vizualizace dat v průmyslových prostorách**
Zadávající katedra:	**Ústav informačních technologií a elektroniky**

Zásady pro vypracování:

1. Seznamte se s 8bitovými mikrořadiči řady PIC, 1-Wire sběrnicí a komunikačním protokolem MicroLAN.
2. Navrhněte elektronické zařízení s mikrořadičem PIC, které vyhledá a načte všechna zařízení přípojené k síti MicroLAN, data zobrazí na LCD displeji a pošle je po sběrnici RS-485 do PLC nebo PC k dalšímu zpracování.
3. Realizujte elektronické zařízení do podoby osazené desky s plošnými spoji, funkčnost zařízení prověřte

Rozsah grafických prací: **Dle potřeby dokumentace**

Rozsah pracovní zprávy: **cca 30-40 stran**

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

[1] **HEROUT, Pavel. Učebnice jazyka C. 3. upr. vyd. České Budějovice: Kopp, 1994, 269 s. ISBN 8085828219**

[2] **MATOUŠEK, David. C pro mikrokontroléry PIC: práce s PIC18F452 a PIC18F1220 v jazyce C. 1. vyd. Praha: BEN - technická literatura, 2011, 367 s. C & praxe. ISBN 978-80-7300-413-2**

[3] **AXELSON, Jan. Serial Port Complete. Second Edition, Lakeview Research LLC Madison, WI 53704, ISBN 978-1931448-07-9**

Vedoucí bakalářské práce: **Ing. Zbyněk Mader, Ph.D.**
Ústav informačních technologií a elektroniky

Datum zadání bakalářské práce: **19. října 2017**

Termín odevzdání bakalářské práce: **14. května 2018**

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

L.S.

prof. Ing. Ondřej Novák, CSc.
vedoucí ústavu

V Liberci dne 19. října 2017

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 10. 5. 2018

Podpis: Šot

Poděkování

Chtěl bych poděkovat především svému vedoucímu projektu Ing. Zbyňkovi Maderovi Ph.D., že mi umožnil realizovat mé vlastní myšlenky a neomezoval mne při výběru použitých technologií.

Anotace

Tato práce se zabývá návrhem modulů pro digitální teplotní senzory DS18B20 od firmy Dallas. Řeší návrh PCB prototypu a obsahuje odladěné části programů do PIC18, PLC a PC. Popisuje princip komunikace po RS-485, přes kterou PLC jako master vysílá dotazy a data, postupně do „libovolného“ počtu modulů v režimu slave. Dále demonstruje možnosti komunikace PLC s PC přes Ethernet. Nastiňuje možné aplikační oblasti tohoto modulu s krátkými ukázkami. Klade si za cíl především nastínit část problematiky z oblasti MaR a ukázat možná řešení.

Nejde tedy o kompletní zapouzdřenou aplikaci, kterou je možné nasadit do provozu. K tomu bude třeba ještě ošetřit možné poruchové stavy a zabezpečit komunikaci a další. Komunikační protokoly jsou v „surovém“ stavu kvůli snadné demonstraci principu. Bude třeba doplnit vlastní algoritmy pro regulaci, které budou pracovat s RTC a s časovými programy, které budou dostupné z uživatelské aplikace vytvořené např. v C# na běžném počítači.

Klíčová slova

1-Wire, MicroLAN, enumerace, sériová komunikace, Ethernet

Annotation

This thesis deals with the design of DS18B20 digital temperature sensors from the company Dallas. The thesis proposes the PCB design of the prototype and contains the debugged parts of the programs into the PIC18 the PLC and the PC. Furthermore it describes the RS-485 communication principle, through which the PLC as a master sends queries and data sequentially to "any" number of slave modules. It also demonstrates the possibilities of PLCs communicating with a PC via Ethernet. It outlines the possible application areas of this module with some short examples. It aims to outline some of the MaR issues and to show possible solutions.

This thesis is not a complete encapsulated application that can be put into operation. In addition, it will be necessary to treat possible faults and to ensure communication and more. Communication protocols are in the "raw" status for easy demonstration of the principle. It will be necessary to add custom control algorithms that will work with RTC and with time programs that will be available from a user-created application such as C # on a regular computer.

Keywords

1-Wire, MicroLAN, enumeration, Serial communication, Ethernet

Obsah

Seznam obrázků

Seznam použitých zkratek

1-Wire	sběrnice navržená firmou Dallas Semiconductor
A/D	převodník analogového signálu na digitální hodnoty do PLC
AIN	analogový vstup (0-10V)
AOUT	analogový výstup
ASCII	standardizovaná tabulka znaků (American Standard Code for Information Interchange)
AWG	označení průřezu vodiče (American Wire Gauge)
C#	C Sharp – vysokoúrovňový objektově orientovaný programovací jazyk
CNC	počítačem řízené synchronní osy, obvykle pro obrábění (Computer Numeric Control)
D/A	převodník digitální hodnoty z PLC na analogovou (0-10V nebo 4-20mA)
DIN	digitální vstup
DIP28	pouzdro IC s 2x14 drátovými vývody (dual in-line package)
DOUT	digitální výstup (log. „1“ odpovídá +24V)
DS18B20	číslicový teploměr Dallas (s 1-Wire komunikací)
EUSART	integrované rozhraní v MCU (Enhanced Universal Synchronous Asynchronous Receiver Transmitter)
FX5..	nejnovější řada kompaktních PLC Mitsubishi (r.2018)
HMI	operátorský panel (Human Machine Interface)
I/O	vstupy a výstupy PLC (Input/Output)
IC	někdy IO, integrovaný obvod (Integrated Circuit)
IEC	mezinárodní asociace pro elektroniku (International Electrotechnical Commission) mimo jiné vytváří programovací standard IEC 61131-2
IoT	Internet věcí (Internet of Things)
kbps	přenosová rychlost (kilobits per second)
LAN	síť, propojující PC příp. technologická zařízení (Local Area Network)
LCD	displej z tekutých krystalů (Liquid Crystal Display)
MaR	měření a regulace technologických celků (teploty, energie aj.)
MicroLAN	síť zařízení 1-Wire
MCU	jednočipový mikropočítač - mikrokontrolér (Micro Controller Unit)
NTC	termistor – s vyšší teplotou má nižší odpor (negative temperature coefficient)
PC	libovolný počítač (Personal Computer)
PCB	někdy DPS, deska plošného spoje (Printed Circuit Board)

PIC	označení jednočipových počítačů firmy Microchip (Peripheral Interface Controller)
PICkit	programátor MCU PIC od firmy Microchip Technology
PLC	programovatelný logický automat (Programmable Logic Controller)
PT100	platinové odporové teplotní čidlo (100Ω při 0°C)
PTC	termistor – s vyšší teplotou má vyšší odpor (positive temperature coefficient)
PWM	pulzně šířková modulace (Pulse Width Modulation)
RISC	architektura procesorů (Reduced Instruction Set Computing)
RS-232	starší standard sériové linky s jinou napěťovou úrovní a doprovodnými signály
RS-485	standard sériové komunikace definovaný v roce 1983 sdružením EIA
RTC	hodiny reálného času a datum (Real Time Clock)
ŘS	řídící systém
SCAN	1 celý kompletní programový cyklus PLC, který se cyklicky opakuje
SMT	PCB pro povrchovou montáž součástek (Surface mount technology)
Spinel	univerzální komunikační protokol sestavený firmou Papouch.cz
THT	PCB s předvrtanými otvory pro součástky s drátovými vývody (Through-hole technology)
TO92	pouzdro pro polovodičové součástky se třemi drátovými vývody

1. Úvod

Toto téma bakalářské práce jsem si vybral proto, že mi sběrnice 1-Wire připadá neprávem opomíjena. Zvláště v dnešní době, kdy se stále více mluví o IoT (internetu věcí), se možná mnohdy zbytečně připojují k LAN všemožná zařízení. Obzvláště ve velkých průmyslových halách je z hlediska úspory kabeláže mnohem výhodnější použití nějaké sběrnice, jako je právě 1-Wire. Teploměry, vlhkoměry a další zařízení s nízkými nároky na datový přenos a napájení, lze výhodně prosmyčkovat obyčejnou dvoulinkou, namísto mnoha větví ethernetových kabelů.

Problematika mě zajímá i z důvodu, že mohu uplatnit nabité znalosti z oblasti programování MCU, PLC a jazyka C a C#. Přesto, že sériová linka ani 1-Wire sběrnice nemohou přenosovou rychlostí konkurovat ethernetu, jsou v mnoha oblastech výhodnější a těžko nahraditelné. Sběr dat ze senzorů je právě jednou z nich. Výhody jsou především ve sběrnicové topologii, velkém dosahu (RS-485 více jak 1km a 1-Wire přes 300m) i při použití běžných nestíněných kabelů a 1-Wire může současně sloužit i k napájení těchto čidel. Sériová linka lze pak již snadno implementovat do PLC nebo PC, které řeší regulaci komplexně v součinnosti s dalšími aspekty požadavky.

S rozvojem automatizace a zvyšujícími se nároky na kvalitu, roste i potřeba zahrnovat do algoritmů více vstupních veličin (včetně teplot). Dnes je prakticky v každém zařízení nějaký ŘS a jejich pořizovací cena je tím pádem stále příznivější. Systémy, které lze použít k řízení nebo k regulaci bych rozdělil do třech skupin: MCU, PLC a PC. Každá má své uplatnění, výhody i nevýhody. I když se oblasti jejich použití částečně překrývají, uchovávají si specifické vlastnosti, pro které může komplexní řešení vyžadovat nasazení všech současně. Pokusím se je stručně charakterizovat v dalších kapitolách.

1.1. Cíl práce

Tato práce navazuje na ročníkový projekt z loňského roku, ve kterém jsem se zabýval prostou komunikací s jedním teplotním čidlem DS18B20. V rámci této práce jsem především upravil PCB modulu a doplnil funkci parazitního napájení přes 1-Wire. Ta umožňuje po jednom vodiči přenášet data i napájení pro více čidel současně. Výhoda je v úspoře jednoho vodiče, prodloužení dosahu, případně napojení odlišných zařízení, jako je iButton (Java Ring) pro autentizaci atd. Další úpravy modulu, jako je doplnění PWM podsvícení LCD a několika dalších součástek, nejsou pro tuto práci podstatné.

Zaměřil jsem se především na princip enumerace, což je vyhledávání všech připojených zařízení na sběrnici MicroLAN. Jde o úžasnou vlastnost, charakteristickou právě pro 1-Wire. Ve své práci popisuji tuto metodu zjištění jedinečných 64bitových adres a detailní princip komunikace.

Použil jsem PLC Mitsubishi z kompaktní řady FX5U, neboť má k dispozici v základní výbavě programovatelný sériový port RS-485. Přes něj posílám dotazy do jednoho či více modulů. Modul jsem pojmenoval „MaRE“, jako zkratku slov „Měření a Regulace Energií“. K jednotlivým MaRE může být pak přes MicroLAN připojen libovolný počet

zařízení, v mém případě teplotních čidel DS18B20. Moduly provedou nejprve enumeraci zařízení a následně postupně a cyklicky spouští měření a čtou teplotu ze všech nalezených teploměrů. Hodnoty postupně zobrazují na LCD displeji a na dotaz PLC pak odešlou přes RS-485.

Modul MaRE jsem navrhl s 8bitovým mikrokontrolérem firmy Microchip PIC18F26K22, pro který je zdarma dostupný překladač C pod označením XC8. PCB jsem navrhl jako oboustranný v provedení THT, kvůli snadnějšímu ručnímu osazování. Snažil jsem se o co nejmenší rozměry PCB kvůli ceně a praktičnosti. Zároveň jsem při návrhu myslel i na univerzálnost, abych mohl využit i k dalším účelům.

