ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA ELEKTROMECHANIKY A VÝKONOVÉ ELEKTRONIKY

BAKALÁŘSKÁ PRÁCE

Implementace komunikačního protokolu USS pro frekvenční měnič Siemens MM430

Filip Štochl **2015**

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2014/2015**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Filip ŠTOCHL**
Osobní číslo:	**E12B0243P**
Studijní program:	**B2644 Aplikovaná elektrotechnika**
Studijní obor:	**Aplikovaná elektrotechnika**
Název tématu:	**Implementace komunikačního protokolu USS pro frekvenční měnič Siemens MM430**
Zadávající katedra:	**Katedra elektromechaniky a výkonové elektroniky**

Zásady pro vypracování:

1. Analyzujte USS protokol společnosti Siemens pro komunikaci s frekvenčními měniči.
2. Prozkoumejte možnosti ovládacího SW STEP7 pro procesní řízení.
3. Sestavte programové vybavení pro práci s protokolem USS.
4. Otestuje navržené softwarové řešení.
5. Diskutujte dosažené výsledky a parametry.

Rozsah grafických prací: **podle doporučení vedoucího**
Rozsah pracovní zprávy: **20 - 30 stran**
Forma zpracování bakalářské práce: **tištěná/elektronická**
Seznam odborné literatury:

1. **Firemní dokumentace Siemens k měniči MM430.**
2. **Firemní dokumentace Siemens protokolu USS.**
3. **Pinker J., Poupa M., Číslicové systémy a jazyk VHDL, ISBN:80-7300-198-5.**

Vedoucí bakalářské práce: **Ing. Aleš Krutina**
Regionální inovační centrum elektrotechniky

Datum zadání bakalářské práce: **15. října 2014**
Termín odevzdání bakalářské práce: **8. června 2015**

Doc. Ing. Jiří Hammerbauer, Ph.D.
děkan

L.S.

Prof. Ing. Václav Kůs, CSc.
vedoucí katedry

V Plzni dne 15. října 2014

Abstrakt

Předkládaná bakalářská práce se zabývá ovládáním a komunikací s frekvenčními měniči řady MicroMaster přes univerzální sériový protokol USS společnosti Siemens. Detailněji je zde popsán způsob sestavení požadované zprávy pro frekvenční měnič řady MicroMaster v protokolu USS, využití knihovny instrukcí USS protokolu používaných softwarem STEP 7-Micro/WIN společnosti Siemens. Součástí bakalářské práce je vytvoření a otestování softwaru pro zjištění a změnu hodnot parametru frekvenčního měniče pomocí USS protokolu.

Klíčová slova

USS protokol, MicroMaster, Siemens, řídící PC, frekvenční měnič, knihovna USS protokolu, instrukce, STEP 7-Micro/WIN, software, řízení, hodnota parametru měniče, Microsoft Visual Studio, C#

Abstract

This bachelor thesis is concerned with operating and communication with the frequency converters series MicroMaster via USS protocol of companies Siemens. Detailed is here described method of composition telegram for frequency converter series MicroMaster in the USS protocol, usage libraries instruction of the USS protocol used by software STEP 7-Micro/WIN of companies Siemens. Part of bachelor thesis is creating and testing software for inquest and change values of parametr in frequency inverter by the USS protocol.

Key words

USS protocol, MicroMaster, Siemens, control PC, frequency converter, library of USS protocol, instruction, STEP 7-Micro/WIN, software, operating, value of parameter converter, Microsoft Visual Studio, C#

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci, zpracovanou na závěr studia na Fakultě elektrotechnické Západočeské univerzity v Plzni.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této bakalářské práce.

Dále prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

.....

podpis

V Hůrkách dne 15.7.2015 Filip Štochl

Poděkování

Tímto bych rád poděkoval vedoucímu bakalářské práce Ing. Aleši Krutinovi za cenné profesionální rady, připomínky a metodické vedení práce.

Obsah

Seznam symbolů a zkratek

ADR.....Address byte, bajt adresy

ASCII.....American Standard Code for Information Interchange, Am. standardní kód

BCC.....Block Check Character, blok kontrolního znaku

BCL.....Basic Class Library, základní knihovna tříd

BOP.....Basic Operator Panel, základní ovládací panel

C#.....Programovací jazyk

CD-ROM.....Compact Disc - Read Only Memory, kompaktní disk – pouze pro čtení

CLR.....Common Language Runtime, společné běhové prostředí

EEPROM.....Electrically Erasable PROM, elektricky vymazatelná paměť PROM

HEX.....Hexadecimální soustava

HW.....HardWare, technické vybavení počítače

LGE.....Telegram length, délka telegramu

MM.....MicroMaster

PC.....Personal Computer, osobní počítač

PKW.....Parameter ID value, hodnota parametru

PLC.....Programmable Logic Controller, programovatelný automat

PPI.....Point-to-point Interface

PZD.....Process data, data procesu

RAM.....Random Access Memory, paměť s náhodným přístupem

STX.....Start of text, začátek textu

USB.....Universal Serial Bus, univerzální sériová sběrnice

USS.....Universal Serial Interface

Úvod

Předkládaná bakalářská práce, se zabývá problematikou ovládání a komunikace s měniči řady MicroMaster od společnosti Siemens přes univerzální sériový protokol USS. Práce je rozdělena do několika hlavních kapitol.

První kapitola se detailně zabývá analýzou USS protokolu od společnosti Siemens pro využití v komunikaci s frekvenčními měniči řady MicroMaster. Jsou zde popsány jednotlivé části telegramu USS protokolu a jeho tvoření. Na závěr první kapitoly je uveden příklad sestavení konkrétní požadované zprávy na dotaz hodnoty parametru frekvenčního měniče a převedení do hexadecimálního kódu pro odeslání zprávy.

Druhá kapitola se zabývá využitím USS protokolu pro komunikaci s frekvenčními měniči řady MicroMaster s využitím softwaru STEP 7-Micro/WIN. Je zde popsán způsob ovládání frekvenčního měniče řady MicroMaster přes USS protokol za nutnosti využití programovatelného automatu, konkrétně automatu S7-200. Detailněji jsou zde popsány instrukce knihovny USS protokolu využívané softwarem STEP 7-Micro/WIN.

Třetí kapitola se zabývá vytvořeným programovým vybavením pro práci s USS protokolem. Popisuje se zde vývojové prostředí Microsoft Visual Studio a založení nového projektu v tomto nástroji. V této kapitole jsou také obsaženy základní informace o programovacím jazyce C#. Posledním bodem této kapitoly je popis pro nastavení hodnot sériové komunikace při používání vytvořeného softwaru pro čtení a zapisování hodnot parametru frekvenčního měniče řady MicroMaster.

Čtvrtá kapitola se zabývá popisem praktických realizací a zkoušení komunikace mezi PC a frekvenčním měničem MM440. Nejprve je zde popsán způsob zapojení PC a měniče pro komunikaci pomocí USS protokolu a oživení frekvenčního měniče MM440. Poté je zde popsána realizace komunikace s frekvenčním měničem MM440 přes sériový terminál, konkrétně program Hercules od společnosti HW-group. Zahrnuto je zde dále testování navrženého softwaru pro komunikaci s frekvenčním měničem MM440 pomocí USS protokolu, pro načtení a změnu hodnoty parametru frekvenčního měniče.

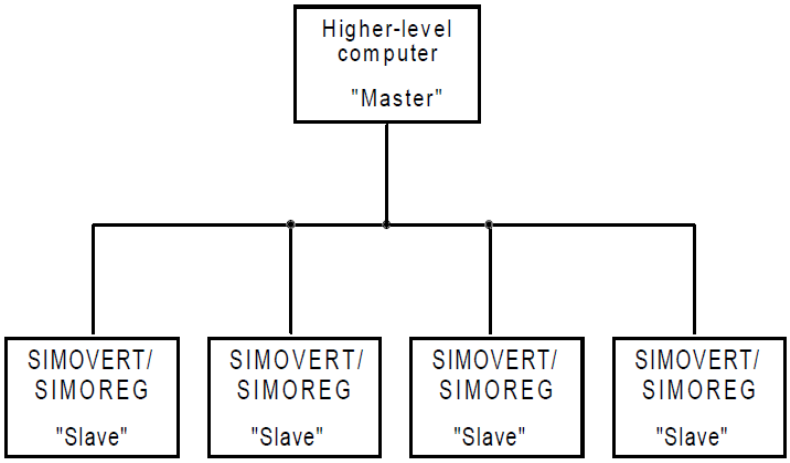
1 Analýza USS protokolu společnosti Siemens

USS protokol (Universal Serial Interface Protocol) je jednoduchý protokol pro sériové přenášení dat, pro více zařízení, přes sériovou sběrnici. Nejčastěji se využívá ke komunikaci mezi hlavním ovládacím prvkem, například řídící PC, PLC... a podřízenými ovládanými prvky, například měničem kmitočtu. V originále se tento řídící princip označuje jako Master – Slave.

Základními vlastnostmi USS protokolu jsou:

- ***Podpora pro vícebodové propojení, RS 485 HW***
- ***Master – slave přístupová metoda***
- ***Maximálně 32 uzlů, respektive maximálně 31 ovládaných prvků (měničů)***
- ***Jednoduché a spolehlivé rámce zprávy***
- ***Jednoduché na realizaci***
- ***Operace buď s proměnou nebo pevnou délkou zprávy***

Všechny podřízené ovládané prvky (měniče) jsou propojeny s řídícím prvkem (PC) pomocí sběrnice. Jednotlivé měniče jsou rozpoznatelné pro řídící prvek pomocí adresy, která je součástí vysílané zprávy. Komunikace mezi zařízeními je realizována v half-duplex (poloviční duplex) modu, což zaznamená, že obě strany, jak master tak slave, mohou vysílat i přijímat, ale nikoli současně. Tato komunikace může pracovat pouze s jedním nadřazeným prvkem PC (single-master systém).



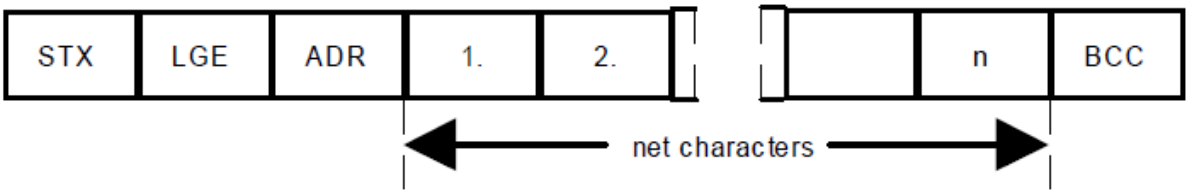
Obr. 1.1 Blokové schéma komunikace po sériové sběrnici

Při této sériové komunikaci pomocí USS protokolu se používá tzv. cyklický přenos zprávy (telegramu). Řídící prvek (master) řídí v jakém pořadí bude probíhat komunikace s jednotlivými měniči (slave) podle adresace, jeden po druhém, v identických časových intervalech.

1.1 Struktura zprávy USS protokolu

Přenášená zpráva pomocí USS protokolu se dělí na části, které jsou časově za sebou:

- ***STX (start of text) – ASCII charakter, obvykle se rovná 02 hex***
- ***LGE (telegram length) – jedná se o informaci o délce přenášené zprávy obvykle o velikosti jednoho bajtu a v binárním čísle***
- ***ADR (address byte) – informace o velikosti 1 bajtu obsahující adresu měniče a informace o typu přenášené zprávy***
- ***Charakter sítě (net characters) – obsah je závislý na požadovaných činnostech měniče a tím se mění i velikost tohoto bloku***
- ***BCC (block check character) – bajtové pole, které potvrzuje konec přenášené zprávy***



Obr. 1.2 Struktura přenášené zprávy USS protokolu

1.1.1 LGE (telegram length)

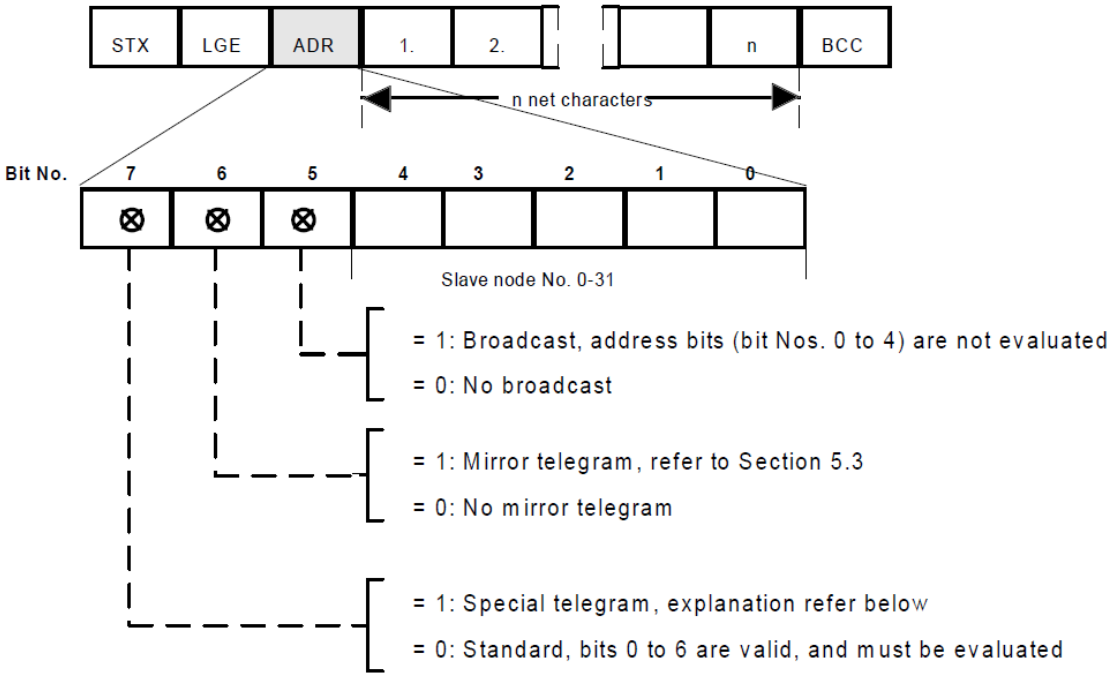
LGE je samostatné bajtové pole, ve kterém se definuje délka přenášené zprávy, respektive definuje se počet bajtů, které budou následovat STX a LGE. Podle specifikace USS protokolu je délka přenášené zprávy variabilní a musí se specifikovat konkrétně v již zmiňovaném druhém bajtu LGE. Maximální celková délka přenášené zprávy může být maximálně 256 bajtů. V délce přenášené zprávy jsou obsaženy pouze části ADR (adresa měniče), požadované činnosti a BCC.

$$LGE = n+2 \{1 \leq LGE \geq 254\} \quad (1.3)$$

Délku přenášené zprávy můžeme mít buď proměnnou nebo pevnou.

1.1.2 ADR (address byte)

ADR je bajt obsahující adresu měniče (uzlu na sběrnici požadovaného měniče). Jednotlivé bity v adresním bajtu jsou adresované tímto způsobem:



Obr. 1.4 Struktura adresového bajtu ADR

Bity 0-4 obsahuje adresu pro komunikaci s měniči č. 0-31.

Bit 5 je vysílaný bit. Pokud bude bit 5 obsahovat logickou 1, zpráva bude vysílána, bude určena všem měničům na sériové sběrnici a bity 0-4 nebudou odvysílány.

Bit 6 indikuje zrcadlenou zprávu, pokud je tento bit nastaven na hodnotu logické 1. Adresa měniče (uzlu na sběrnici) je vyhodnoceno a adresovaný měnič vrátí nezměněnou zprávu zpět do řídícího PC. Nepoužitý 7. a 5. bit by měl být nastaven na hodnotu 0. Zrcadlená zpráva je prospěšná při komunikaci krok za krokem, při inicializaci zařízení nebo hledání závady.

Bit 7 vytváří speciální přenášenou zprávu, pokud je tento bit nastaven na hodnotu logické 1, která se používá pro speciální aplikace, které požadují síťovou strukturu dat odlišnou od standardní struktury datové oblasti USS protokolu. Při používání normální přenášené zprávy a speciální přenášené zprávy na jedné sběrnici, musí být speciální zpráva dobře rozpoznatelná,

aby je měniče, pracující na základní strukturu, odmítly. Struktura speciální zprávy je však úplně stejná jako normální zpráva a speciální zprávu určujeme tedy pouze v bajtu adresy ADR. Pokud bychom chtěli poslat speciální zprávu všem měničům k tomu určeným, využijeme nastavení bitu 7 a 5 na hodnotu logické 1. Zrcadlená zpráva v případě nastavení speciální zprávy (bit7 = log 1) nelze použít.

1.1.3 BCC (block check character)

BCC je bajtové pole, které potvrzuje konec zprávy. Velikost BCC pole je dána součtem XOR respektive EXOR (exclusive OR logic operation) všech předchozích bajtů obsažených ve zprávě.

