TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Implementace komunikačního protokolu Siemens USS na platformě PLC automatů BR

Bakalářská práce

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

Autor práce: **Daniel Matocha**
Vedoucí práce: Ing. Martin Diblík Ph.D.

Liberec 2019

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

Implementation of Siemens USS communication protocol with BR-Automation PLC controller

Bachelor thesis

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

Author: **Daniel Matocha**
Supervisor: Ing. Martin Diblík Ph.D.

Liberec 2019

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Zadání bakalářské práce

Implementace komunikačního protokolu Siemens USS na platformě PLC automatů BR

Jméno a příjmení: **Daniel Matocha**
Osobní číslo: M16000083
Studijní program: B2612 Elektrotechnika a informatika
Studijní obor: Elektronické informační a řídicí systémy
Zadávající katedra: Ústav mechatroniky a technické informatiky
Akademický rok: **2018/2019**

Zásady pro vypracování:

1. Prostudujte strukturu a způsob použití komunikačního protokolu USS firmy Siemens pro ovládání frekvenčních měničů dle dostupné dokumentace.
2. Seznamte se s programovatelnými automaty a s vývojovým prostředím Automation Studio firmy BR-Automation.
3. V prostředí Automation Studio navrhněte vhodné datové struktury a program, který bude zajišťovat ovládání jednoduchých frekvenčních měničů Siemens pomocí USS protokolu.
4. Na vybraném typu frekvenčního měniče program otestujte.

Rozsah grafických prací:	dle potřeby dokumentace
Rozsah pracovní zprávy:	30–40 stran
Forma zpracování práce:	tištěná/elektronická

Seznam odborné literatury:

[1] JOHN, Karl-Heinz; TIEGELKAMP, Michael. IEC 61131-3: programming industrial automation systems : Concepts and Programming Languages, Requirements for Programming Systems, Decision-Making Aids. 2nd ed. Springer, 2010. ISBN 9783642120145.

[2] SIEMENS A.G. Universal Serial Interface Protocol USS: Specification. 09.94. 1994. E20125-D0001-S302-A1-7600.

[3] Siemens A.G. Sinamics G110 120W – 3kW Seznam parametrů, příručka uživatele. Vydání 04/03. 2004.

Vedoucí práce:	Ing. Martin Diblík, Ph.D. Ústav mechatroniky a technické informatiky
Datum zadání práce:	10. října 2018
Předpokládaný termín odevzdání:	30. dubna 2019

L. S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci 10. října 2018

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 texty tištěné verze práce a elektronické verze práce vložené do IS STAG se shodují.

29. 4. 2019 Daniel Matocha

Poděkování

Rád bych touto cestou poděkoval vedoucímu práce panu Ing. Martinovi Diblíkovi Ph.D. za jeho cenné rady, ochotu a odborné vedení při vypracování této bakalářské práce.

Dále bych rád poděkoval Ing. Petru Skalickému za týdenní školení v prostředí B&R Automation Studio.

Nakonec bych rád poděkoval své rodině a blízkým přátelům za trpělivost a projevenou podporu po dobu mého studia.

Implementace komunikačního protokolu Siemens USS na platformě PLC automatů BR

Abstrakt

Cílem bakalářské práce byla implementace komunikačního protokolu Siemens USS na platformě PLC automatů B&R-Automation. Začátek práce je věnován studiu dokumentace USS protokolu, jeho struktuře a způsobu použití. Jelikož se jedná o protokol vyvinutý firmou Siemens, nemá podporu na konkurenčních řídicích systémech. Proto bylo nutné navrhnout vhodnou datovou strukturu a vytvořit program v prostředí B&R Automation Studio, který bude zajišťovat veškeré náležitosti USS protokolu. Výsledný program řeší veškerou stavbu, odeslání a přijetí telegramu. Program je vytvořen s cílem ovládání jednoduchých frekvenčních měničů Siemens, ale je s ní možné řídit i složitější frekvenční měniče.

Klíčová slova: USS protokol, B&R-Automation, Siemens, Frekvenční měniče, PLC, řídicí automat

Implementation of Siemens USS communication protocol with BR-Automation PLC controller

Abstract

The thesis deals with implementing a Siemens USS communication protocol on the B&R-Automation PLC platform. The first part focuses on the study of the USS protocol's documentation, it's structure and means of use. Since the protocol was developed by the Siemens company it does not have the support of other competing control systems. Developing a suitable data structure and programme in the B&R Automation Studio environment was necessary for meeting every requirement of the USS protocol. The final programme is meant to deal with the build, sending and receiving of the telegram. This programme is developed with the goal of controlling the Siemens variable frequency drive but also allows to be used with more advanced ones.