1.2. Vývoj řízení s MCU

Regulace pomocí vlastního modulu s MCU obnáší návrh PCB dle konkrétních požadavků. Pro malá autonomní zařízení a velké série je obvykle nejvýhodnější. Vyplatí se tam, kde neočekáváme potřebu rozšiřování o další I/O, nebo jiné periferie. Při malých rozměrech může obsahovat přesně potřebnou kombinaci analogových a digitálních I/O. Výhodou je velmi příznivá cena. Malé 8 bitové MCU lze pořídit už za několik desítek korun a náklady na vývoj a výrobu PCB se „rozmělní“ do opakované výroby.

Na rozdíl od PLC jsou velmi omezené programové možnosti. Malá MCU v architektuře RISC mají k dispozici typicky méně než 50 instrukcí. Nejsou zde žádná „hotová“ řešení výrobce (pro připojení dalších periferií, jako je HMI atd.) a konstruktér se musí vypořádat od základů se vším, od ošetření vstupů proti rušivým zákmitům, až po způsob programování a ladění (připojení externího programátoru, jako např. v mém případě PICkit™ 3).

1.3. Řízení pomocí PLC

Jde o autonomní řídící systém s možností rozšiřujících modulů (případně vzdálených I/O) od nějakého renomovaného výrobce (Mitsubishi, Omron, Siemens, TECO, AMiT aj.). Pro rozsáhlejší nebo jednoúčelové výrobní stroje, kde lze očekávat změny na požadavky výroby, variantní zařízení, je výhodnější a mnohdy nezbytné. Je obvykle v průmyslovém provedení, výrobce musí počítat s náročnějšími podmínkami, jako větší rozsah pracovních teplot, vibrace atd. Programování a oživování zejména velkých projektů je díky podpoře rychlejší a komfortnější. Jsou k dispozici knihovny, rozsáhlá instrukční sada, aplikované instrukce a předdefinované programové funkční bloky. Rovněž programování a ladění je přívětivější. PLC umožňují snadné přímé připojení k PC, online monitoring, online změny programu a další. Díky modulární rozšiřitelnosti není problém připojení dalších I/O, se kterými se v zadání nepočítalo. Navíc existují různé speciální moduly, např. pro komunikaci s firemní databází, CNC řízení atd. Pořizovací náklady nejmenších kompaktních PLC se pohybují v jednotkách tisíc Kč, ale se speciálními moduly může cena PLC přesahovat několik set tisíc Kč. Většina PLC nemá v základním modulu analogové I/O (nebo jen velmi omezený

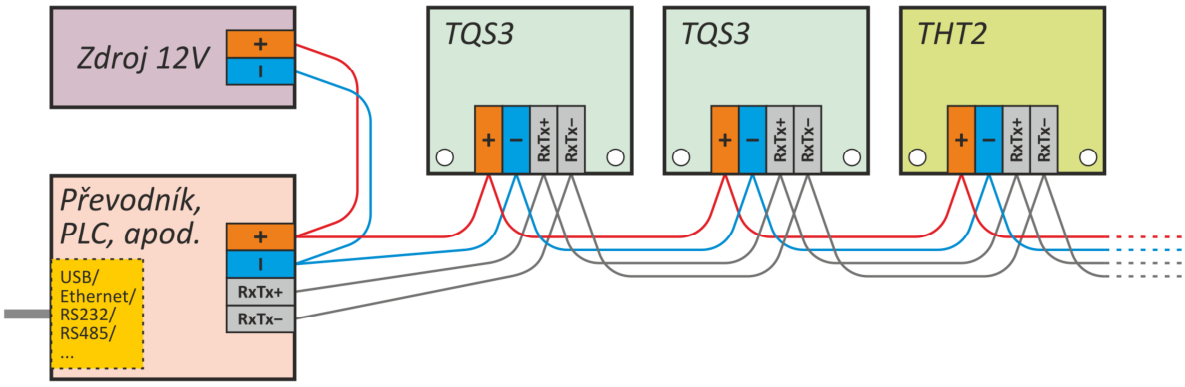
počet) a jsou orientovány především na digitální I/O. Analogové I/O se řeší rozšiřujícími moduly k PLC. Výhodou je rychlá implementace na straně PLC (řešení podporované výrobcem), rychlé vzorkování (obnova hodnoty prakticky každý scan PLC). Velký důraz je v poslední době kladen i na bezpečnostní funkce. Selhání řízení může mít někde za následek poškození technologie, nebo dokonce zranění či usmrcení osob. Podle stupně rizika je možné nasadit dražší bezpečnostní (safety) PLC, které výrobce atestuje na řízení a hlídání kritických oblastí. Nevýhodou je pak především cena, která roste s každým rozšiřujícím modulem a omezený maximální počet podle typu PLC.

1.4. Řízení pomocí PC

Počítač je pravděpodobně nejrozšířenější řídící systém. Výpočetním výkonem daleko přesahuje MCU i PLC, ale pro automatizaci a průmyslové řízení technologií je oblast jeho použití velmi omezená. I když je v zásadě použitelný pro řízení libovolného procesu, v praxi se tak příliš neděje. Důvody jsou prosté. Počítač potřebuje pro vlastní chod složitý operační systém, který není dlouhodobě stabilní, vyžaduje aktualizace, restarty atd. Chod řídící aplikace může fatálně ovlivnit jiný banální proces, který v PC běží. Při napojení na síť může být terčem hackerských útoků, napaden viry atd. Na rozdíl od MCU a PLC, které využívají převážně procesory s harvardskou architekturou, mají PC své procesory založené na Von Neumann architektuře. Ta využívá společnou paměť pro data i instrukce, což může opět nepříznivě ovlivnit stabilitu. Pro řízení procesů tedy nejsou vhodné a využívají se především k vizualizaci a ukládání dat, kde samotné PLC nestačí, ale přímo k řízení se nevyužívají.

1.5. Dostupné řešení na trhu

Firma Papouch s.r.o. vyrábí a prodává za cenu kolem 1000,-Kč bez Dph teploměry pod označením THT a TQS. Jde o teploměry a vlhkoměry s komunikací přes RS-485. Jde ale pouze o čidla se sériovou linkou, komunikující protokolem MODBUS nebo Spinel, který si sami navrhli. Čidla nemají displej pro zobrazení. Navíc musí být před zapojením dle Obr. 1 nastavena jejich jedinečná adresa. To kromě omezeného počtu (max. 32 teploměrů) přináší i komplikace s možnou kolizí při shodných adresách, výměně čidel atd. Nelze je nijak dále rozšiřovat, mají pouze jednu LED, která signalizuje čtení teploty. Výrobce neposkytuje zdrojový kód pro MCU, pouze popis jejich komunikačního protokolu. Dalším omezením je 12V napájení. Na větší vzdálenosti (stovky metrů) může být problém s úbytky napětí. Cenově dostupný datový kabel, který by byl na instalaci vhodný, UTP CAT 5E 4x2xAWG24 má udávaný odpor $\leq 0.188\Omega/m$. Teploměr má max. odběr 20mA, což odpovídá ohmické zátěži 600Ω. Pokud budeme chtít na linku zapojit např. 30 teploměrů, jsme na 20Ω a na 100m vedení (18,8Ω) bude úbytek téměř 6V. To už nebude dostatečné napětí pro funkci teploměrů. Z tohoto důvodu jsem na svých modulech použil DC/DC regulátor AMSR7805, který umožňuje rozsah vstupního napětí 6,5-32V, čímž lze použít větší napájecí napětí a úbytky na vedení nám nebudou vadit.



Obr. 1 – princip zapojení modulů a teploměrů, zdroj [4]

Komunikační protokol Spinel je detailně v manuálech firmy Papouch popsán, takže jsem se rozhodl pro své moduly vytvořit stejné kompatibilní jádro, aby bylo možné kombinovat na jedné lince RS-485 i s kupovanými teploměry Papouch.

2. Mikrořadič PIC18F26K22

Modulu MaRE obsahuje 8 bitový mikropočítač z rodiny PIC18, který je optimalizován pro překladače C. Obvod je navržen v architektuře RISC a obsahuje tuto HW výbavu:

Programová paměť Flash	65536 Bytes (32768 instrukcí po 16 bitech)
Datová EEPROM	1024 Bytes
Datová SRAM	3896 Bytes
CPU maximální takt	16 MIPS (při 64 MHz s PLL)
Komunikační periferie	2-UART, 2-SPI, 2-I^2C, 2-MSSP(SPI/ I^2C)
PWM	2-CCP, 3-ECCP
Časovače	3x 8-bit, 4x 16-bit
Analogový vstup	17 ch, 10-bit
Analogový komparátor	2x
Teplotní rozsah	-40 až 125°C
Pracovní napětí	1,8 až 5,5V
Počet IO pinů	28
Maximální proud 1 výstup	25mA
Maximální proud z Vss	300mA
Maximální proud do Vdd	200mA
Možné zdroje přerušení	33
Vektory přerušení	2
Základní instrukční sada	75 instrukcí

Z další výbavy stojí za zmínku interní oscilátory 16MHz, 500kHz a 31,25kHz, Watchdog časovač a režim spánku s minimální spotřebou. Vše je detailně popsáno v manuálu od výrobce. Já tyto funkce ve své práci nevyužívám, proto se jim zde detailně nevěnuji.

Pro jednoduchost obvodu lze snadno programovat i bez překladače přímo v assembleru. Jediná svízel může snad nastat při nedůsledném hlídání přepínání Bank datové paměti (0 – 15), neboť musí být kvůli 8bitové adresaci takto rozdělena. Případná chyba v programu (špatně nastavené BSR) se velmi obtížně hledá. Z tohoto pohledu je zajisté lepší použít překladač C, který je dnes již zdarma a tyto problémy řeší za nás. Výsledný kód je sice o něco delší, zato ale překladač nabízí funkce a matematické operace přímo s datovými typy jako je float, dint a další. To se v assembleru provádí někdy dost těžko.

2.1. Konfigurace MCU

Pro správnou funkci je třeba nejprve v programovém prostředí nastavit tzv. konfigurační bity. K tomu slouží v MPLAB v záložce Production -> Set Configurations bits. Jde o nastavení klíčových funkcí, které nelze měnit za chodu z programu, jako je volba oscilátoru, zamknutí programu atd. Nachází se v MCU na adresách 300001 – 30000D. Po nastavení se vygeneruje tlačítkem kód, který se přidá programu. Nastavení detailně popisovat nebudu, je poměrně rozsáhlé a zřejmé.

Další nastavení, které lze ovládat přímo z programu je především konfigurace vstupů a výstupů a povolení a nastavení přerušení. Konkrétní nastavení je dostupné ve zdrojovém kódu, v této práci pouze slovně popíšu, co bude potřeba k vysvětlení.