Generování BCC kódu:

- ***hodnota BCC je před přijmutím prvního znaku telegramu (STX) rovna 0 → BCC = 0000 0000***
- ***po přijetí prvního znaku dochází k součtu BCC = 0 a STX = 02hex***
  → BCC_{old} = 0000 0000
  EXOR
  První znak (STX) = 0000 0010 (02 hex)
  BCC_{new} = 0000 0010
- ***po každém přijetí speciálního znaku dochází k součtu BCC_{old} a druhým znakem***
  → BCC_{old} = 0000 0010
  EXOR
  Druhý znak = 1101 0110
  BCC_{new} = 1101 0100

Výsledkem je BCC po posledním přijatém síťovém znaku.

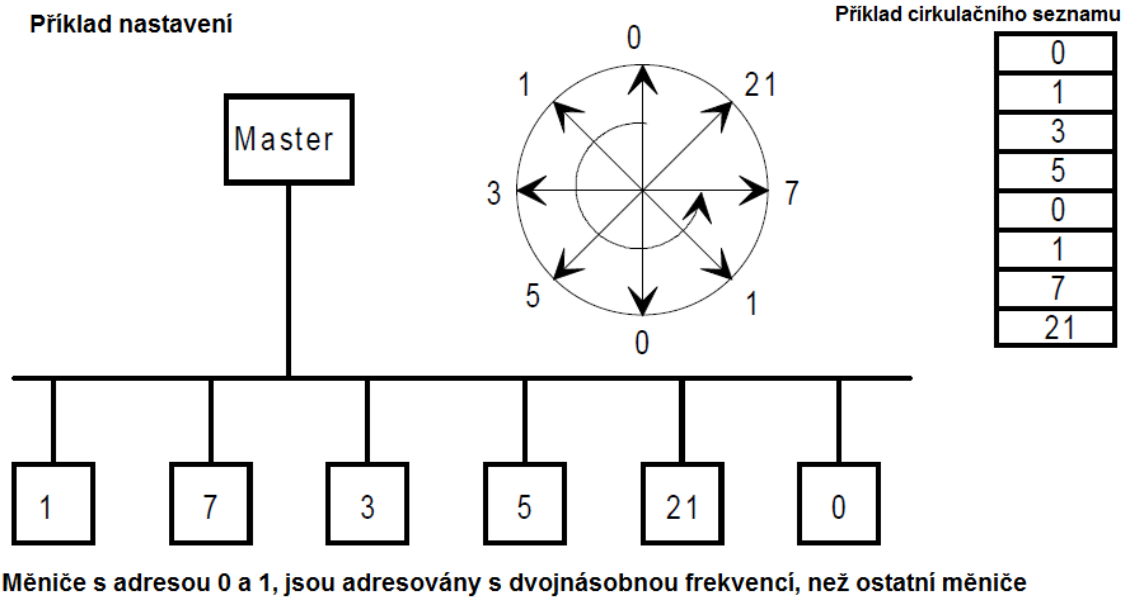
1.2 Postup přenosu dat

Hlavní řídící prvek (PC) realizuje přenos dat pomocí cyklického přenosu zprávy. Řídící počítač postupně, za sebou a podle adresy, posílá měničům úkolové zprávy. Měniče na tyto zprávy reagují zpětnou odezvou. V souladu s master-slave procesem musí měnič odeslat

zpětnou zprávu, jako odezvu o přijetí zpět do řídícího PC, než odešle řídící PC další zprávu do měniče, který následuje podle adresace.

1.2.1 Řízení přenosu dat

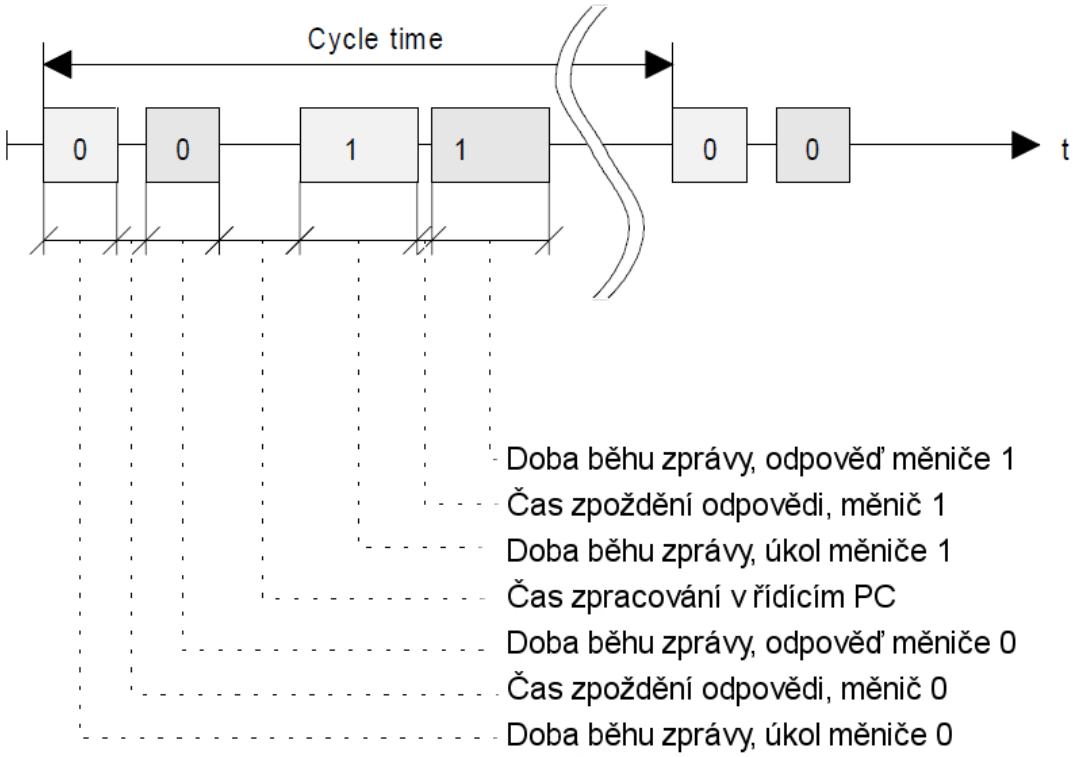
Pořadí měničů může být určeno například vložením adresového čísla (ADR) do takzvaného cirkulačního seznamu (seznam výzev) řídícího PC. Pokud chceme, aby nějaký měnič byl adresovaný v rychlejším cyklu než ostatní měniče, pak můžeme zařadit adresové číslo (ADR) častěji do cirkulačního seznamu. Komunikace point-to-point (z bodu do bodu) může být realizována pomocí cirkulačního seznamu pouze v případě, že v cirkulačním seznamu je obsažen pouze jeden měnič nebo uzel.



Obr. 1.5 Cirkulační seznam

1.2.2 Doba cyklu

Doba cyklu je definovaná časem potřebným pro přenos dat s jednotlivými měniči. Čas cyklu nemůže být přesně definovaný z důvodu rozdílných zpoždění odezvy měniče a rozdílných časů zpracování zprávy. Řídící PC může realizovat cyklus s pevným časem, určením maximálního času cyklu pro jedno nastavení a pak podle toho definovat absolutní čas cyklu. Po přenesení dat k poslednímu měniči musí řídící PC počkat do vypršení definovaného času cyklu, než bude moci odeslat zprávu pro měniče v cyklu znovu.



Obr. 1.6 Čas cyklu

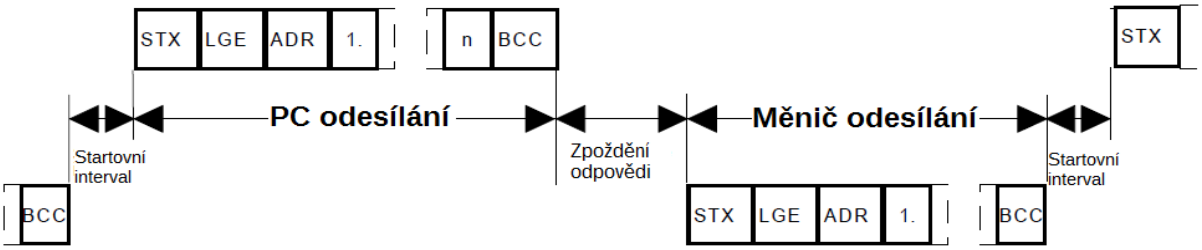
1.2.3 Startovní interval

Začátek přenášené zprávy (STX = 02hex) není pro měniče dostačující pro zřetelné identifikování začátku přenášené zprávy, protože bitová kombinace 02hex se může také vyskytovat v síťových znacích. Před STX se proto zařadí startovní zpoždění nejméně dobu dvou znakových dob chodu, toto zpoždění je specifikované pro řídící PC. Startovní zpoždění, nebo také startovní interval je součástí úkolového telegramu. Pouze STX s předchozím startovním intervalem určí (identifikuje) platný začátek přenášené zprávy.

Tab. 1.1 Minimální startovní interval pro různou přenosovou rychlost

Přenosová rychlost [bit/s]	Startovní interval [ms]
9600	2,3
19200	1,15
38400	0,58
187500	0,12

Přenášená data jsou obvykle realizována v principiálním schématu zobrazeném na *Obr. 1.7*. Jedná se o přenos v činnosti half-duplex (poloviční duplex).



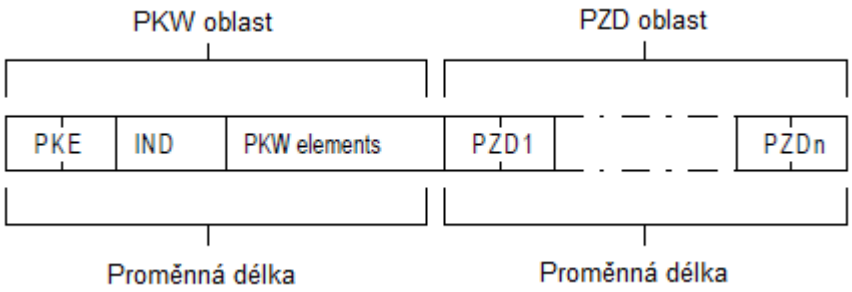
Obr. 1.7 Principiální schéma odesílání zprávy

1.3 Struktura bloků síťových dat

Blok síťových dat je rozdělený do dvou částí (oblastí), PKW oblast a PZD oblast.

PKW oblast odkazuje na hodnotu parametru ID (PKW) rozhraní. PKW rozhraní neobsahuje fyzické rozhraní, ale definuje mechanismus, který zajišťuje přenos parametru mezi dvěma komunikačními partnery. To znamená, že se zde budou číst a zapisovat hodnoty parametru, definovat parametr a přidružený text, zajišťuje se zde změna parametru a použití parametru. Celým tímto blokem se mění význam vysílané zprávy. Všechny úkoly, které jdou přes PKW jsou operátorem kontrolovány, zobrazovány, obsluhovány a vyhodnocovány. PKW oblast používá tzv. úkolů a odpovědí, v této oblasti se využívá k přístupu k parametrům dostupných v měniči přes USS protokol a jsou skryté.

PZD oblast obsahuje signály požadované pro automatizaci. Skládá se z řídícího slova (control word) a zadané hodnoty (setpoint) z řídícího PC do měniče. Stavové slovo (status word) a aktuální hodnoty (actual value) z měniče do řídícího PC. Úkolový telegram je přenos celého síťového bloku dat z řídícího PC do měniče a odezvový telegram je přenos celého síťového bloku dat z měniče do řídícího PC.

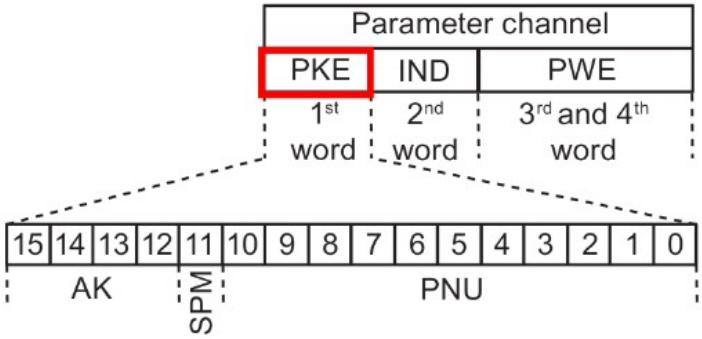


Obr. 1.8 Struktura bloků síťových dat

1.3.1 PKE oblast

ID parametru (PKE) je využívaný pro identifikaci a uvedení úkolu a odpovědi pro zpracování parametru. PKE má vždy délku jednoho slova (16 bitů). V PKE je obsaženo číslo parametru. Dělí se do tří částí:

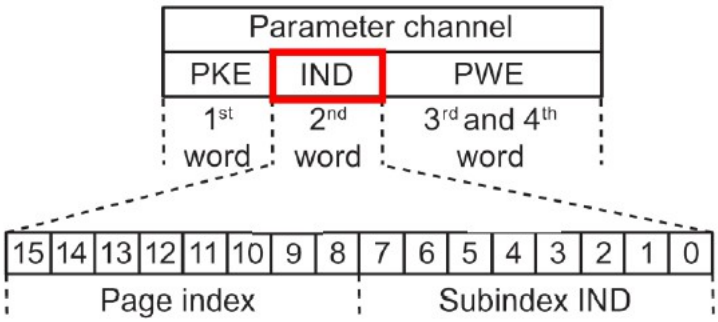
- ***AK – Bity 12 až 15, jedná se o identifikátor odezvy při komunikaci řídícího PC s měničem. Například hodnota 0000 je označeno jako „bez úkolu“, hodnota 0001 je požadavek na navrácení hodnoty parametru specifikovaném v části PNU***
- ***SPM – Bit 11 je rezerva a většinou bývá rovno 0***
- ***PNU – Bity 0 až 10, obsahují číslo parametru měniče. Pro parametry s hodnotou ≥ 2000, je zapotřebí přidat kompenzaci, která je definována pomocí indexu IN.***



Obr. 1.9 Struktura PKE oblasti

1.3.2 IND oblast

IND je oblast o velikosti jednoho slova (wordu). Velké množství parametrů měniče využívá pro své detailnější nastavení v určitých situacích indexy. Ty se využívají ke specifikaci požadovaného parametru vybraném v oblasti PKE a jsou určeny tímto polem. Další využití tohoto pole nastává v případě, že využíváme parametry vyššího čísla než 1999.



Obr. 1.10 Struktura IND oblasti

Jak vidíme na *Obr. 1.10,* oblast IND se dělí na dva jednotlivé bajty. Ten první označovaný jako „Subindex IND“ je využíván jako určení indexu parametru. Například pro nastavení parametru P0700 s indexem 1 (P0700[1]) měniče MicroMaster, bude vypadat tato část IND takto: 0000 0001 (01Hex).

Druhý bajt této oblasti, označovaný jako „Page index“ se využívá při zadávání parametru s číslem ≥ 2000. To jakou hodnotu toto pole bude mít určuje *Tab. 1.2*.

Tab. 1.2 Pravidla pro nastavení hodnoty „Page index“

Hodnota parametru	Bity								Hodnota Hex
[redacted]	15	14	13	12	11	10	9	8	[redacted]
0000 - 1999	0	0	0	0	0	0	0	0	0
2000 - 3999	1	0	0	0	0	0	0	0	80
4000 - 5999	0	0	0	1	0	0	0	0	10
6000 - 7999	1	0	0	1	0	0	0	0	90
8000 - 9999	0	0	1	0	0	0	0	0	20

Například pro nastavení parametrů o hodnotě větší než 2000, bude hodnota vypadat takto: 1000 0000 (80Hex). Celková hodnota oblasti IND podle příkladů, jaké jsme si ukázali, bude mít hodnotu 80 01 Hex.

1.3.3 PWE oblast

PWE oblast obsahuje informaci týkající se zadaného úkolu měniči nebo odpovědi měniče, jako hodnotu parametru, text nebo popis parametru. Tato oblast může být proměnná v délce slova a je to závislé na zadaném úkolu nebo odpovědi. Tato délka, respektive délka PKW oblasti musí být předem definovaná a to na variabilní délku nebo pevnou délku

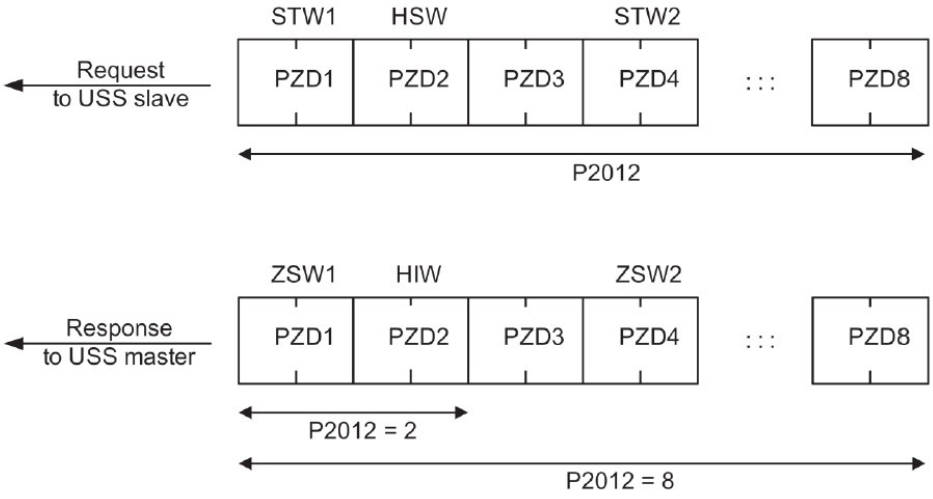
například 4 slov (wordů). V případě 4 slov PKW bude zpráva obsahovat dvě slova PWE (PWE1, PWE2). V případě, že PZD data mají být přenášena v síťovém bloku dat, je počet PKW elementů nastaven na hodnotu 0.