Keywords: USS protocol, B&R-Automation, Siemens, variable frequency drive, PLC, programmable controller

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

ADR	Address byte
ASCII	American Standard Code for Information Interchange
BCC	Block Check Character
CPU	Central processing unit
HEX	Hexadecimální soustava
HMI	Human-machine interface
IND	Index
LGE	Telegram length
net characters	síťový blok dat
PC	Personal Computer
PKW	Parameter ID Value
PKE	Parameter ID
PWE	Parameter Value
PZD	Process Data
PLC	Programmable Logic Controller
RAM	Random Access Memory
ST	Structured Text
STX	Start of Text
USB	Universal Serial Bus
USS	Universal Serial Interface Protocol

1 Úvod

Siemens SINAMICS G110 je frekvenční měnič se základními funkcemi pro různé průmyslové aplikace vyžadující proměnné otáčky. Jedná se o zařízení menších rozměrů k provozu skalárního řízení třífázových motorů na jednofázových sítích. Tato řada je ideálním řešením pro řízení nižších výkonů a zároveň patří mezi cenově dostupnější.

Možnosti řízení tohoto jednoduchého měniče jsou pouze dvě. V případě jednoduší aplikace je řízení analogově nebo digitálně přes dostupné svorkovnice dostačující, ale pro úplné řízení v automatizovaných provozech je zapotřebí využít sériovou komunikaci a protokol USS. Pomocí této varianty řízení lze uvést měnič do provozu, nastavit parametry a ovládat ho. Jelikož se jedná o protokol vytvořený firmou Siemens, nemá podporu na konkurenčních řídicích systémech. Proto je nutné vytvořit vlastní program pro platformu B&R-Automation, pro řízení měniče pomocí USS protokolu. Tato implementace rozšíří možnost výběru vhodného frekvenčního měniče o firmu Siemens.

Na začátku práce bylo nutné nastudovat strukturu a způsob užití USS protokolu. Pro lepší pochopení funkčnosti protokolu bylo provedeno testování pomocí počítače s převodníkem USB-RS232 a frekvenčním měničem. Následně byla zvolena Testovací konfigurace, na které se vyvíjel a testoval výsledný program. Navržené datové struktury a program byl vytvořen ve vývojovém prostředí B&R Automation Studio. V poslední části práce je popis Tvorba programu, jeho uspořádání a výsledná funkčnost programu.

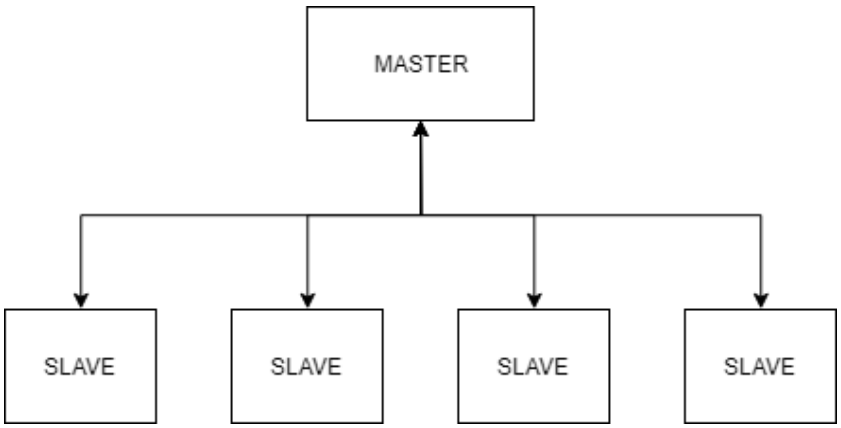
Toto téma bakalářské práce jsem si vybral z důvodu možnosti programování řídicích automatů. Už od počátku studia mi je programování blízké a zároveň pro mě představuje možnost získat nové zkušenosti, které mohu v budoucí praxi využít. Současně bych rád porozuměl problematice komunikace po sériové lince na které si vyzkouším implementovat existující protokol. Proto doufám, že výsledný program bude přínosem pro aplikace s konfigurací PLC od B&R-Automation a frekvenčního měniče od společnosti Siemens.

2 USS protokol

2.1 Specifikace

Protokol USS je jednoduchý standard pro sériový přenos dat definovaný společností Siemens, který je určený pro komunikaci s frekvenčními měniči od stejnojmenné firmy. Využívá model komunikace master-slave (viz Obrázek 2.1). Role master může zaujmout pouze jedno zařízení, které ovládá celý proces řízení (např. PC, PLC). V roli slave jsou jednotlivé frekvenční měniče, kterých může být maximálně 31 a jsou od sebe odlišeny adresou.