2.2. Výběr krystalu

Jelikož se komunikační rychlost asynchronního sériového přenosu odvíjí od taktovací frekvence obvodu, je třeba navrhnout správný krystal a správně nastavit příslušné registry. Obvod má dokonce i automatickou funkci detekci přenosové rychlosti, ale tu nepoužívám. Vycházím ze vztahu (zdroj [1]):

$$BaudRate = \frac{Fosc}{64([SPBRGH1{:}SPBRG1]) + 1)}$$

Abych se dostal na standardní komunikační rychlost 19200bps (případně 9600) a hodnoty 8bitových registrů **SPBRGH1** a **SPBRG1** vycházely na celé číslo (nulová časová odchylka), použil jsem krystal s frekvencí 18,432MHz. Pak stačí nastavit **SPBRG1 = 14. SPBRGH1** je výchozí v nule a další související registry mohou zůstat rovněž ve výchozím nastavení. Detailní principu je patrný z obrázku výrobce níže. Jelikož obvod obsahuje EUSART 2x, ale princip je shodný, obsahují registry písmeno „x“, které se nahradí v mém případě číslem jedna. S nastavení EUSART souvisí ještě mnoho dalších registrů, jako **IPRx, PIEx, PIRx** atd. ale všechny jsou především kvůli přerušení, které může být zavoláno přijetím i

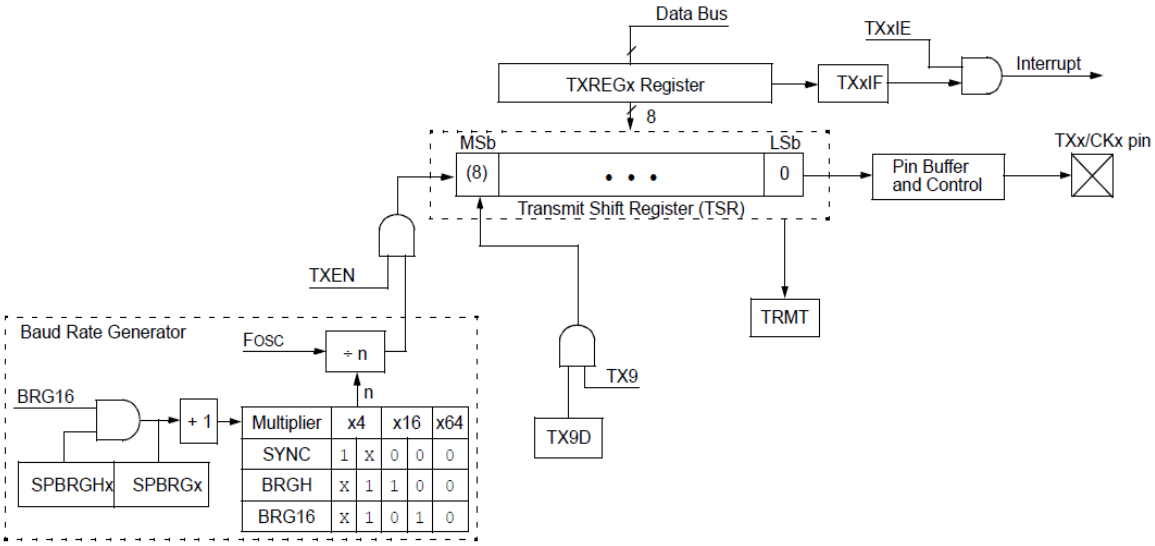
dokončením odeslání celého bajtu. Já používám pouze přerušení od příchozích dat, které je signalizováno flagem **RC1IF**.

2.3. Odesílání dat přes RS-485

Před odesíláním dat se výstup RC5 musí přepnout do HI, což přepne budič RS-485 na vysílání. Vyslání bajtu se provede jeho prostým zápisem do **TXREG1**. Dokončení odeslání je pak signalizováno flagem **TXSTA1,TRMT** do HI (Transmit Shift Register is empty). Abych mohl z programu odesílat najednou celé textové řetězce, vytvořil jsem si funkci **write_uart1.**

```
void uart1_xmit(unsigned char mydata_byte) // tato funkce odesle 1 char na EUART
{
    while(!TXSTA1bits.TRMT);               // ceka na uvolneni portu
    TXREG1 = mydata_byte;                  // pak zahaji vysilani
}

void write_uart1(const char *txt)          // tato funkce odesle textovy retezec
{
    EN_TX = true;                          // prepni budic RS-485 na vysilani
    while(*txt != 0) uart1_xmit(*txt++);   // odesilej string po charech
    while(!TXSTA1bits.TRMT);               // počkej na dokonceni vysilani
    EN_TX = false;                         // pak prepni budic RS-485 na prijem
}
```

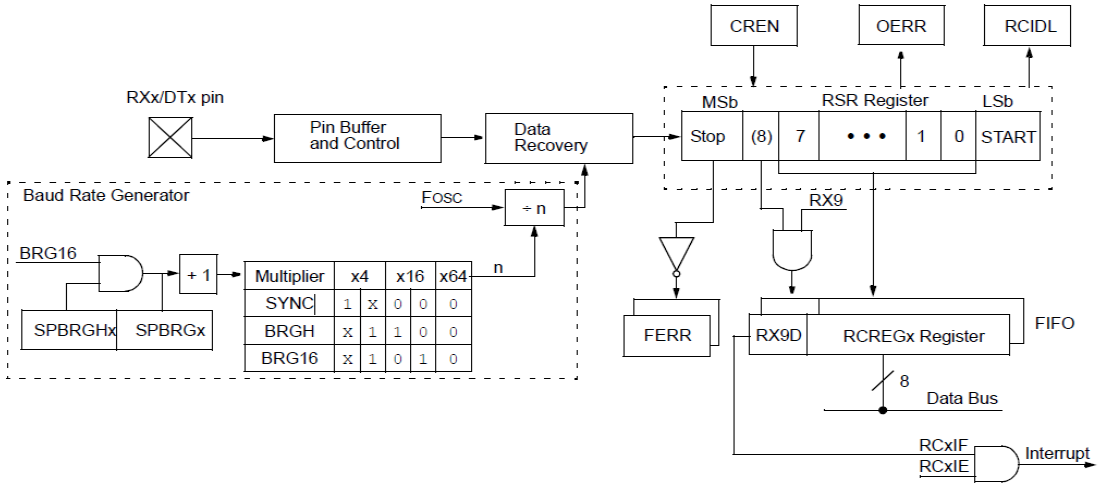


Obr. 2 – EUSART blokový diagram vysílání dat, zdroj [1]

2.4. Příjem dat přes RS-485

Pokud PIC nevysílá, je trvale v „naslouchacím“ režimu (RC5 - LOW). Přijetím 1B dat přeteče buffer **RCREG1**, který pomocí **RC1IF** vyvolá přerušení. Toto přerušení mám nakonfigurováno na „low priority“ (programový skok na adresu 0018h) a k jeho obsloužení jsem v XC8 použil takovouto funkci:

```
void interrupt low_priority low_isr(void)   // Low priority interrupt
{
    if (PIR1bits.RC1IF)
    {
        rxByte = RCREG1;                    // RC1IF se shodi prectenim RCREG1
        if (rxIndex > 1) rxIndex++;         // pokud bezi prenos, inkrementuj
        if (rxByte == 0x02) rxIndex = 1;    // zacatek prenosu
    }
    return;
}
```

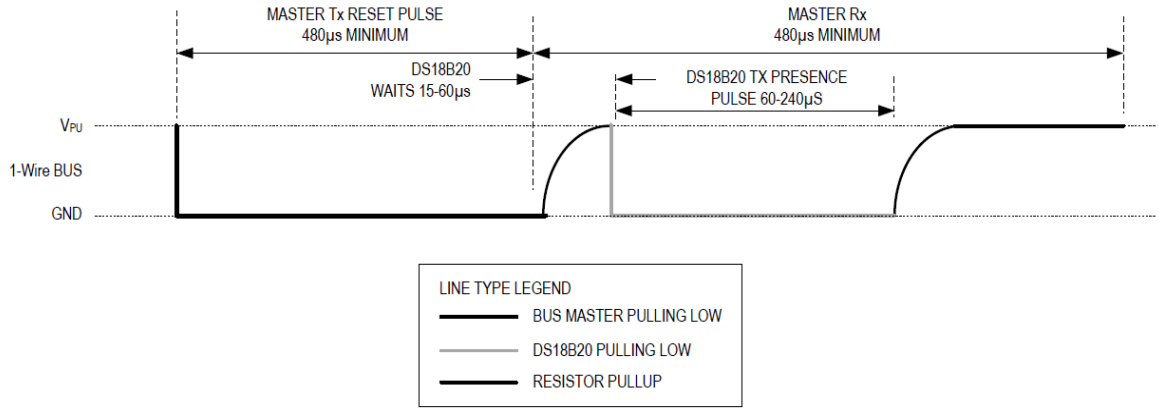


Obr. 3 – EUSART blokový diagram příjmu dat, zdroj [1]

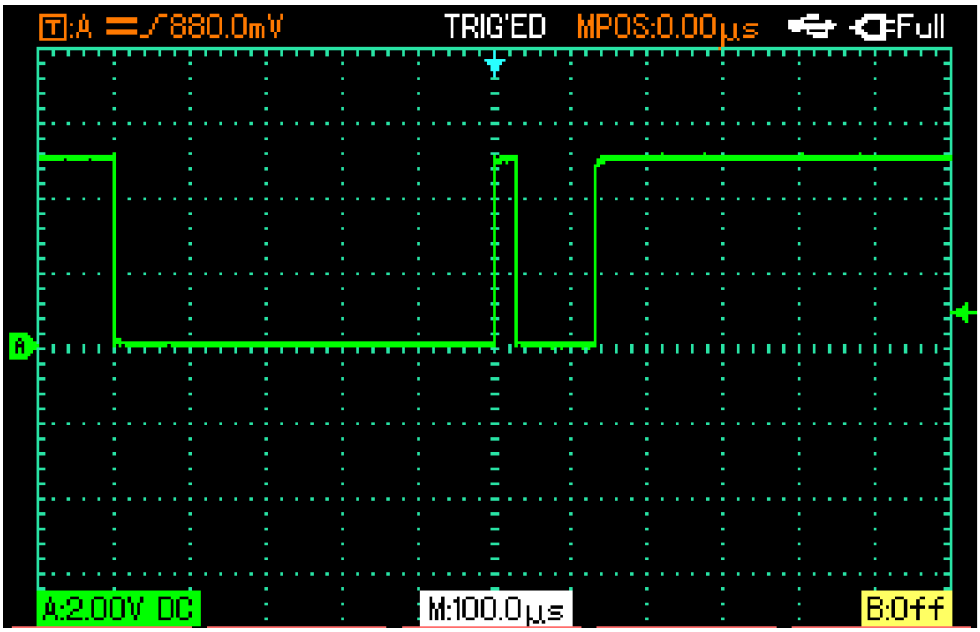
3. Komunikace 1-Wire

Jak je patrné ze schématu zapojení níže, ke komunikaci využívám 2 vývody z portu B. RB5 připíná napájecí napětí v čase, kdy neprobíhá komunikace. RB7 se přepíná mezi vstupem a výstupem podle toho, zda právě vysílá. Jak je patrné ze zapojení, sběrnice je v klidovém stavu v HI, kvůli napájení čidel. Zařízení, které komunikuje, spíná vodič sběrnice impulzně k nule. Komunikaci zahajuje vždy master, v mém případě tedy modul MaRE. Ten nejprve vyšle resetovací impulz. To provede zavřením napájecího tranzistoru a stáhnutím sběrnice na

480µs k nule. Pokud je na sběrnici 1 nebo více zařízení, odpoví rovněž spojením sběrnice se zemí na čas 60 - 240µs.