1.3.4 PZD oblast

PZD oblast je oblast dat procesu, které jsou spojitě přenášeny mezi řídícím PC a měničem v této oblasti. Tyto data obsahují signály požadované pro automatizaci. V závislosti na směru obsahuje kanál dat data pro žádost ze strany řídícího PC k jemu podřízenému měniči. V těchto žádostech jsou obsaženy řídící slova (control words) a zadané hodnoty (setpoints) pro měniče a v odpovědích jsou obsaženy stavová slova (status words) a skutečné hodnoty (actual value) pro řídící PC. Počet PZD slov se určuje pomocí parametru měniče P2012 a může se pohybovat v rozsahu 0 až 8.

- ***Žádost ze strany PC: Řídící slovo (STW1) a Zadaná hodnota (HSW)***
- ***Odpověď ze strany měniče: Stavové slovo (ZSW1) a Aktuální hodnota (HIW)***

Pokud je parametr P2012 nastaven na hodnotu 4, bude řídící slovo (STW2) přeneseno jako čtvrté slovo (PZD4). Zdroj všech dalších PZD se určuje v parametru P2019 pro komunikaci přes rozhraní RS485 nebo v parametru P2016 pro komunikaci přes rozhraní RS232.



Obr. 1.11 Struktura přenášení procesních dat PZD

1.3.5 Příklad telegramu pro dotaz na hodnotu parametru

Příklad telegramu pro dotaz na hodnotu parametru P0700 MicroMaster 440 Siemens:

- **STX = 02 hex**
- **LGE = 14 bajtů » 0E hex**
    - **PKW oblasti bude mít velikost 4 slova (8 bajtů)**
    - **PZD oblast bude mít velikost 2 slova (4 bajty)**
    - **ADR má velikost jednoho bajtu**
    - **BCC má velikost jednoho bajtu**
- **ADR = 00 hex**
    - **Adresa měniče je bez speciálních nastavení jako například zrcadlená zpráva**
- **PKE = 12 BC hex**
    - **Parametr P0700 převeden do šestnáctkové soustavy » 2BC hex**
    - **Požadavek na hodnotu parametru je vyjádřen hodnotou 1 hex**
- **IND = 00 00 hex**
    - **Dotaz pro daný parametr není specifikován žádným indexem a hodnota parametru je menší než 2000, proto je hodnota parametru 0**
- **PWE1 = 00 00 hex**
    - **Žádné určení hodnoty, pouze dotaz**
- **PWE2 = 00 00 hex**
    - **Žádné určení hodnoty, pouze dotaz**
- **PZD1 = 00 00 hex**
    - **Žádné určení hodnoty, pouze dotaz**
- **PZD2 = 00 00 hex**
    - **Žádné určení hodnoty, pouze dotaz**
- **BCC = A2 hex**
    - **Jedná se o XOR postupně všech polí s hodnotou jinou než 0, v případě součtu s hodnotou 0, je výsledek stejný jako v předchozím případě**
    - **02 XOR 0E XOR 12 XOR BC**

Výsledný odesílaný telegram vypadá následovně:

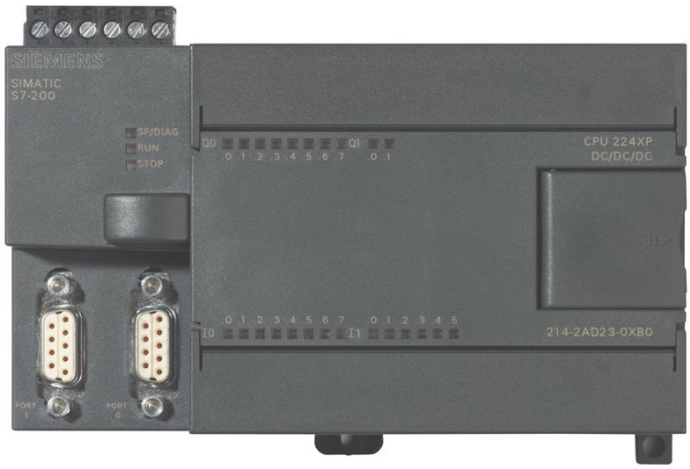
{02}{0E}{00}{12}{BC}{00}{00}{00}{00}{00}{00}{00}{00}{00}{00}{A2} hex

2 Možnosti ovládacího SW STEP 7 pro procesní řízení

Řízení pohonů typu MicroMaster velmi usnadňuje software STEP 7-Micro/WIN, pomocí knihovny instrukcí tohoto softwaru. Tyto knihovny obsahují již nakonfigurované podprogramy přerušení a jiné podprogramy, které jsou speciálně navržené pro použití USS protokolu společnosti Siemens při komunikaci s pohony. Pomocí instrukcí USS protokolu můžeme řídit pohon, číst a zapisovat jeho aktuální hodnoty a parametry.

2.1 Využití SW STEP 7 pro řízení měniče řady MicroMaster

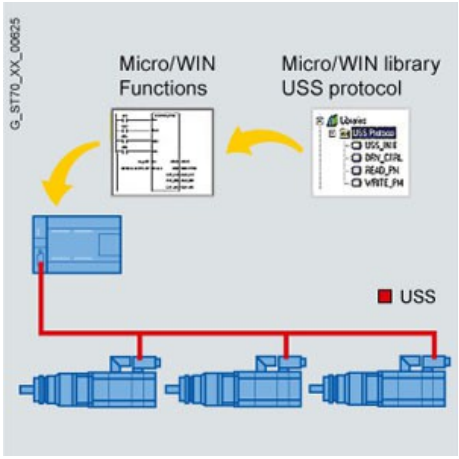
Při použití softwaru STEP 7-Micro/WIN pro řízení měniče MicroMaster, nelze řídit měnič přímo z PC pomocí tohoto softwaru. Je zapotřebí dalšího komponentu, konkrétně programovatelného automatu od Společnosti Siemens. Jedním z těchto automatů je například automat S7-200. Tento automat je určen pro řízení malých aplikací, ale je velmi silný v komunikačních funkcích. Tento model disponuje i možností velkého rozšíření o další moduly podle potřeby, například vstupů a výstupů.



Obr. 2.1 S7-200 s jednotkou CPU 224 XP

Při využití SW STEP 7 je tedy vytvořen program, pomocí níže uvedených instrukcí USS protokolu. Celý program je pak nahrán přes komunikační kabel do automatu S7-200. Jako komunikační kabel mezi PC a automatem může sloužit například rozhraní RS485 nebo RS232. Po nahrání již může být řízen měnič, například řady MicroMaster, pouze za pomoci

automatu a detekce přivedených vstupů a výstupů. Komunikace mezi automatem a měničem je v tomto případě také realizována pomocí RS485, nebo pokud by byl na řízené sběrnici pouze jeden měnič o jedné adrese, lze taktéž využít rozhraní RS232. PC můžeme dále využít jako čtení hodnot a dat v programu STEP 7 v režimu „online“. Na sériové lince mezi automatem a měničem dochází k přenosu naprogramovaných dat pomocí USS protokolu.

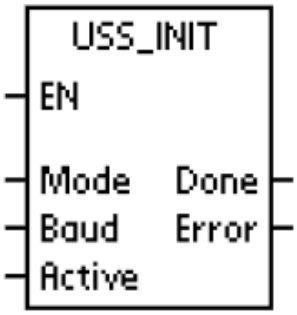


Obr. 2.2 Způsob ovládání měničů pomocí STEP 7 a S7-200

2.2 Instrukce USS protokolu SW STEP 7

2.2.1 Instrukce USS_INIT

Instrukce USS protokolu USS_INIT slouží k povolení, inicializaci a ukončení komunikace s měniči MicroMaster (pohony). Před po užitím jakékoliv další instrukce USS protokolu je nutné, aby byla instrukce USS_INIT dokončena v pořádku a bez chyb. Další podmínkou, pro možnost provedení další instrukce, je ukončení instrukce USS_INIT a nastavení bitu DONE. Pro každou změnu stavu v komunikaci s měničem se mění instrukce USS protokolu USS_INIT pouze jednou. Z tohoto důvodu se na vstup EN připojují pulzy vyhodnocující náběžnou hranu. Instrukce se provádí v každém programovém cyklu, při zapnutém vstupu EN. Nová instrukce USS_INIT se provádí v případě, že chceme změnit parametry inicializace.



Obr. 2.3 Blokové schéma instrukce USS_INIT

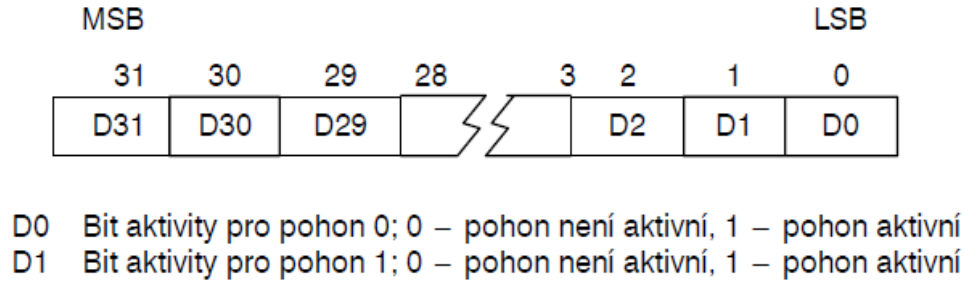
Parametry:

- ***EN – Tzv. Povolovací bit, musí být stále aktivní pro chod programu***
- ***MODE – Volba komunikačního protokolu pro port 0:***
  - ***1 – USS protokol, jeho aktivace***
  - ***0 – PPI protokol, blokování USS protokolu***
- ***Baud – Nastavení přenosové rychlosti, pro měniče MicroMaster se obvykle nastavuje na rychlost 19200 bps***
- ***Active – Indikuje, které měniče jsou aktivní, některé druhy měničů podporují pouze adresy 0 až 30***
- ***Error – indikuje chybu přenosu instrukce USS_INIT***

Tab. 2.1 Operandy platné pro USS_INIT

Vstupy / Výstupy	Typ dat	Operandy
MODE	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, Konstanta, *VD, *AC, *LD
BAUD	DWORD	VW, IW, QW, MW, SW, SMW, LW, T, C, AIW, Konstanta, AC, *VD, *AC, *LD
ACTIVE	DWORD	VD, ID, QD, MD, SD, SMD, LD, AC, Konstanta, *VD, *AC, *LD
DONE	BOOL	I, Q, M, S, SM, T, C, V, L
ERROR	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD

Pokud je měnič (pohon) označen jako aktivní, je automaticky tázán na pozadí tak, aby řídil činnost pohonu, dokumentoval stav měniče a zabránil časovému dopojení od seriové linky.

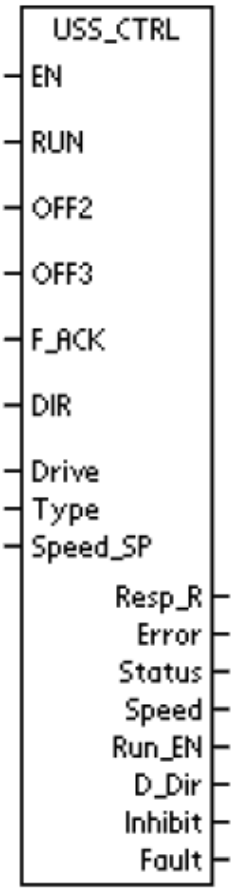


Obr. 2.4 Rozměr parametru aktivního pohonu

Po dokončení instrukce USS_INIT se sepne výstup DONE. V případě, že nebude instrukce dobře odeslána, bude výstupní bajt ERROR obsahovat výsledek, možné chyby přenosu jsou uvedeny v tabulce *Tab. 2.5*.

2.2.2 Instrukce USS_CTRL

K řízení aktivního pohonu MicroMaster slouží instrukce USS_CTRL. Ke každému měniči (pohonu) by měla být přiřazena pouze jedna tato instrukce. Podmínkou je, aby byl daný měnič (pohon) aktivní. Stejně jako u instrukce USS_INIT, musí i instrukce USS_CTRL mít zapnutý bit EN. Tato instrukce musí být povolena v každém případě. Měniče (pohony) většinou udávají svojí rychlost se znaménkem + nebo – podle směru otáčení. Některé pohony, ale tuto možnost nepovolují a daný pohon hlásí aktuální rychlost pouze v kladných hodnotách, ale dochází k invertování bitu D_Dir (směr).



Obr. 2.5 Blokové schéma instrukce USS_CTRL

Parametry:

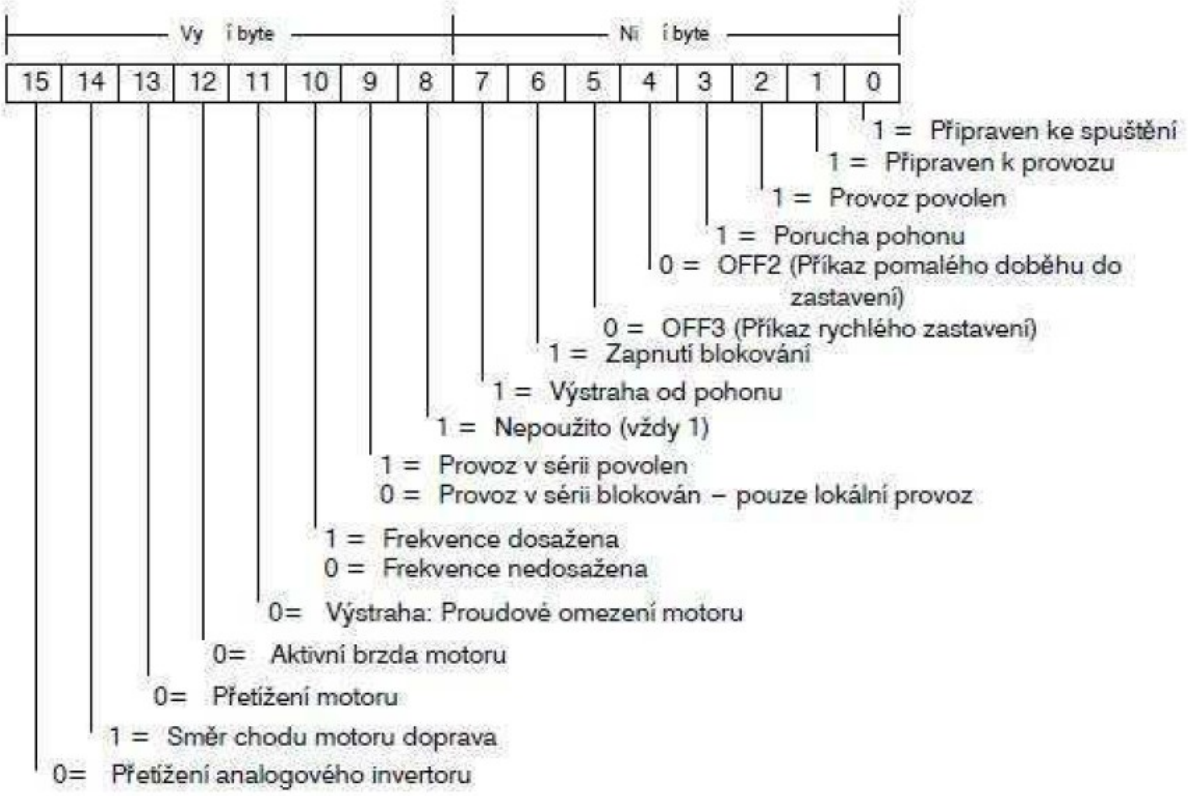
- ***EN – Tzv. Povolovací bit, musí být stále aktivní, aby mohla být instrukce provedena***
- ***RUN:***
  - ***Indikuje, zda je pohon v chodu (log 1) nebo v klidu (log 0)***
  - ***Při indikaci pohonu v chodu, dostane měnič MicroMaster povel pro rozběhnutí se zadanou rychlostí a směrem***

- ***Aby mohl být motor uveden do provozu, musí být splněny následující požadavky:***
  - ***Vybrání pohonu jako aktivní v instrukci USS_INIT***
  - ***Parametry OFF2 a OFF3 musí být na staveny na hodnotu 0***
  - ***Parametry Fault (porucha) a Inhibit (blokování) musejí být nastaveny na hodnotu 0***

- ***OFF2 – Bit, který zajišťuje zastavení pohonu s doběhem***
- ***OFF3 – Bit, který zajišťuje okamžité zastavení pohonu***
- ***F_ACK – Bit, který se využívá pro potvrzení porucha pohonu. Při přechodu F_ACK z log 0 na log 1 se daná porucha resetuje***
- ***DIR – Bit, který určuje na kterou stranu by se měl pohon otáčet***
- ***Drive – Adresa pohonu MicroMaster, obvykle adresy 0 – 31. Na tuto adresu má být odeslána instrukce USS_CTRL***
- ***Type – Určuje o jaký typ pohonu se jedná. Pro nastavení pohonu MicroMaster 3 a starší veze musí dojít k nastavení tohoto bitu na hodnotu 0. Pro pohon MicroMaster 4 nastavení bitu na hodnotu 1***
- ***Speed_SP – Nastavení rychlosti otáčení motoru, jako procento plných otáček. Záporná hodnota údaje znamená otáčení motoru na druhou stranu. Rozsah rychlosti otáčení je -200% - 200%.***
- ***Resp_R – Informace o stavu pohonu***
- ***Error – Indikuje chybu přenosu instrukce USS_CTRL***
- ***Status – Zpětná odezva motoru, stavové bity jsou zobrazené na Obr. 2.4.***
- ***Speed – Rychlost pohonu jako procento plných otáček s rozsahem -200% až 200%***
- ***Run EN – Bit, který indikuje, zda je motor v chodu (log 1) nebo v klidu (log 0)***
- ***D_Dir – Bit, který indikuje, kterým směrem se daný pohon otáčí***
- ***Inhibit – Indikuje stav blokovacího bitu, log 0 - není blokován, log 1 - je blokován***
- ***Fault – Indikuje stav bitu pro poruchu, log 0 - není porucha, log 1 - je porucha***

Tab. 2.2 Operandy platné pro USS_CTRL

Vstupy / Výstupy	Typ dat	Operandy
RUN	BOOL	I, Q, M, S, SM, T, C, V, L, Power Flow
OFF2	BOOL	I, Q, M, S, SM, T, C, V, L, Power Flow
OFF3	BOOL	I, Q, M, S, SM, T, C, V, L, Power Flow
F_ACK	BOOL	I, Q, M, S, SM, T, C, V, L, Power Flow
DIR	BOOL	I, Q, M, S, SM, T, C, V, L, Power Flow
DRIVE	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, Konstanta, *VD, *AC, *LD
SPD_SP	REAL	V, I, Q, M, S, SM, LD, AC, Konstanta, *VD, *AC, *LD
RSP_RCVD	BOOL	I, Q, M, S, SM, T, C, V, L
ERR	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD
DRV_STATUS	WORD	VW, T, C, IW, QW, SW, MW, SMW, LW, AC, AQW, *VD, *AC, *LD
DRV_SPEED	REAL	VD, ID, QD, MD, SD, SMD, LD, AC, *VD, *AC, *LD
DRV_RUN	BOOL	I, Q, M, S, SM, T, C, V, L
DVR_DIR	BOOL	I, Q, M, S, SM, T, C, V, L
DRV_INH(Inhibit)	BOOL	I, Q, M, S, SM, T, C, V, L
DRV_FLT (Fault)	BOOL	I, Q, M, S, SM, T, C, V, L



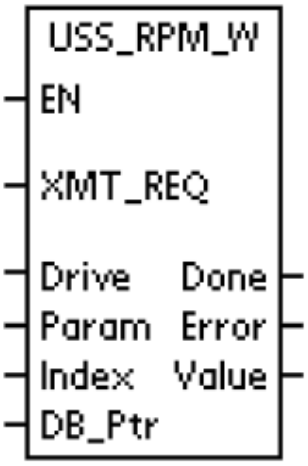
Obr. 2.6 Stavové bity DRV_STATUS

2.2.3 Instrukce USS_RPM_x

USS_RPM je instrukce čtení parametrů. V sériovém protokolu USS od společnosti Siemens existují pro čtení tři instrukce:

- ***USS_RPM_W – Instrukce, která čte parametr WORD bez znaménka***
- ***USS_RPM_D – Instrukce, která čte parametr Double WORD bez znaménka***
- ***USS_RPM_R – Instrukce, která čte parametr s plovoucí desetinnou čárkou***

U této instrukce je důležité, že v souvislosti s instrukcí pro zápis USS_WPM, nemůžou být zároveň v jednom časovém úseku aktivní obě instrukce. Aktivní může být tedy buď instrukce pro čtení (USS_RPM) nebo instrukce pro zápis (USS_WPM). Přenos USS_RPM je ukončen v případě, když pohon MicroMaster zpětně potvrdí příjem zprávy nebo dojde k chybě přenosu. Cyklus programu pokračuje i v případě, kdy se čeká na odezvu pohonu. Stejně jako u předchozích instrukcí, musí i u této být po celou dobu přenosu sepnutý bit EN, zapnutý musí být až do nastavení bitu DONE, který určuje konec přenosu. Instrukce USS_RPM je přenášena v každém přenosovém cyklu, pokud je zapnutý bit XMT_REQ. XMT_REQ by měl být spínán pulzem s náběžnou hranou, aby byl požadavek přenášený pouze v kladném přechodu vstupu parametru EN.



Obr. 2.7 Blokové schéma instrukce USS_RPM_x

Parametry:

- ***EN – Tzv. Povolovací bit, musí být stále aktivní, aby mohla být instrukce provedena***
- ***XMT_REQ – Vstup, který musí být sepnutý pro vysílání USS_RPM k pohonu MicroMaster. Tento vstup musí být spouštěn pulzy s náběžnou hranou.***

- ***Drive – Adresa pohonu MicroMaster, obvykle adresy 0 – 31. Na tuto adresu má být odeslána instrukce USS_RPM.***
- ***Param – Tato hodnota je číslo požadovaného parametru***
- ***Index – Indexová hodnota parametru, který bude načten***
- ***DB_Ptr – Na tento vstup musí být zadána adresa bajtového zásobníku***
- ***Done – Výstup, který se sepne při bezproblémovém přenosu instrukce***
- ***Error – Indikuje chybu přenosu instrukce USS_RPM_x***
- ***Value – Hodnota parametru, která je vrácená zpět pohonem***

V případě, že nebude instrukce dobře odeslána, bude výstupní bajt ERROR obsahovat výsledek, možné chyby přenosu jsou uvedeny v tabulce *Tab. 2.5*.

Tab. 2.3 Operandy platné pro USS_RPM_x

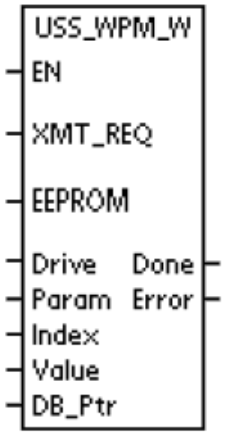
Vstupy / Výstupy	Typ dat	Operandy
XMT_REQ	BOOL	I, Q, M, S, SM, T, C, V, L, Signálový tok podmíněný detekcí náběžné hrany
DRIVE	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD, Konstanta
PARAM	WORD	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, AIW, *VD, *AC, *LD, Konstanta
INDEX	WORD	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, AIW, *VD, *AC, *LD, Konstanta
DB_PTR	DWORD	&VB
VALUE	WORD DWORD, REAL	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, AQW, *VD, *AC, *LD VD, ID, QD, MD, SD, SMD, LD, *VD, *AC, *LD
DONE	BOOL	I, Q, M, S, SM, T, C, V, L
ERROR	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD

2.2.4 Instrukce USS_WPM_x

USS_WPM je instrukce zapisování parametrů. V sériovém protokolu USS existují pro čtení tři instrukce:

- ***USS_WPM_W – Instrukce, která zapisuje parametr WORD bez znaménka***
- ***USS_WPM_D – Instrukce, která zapisuje parametr Double WORD bez znaménka***
- ***USS_WPM_R – Instrukce, která zapisuje parametr s plovoucí desetinnou čárkou***

U této instrukce, je důležité, že v souvislosti s instrukcí pro čtení USS_RPM, nemůžou být zároveň v jednom časovém úseku aktivní obě instrukce. Aktivní může být tedy buď instrukce pro zápis (USS_WPM) nebo instrukce pro čtení (USS_RPM). Přenos USS_WPM je ukončen v případě, když pohon MicroMaster zpětně potvrdí příjem zprávy nebo dojde k chybě přenosu. Cyklus programu pokračuje i v případě, kdy se čeká na odezvu pohonu. Stejně jako u předchozích instrukcí, musí i u této být po celou dobu přenosu sepnutý bit EN, zapnutý musí být až do nastavení bitu DONE, který určuje konec přenosu. Instrukce USS_WPM je přenášena v každém přenosovém cyklu, pokud je zapnutý bit XMT_REQ. XMT_REQ by měl být spínán pulzem s náběžnou hranou, aby byl požadavek přenášený pouze v kladném přechodu vstupu parametru EN.



Obr. 2.8 Blokové schéma instrukce USS_WPM_x

Parametry:

- ***EN – Tzv. Povolovací bit, musí být stále aktivní, aby mohla být instrukce provedena***
- ***XMT_REQ – Vstup, který musí být sepnutý pro vysílání USS_WPM k pohonu MicroMaster. Tento vstup musí být spouštěn pulzy s náběžnou hranou.***
- ***EEPROM – Paměť pohonu MicroMaster***
- ***Drive – Adresa pohonu MicroMaster, obvykle adresy 0 – 31. Na tuto adresu má být odeslána instrukce USS_WPM.***
- ***Param – Tato hodnota je číslo požadovaného parametru***
- ***Index – Indexová hodnota parametru, který bude načten***
- ***DB_Ptr – Na tento vstup musí být zadána adresa bajtového zásobníku, zásobník se využívá k uložení výsledku provádění příkazu***
- ***Done – Výstup, který se sepne při bezproblémovém přenosu instrukce***

- ***Error – Indikuje chybu přenosu instrukce USS_WPM_x***
- ***Value – Hodnota parametru, která je vrácená zpět pohonem***

U měničů MicroMaster třídy 3 se může hodnota parametru Value ukládat do paměti EEPROM pohonu. Záleží na nakonfigurování příslušného parametru měniče. Při zapnutí vstupu EEPROM se hodnota ukládá jak do paměti RAM tak do EEPROM pohonu. Měniče MicroMaster 3, ale tuto možnost nepodporují, tudíž musí být tato možnost vypnuta.

V případě, že nebude instrukce dobře odeslána, bude výstupní bajt ERROR obsahovat výsledek, možné chyby přenosu jsou uvedeny v tabulce *Tab. 2.5*.

Tab. 2.4 Operandy platné pro USS_WPM_x

Vstupy / Výstupy	Typ dat	Operandy
XMT_REQ	BOOL	I, Q, M, S, SM, T, C, V, L, Signálový tok podmíněný detekcí náběžné hrany
EEPROM	BOOL	I, Q, M, S, SM, T, C, V, L, Signálový tok
DRIVE	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD, Konstanta
PARAM	WORD	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, AIW, *VD, *AC, *LD, Konstanta
INDEX	WORD	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, AIW, *VD, *AC, *LD, Konstanta
DB_PTR	DWORD	&VB
VALUE	WORD DWORD, REAL	VW, IW, QW, MW, SW, SMW, LW, T, C, AC, AQW, *VD, *AC, *LD VD, ID, QD, MD, SD, SMD, LD, *VD, *AC, *LD
DONE	BOOL	I, Q, M, S, SM, T, C, V, L
ERROR	BYTE	VB, IB, QB, MB, SB, SMB, LB, AC, *VD, *AC, *LD

2.2.5 Chybové kódy přenosu instrukcí USS protokolu

Tab. 2.5 Chybové kódy přenosu instrukcí USS protokolu

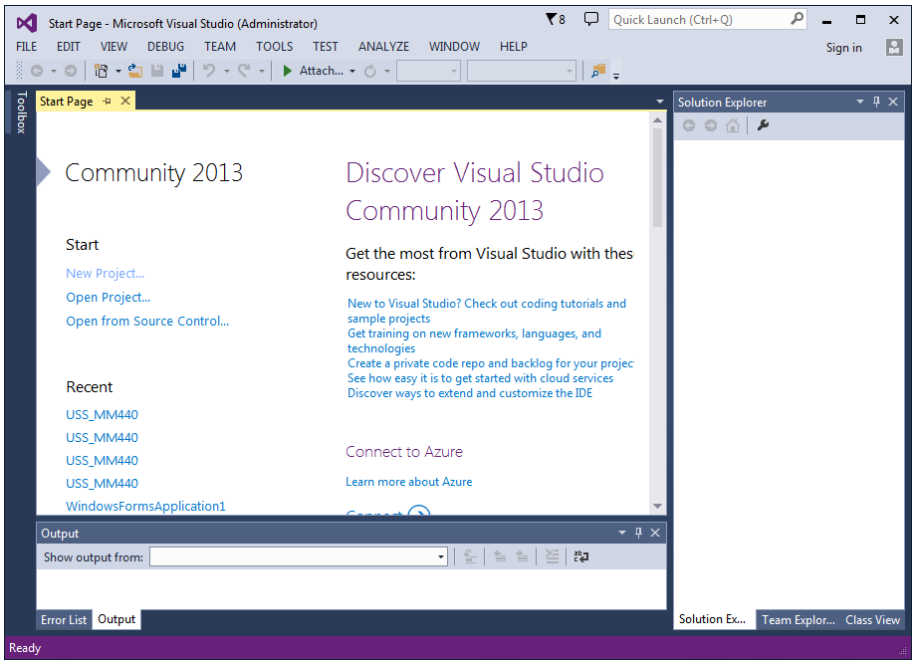
Chybové kódy	Popis
0	Bez chyby
1	Pohon neodpověděl
2	Byla zjištěna chyba kontrolního součtu v odpovědi pohonu
3	Byla zjištěna chyba parity v odpovědi pohonu
4	Chyba způsobená zásahem uživatelského programu
5	Pokus o nepřístupný příkaz
6	Zadána nepřípustná adresa pohonu
7	Komunikační port nebyl nastaven pro USS protokol
8	Komunikační port je zaneprázdněn zpracováváním instrukce
9	Vstupní rychlost pohonu je mimo rozsah
10	Nesprávná délka odpovědi pohonu
11	Nesprávný první znak v odpovědi pohonu
12	Znak délky v odpovědi pohonu není podporován instrukcemi USS
13	Odpověděl nesprávný pohon
14	Zadaná adresa DB_Ptr je nesprávná
15	Zadané číslo parametru je nesprávné
16	Byl vybrán neplatný protokol
17	USS je aktivní; změna není povolena
18	Byla specifikována neplatná přenosová rychlost
19	Nekomunikuje se: pohon není AKTIVNÍ
20	Parametr nebo hodnota v odpovědi pohonu jsou nesprávné nebo obsahují chybový kód
21	Namísto požadované hodnoty word byla vrácena hodnota double word
22	Namísto požadované hodnoty double word byla vrácena hodnota word

3 Programové vybavení pro práci s USS protokolem

Vytvořený software slouží pro ulehčení komunikace s frekvenčním měničem řady MicroMaster přes grafické uživatelské rozhraní. Sestavování požadovaného telegramu pomocí USS protokolu je velmi zdlouhavé, proto by měl tento program tuto komunikaci zjednodušit. Program slouží ke čtení a zapisování hodnot k daným parametrům frekvenčního měniče řady MicroMaster. Software vznikl ve vývojovém prostředí „Microsoft Visual Studio-Community 2013“, v programovací jazyku C#.