Obr. 4 – časová inicializace 1-wire reset



Obr. 5 – 1-wire reset osciloskopem

V programu mám definováno a zajištěno takto:

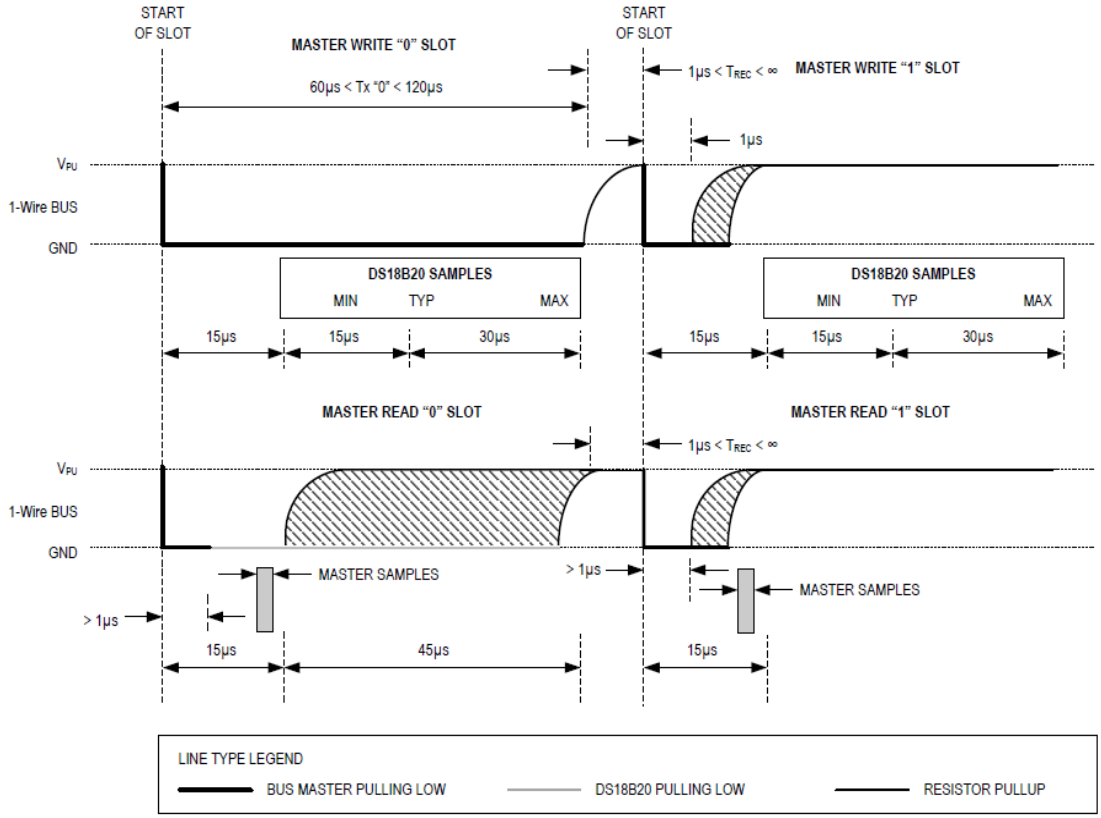
```
#define PWR_OW      PORTBbits.RB5
#define OW          PORTBbits.RB7
#define OW_TRIS     TRISBbits.TRISB7


bit owReset()
{
```

```
    bool ds_present = false;
    OW_TRIS = 0;        // prepni RB7 na vystup
    OW = 0;             // posli k nule
    __delay_us(500);    // prodleva
    OW = 1;             // znovu zvedni
    OW_TRIS = 1;        // a prepni na pin na vstup
    __delay_us(80);
    ds_present = !OW;   // na lince je pripojeno alespon 1 cidlo
    while (OW == 0);    // cekej na konec inicializacniho pulzu cidla
    return ds_present;
}
```

Nyní jsou všechna zařízení na sběrnici připravena v naslouchacím režimu a očekávají ROM povel od masteru. Těch existuje několik podle zapojení. Záleží, zda máme na sběrnici pouze 1 zařízení, nebo jich můžeme očekávat více. V mém případě počítám s více teploměry, takže příkaz přímého čtení READ (33h) bez adresace nemohu používat. Musím nejprve zjistit všechny 64bitové adresy na sběrnici (počet teploměrů) pomocí enumerace příkazem SEARCH (F0h) a pak mohu postupně komunikovat již pomocí těchto adres s jednotlivými teploměry pomocí příkazem MATCH (55h). Příkaz ALARM SEARCH (ECh), který slouží k rychlé detekci teploty mimo přednastavené meze, nepoužívám.

Každý příkaz je dlouhý 1 Bajt a odesílá se postupně po bitech od bitu 0 po bit 7. Log „1“ představuje krátké stažení k nule a dlouho trvající návrat k „1“, log „0“ pak přesně obráceně. Lepší pochopení je opět z obrázku výrobce.



Obr. 6 – časový diagram R/W jednoho bitu, zdroj [1]

V programu řeším funkcemi `owReadBit` a `owWriteBit` v principu takto:

```
bool owReadBit(void)
{
    bool log;
    OW_TRIS = 0;        // prepni RB7 na vystup
    OW = 0;
    __delay_us(2);
    OW_TRIS = 1;
    __delay_us(8);
    log = OW;
    __delay_us(60);
    return log;
}

void owWriteBit(bool wrBit)
{
    OW_TRIS = 0;        // prepni RB7 na vystup
    OW = 0;
    if (wrBit)
    {
        __delay_us(2);
        OW = 1;
        OW_TRIS = 1;
        __delay_us(68);
```

```
    }
    else
    {
        __delay_us(68);
        OW = 1;
        OW_TRIS = 1;
        __delay_us(2);
    }
}
```

V programu využívám vyšší funkce `owReadByte` a `owWriteByte`, které tyto bitové funkce vždy 8x zavolají a mohu pracovat s celými bajty. Jsou jednoduché, takže je zde nebudu ani podrobně a popisovat.

3.1. Řízení RX / TX

V případě, že už jsou všechny adresy vyhledány, master odesílá společný příkaz na konverzi (měření teploty) a následně přečte teplotu ze všech teploměrů, podle následující tabulky. V programu jsou funkce pojmenovány `owConvert` a `owReadTempFromAddr`. Jedná se o zjednodušený rychlý princip, který není popsaný v manuálu. Funguje spolehlivě, číst celý Scratchpad (9B) je zbytečné.

Read Rx / transmit Tx	data	popis
Tx	Reset pulz	
Rx	potvrzení	
Tx	CCh	Příkaz Skip ROM
Tx	44h	Příkaz Convert T
Tx	Reset pulz	
Rx	potvrzení	
Tx	55h	Příkaz Match ROM
Tx	64bit ROM code	adresa DS18B20
Tx	BEh	Příkaz čtení Scratchpad
Rx	Temperature LSB	Přečtu spodní bajt
Rx	Temperature MSB	A horní bajt

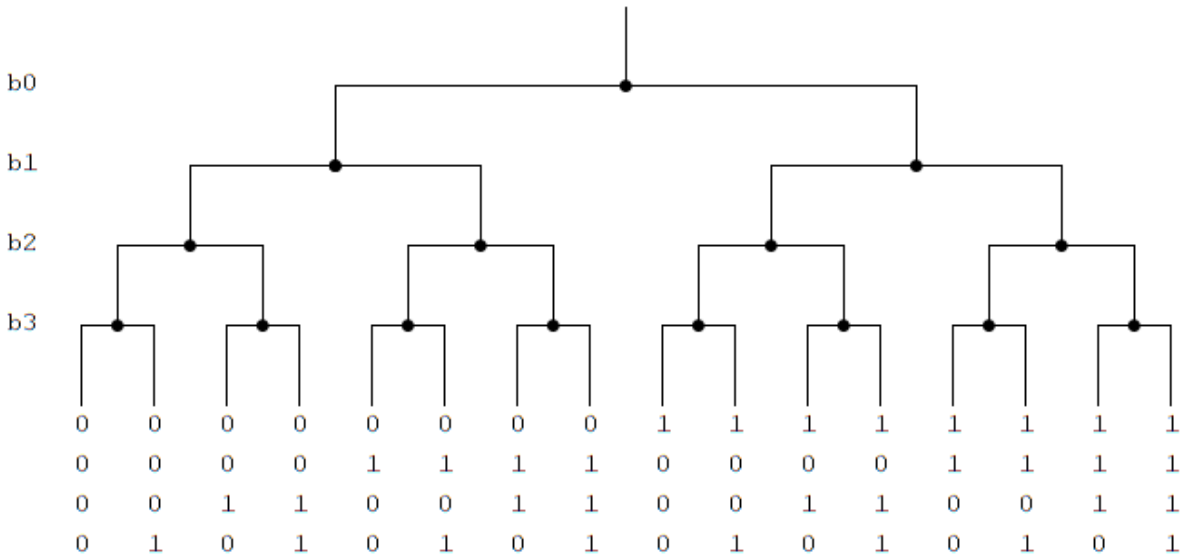
Další úspora v komunikaci je vysílání příkazu na měření teploty 44h. Ten lze posílat rovnou do všech teploměrů, což manuál opět nezmiňuje.

3.2. Popis enumerace

Jak jsem již zmiňoval, tato vlastnost mi připadá na celé sběrnici naprosto unikátní a dokonalá. Bez nutnosti nějaké jednotlivé konfigurace nebo postupného připojování lze prostě zjistit všechna připojená čidla na sběrnici. Komunikace začíná opět resetem a následně příkazem F0h SEARCH ROM. Poté přečtu 1 bit pomocí `owReadBit`, kde mi odpoví pouze

zařízení, která mají ve své adrese na bitu 0 log „1“ (vyšlou „1“). Pak přečtu další bit ze zběrnice, kde mi odpoví všechna zařízení, která mají v tomto bitu log „0“ (opět vyšlou „1“). Takto jsem zjistil, zda je na sběrnici jedno či více zařízení, jehož adresa má v nejnižším bitu log „1“ a zda je tam jedno či více zařízení, které tam má „0“. Zda odpovídá jedno zařízení, nebo více se shodnou hodnotou příslušného bitu přirozeně nepoznám. To ale nevadí. Pokud je odpověď pouze na 1 bit (ať už 0, nebo 1), tuto hodnotu bitu zopakuji. Vyšlu pomocí `owWriteBit`, a uložím si ji. Tím se posouvám na kontrolu dalšího vyššího bitu adresy. V mém případě tedy na kontrolu bitu 1 z adresy. Celý cyklus se opakuje, přečtu jeden bit, kde odpoví zařízení s log „1“ a druhý bit, kde odpoví zařízení s log „0“. Pokud je na sběrnici pouze 1 zařízení, děje se takto až do bitu 63 a takto přečtu adresu. Jelikož jsou ale adresy jedinečné, musím při více jak jednom zařízení logicky narazit na bit, kde se adresa liší. Pak jde o větev, kde si mohu vybrat, který bit zopakuji (kudy se vydám). Zařízení s opačným bitem v adrese, než vyšlu, se už dále hlásit nebudou. Proto si místo musím zapamatovat a zopakovat celý proces ještě jednou s opačnou hodnotou. Takto čtu bity adresy až do nejvyššího bitu 63. Každým cyklem najdu jedno zařízení (získám 1 unikátní adresu). Délka trvání vyhledání jedné adresy na sběrnici je teoreticky přibližně součet: 1000µs RESET + 8x80µs COMMAND + 3x64x80µs ENUMERACE = 17ms. Vzhledem k prodlevám mezi bity je ale třeba počítat s časem o něco delším.