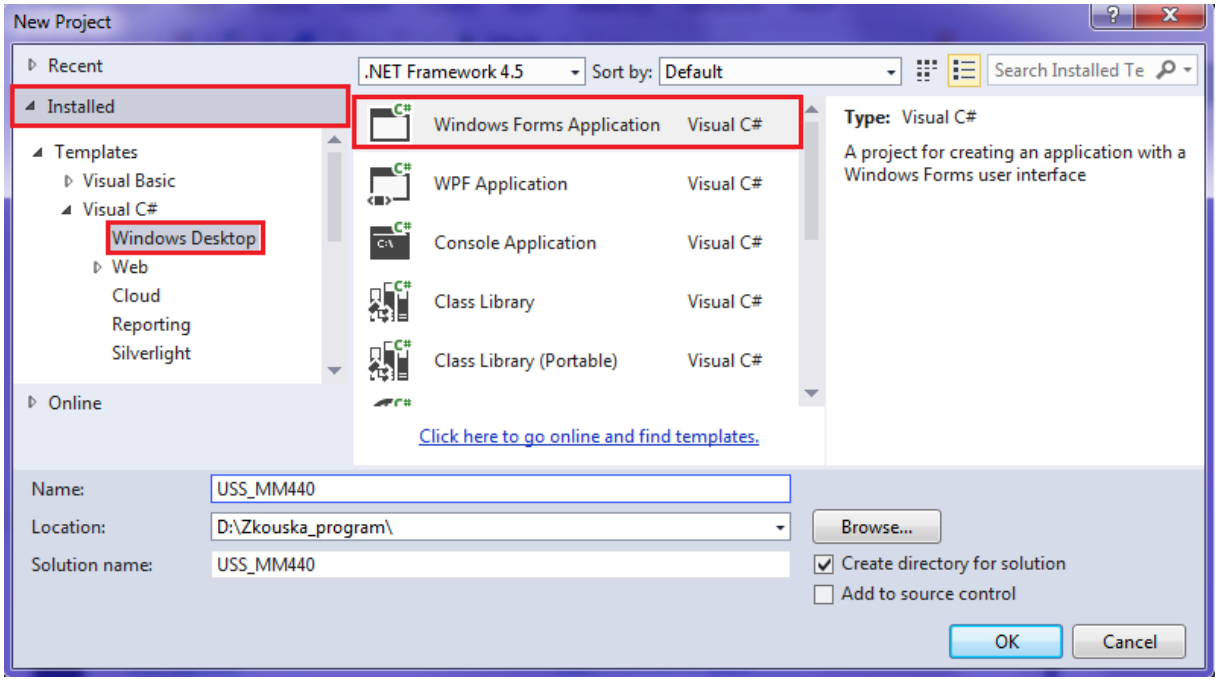
3.1 Vývojové prostředí Microsoft Visual Studio

Microsoft Virtual Studio je nástroj pro tvorbu aplikací pro systém Windows, nazývaný „Windows Forms“. Výhodou tohoto prostředí pro tvorbu aplikací je tvorba formulářových aplikací pomocí grafického designeru. Je zde na výběr velké množství kontrolek, textových polí, menu a dalších možností, včetně vlastní tvorby těchto komponentů. Výhodou vývojového prostředí Windows Visual Studio Community 2013 je jeho volná a bezplatná verze. Program je volně ke stažení a lze instalovat rovnou z webu. První měsíc užívání je zdarma, pro další volné užívání tohoto programu je nutné se po uplynutí jednoho měsíce zaregistrovat.



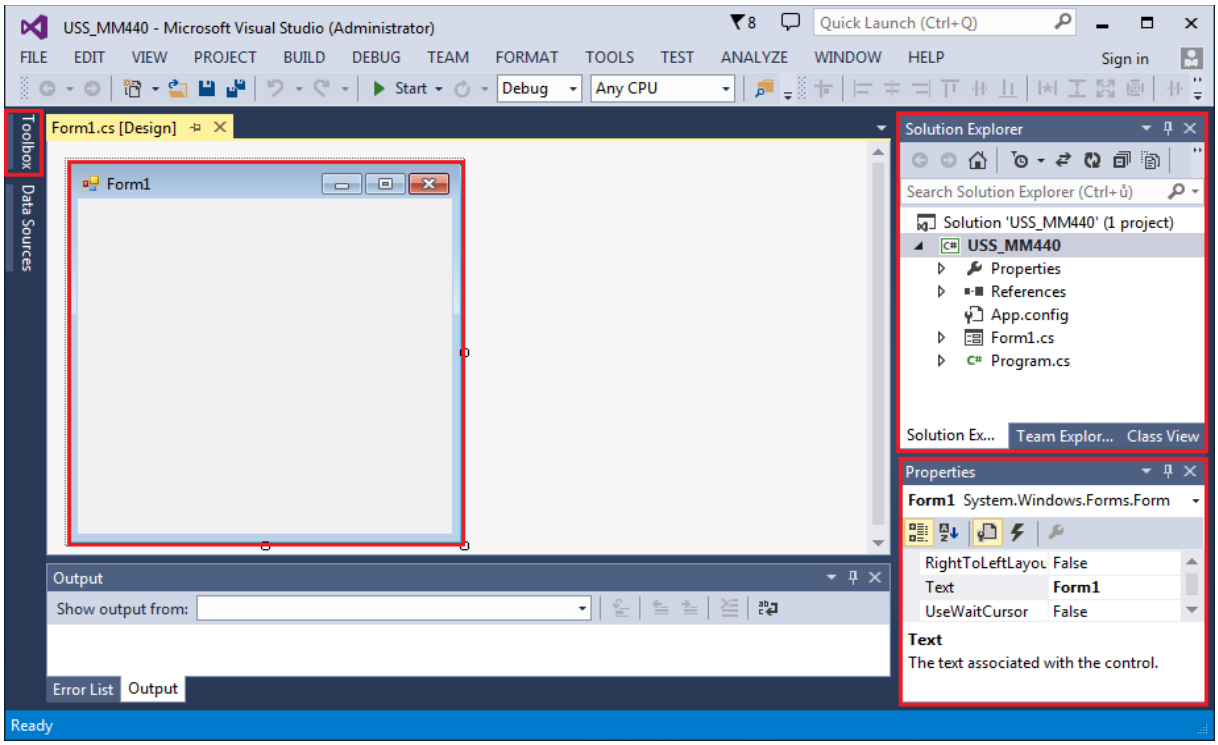
Obr. 3.1 Grafické prostředí Microsoft Visual Studia Community 2013

Vytvoření nového programu se prování přes tlačítko „FILE“ v horní liště programu, jak je vidět na *Obr. 3.1*. Dále se pokračuje přes tlačítko „New“ a „Project“. Pro vytvoření projektu se později objeví uživatelské okno pro vybrání příslušných možností nastavení nového projektu.



Obr. 3.2 Grafické prostředí Microsoft Visual Studia – Založení nového projektu

Pro vytvoření mého softwaru pro práci s USS protokolem jsem si zvolil programovací jazyk C# a využil jsem možností Windows Forms aplikací, jak je vidět na *Obr. 3.2*. Po vybrání vyznačených možností a vyplnění názvu tvořeného programu a uložiště se dostaneme do základního zobrazení programu Visual studio pro tvorbu aplikace.



Obr. 3.3 Grafické prostředí Microsoft Visual Studia – Vytváření vlastní aplikace

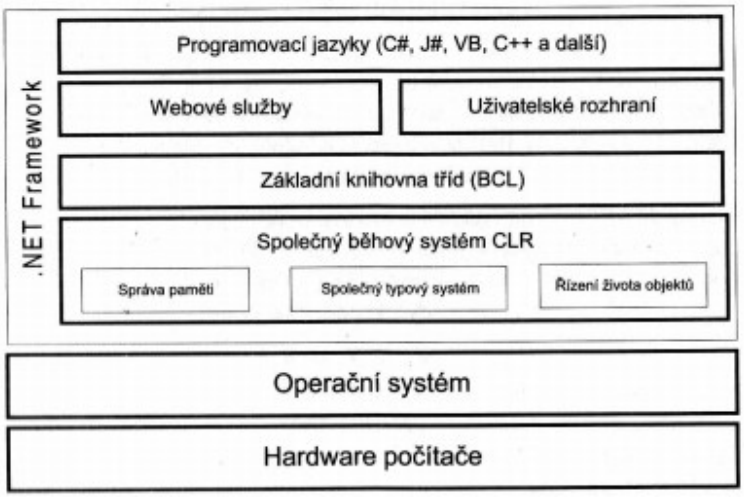
Na *Obr. 3.3* jsou červeně označeny důležité části, které se při tvorbě formulářových aplikací využívají:

- **Designer (Grafický návrhář) – Zde se umisťují požadované komponenty, jako například tlačítka, textové pole a tvoří se zde design grafického prostředí softwaru**
- **Properties (Vlastnosti) – Zde se nastavují vlastnosti jednotlivých prvků námi umístěných v grafickém návrháři**
- **Toolbox (Panel nástrojů) – Slouží jako databáze s jednotlivými kontrolkami a dalšími prvky, které lze umístit do grafického návrháře**
- **Solution Explorer – Zobrazuje aktuálně otevřený projekt a jeho soubory**

3.2 Programovací jazyk C#

Programovací jazyk C# vyvinula společnost Microsoft a jde o programovací jazyk, který je objektově orientovaný. Programový jazyk C# se využívá hlavně u formulářových aplikací v systému Windows, také nazývaných jako Windows Forms aplikace a dále u databázových programů a webových aplikací.

Důležitým prvkem při programování v programovacím jazyku C# je prostředí .NET Framework. Toto prostředí je velmi důležité z hlediska spouštění programů vytvořených v programovacím jazyce C#. Prostředí .NET Framework se skládá z několika částí.



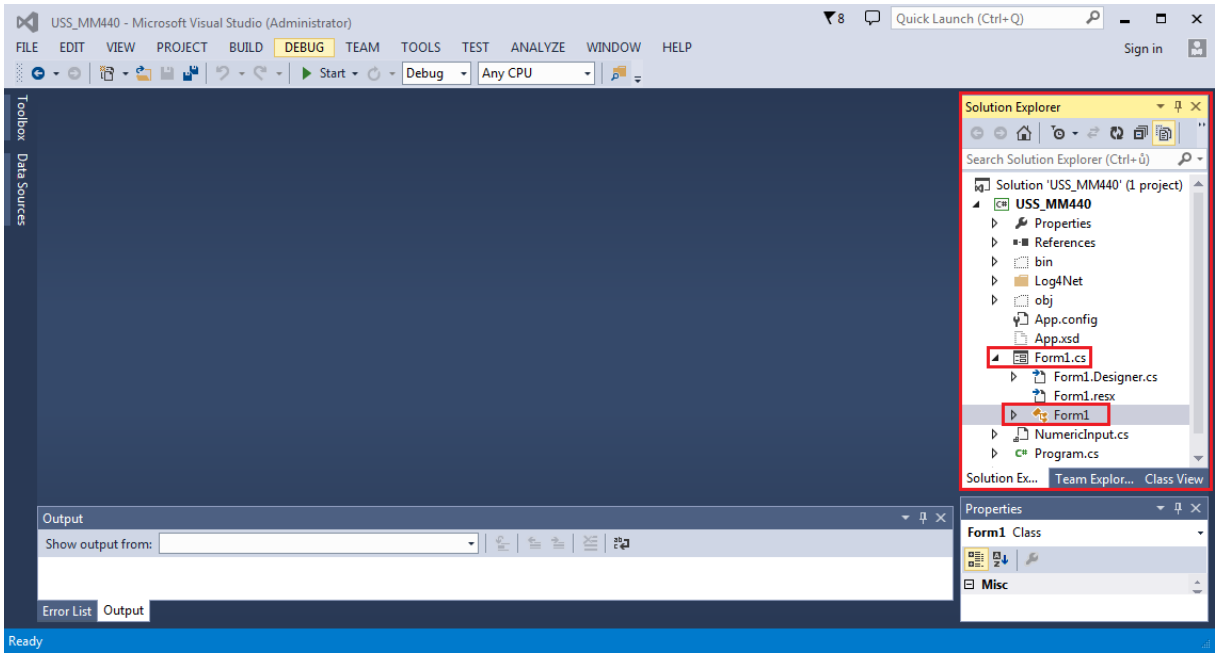
Obr. 3.4 Struktura prostředí .NET Framework

Jak je vidět na *Obr. 3.4* prostředí .NET Framework se skládá z několika částí. Nejdůležitější částí tohoto prostředí je společný běhový systém (CLR). CLR zajišťuje chod programů, které byly napsány v různých programovacích jazycích. Další nejdůležitější částí prostředí .NET Framework je tzv. knihovna tříd (BCL). Ta obsahuje vše od vstupních a výstupních operací až po třídy, které slouží pro ukládání různých dat. Z tohoto důvodu programový jazyk C# nepotřebuje své knihovny. Nadřazené nad knihovou BCL jsou knihovny webových služeb a grafického rozhraní pro tvoření programů.

V programovacím jazyce C# se rozlišují dva datové typy. Do tzv. Hodnotových typů patří například čísla, znaky, logické hodnoty a struktury. Druhým jsou tzv. Objektové typy, což jsou tzv. Třídy. V tomto programovacím jazyce, jelikož jde o čistě objektově orientovaný typ jazyka, je velké množství předdefinovaných tříd, které jsou odvozeny od společné třídy „object“.

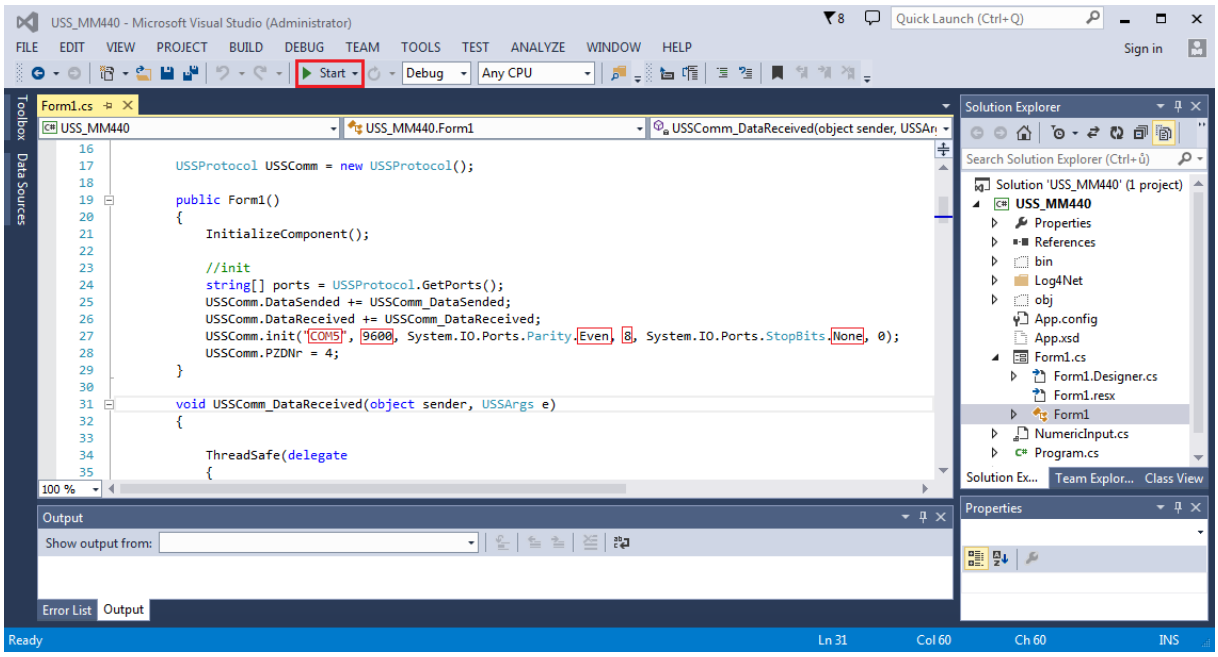
3.3 Nastavení parametrů sériového portu při prvním využití softwaru pro práci s USS protokolem

Při používání vytvořeného softwaru pro komunikaci s frekvenčními měniči řady MicroMaster pomocí USS protokolu, je nutné před prvním spuštěním nastavit parametry využívaného sériového portu, přímo ve zdrojovém kódu vytvořeného softwaru. Všechny potřebné soubory k programu jsou uloženy na přiloženém CD-ROM a je zapotřebí před požadovanými změnami uložit celou složku „Software“ do uložiště na PC, aby mohly být uloženy veškeré změny. Pro změnu těchto parametrů je zapotřebí spustit soubor „USS_MM440.sln“, v programu Microsoft Visual Studio, nejlépe ve výše zmiňované verzi. Po spuštění tohoto souboru se zobrazí grafické prostředí programu Microsoft Visual Studio. V okně „Solution Explorer“ vybereme jedním kliknutím rozbalení záložky u soubor „Form1.cs“, což je grafický návrh vytvořeného programu a po jeho rozbalení dvojklikem vybereme položku „Form1“, jednotlivé položky jsou červeně vyznačeny na *Obr. 3.5*.



Obr. 3.5 Nastavení parametrů sériového portu v programu Microsoft Visual Studio

Na hlavní ploše programu Microsoft Visual Studio se nám objeví zdrojový kód daného souboru, který slouží k dalšímu nastavování a využívání komponentů zobrazených na grafickém návrhu vytvořeného softwaru. V tomto zdrojovém kódu lze měnit parametry přenosu na sériovém portu, konkrétně lze tyto parametry změnit na řádku 27 jak je vyznačeno na *Obr. 3.6*.



Obr. 3.6 Změna hodnot parametrů sériového portu v programu Microsoft Visual Studio

Pro nastavení vlastností sériového portu lze přenastavit tyto parametry:

- **Komunikační port – název portu PC**
- **Přenosová rychlost sériového komunikace**
- **Parita**
- **Velikost dat**
- **StopBits**

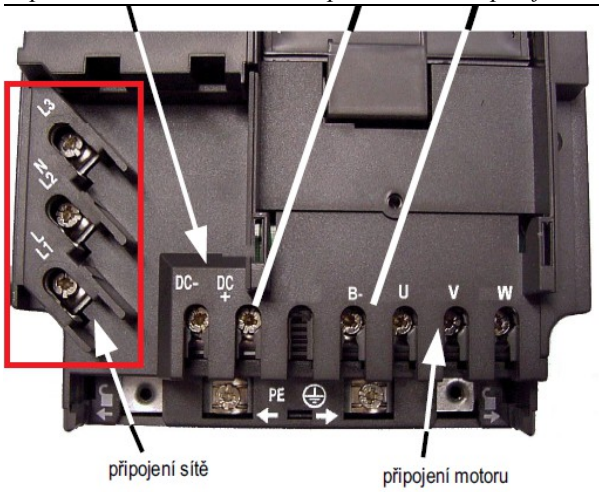
Po nastavení námi požadovaných parametrů sériové komunikace necháme program přeložit pomocí tlačítka „Start“ vyznačeném na *Obr. 3.6*. Po otevření grafického okna softwaru pro práci s USS protokolem můžeme toto okno vypnout a vypnout i program Microsoft Visual Studio. Software je nyní nastaven na požadovaný sériový port a jeho přenosové parametry a lze ho spustit souborem „USS_MM440.exe“.