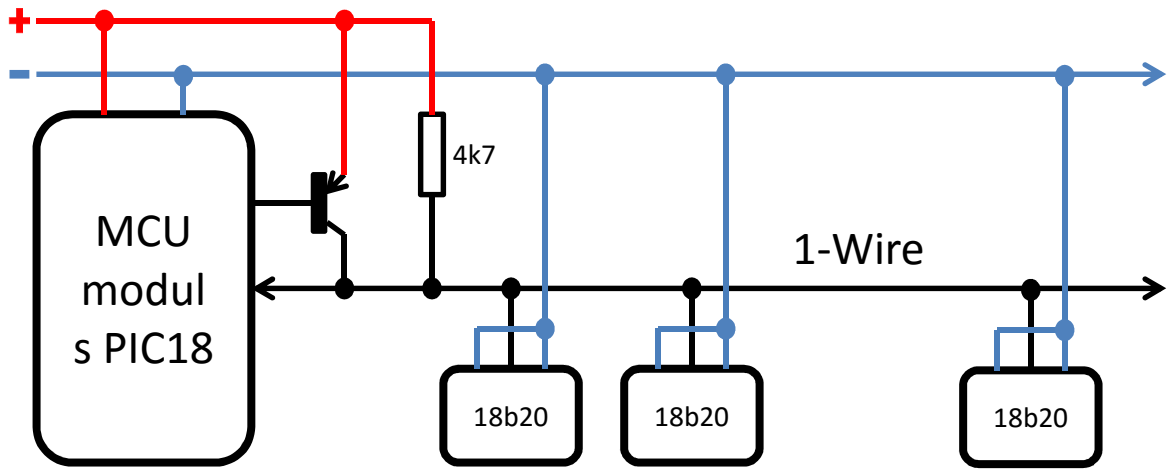
Lze si představit jako rozhodovací větve, které musím postupně všechny projít, abych nalezl adresy zařízení. Na demonstračním obrázku je zachycen princip pro čtyři bity adresy se všemi možnostmi. Program musí dokázat postupně najít a projít všechny větve v 64b adrese (podle počtu zařízení), což není pro PIC18 s necelými 4kB RAM úplně snadný úkol.



Obr. 7 – enumerační strom

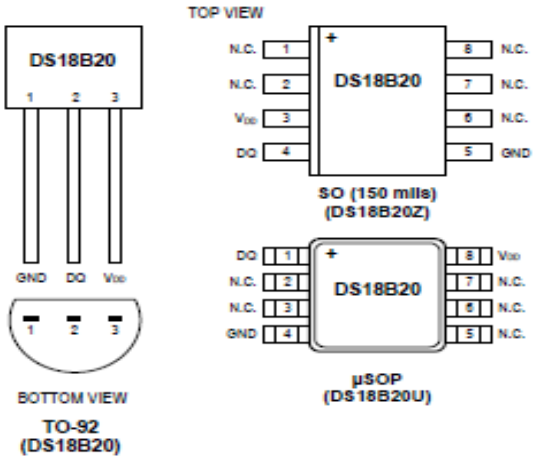
Možností, jak programově řešit je určitě více. Já zvolil zřejmě ne nejlepší cestu dvourozměrné proměnné `volatile unsigned char addr[8][10]`, kam ukládám po bajtech nalezené adresy. Překladač mě ale dovolí maximálně deset zařízení. Na testování principu je ale dostatečné. Funkci `owSearch` zde vypisovat nebudu, je poměrně dlouhá. Funguje ale dle popsaného principu a vrací počet nalezených adres. Vlastní adresy zapisuje do globální proměnné `addr`.

3.3. Teploměry DS18B20



Obr. 8 – zapojení 1-Wire

Teploměry se připojují přímo k PIC, jak je patrné z katalogového zapojení. Možné je plné 3vodičové zapojení, nebo tzv. parazitní napájení. Pak stačí pro připojení čidla pouze 2 vodiče. Čidla pracují na principu porovnávání dvou čítačů frekvencí od oscilátorů s odlišnými teplotními kompenzacemi. Obvod je možné přepínat na 4 různě dlouhá a přesná měření pomocí dvou bitů v 4. bajtu registru scratchpad. 9 bitové nastavení vypisuje hodnoty v rozlišení 0,5°C a doba měření je max. 93,75ms. 10b..0,25°C, 11b..0,125°C a nejdelší 12 bitové je s rozlišením 0,0625°C a trvá max. 750ms. Toto



Obr. 9 – pouzdra teploměrů, zdroj [2]

nejpřesnější je výchozí a v programu ho na nižší neměním. Z této vlastnosti vyplívá i omezení periody čtení teploty, která nemůže být logicky kratší než čas měření čidla.

Maximální teplotní rozsah DS18B20 je od -55°C do +125°C. V rozsahu od -10°C do +85°C je přesnost čidel ±0,5°C. V krajních částech celkového rozsahu se pak přesnost může zhoršit na ±2°C. Pro jedno konkrétní čidlo je ale odchylka opakovaně stejná, takže lze kalibrovat a přesnější měření není zbytečné. Úplné stanovení chyby měření je patrné z chybové křivky, viz datasheet DS18B20. Sensor pracuje s napájecím napětím 3.0 až 5,5V. Max proud při měření je 1,5mA a mimo tuto dobu max. 1µA. Možnou délku kabelu sběrnice 1-Wire uvádí výrobce až 300m, ale já jsem testoval bez problémů pouze několik desítek. Záleží samozřejmě na typu vedení. Vlastní teplotu vypisuje čidlo přes 1-wire v běžném (binárně kódovaném) formátu ve dvou bajtech. Horní 4 bity tvoří znaménko (0 je plus a 1 mínus), bit 4 až bit 10 udávají celočíselnou hodnotu a bit 3 až bit 0 mají zpřesňující binární váhu 0,5; 0,25; 0,125; 0,0625°C.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
2^3	2^2	2^1	2^0	2^{-1}	2^{-2} *	2^{-3} *	2^{-4} *
Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
S	S	S	S	S	2^6	2^5	2^4

* tyto bity se nastavují pouze při měření v odpovídající přesnosti

4. Vývoj modulu MaRE

Schéma a DPS jsem kreslil v programu AUTODESK EAGLE 8.6.3, který je pro nekomerční použití zdarma. Modul je možné osadit dvěma stabilizátory, pokud by bylo zapotřebí pomocné napětí 12V pro případné externí zařízení. Vlastní modul s LCD potřebuje pouze 5V, takže pro funkci teploměru a zobrazovače se první stabilizátor IC3 nemusí osazovat a jeho vývody 1 a 3 se pouze spojí propojkou. Místo lineárních stabilizátorů 78xx jsou vhodnější DC/DC spínané stepdown měniče AMSR (viz rozpiska), které mají menší tepelné ztráty a větší pracovní rozsah napájení (6,5 - 32V) a ochranu proti zkratu a přehřátí. Velký napájecí rozsah tak bez problémů pokryje případné ztráty na dlouhém vedení. S výhodou lze použít pro napájení 24VDC - vnitřního zdroje PLC FX5U. PIC má integrovaný oscilátor, ale na DPS jsem raději osadil krystal 18,432MHz, abych zajistil co nejspolehlivější komunikaci. Programovací konektor SV1 je pro programátor PRESTO firmy ASIX, ale drátovou redukcí lze připojit i PICkit. Dioda D2 je kvůli omezení programovacího napětí programátoru PRESTO, pro PICkit je možné nahradit drátovou propojkou. Na desce je jedno tlačítko S1, které může mít funkci MCLR (Main Clear), nebo libovolnou dle programu. Na svorkách A4, A5 je datová sběrnice RS-485, kterou obsluhuje IC2. Výhodou je, že není potřeba žádné další diskrétní součástky. Na posledním modulu by měl být mezi vývody zapojen terminační odpor 330R, ale při testování s běžnými délkami kabelů je parazitní kapacita zanedbatelná a odpor zbytečný. Na svorky A6,8,11,12 je možné připojit variantně další zařízení, např. ovládací tlačítka. Testoval jsem čtecí hlavu RFID s protokolem Wiegand.

Jeden interní teploměr DS18B20 lze připojit na A3,7,10 a další externí spojené do sítě MicroLAN pak dvoudrátově na A3,9. Svorky DIN a DOUT (A13,14,15,16) se mohou použít pro přímou regulaci. V této práci ale nijak nevyužívám. Dále je na DPS podsvícený displej LCD 2x8 znaků. Trimrem R1 se seřizuje kontrast LCD. Desku není nutné samozřejmě osazovat kompletně, nepoužívané prvky (včetně displeje) je možné variantně vynechávat.

Okomentovaný zdrojový kód do PIC je v příloze. Psal jsem ho v MPLAB X IDE v4.00 a použil kompilátor XC8 v1.43. Obojí je freeware přímo od Microchip Technology Inc.

4.1. Komunikace s LCD

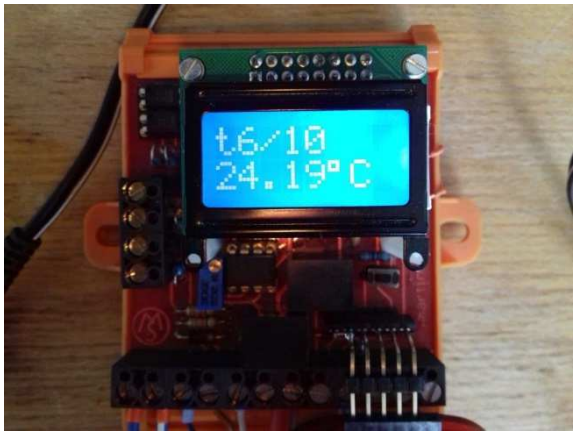
Displej používá k řízení obvod typu HD44780. Mohou přenášet data po 4 nebo 8 vodičích. Já používám komunikaci po čtyřech vodičích, kvůli úspoře pinů MCU. Zapojení je patrné z výkresu. Vývod **RS** řídí výběr přenosu dat (log „1“) nebo příkazu (log „0“), **R/$\overline{W}$** čtení (to nepoužívám) mám trvale zápis (log „0“) a **E** je povolení (řídící signál).

Komunikace probíhá tak, že se přivede log „1“ na E, nastaví se horní 4 bity z bajtu, který chci přenést do DB4-7 a do LCD se zapíší sestupnou hranou na **E**. Poté totéž s dolními čtyřmi bity.

LCD umožňuje vytvořit i vlastní speciální znaky a další srandy, ale já používám pouze běžný režim zobrazení a jen znaky spodní části ASCII tabulky. Pro tuto funkci je ovšem po zapnutí třeba ještě inicializační sekvence. Ta vypadá v assembleru v principu takto:

```
DelayMs(15);              //pauza po zapnutí
LCDPORT = 0b0000011;      //čtyř-bit
E_PIN = 1; E_PIN = 0;     //zapíše do LCD
DelayMs(5);
E_PIN = 1; E_PIN = 0;
DelayMs(1);
E_PIN = 1; E_PIN = 0;
DelayMs(1);
LCDPORT = 0b0000010;     //dvouřádkový režim
E_PIN = 1; E_PIN = 0;
//dále už funkce LcdWr zapisují postupně horní a dolní 4 bity
LcdWrCmd(0x28);          //počet bitů, 2 řádky, znak 5x7
LcdWrCmd(0x0C);          //dislej ON, kurzor OFF, blikání OFF
LcdWrCmd(0x01);          //smaže displej, kurzor na pozici 0
LcdWrCmd(0x06);          //směr kurzoru – posunu displeje
```

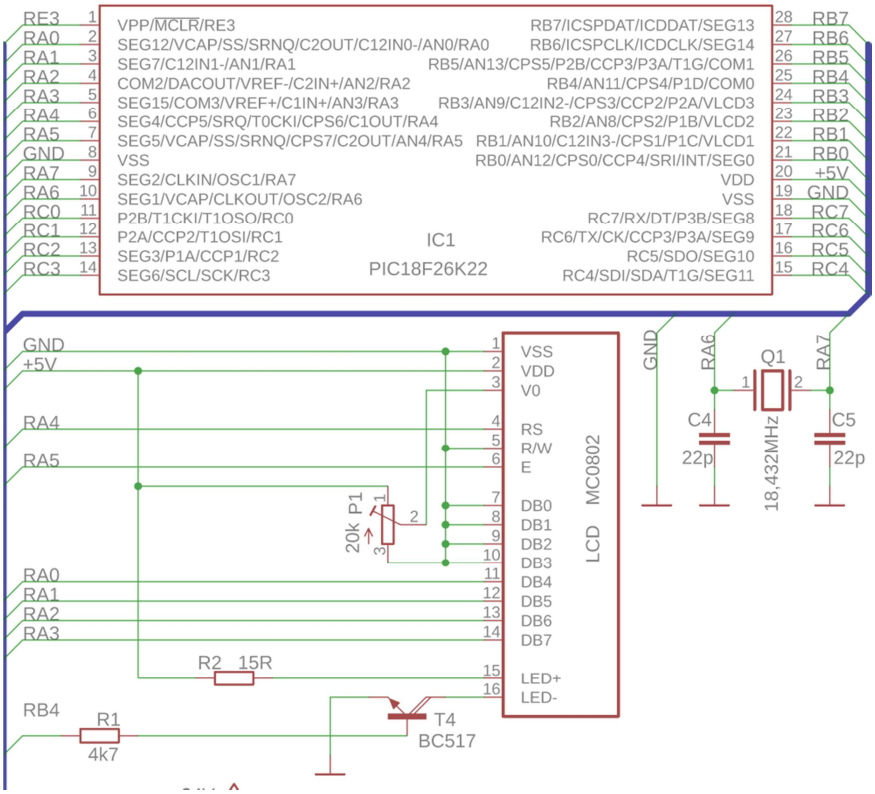
Já ale používám v překladači XC8 knihovnu výrobce `lcd`, která vše řeší za mě. Pracuji s jejími funkcemi `LCDGoto` pro skok na znak a řádek a `LCDPutStr` pro vykreslení řetězce. Zdrojový program v příloze vykresluje na 1. řádek LCD postupně pořadové číslo čidla / z kolika nalezených a na 2. Řádku teplotu.



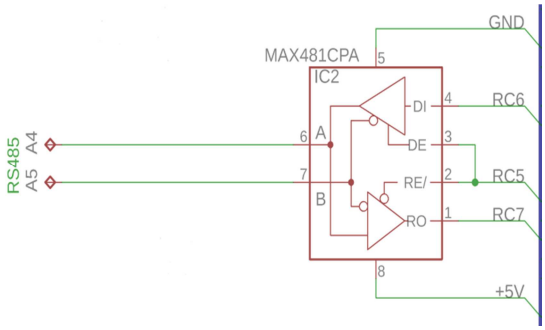
Obr. 10 – ukázka zobrazení LCD

4.2. Schéma a návrh PCB

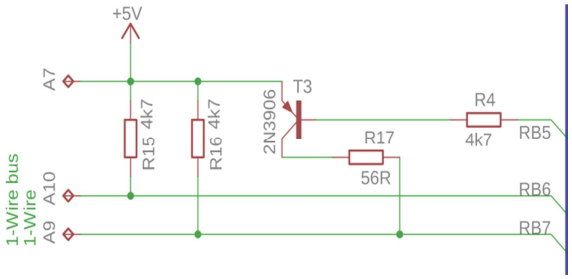
Kompletní schéma zapojení modulu MaRE je v příloze. Zde jsou obrázky pouze těch částí schématu, které jsou pro tuto práci důležité.



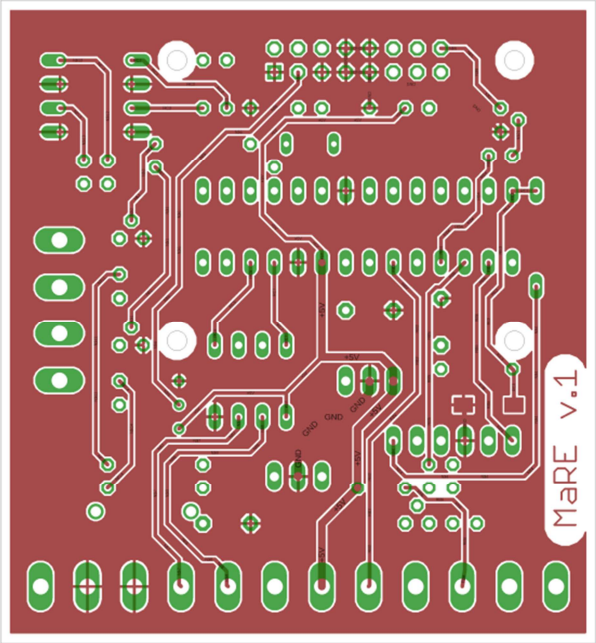
Obr. 11 – zapojení PIC a LCD



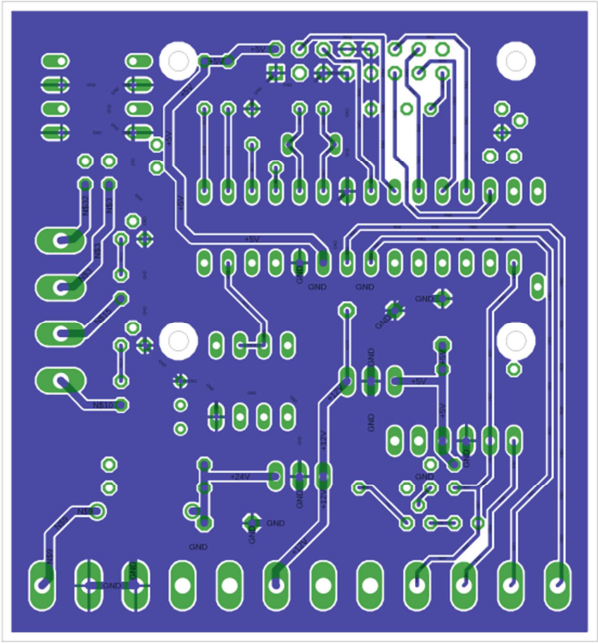
Obr. 12 – zapojení RS-485



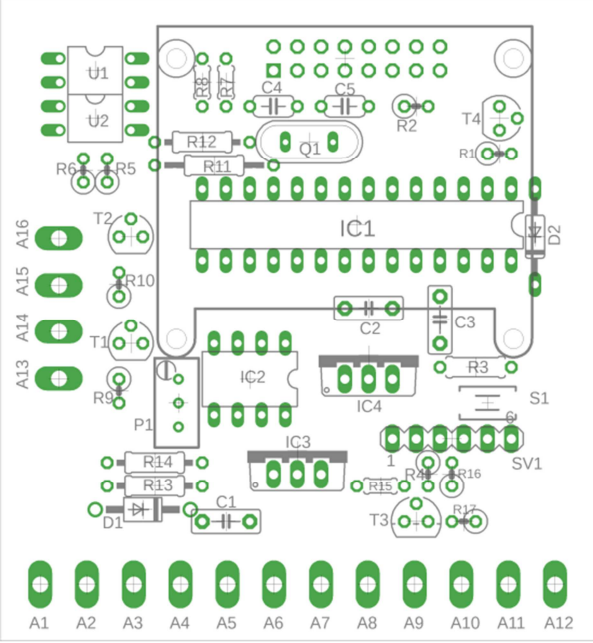
Obr. 13 – zapojení 1-Wire



Obr. 15 – náhled PCB top



Obr. 14 – náhled PCB bottom



Obr. 16 – náhled PCB osazení top

4.3. Rozpiska materiálu

1	ks	PCB MaRE v.1 64x69		
1	ks	DIODA 1N4007	220-002 GM	D1
1	ks	DIODA ZENEROVA BZX85V005.1	222-047 GM	D2
1	ks	DISPLEJ LCD MC0802A-SYL/H	513-122 GM	
3	ks	KONDENZÁTOR CK 22P /500V	120-018 GM	C4,5
3	ks	KONDENZÁTOR CK 100N / 63V	120-060 GM	C1,2,3
2	ks	kolík přímý ASS12038G	832-162 GM	
1	ks	lišta 2x20 pin BL240G	832-072 GM	
8	ks	SVORKOVNICE ARK 210/2	821-003 GM	
1	ks	KRYSTAL Q 18.432 MHz	131-022 GM	Q1
1	ks	PIC18F26K22-I/SP DIP28	TME	IC1
1	ks	MIKROSPÍNAČ DTSM31N-F 3,5x6	TME	S1
1	ks	OBVOD MAX485CPA+ DIP8	TME	IC2
1	ks	ODPOR RR 15R	110-029 GM	R2
8	ks	ODPOR RRU 4k7	119-074 GM	R1,4,11,12,13,14,15,16
3	ks	ODPOR RR 56R	110-043 GM	R9,10,17
3	ks	ODPOR RR 10K	110-097 GM	R3,7,8
2	ks	ODPOR RR 1K	110-073 GM	R5,6
2	ks	OPTOČLEN PC817	523-052 GM	U1,2
2	ks	PATICE DIL08PZ	824-002 GM	U1,2
2	ks	PATICE DIL14PZ	824-003 GM	IC1
1	ks	STABILIZÁTOR AMSR1-7805-NZ	TME	IC4
1	ks	STABILIZÁTOR AMSR1-7812-NZ	TME	IC3
5	ks	SLOUPEK DISTANČNÍ FIX-HP2-21	TME	
10	ks	ŠROUB M2 X 4 válc.hl.	uchycení LCD	
3	ks	TRANZISTOR darling NPN BC517	210-100 GM	T1,2
1	ks	TRANZISTOR PNP 2N3906	215-575 GM	T3
1	ks	TRIMR T910W-20K 0,5W 28 OT.	112-482 GM	P1
6,5	cm	PROFIL PF RS 80 OR NA DPS	415744 WEID	
2	ks	BOČNICE AP 80 D OR	132436 WEID	

Přibližná cena materiálu při plném osazení je 800 Kč

5. PLC Mitsubishi FX5U

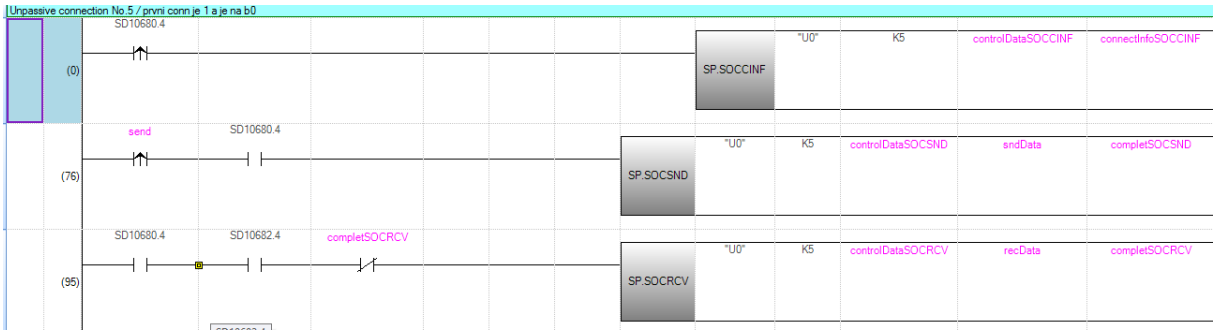
- Základní jednotky v provedení s 32, 64 a 80 I/O
- Napájecí napětí 85 – 264 VAC
- Vnitřní zdroj 24 VDC (400/600mA) pro libovolné periferie
- Reléové nebo tranzistorové výstupy
- 1-fázové a 2-fázové vysokorychlostní čítače (8 kanálů)
- Pulzní výstupy 200 kHz, 4 osy (Open collector), jednoduchá interpolace PWM modulované výstupy s rozlišením 1µs. (4 kanály)
- Slot pro paměťové karty SD
- 2x analogový vstup 0-10V (rozlišení 12-bit)
- 1x analogový výstup 0-10V (rozlišení 12-bit)
- Ethernetový port – volně programovatelný – max. 9 otevřených portů současně
- Sériový port RS-485

- Sloty pro rozšiřující adaptéry
- Paměť programu 64000 kroků
- Rychlost 34ns / logickou instrukci
- 29 sekvenčních a 124 aplikovaných instrukcí
- 7680 vnitřních bitových proměnných

Výhodou těchto PLC je příznivá pořizovací cena, široká výbava v základní jednotce a rozsáhlá instrukční sada. Za nevýhodu považuji ukončení podpory oblíbeného Structured ladder/FBD a nahrazení FBD/LD v GXworks3. Ten už není strukturován do ladder blocks a pro větší programy je nepřehledný. Dále zde stále chybí implementace vyšší komunikace, jako např. EtherCAT nebo PROFINET, pro snadnou implementaci dalších zařízení jiných výrobců. To ale není pro naše účely zapotřebí.