4 Praktické využití USS protokolu na měniči řady MM

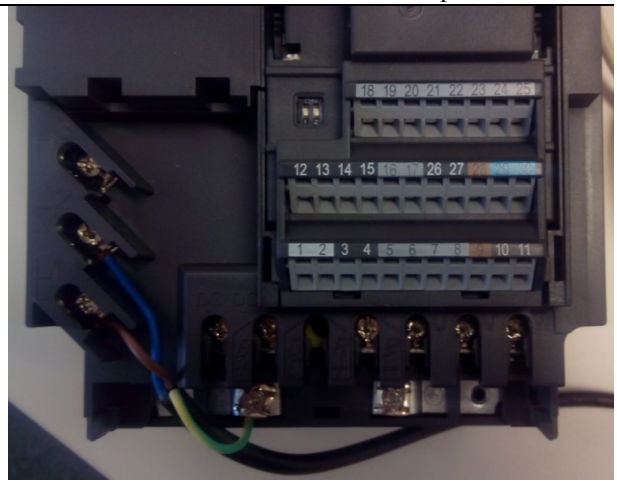
Pro praktické vyzkoušení USS protokolu na měničích řady MicroMaster jsem využil dvou možností. Nejdříve jsem otestoval a zprovoznil komunikaci s měničem pomocí sériového terminálu, abych nejdříve otestoval komunikaci s měničem a mé poznatky USS protokolu. Později jsem otestoval komunikaci pomocí mnou navrženého softwaru pro čtení a zapisování hodnot parametrů měniče řady MicroMaster, pomocí USS protokolu. Celé praktické zkoušení bylo provedeno na měniči řady MicroMaster, konkrétně na měniči MM440.

4.1 Zapojení měniče pro testování

Nejprve jsem musel oživit daný měnič. Jelikož daný typ měniče nepodporuje pouze oživení ovládacích obvodů pomocí napájecího napětí 24V, proto jsem musel připojit pod napájecí napětí 230 V celý měnič.

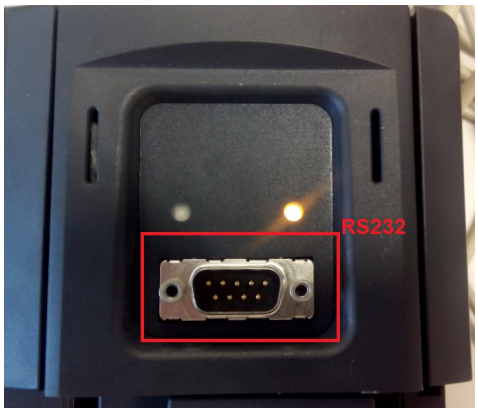


Obr. 4.1 Napájecí svorkovnice



Obr. 4.2 Zapojení napájecího napětí 230V

Komunikaci jsem zkoušel pouze na jednom měniči (point-to-point), proto jsem mohl propojit PC s měničem přes rozhraní RS232. Jelikož na mém PC chybí port RS232, musel jsem připojit kabel přes převodník k portu USB. Na druhé straně byl kabel připojen k měniči přes port RS232 umístěném na konektoru pro připojení ovládacího panelu BOP.



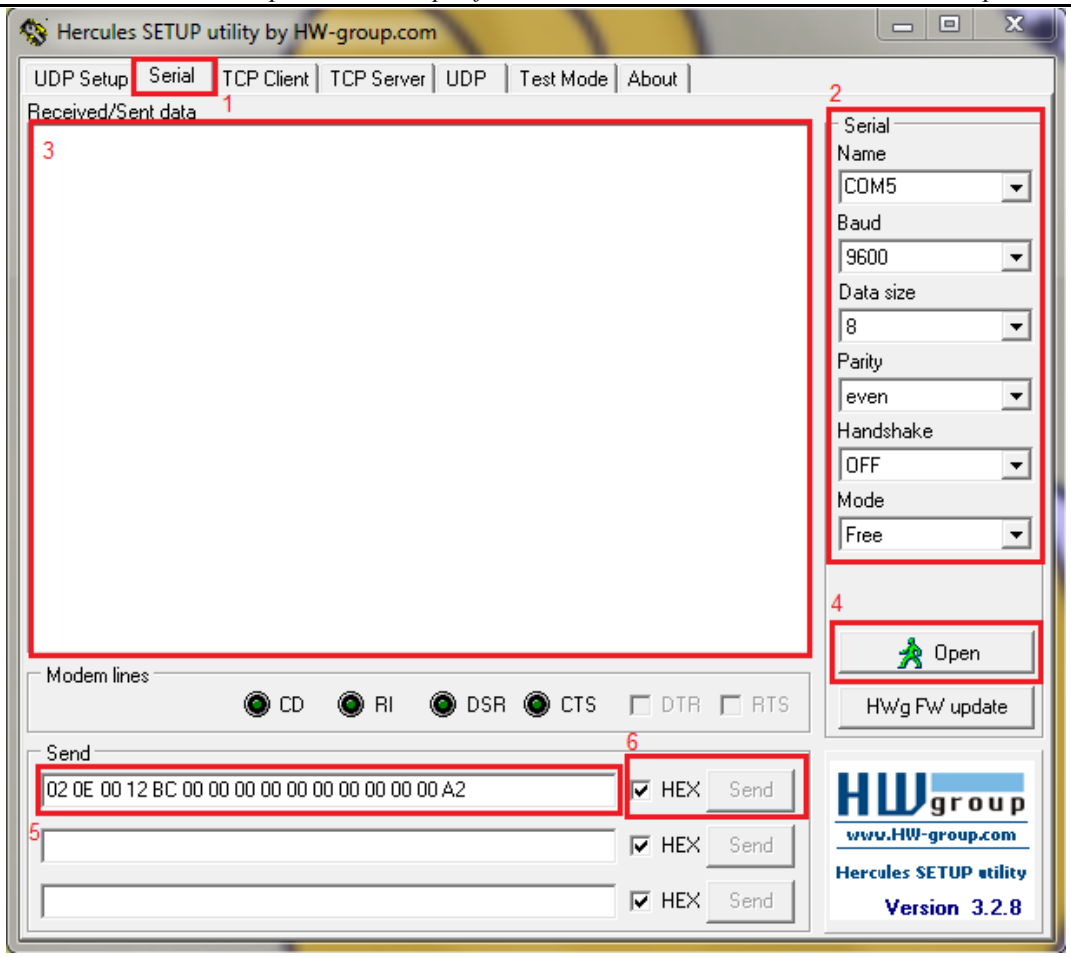
Obr. 4.3 Konektor pro připojení RS232

Obr. 4.4 Celkové zapojení pro testování měniče MM440

S takto zapojeným měničem jsem mohl začít testovat komunikaci mezi prvky pomocí USS protokolu.

4.2 Komunikace s měničem řady MicroMaster pomocí sériového terminálu

Na začátku praktického zkoušení komunikace s měničem MM440 jsem nejdříve zkoušel komunikaci s PC pomocí sériového terminálu. Jako sériový terminál jsem si zvolil program Hercules od společnosti HW-group. Tento program je volně šiřitelný a nabízí velké množství možností komunikace.



Obr. 4.5 Náhled programu Hercules společnosti HW-group

Na *Obr. 4.5* můžeme vidět grafické prostředí programu Hercules a vyznačené důležité části tohoto programu:

- **1 – Nastavení programu na volbu sériového terminálu**
- **2 – Nastavení parametrů přenosu:**
- **3 – Okno pro zobrazování provedených akcí**
- **4 – Tlačítko pro otevření komunikačního portu**
- **5 – Okno pro zapsání požadované zprávy ve tvaru hex**
- **6 – Vybrání možnosti zobrazit číslo v hex tvaru a tlačítko pro odeslání zprávy**

Pro vyzkoušení první komunikace s měničem MM440 jsem si sestavil pomocí USS protokolu požadovaný telegram, který se dotazoval na hodnotu parametru P0700. Jedná se o parametr pro způsob ovládání měniče a může nabývat hodnot 0 – 6. Podrobné zpracování

tohoto dotazu do hexadecimálního kódu můžete vidět v *Kapitole 1.3.5*. Výsledný požadovaný kód vypadal takto: 02 0E 00 12 BC 00 00 00 00 00 00 00 00 00 A2.

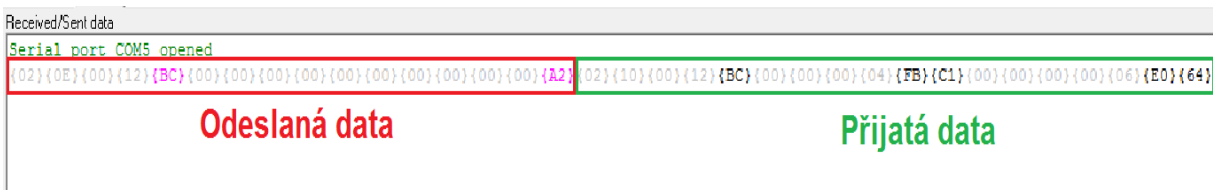
Před odesláním tohoto požadavku však bylo ještě zapotřebí nastavit zobrazovací okno na formát Hex a nastavit komunikační vlastnosti v programu Hercules a to následovně:

- **Name – nastavení portu – COM5**
- **Baud – nastavení přen. rychlosti – 9600**
- **Data size – velikost dat – 8**
- **Parity – nastavení parity – none**
- **Handshake – nastavení řízení komunikace – OFF**
- **Mode – nastavení módu komunikace – Free**

Po otevření portu a odeslání zprávy však nepřišla žádná odpověď ze strany měniče. Pro ověření přenosové cesty k měniči jsem se pokusil o komunikaci s měničem pomocí softwaru STARTER od společnosti Siemens. Komunikace přes tento software byla bez problému navázána, ověřil jsem si, že přenosová cesta mezi PC a měničem je v pořádku. Pečlivě jsem prostudoval znovu dokumentaci k měniči MM440 a USS protokolu a zjistil, že nastala chyba v Paritě (paritním bitu). Paritní bit je určen k jednoduché detekci chyb ve slově. V našem případě musela být parita nastavena na sudý paritní bit (even), což znamená sudý počet jedničkových bitů ve slově (paritní bit tedy doplňoval jedničku, aby byl výsledný počet sudý).

Po přenastavení parity na hodnotu „even“ (sudá) byl znovu otevřen port a odeslána námi požadovaná zpráva.

Zpětná odpověď od měniče: 02 10 00 12 BC 00 00 00 04 FB C1 00 00 00 00 06 E0 64. Podle USS protokolu je vidět, že výsledná hodnota by se měla objevit na pozici 18. čísla a hodnota parametru P0700 je číslo 4. Správnost hodnoty parametru jsem ověřil pomocí softwaru STARTER Siemens a tato hodnota souhlasila.



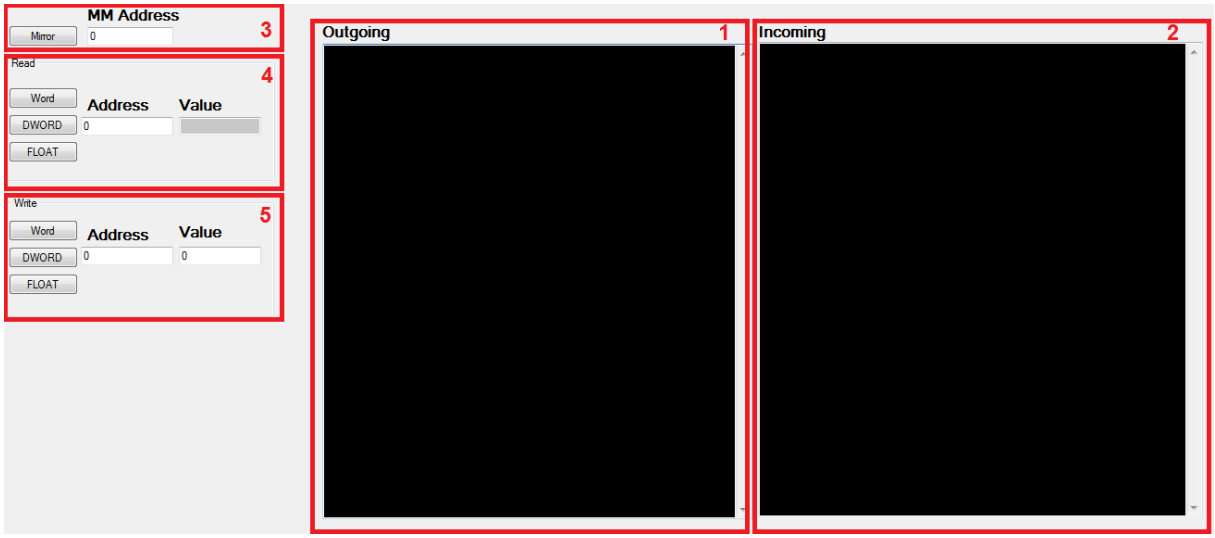
Obr. 4.6 Náhled odeslaných a přijatých dat v programu Hercules

Pro další ověření jsem přes USS protokol sestavil a otestoval více takovýchto zpráv, jak pro čtení parametru, tak i pro zápis hodnoty parametru a zrcadlené zprávy (Mirror telegram).

4.3 Komunikace s měničem řady MicroMaster pomocí navrženého softwaru

Pro testování navrženého softwaru bylo zapotřebí oživení frekvenčního měniče MM440 a propojení PC s měničem. Zapojení měniče a propojení s PC je popsáno v *Kapitole 4.1*. Testování navrženého softwaru proběhlo stejně jako v předchozím případě na frekvenčním měniči MM440. Vytvořený software se spouští pomocí souboru „USS_MM440.exe“ umístěném na přiloženém CD-ROM.

Vytvořený software využívá komunikačního protokolu USS od společnosti Siemens a slouží ke čtení a zapisování hodnot vybraného parametru frekvenčního měniče řady MicroMaster. Na *Obr. 4.7* můžeme vidět grafické prostředí softwaru.



Obr. 4.7 Náhled grafického prostředí navrženého softwaru pro práci s USS protokolem

Na *Obr. 4.7* jsou vyznačeny důležité části tohoto programu:

- **1 – Outgoing, grafické okno pro zobrazování odeslaných zpráv do frek. měniče v hexadecimálním kódu**
- **2 – Incoming, grafické okno pro zobrazování přijatých zpráv od frek. měniče v hexadecimálním kódu**
- **3 – MM Address, textové okno pro zadání adresy frek. měniče a otestování této adresy odesláním zprávy „Mirror“ pomocí označeného tlačítka**

- **4 – Read, textové okno „Address“ pro zadání adresy parametru frek. měniče a odeslání našeho požadavku příslušným tlačítkem podle typu předpokládané hodnoty, hodnota parametru se zobrazí v textovém poli „Value“**
- **5 – Write, textové okno „Address“ pro zadání adresy parametru frek. měniče a textové okno „Value“ pro zadání požadované hodnoty vybraného parametru, odeslání našeho požadavku příslušným tlačítkem podle typu předpokládané hodnoty**

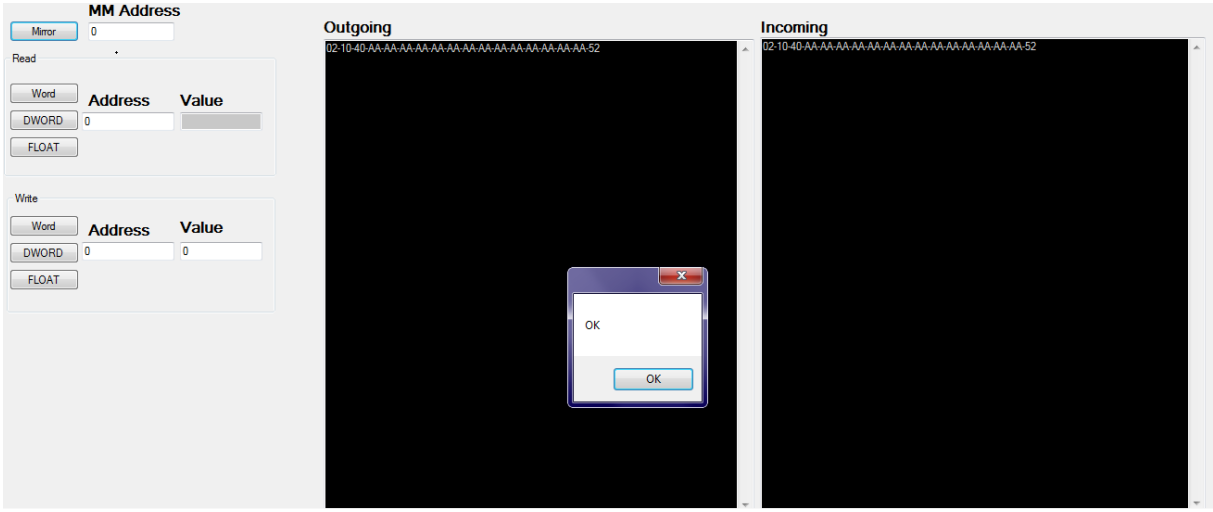
Pokud se tento program testuje na zařízení poprvé, je nutné provést před samotným používání programu nastavení komunikačních parametrů:

- **Komunikační port, v našem případě – COM5**
- **Přenosová rychlost, v našem případě – 9600**
- **Parita, v našem případě – EVEN (sudá)**
- **Velikost dat, v našem případě – 8**
- **StopBits, v našem případě – NONE (žádné)**

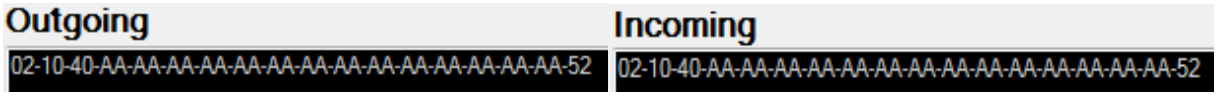
Tyto parametry se nastavují přímo ve zdrojovém kódu tohoto softwaru a pro jejich změnu je zapotřebí využití programu Microsoft Visual Studio nebo jiného podobného programu pro tvorbu aplikací pomocí Windows forms. Popis postupu pro změnu těchto parametrů najdete v *Kapitole 3.3.* V případě, že nebudou parametry sériového portu správně nastaveny, nebude daný program funkční a vypíše se chybová hláška „Error“.