5.1. Konfigurace ethernetu

Je v rozbalovacím navigátoru pod Parameter -> FX5UCPU -> Module Parameter -> Ethernet Port -> Detailed Setting. PLC umožňuje na ethernetovém portu kromě jednoho výchozího MELSOFT spojení nakonfigurovat dalších osm, maximálně tedy může být současně 9 navázaných spojení.
V našem případě budu využívat pouze Unpassive spojení no.5, pro navázání spojení z nadřazeného PC. Connect no.1 je dobré doplnit vždy, aby bylo možné mít současně připojený panel HMI a pracovat s GX works3. Ostatním možnostem se věnovat nebudu, nejsou pro naše potřeby důležité. Pozor, pro převzetí změn je někdy třeba PLC vypnout a zapnout.

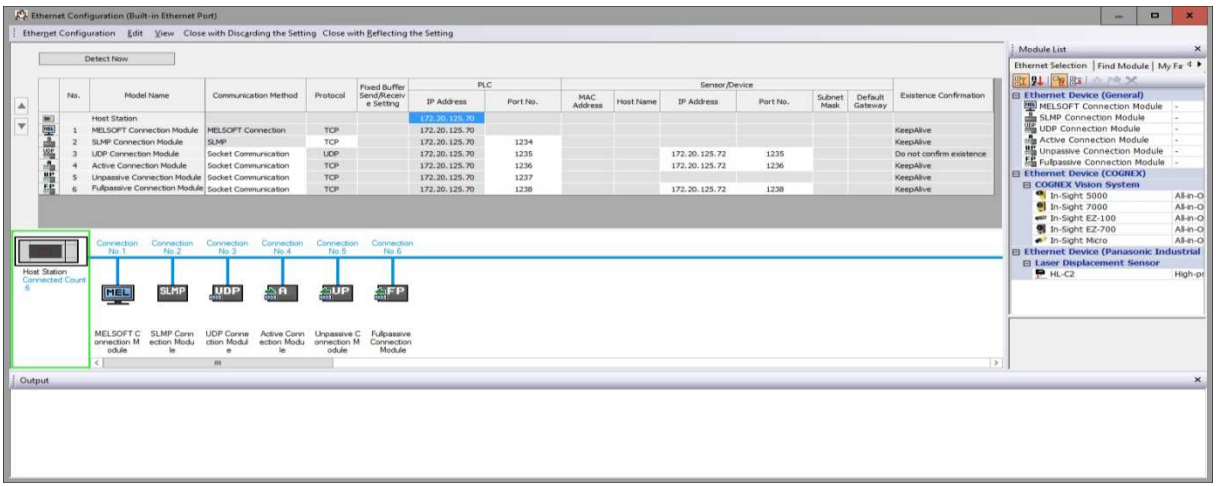


Obr. 17 - hlavička prog. eth. PLC

Ke komunikaci se používají instrukce `SP.SOCSND` pro odeslání dat a `SP.SOCRCV` pro příjem. Dále ještě bity systémových registrů `SD10680.4` (navázané spojení) a `SD10682.4` (přijatá data). `SP.SOCCINF` je pro zjištění detailů o navázaném spojení.

Při navázání spojení z libovolného zařízení do PLC, v našem případě na IP 172.20.125.70 na port 1237 lze jednoduše číst a zapisovat do vnitřních registrů D. Např. dotaz na přečtení dat z registrů D100 – D104 vypadá takto:

```
44h 52h 64h 00h 68h 00h
```



Obr. 18 - konfigurace ethernetu PLC

Zvolil jsem pro snadné zapamatování - první dva znaky jsou v ASCII „D“ a „R“ (čtení oblasti D) 0064h je začátek oblasti (dekadicky 100) a 0068h konec (104).

Pokud překvapí obrácený zápis bajtů (64h 00h) je to pro to, že Mitsubishi přijímá a vysílá nejdříve spodní bajt, který má nižší váhu, takže číslo se čte odzadu. Je to obráceně než např. u SIMATIC (SIEMENS). Jde o problematiku zvanou Endianita (pořadí bajtů, anglicky byte order). Mitsubishi je tzv. Little-endian. Samozřejmě není problém jednou instrukcí SWAP v kódu změnit.

PLC odpoví (zopakuje a připojí hodnoty z registrů) ve tvaru:

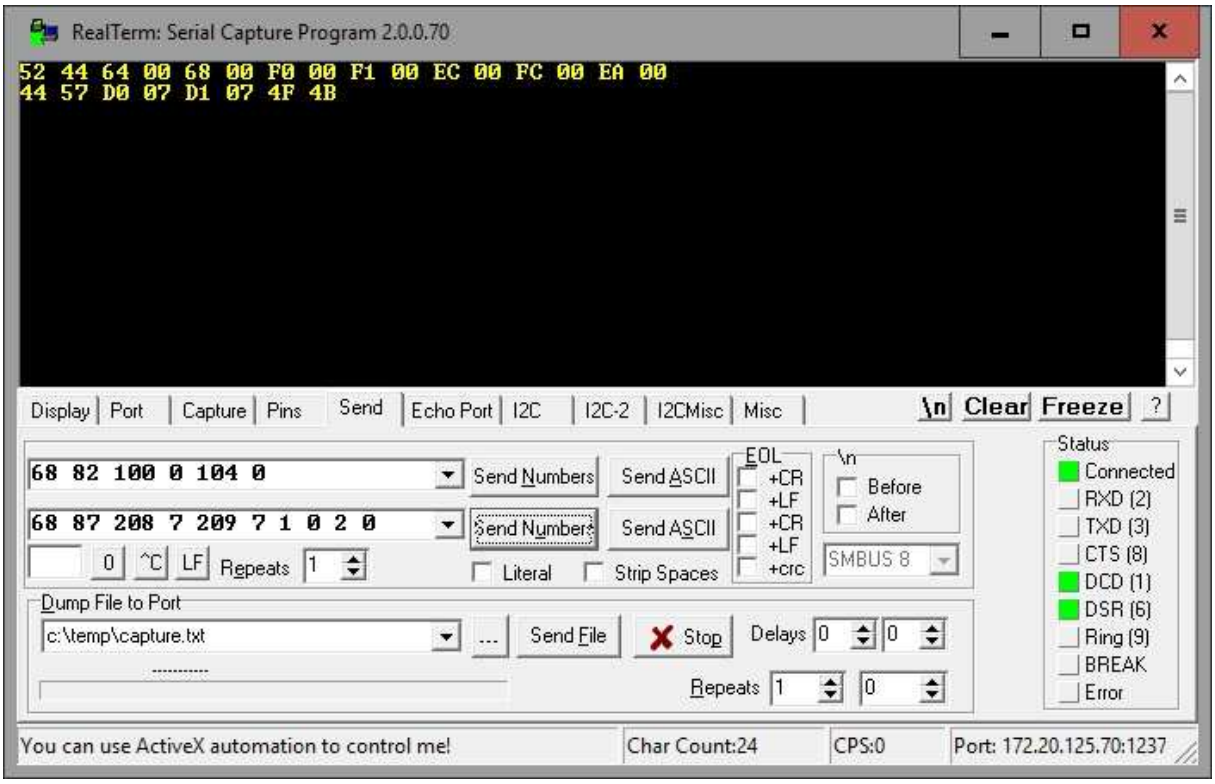
```
44h 52h 64h 00h 68h 00h F0h 00h F1h 00h ECh 00h FCh 00h EAh 00h

D100 hodnota 00F0h což je 240, tedy teplota 24,0°C
D101 hodnota 00F1h což je 241, tedy teplota 24,1°C
D102 hodnota 00ECh což je 236, tedy teplota 23,6°C
D103 hodnota 00FCh což je 252, tedy teplota 25,2°C
D104 hodnota 00EAh což je 234, tedy teplota 23,4°C
```

Podobně lze data do nějaké oblasti PLC zapisovat (např. časové programy, ekvitermy, aktuální čas a další…). Např. příkaz zapiš do D2000 a D2001 hodnoty 1 a 2 vypadá takto:

```
44h 57h D0h 07h D1h 07h 01h 00h 02h 00h
 D   W    2000   2001      1       2
```

Funkčnost jsem ověřil freeware programem RealTerm. V praxi předpokládám takovouto socketovou komunikaci vytvořenou v C#, která poběží na nějakém serveru a bude data vyměňovat s nějakou databází. Tam již bude současně možno přistupovat prakticky neomezený počet uživatelů.



Obr. 19 - ukázka socket comm. přes RealTerm

5.2. Konfigurace RS-485

Communication Protocol Type	**Set communication protocol type.**
Communication Protocol Type	Non-procedural
Advanced Settings	**Set detailed setting.**
Data Length	8bit
Parity Bit	None
Stop Bit	1bit
Baud Rate	9,600bps
Header	Not Added
Header Setting Value	2A000000
Terminator	Not Added
Terminator Setting Value	0D000000
Control Mode (RS - 232C)	No Control Line
Control Mode (RS - 485)	Half Duplex Bi-directional Communication
Sum Check Code	Not Added
Control Procedure	Format 1 (CR, LF Not Added)

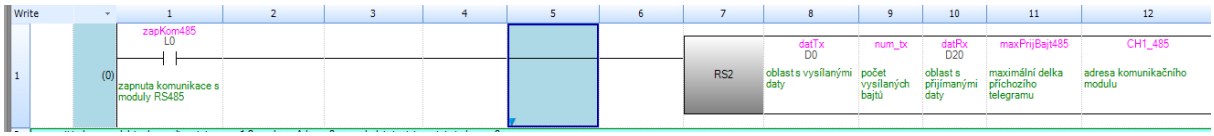
Obr. 20 - konfig. sériového portu PLC

Je v rozbalovacím navigátoru pod Parameter -> FX5UCPU -> Module Parameter -> 485 Serial Port. Zde je možné používat port několika předdefinovanými protokoly pro připojení HMI, měničů a dalších. Pro naše účely ale potřebujeme otevřenou NON-procedural komunikaci, kde budeme kompletně řídit vysílaná a přijímaná data z programu. Nastavení délky, parity atd. je otázka volby, ale musí být stejná v modulech jako v PLC. Paritu není nutné v tomto případě používat. Nepravděpodobná chyba v přenášeném bajtu nezpůsobí asi zásadní problém. Hlavičku a terminační bajt rovněž nepřenáším automaticky, ale doplňuji do vysílaného řetězce ručně. Důležité je nezapomenout pro naše dvoudrátové zapojení propojit

SDA s RDA a SDB s RDB. Sum Check Code bohužel funguje jinak, než potřebujeme pro protokol Spinel, takže je nutné počítat rovněž v programu. CR LF se také nepoužívají. Ještě na záložce Fixed setting je defaultně 16bit Mode. Říká, že z 16 bitového registru se odesílá postupně dolní a horní bajt. Stejně tak i přijímá. Data zabírají menší prostor, ale práce s nimi je obtížnější, musí se vymaskovávat. Veškeré nastavení je přístupné i z programu v systémových parametrech, ale takováto konfigurace je pohodlnější a není ji třeba měnit.

5.3. Komunikace RS-485

Je mírně odlišná od ethernetové komunikace, používá pouze 1 instrukci RS2. Ta má parametry, viz komentáře v obrázku. Předposlední K50 je maximální délka příchozího telegramu a K1 adresa komunikačního modulu (první build-in).



Obr. 21 - instrukce RS2 v programu PLC

Komunikace se ovládá pomocí dvou systémových markerů. Nasetováním SM8561 se vyšle nastavený počet bajtů v *num_tx* z pole *datTx* (D0,D1..). Po odeslání dat si SM8561 PLC resetuje. SM8562 se nastaví do 1, pokud nějaká data naopak přijdou. Po přečtení dat se musí v programu resetovat ručně a PLC je připraveno na příjem dalších. Pokud nějaká data přijdou před shozením SM8562, budou ignorována. Celý funkční a okomentovaný příklad je v příloze - úloha *P01_485_spinel_comm*.

5.4. Protokol Spinel

Protokol byl sestaven firmou Papouch s.r.o. pro komunikaci s různými moduly, měřidly a převodníky, které vyrábějí. Byl navržen tak, aby bylo možné jej dále rozšiřovat nebo modifikovat, a bez kolizí propojovat zařízení s různými modifikacemi protokolu. Je dispozici Spinel terminál, usnadňující ladění komunikace pod OS Windows. Protokol je založen na principu dotaz – odpověď. Já používám pouze část pro čtení měřených dat a zápis, kterou zde vysvětlím. Další speciální funkce, jako změna adresy modulu, změna komunikační rychlosti a další lze doprogramovat, ale zde se s nimi nezabývám. Detailní manuál výrobce je v příloze.
jednoduchý dotaz do modulu:

```
PRE FRM NUM NUM ADR SIG INS DAT SUM CRC
```

Odpověď:
```
PRE FRM NUM NUM ADR SIG ACK DAT SUM CRC
```

Vysvětlivky zkratek protokolu Spinel:
`PRE` Prefix, 2Ah (znak “*“)

FRM formát - 61h pro binární komunikaci, 65h ASCII pro terminál
NUM Počet bajtů instrukce od následujícího bajtu do konce rámce
ADR Adresa modulu, kterému je posílán dotaz, nebo který odpovídá
SIG Podpis zprávy - libovolné číslo od 00h do FFh. Stejné číslo, které se vrátí v odpovědi
INS kód instrukce
ACK Potvrzení dotazu (Acknowledge), zda a jak byl proveden
DAT data k instrukci (velikost záleží na instrukci, nemusí být vůbec)
SUM kontrolní součet (FFh mínus suma všech předchozích
CRC zakončovaní znak 0Dh
VAL vlastní naměřená data (např. naměřená aktuální teplota)

Příklad - dotaz na teplotu do modulu s ASCII adresou „A“:

```
PRE FRM NUM NUM ADR SIG INS DAT SUM CRC
2Ah 61h 00h 05h 41h 02h 51h     DBh 0Dh
```

Odpověď:

```
PRE FRM NUM NUM ADR SIG ACK VAL VAL SUM CRC
2Ah 61h 00h 07h 41h 02h 00h 03h 28h FFh 0Dh
```

Teplota se v PLC vypočítá z 0328h. Musí se trochu nelogicky vydělit hodnotou 32 a vyjde výsledná teplota 25,3°C.
Přestože PLC FX5 umí pracovat s plovoucí desetinnou čárkou, je výhodnější pracovat s teplotami ve formátu INT16 v desetinách stupňů

Pokud je v zařízení více teplot, případně čidlo vlhkosti, vypadá dotaz na vypsání všeho trochu jinak. Tvar mi nepřipadá logický a šťastně zvolený, ale jelikož takto funguje např. u čidel THT2, ponechal jsem tuto podobu i pro své moduly. V dotazu přibyde pouze byte DAT, ale čidla používající 1. formát neumí druhý a obráceně.

Dotaz 2:

```
PRE FRM NUM NUM ADR SIG INS DAT SUM CRC
2Ah 61h 00h 06h 42h 02h 51h 00h D9h 0Dh
```

Odpověď 2 – více měřených hodnot:

```
PRE FRM NUM NUM ADR SIG ACK IDE STA VAL VAL
2Ah 61h 00h 11h 42h 02h 00h 01h 80h 00h DFh

IDE STA VAL VAL IDE STA VAL VAL SUM CRC
02h 80h 01h 0Eh 03h 80h 00h 18h 93h 0Dh
```

Hodnoty ve VAL jsou zde už rovnou v desetinách stupňů, takže stačí převést do dekadického tvaru 00DFh je 22,3°C, 010Eh je 27,0°C a 0018h je 2,4°C.

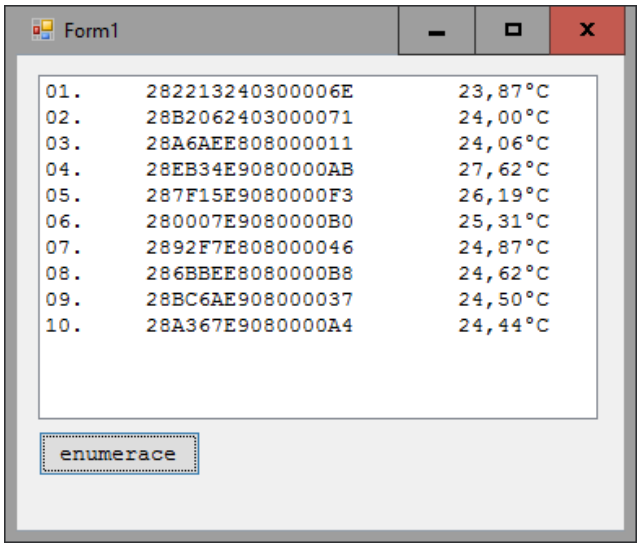
6. Vizualizace v PC

Nyní zbývá už jen uživatelská vizualizace na nezávislém PC, které bude připojené v LAN. Navrhl jsem pouze funkční princip aplikace, která pomocí socketové komunikace čte a zapisuje data do PLC. Krafickou stránku jsem dořešit zcela nestihl. Program v příloze pouze cyklicky čte z PLC registry D0 – D100, kde jsou po pěti registrech uloženy data jednotlivých čidel (adresa 4 word a teplota 1 word). Využil nabité znalosti z kurzů programování C# a vytvořil si k tomuto účelu 2 funkce:

```
public int ReadDregFromTo(Int16 from, Int16 to)
public bool WriteDregFromTo(Int16 from, Int16 to)
```

které pracují s direktivou „`using System.Net.Sockets;`“ Obě používají stejnou globální proměnnou `Int16[] Dreg = new Int16[2048];` přes kterou přenáším data do/z PLC. Komunikaci navazuji z PC. Po zavolání funkce otevřou port na PLC, odešlou data, zpracují odpověď a zavřou komunikaci. Pracují s proměnnými typu `Int16[]`,ale pro přenos provádí konverzi na `byte[]`. Takto vypadá funkce pro čtení PLC datové oblasti Dx – Dy.

```
public int ReadDregFromTo(Int16 from, Int16 to)
{
    if (!PLC_comm_zap) return 0;
    byte[] ByteFrom = BitConverter.GetBytes(from);  //zacatek D reg
    byte[] ByteTo = BitConverter.GetBytes(to);      //konec
    byte[] SdataBytesArr = { Convert.ToByte('D'), Convert.ToByte('R'), ByteFrom[0],
                             ByteFrom[1], ByteTo[0], ByteTo[1] };
    try
    {
        TcpClient client = new TcpClient("172.20.125.70", 1237);
        NetworkStream stream = client.GetStream();
        stream.Write(SdataBytesArr, 0, 6);
        int num = 0;
        int maxCas = 5;
        do
        {
            maxCas--;
            num = stream.Read(RdataBytesArr, 0, 4096);
        }
        while (num == 0 && maxCas > 0); //cekej, dokud nejaka data neprijdou
        stream.Close();
        client.Close();
        for (int i = 6; i < num; i += 2)
        {
            Dreg[(i - 6) / 2] = BitConverter.ToInt16(RdataBytesArr, i);
        }
        return num;
    }
    catch (Exception ee)
    {
        MessageBox.Show(ee.Message);
        return 0;
    }
}
```



Obr. 22 – ukázka aplikace

7. Závěr

Navržený modul vesměs splnil všechny vytyčené cíle a předpoklady. Především jsem měl možnost si prakticky vyzkoušet a přehodnotit některé teoretické předpoklady.

Nejvíce jsem byl zaskočen složitostí principu enumerace, což se začalo ukazovat až s připojováním více teplotních sensorů. Tam rovněž cítím další programové možnosti a rezervy. Adresy by šlo jistě ukládat úsporněji, nebo využít programovou paměť, aby bylo možné na sběrnici připojit více zařízení. K těmto testům jsem se už bohužel nedostal.

Modul není rovněž dotažen do podoby, aby mohl být puštěn do sériové výroby. Šlo mi jen o odzkoušení principů. Pro dlouhodobý a spolehlivý provoz by bylo třeba vyřešit mechanické provedení, ochrany před náhodným přepětím atd., možná foliovou klávesnici atd. Rovněž z PCB bych odstranil nepoužívané prvky a zřejmě celý navrhl znovu pro úspornější SMT technologii.

Seznam použitého software

- **GX works 3** - od Mitsubishi Electric. Vývojové prostředí pro nejnovější řadu PLC Mitsubishi. Jediný placený, ale po dohodě s dodavatelem přikládám na doprovodném nosiči, distributor souhlasí s nekomerčním využitím v rámci TUL.
- **MPLAB X IDE** – Microchip Technology, Inc. - Freeware
- **MPLAB XC8** – kompilátor C - Freeware
- **RealTerm** - terminál pro ladění komunikace. BSD licence – volně šiřitelný

Seznam použité literatury

1. 40001412G.pdf - Datasheet Microchip PIC18(L)F2X/4XK22, ISBN: 978-1-5224-0907-6 [Online] 2. Březen 2018. http://www.microchip.com/wwwproducts/en/PIC18F26K22
2. DS18B20.pdf, 19-7487, Rev 4, [Online] 5. Březen 2018. https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf
3. jy997d55801B.pdf, Programming Manual MELSEC iQ-F [Online] 5. Březen 2018. https://cz3a.mitsubishielectric.com/fa/cs/service/download
4. tx20xxx-spinel.pdf [Online] 5. Březen 2018. https://www.papouch.com/cz/website/mainmenu/spinel/

Obsah CD

bakalarska_prace_Martin_Šetina.docx
bakalarska_prace_Martin_Šetina.pdf
modul.brd – PCB Eagle
modul.sch – schéma Eagle
/programy – zde jsou uloženy zdrojové programy pro PLC, PIC a PC
/datasheets – manuály PIC, PLC a DS18B20