Nyní už jsem mohl přikročit k vlastnímu testování vytvořeného softwaru. Po spuštění programu jsem otestoval funkci „Mirror“. Ta slouží pro ověření a nastavení adresy měniče. Adresa měniče bývá defaultně nastavena na hodnotu 0, ale může být tato hodnota jiná. V mém případě byla adresa frekvenčního měniče MM440 skutečně nastavena na hodnotu 0 a dokazuje to i testování této funkce. Pro ověření je nutné zadat hodnotu adresy měniče do textového pole „MM Address“ v rozmezí 0 – 31 a stisknutí tlačítka „Mirror“. V zápětí se odešle zpráva k měniči ve tvaru tzv. Mirror mesage (zrcadlená zpráva), popsaná v *Kapitole 1.1.2*, to znamená, že měnič by měl tuto zprávu pouze beze změny vrátit zpět. Nastavil jsem

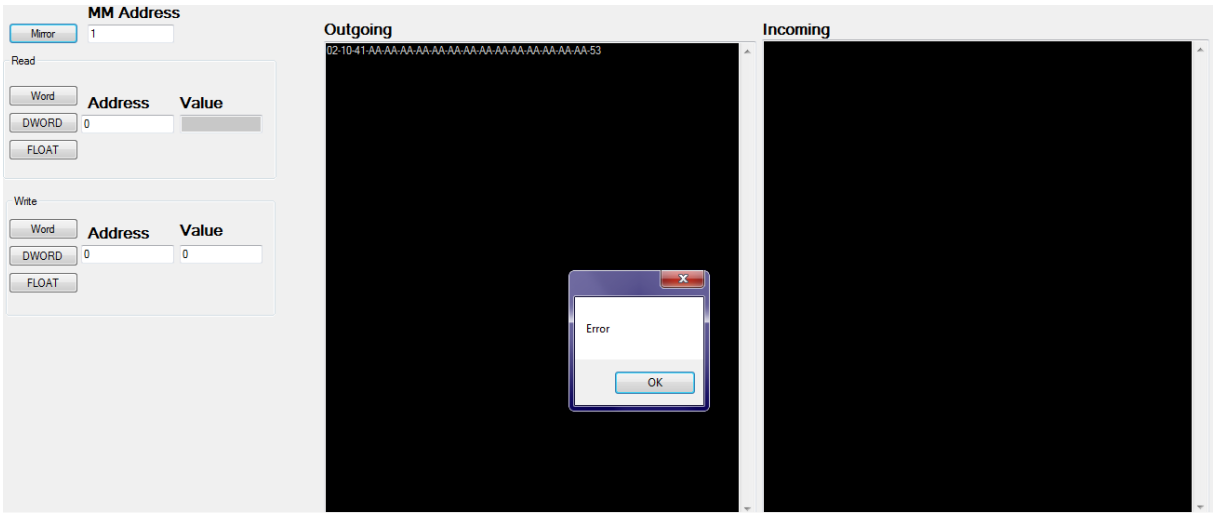
proto adresu frekvenčního měniče MM440 nejprve na hodnotu 0 a pak na hodnotu 1. V případě shodující se hodnoty s reálnou adresou měniče MM440, hodnotou 0, odpověděl měnič zpětně stejnou zprávou a zobrazila se informační zpráva „OK“, jak je vidět na *Obr. 4.8*. Ve druhém případě nedošlo k odpovědi ze strany měniče a zobrazila se informační zpráva „Error“, jak je vidět na *Obr. 4.10*. Odeslaná a přijatá zpráva se v hexadecimálním kódu zobrazí v grafických oknech „Outgoing“ a „Incoming“. Mezi odesláním požadované zprávy a odpovědí od měniče je určité zpoždění. Toto čekání na odpověď jsem vytvořil z důvodu případného zpoždění odpovědi frekvenčního měniče a je nastaveno na hodnotu 1 sekunda. Po uplynutí této doby se vypíše informační zpráva o stavu podle toho, zda v tomto časovém intervalu obdrží program zprávu od měniče zpět nebo ne.



Obr. 4.8 Náhled grafického prostředí softwaru při správném nastavení adresy měniče MM440



Obr. 4.9 Náhled grafických oken při správném nastavení adresy měniče MM440

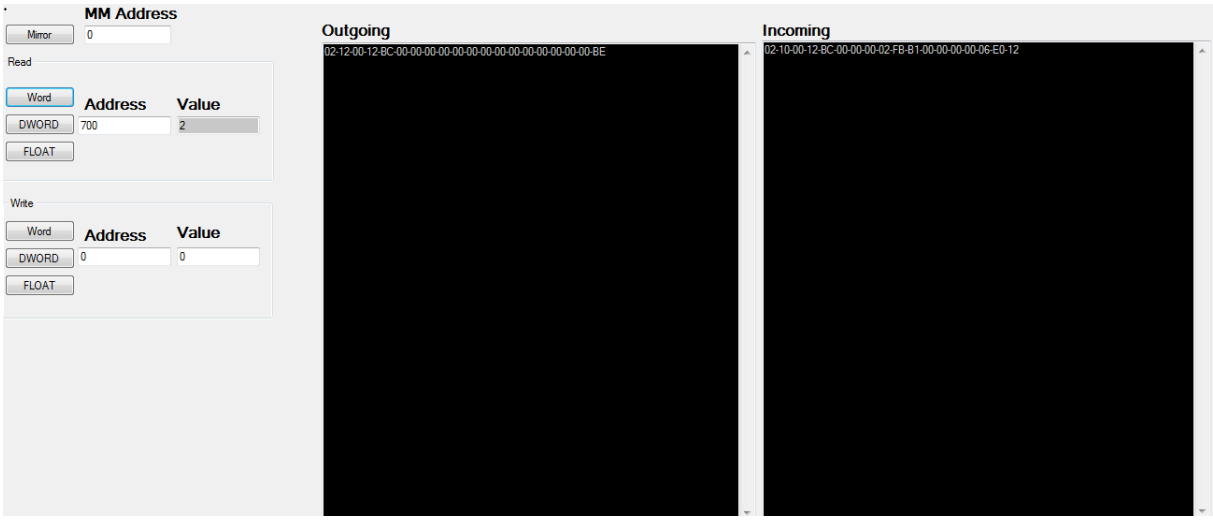


Obr. 4.10 Náhled grafického prostředí softwaru při nesprávném nastavení adresy měniče MM440

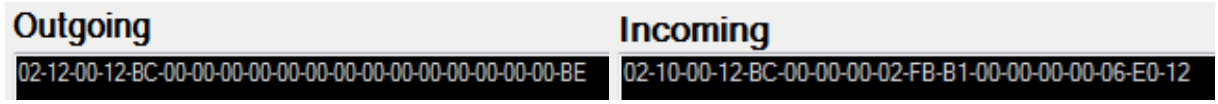
Při testování softwaru jsem se dále zaměřil na funkci „Read“, čtení hodnoty parametru frekvenčního měniče MM440. Tato funkce umožňuje přečtení hodnoty určitého parametru frekvenčního měniče řady MicroMaster. Pro vyzkoušení jsem si zvolil parametr P0700, který určuje způsob ovládání měniče a může tento parametr nabývat hodnot 0-6. Zapsal jsem proto hodnotu 700 do textového pole „Address“. Tlačítka pro odeslání našeho požadavku na přečtení hodnoty jsou označeny podle druhu čísla:

- **Word – celočíselný typ o velikosti dva bajty a rozsahu hodnot 0 až 65 535**
- **Dword – celočíselný typ o velikosti čtyři bajty a rozsahu hodnot 0 až 4 294 967 295**
- **Float – číselný typ s pohybující se desetinnou čárkou o velikosti čtyři bajty a možnosti až 6 desetinných míst**

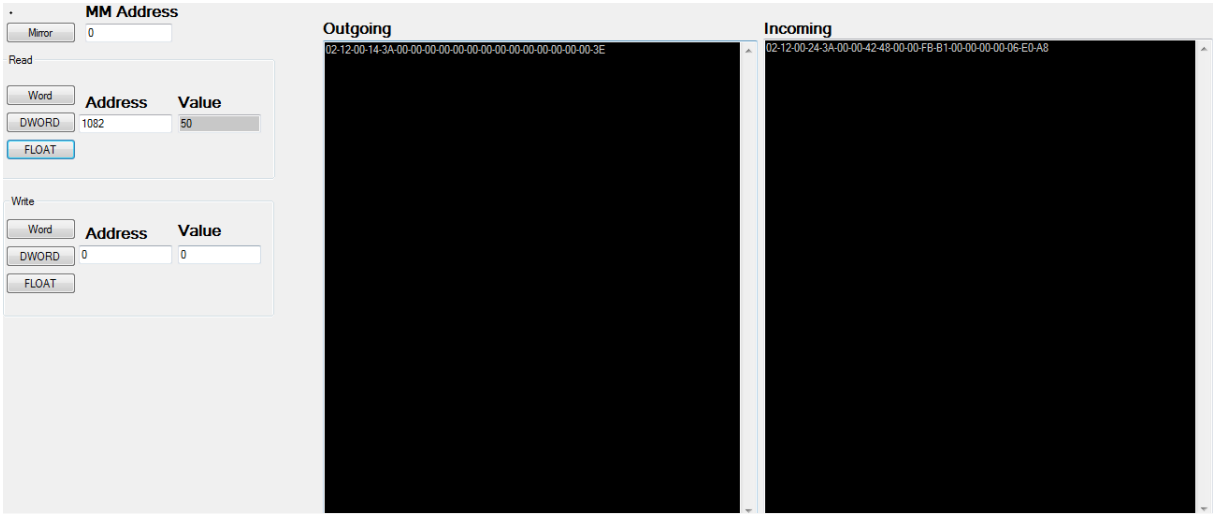
V mém případě jsem zvolil tlačítko „Word“. Je vždy nutné se podívat do manuálu o jakou hodnotu parametru půjde. Po odeslání požadovaného dotazu jsem obdržel zpětnou odpověď od frekvenčního měniče MM440 s danou hodnotou a zobrazila se informační zpráva „OK“. Následně po vypnutí této informace se hodnota parametru P0700 zobrazila v textovém okně „Value“ a to hodnota parametru 2, jak je vidět na *Obr. 4.11*. Odeslaná zpráva a přijatá zpráva se opět v hexadecimálním kódu zobrazila v grafických oknech „Outgoing“ a „Incoming“. V případě nesprávného vybrání daného typu hodnoty se vypíše informační zpráva „Error“, jak je vidět na *Obr. 4.15* a hodnota parametru se nevypíše. Otestoval jsem i čtení hodnoty float (hodnoty s desetinnou čárkou) a to na parametru P1082, což je parametr pro nastavení maximální hodnoty výstupního kmitočtu měniče MM440. Výsledná hodnota byla 50 Hz, což je vidět na *Obr. 4.13*. Testování proběhlo stejným způsobem jako u parametru P0700.



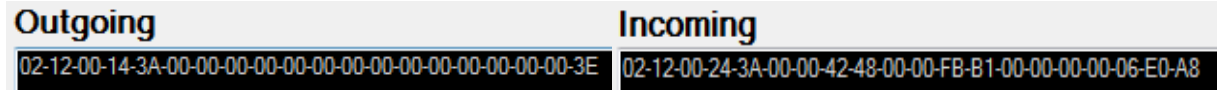
Obr. 4.11 Náhled grafického prostředí softwaru při správném načtení hodnoty parametru P0700



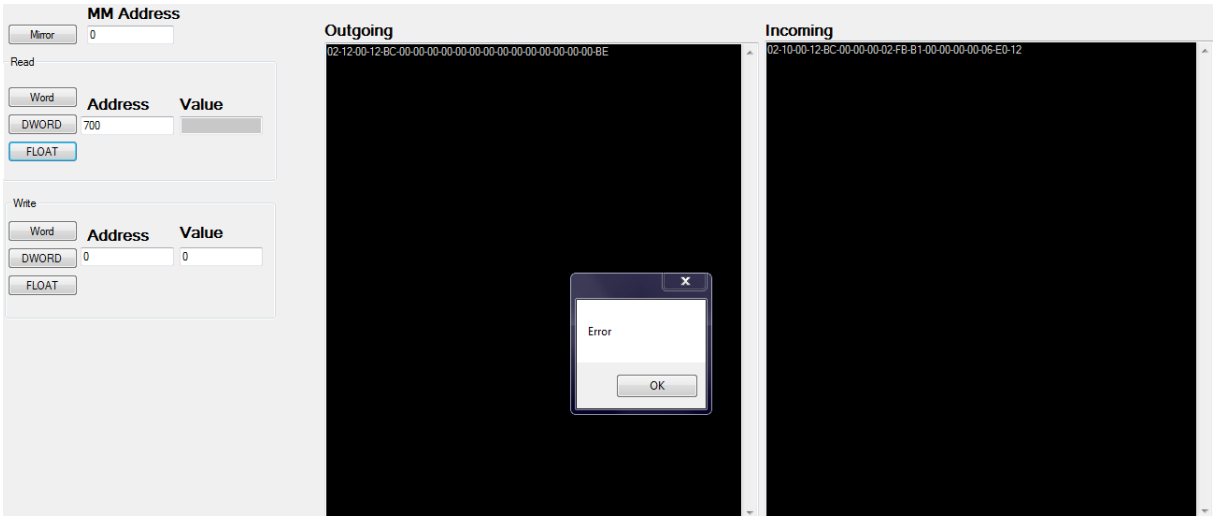
Obr. 4.12 Náhled grafických oken při správném načtení hodnoty parametru P0700 měniče MM440



Obr. 4.13 Náhled grafického prostředí softwaru při správném načtení hodnoty parametru P1082

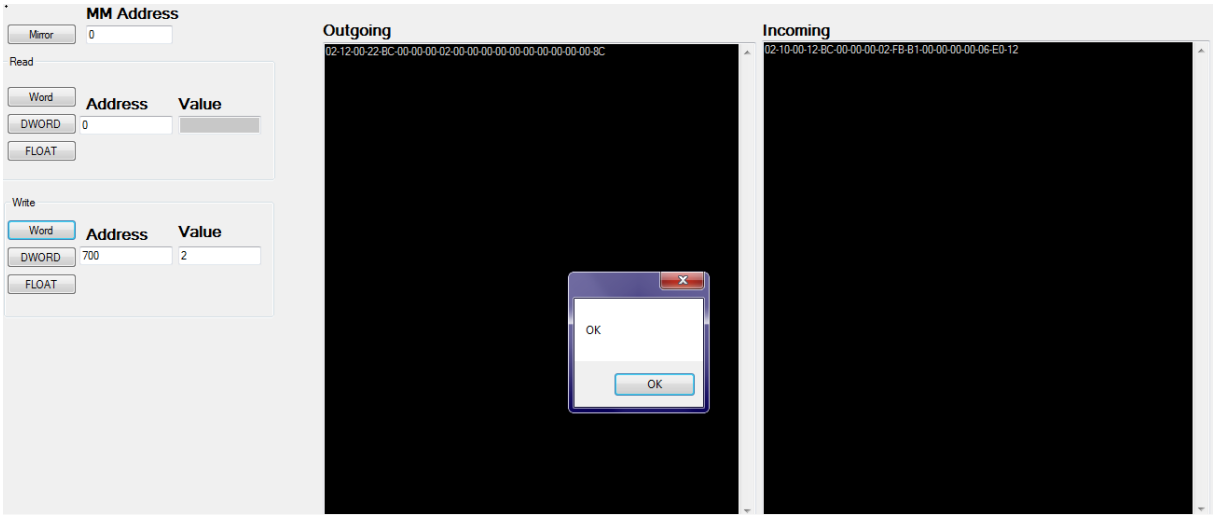


Obr. 4.14 Náhled grafických oken při správném načtení hodnoty parametru P1082 měniče MM440

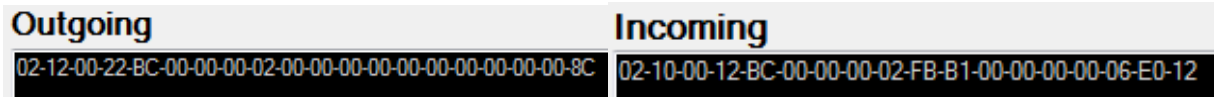


Obr. 4.15 Náhled grafického prostředí softwaru při nesprávném načtení hodnoty parametru

Jako poslední bod testování jsem měl otestovat funkci „Write“. Tato funkce slouží pro přepsání (přenastavení) hodnoty parametru frekvenčního měniče MM440. Stejně jako u předchozí funkce „Read“ zapíšeme do textového pole „Address“ adresu parametru měniče řady MicroMaster. Opět jsem si pro testování zvolil parametr P0700. Do textového pole jsem zapsal požadovanou hodnotu parametru a to konkrétně číslo 2, aktuální hodnota parametru bylo číslo 4. Podle hodnoty jsem vybral tlačítko „Word“ a odeslal můj požadavek. O provedení zápisu požadované hodnoty mě informovala informační zpráva „OK“, jak je vidět na *Obr. 4.16*. V případě špatného výběru typu hodnoty, mi stejně jako u funkce „Read“, čtení hodnoty parametru, vyskočila informační zpráva „Error“ a daný příkaz se neprovedl. Výběr typu hodnoty pomocí tlačítka je v obou případech, jak při čtení tak při zápisu totožné a podrobněji jsou tyto typy hodnot popsány u testování funkce „Read“.

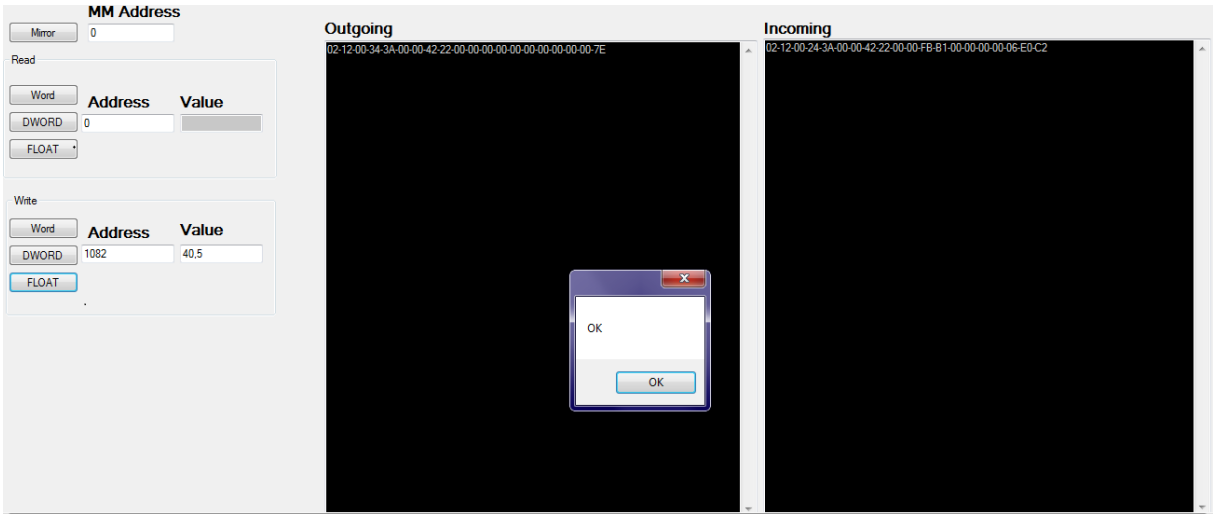


Obr. 4.16 Náhled grafického prostředí softwaru při správném zapsání hodnoty parametru P0700

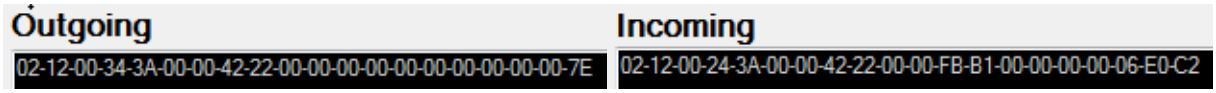


Obr. 4.17 Náhled grafických oken při správném zapsání hodnoty parametru P0700 na hodnotu 2

Ani při zkoušení zapisování hodnot parametru jsem nezapomněl na otestování typu hodnoty Float. Konkrétně jsem vyzkoušel zapsat hodnotu s desetinnou čárkou pro parametr P1082, pro nastavení maximální výstupní frekvence měniče a nastavil jsem tento parametr na hodnotu 40.5 Hz, jak je vidět na *Obr. 4.18*.

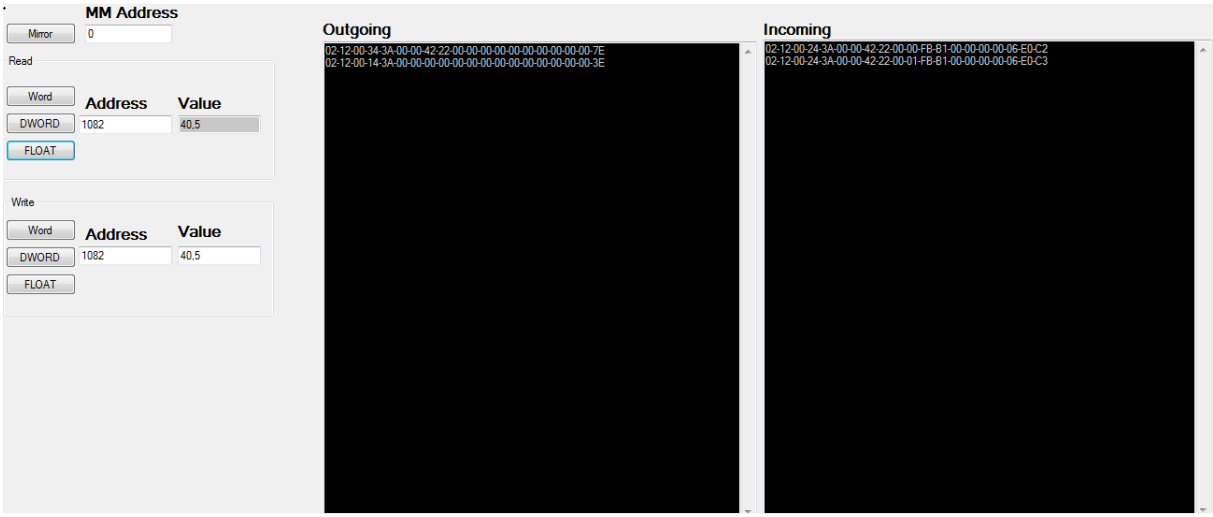


Obr. 4.18 Náhled grafického prostředí softwaru při správném zapsání hodnoty parametru P1082

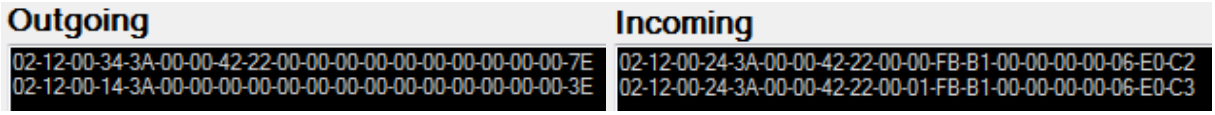


Obr. 4.19 Náhled grafických oken při správném zapsání hodnoty parametru P1082 na hodnotu 40,5

Při testování zápisů hodnot parametrů frekvenčního měniče MM440 jsem ověřoval zapsané hodnoty zpětně pomocí funkce „Read“, čtení hodnoty parametru. Příklad ověření zapsané hodnoty, konkrétně hodnoty parametru P1082, můžete vidět na *Obr. 4.20*.



Obr. 4.20 Náhled grafického prostředí softwaru při ověření zapsání hodnoty parametru P1082



Obr. 4.21 Náhled grafických oken při ověření zapsání hodnoty parametru P1082 na hodnotu 40,5

I pro testování funkce „Write“ se zapisují odeslané zprávy z PC a přijaté zprávy od měniče MM440 v grafických oknech „Outgoing“ a „Incoming“. Při testování jsem vyzkoušel v obou režimech i hodnotu typu Dword, ale nebylo to pro tento typ měniče důležité z důvodu malých hodnot parametrů. Co se týče velikosti parametrů, nastavil jsem a otestoval tento program do hodnoty parametrů 4000. Frekvenční měnič, na kterém probíhalo testování, má nejvyšší možný nastavitelný parametr P3980. V rámci testování bylo vyzkoušeno, pro zapisování a čtení, více parametrů frekvenčního měniče MM440, zde jsou uvedeny pouze některé z nich pro přiblížení funkčnosti vytvořeného softwaru.

5 Závěr

V této bakalářské práci jsem se zabýval problematikou Univerzálního sériového protokolu USS vyvinutém společností Siemens a jeho aplikování na frekvenční měniče řady MicroMaster. Práce je rozdělena do několika hlavních částí.

V první části jsem se zabýval analyzováním USS protokolu, postupně zjišťoval jeho vlastnosti a prozkoumával jeho jednotlivé části a skládání těchto částí do výsledného celku, čímž je hexadecimální kód, tvořící náš požadavek, či dotaz směrem k frekvenčnímu měniči nebo zpět. Na závěr první kapitoly jsem pro ukázku postupně sestavil hexadecimální kód pro dotaz k frekvenčnímu měniči na hodnotu parametru frekvenčního měniče P0700.

Ve druhé části, po analyzování USS protokolu, jsem se zaměřil na využití tohoto protokolu pro řízení frekvenčního měniče řady MicroMaster, konkrétně s využitím softwaru STEP 7-Micro/WIN. Tento software má instalovanou knihovnu USS protokolu a v této knihovně již nadefinované instrukce pro ovládání frekvenčního měniče a tvorbu případného programu v tomto softwaru. V této kapitole jsem popsal jednotlivé instrukce USS protokolu pro software STEP 7-Micro WIN. Zjistil jsem, že pro využití tohoto softwaru je zapotřebí třetího prvku pro ovládání frekvenčního měniče. Kromě PC a frekvenčního měniče je v tomto případě nutné umístění některého z programovatelných automatů, například automatu S7-200, do kterého musí být vytvořený program v prostředí STEP 7 nahrán a z něho je pak frekvenční měnič ovládán. Myslím si, že toto řešení je z hlediska USS protokolu nevýhodné z důvodu dalšího prvku. Naopak výhodou mi přijdou již předdefinované instrukce USS protokolu.

Ve třetí části jsem se zaměřil na popis vývojového prostředí Microsoft Visual studio, konkrétně na založení nového projektu pro vytvoření aplikace Windows Forms a na stručný popis použitého jazyka C#, jelikož jsem tyto části použil pro tvorbu softwaru pro práci s USS protokolem. Na konci třetí kapitoly jsem se zabýval popisem nastavení parametrů sériové komunikace pro komunikaci s měničem řady MicroMaster, konkrétně frekvenčního měniče MM440. Tyto změny je zapotřebí provést ve zdrojovém kódu vytvořeného softwaru, jak je popsáno v *Kapitole 3.3*. Toto nastavování se nastavuje při prvním použití tohoto programu s frekvenčním měniče a toto nastavování je nevýhodou tohoto programu. Výhodnější by bylo

rozšíření tohoto programu a nastavování těchto parametrů přímo z uživatelského grafického prostředí programu.

Ve čtvrté části jsem se zaměřil na celkové testování USS protokolu při komunikaci s frekvenčním měničem MM440, ale využít lze kterýkoliv měničů řady MicroMaster. Nejprve jsem popsal oživení a zapojení měniče a komunikační propojení s PC. Dále jsem se zaměřil na otestování komunikace s měničem MM440 pomocí USS protokolu přes sériový terminál Hercules. Při tomto testování jsem narazil na problémy s komunikací, ale nakonec jsem problém vyřešil. Příčinou těchto problémů bylo špatné nastavení sériové komunikace, konkrétně chybné nastavení Parity. Po přenastavení vše fungovalo jak mělo, ale využívání tohoto způsobu komunikace je velmi nevýhodné z důvodu počítání jednotlivých telegramů podle USS protokolu a to je velmi časově náročné. Ulehčením pro tuto komunikaci byl vytvořený software, který slouží pro komunikaci s měničem MM440 a jiným měničem řady MicroMaster. Konkrétně jde o aplikaci čtení a zapisování hodnot parametrů frekvenčního měniče. Tento program si podle zadané hodnoty sám vytvoří hexadecimální kód přes který s měničem komunikujeme. Popis programu a jeho použití je popsáno v *Kapitole 4.3*. Při testování jsem nenarazil na nějaké větší komplikace.

Na závěr této práce bych chtěl zhodnotit využití USS protokolu v praxi. V praxi se tento protokol moc nevyužívá. Výhodné používat USS protokol je podle mého názoru na malých aplikacích o malém počtu komponentů. Pro velké projekty je velmi využívaná sběrnice PROFIBUS.

Seznam použité literatury a internetových zdrojů

[1] MÖLLER-NEHRING, Walter a Wolfgang BOHRER. *Universal Serial Interface Protocol, USS protocol: Specification* [online]. 1994 [cit. 2015-07-12]. Dostupné z: https://cache.industry.siemens.com/dl/files/253/24178253/att_101238/v1/uss_24178253_spec_76.pdf

[2] *Using USS Protocol with MICROMASTER MM420* [online]. 2005 [cit. 2015-07-12]. Dostupné z: https://www.automatyka.siemens.pl/docs/MM4_USS_en.pdf

[3] SIEMENS. *Programovatelný automat S7-200 Systémový manuál* [online]. 3. 2005 [cit. 2015-07-12]. Dostupné z: https://cache.industry.siemens.com/dl/files/582/1109582/att_22074/v1/S7200CZ.pdf

[4] SIEMENS. *MicroMaster 440 - Návod k obsluze (stručný)* [online]. 2005 [cit. 2015-07-13]. Dostupné z: http://www.elprim.cz/navody/Siemens_MM440/MM440_strucny_navod.pdf

[5] SIEMENS. *MicroMaster 440 - Návod k obsluze a údržbě* [online]. Praha, 2005 [cit. 2015-07-13]. Dostupné z: http://stest1.etnetera.cz/ad/current/content/data_files/technika_pohonu/menice/stridave_menice/nizkonapetove_menice/micromaster/micromaster_440/_manualy/opi_micromaster_440_03-2003_cz.pdf

[6] BĚHÁLEK, Marek. *Programovací jazyk C#* [online]. [cit. 2015-07-13]. Dostupné z: http://www.cs.vsb.cz/behalek/vyuka/pcsharp/text.pdf

[7] SHARP, John. *Step by step Microsoft Visual C# 2013* [online]. 2013 [cit. 2015-07-13]. Dostupné z: https://ptgmedia.pearsoncmg.com/images/9780735681835/samplepages/9780735681835.pdf

[8] SPŠ BRNO. *Struktura programu, proměnné, konstanty, výrazy v jazycích C a C#* [online]. 2010 [cit. 2015-07-13]. Dostupné z: https://moodle.sspbrno.cz/pluginfile.php/5468/mod_resource/content/0/CSHARP/ZakladniPojmy_C_a_CSharp.pdf

[9] VIRIUS, Miroslav. *C# pro zelenáče*. Praha: Neocortex, c2002, 255 s. Bestseller for all. ISBN 80-863-3011-7.

[10] SELLS, Chris. *Windows forms programming in C#*. Boston: Addison-Wesley, ©2004. xliii, 681 s. Microsoft .NET development series. ISBN 0-321-11620-8.

[11] SELLS, Chris. *C# a WinForms: programování formulářů Windows*. Vyd. 1. Brno: Zoner Press, 2005. 648 s. Encyklopedie Zoner Press. ISBN 80-86815-25-0.

[12] ELLER, Frank. *C# - začínáme programovat: podrobný průvodce začínajícího uživatele*. 1. vyd. Praha: Grada, 2002, 240 s. ISBN 80-247-0324-6.

Přílohy

Příloha A:

- **Soubory k vytvořenému softwaru pro práci s USS protokolem**
- **Tyto soubory jsou umístěny na přiloženém CD-ROM ve složce „Software“**