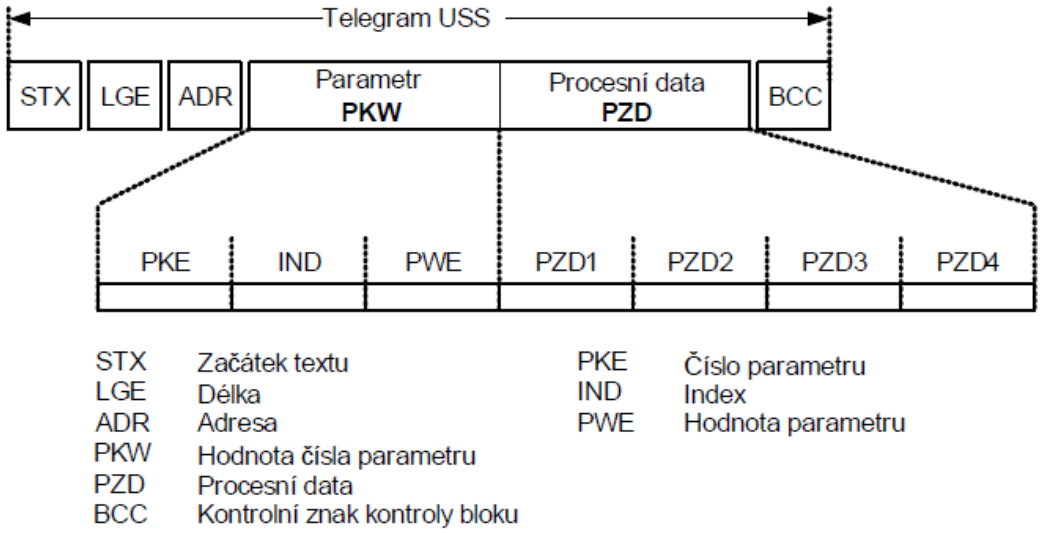
Komunikace probíhá v poloduplexním (half-duplex) režimu, takže v jednom okamžiku může data posílat pouze jedno zařízení. Slave nikdy nemůže poslat data bez předchozího požádání a přenos dat mezi slave není možný. Master vždy zahajuje komunikaci jako první a může poslat data pouze na jednu adresu nebo všem adresám na sběrnici.



Obrázek 2.1: Model Master-Slave komunikace

2.2 Struktura telegramu

Každý telegram se skládá z šesti částí (viz Obrázek 2.2). V záhlaví se nachází STX, LGE, ADR. Následuje část net characters, kterou tvoří dvě hlavní nezávislé datové části PKW a PZD oblast. Tato část může mít nulovou nebo proměnnou délku. Na konci telegramu v zápatí se vyskytuje poslední část BCC. Podrobnější struktura telegramu je zobrazena v příloze B.



Obrázek 2.2: Základní struktura USS telegramu; zdroj [12]

2.2.1 STX byte

Část STX (Start of text) značí počátek telegramu. Má velikost 1 byte a vždy obsahuje ASCII znak STX (02 hex).

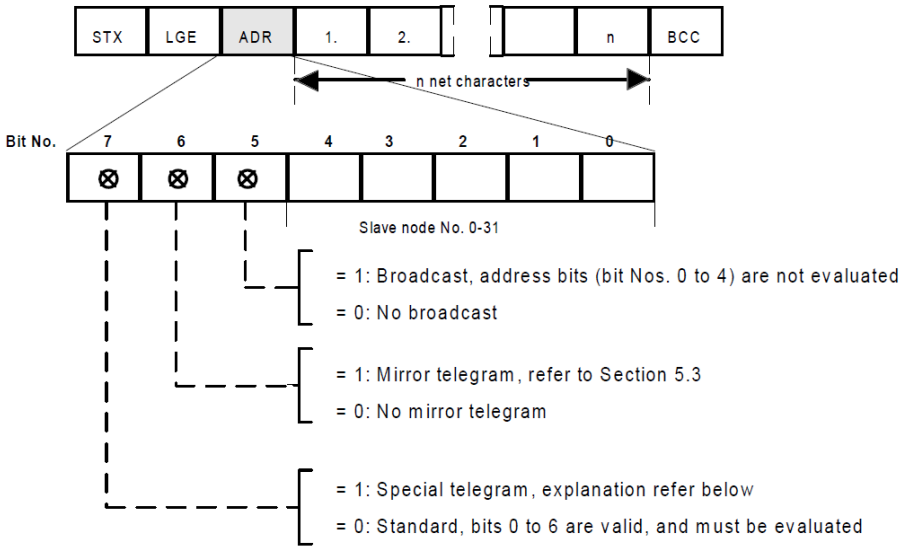
2.2.2 LGE byte

V druhém bytu telegramu se nachází LGE část, ve které se definuje délka přenášené zprávy. Hodnota délky je vypočítaná pomocí vzorce 2.1 z počtu bajtů, které budou následovat za LGE. Obsahuje velikost ADR, BCC a net characters části, která je ve vzorci znázorněna proměnou n. Skutečná délka celkového telegramu je o dva byty delší, jelikož se do výpočtu nezahrnuje velikost STX a LGE. Maximální celková délka zprávy může tedy být 256 bytů a největší možná délka net charakters je 252 bytů. V závislosti na konfiguraci může být délka telegramu pevná nebo proměnná.

$$LGE = n + 2 \quad \{1 \leq LGE \leq 254\} \tag{2.1}$$

2.2.3 ADR byte

Každý telegram obsahuje bajt ADR, který obsahuje adresu frekvenčního měniče na 0. až 4. bitu (viz Obrázek 2.3). Hodnota adresy je limitovaná maximálním počtem připojených zařízení, takže může nabývat hodnot v mezích 0 až 31. Nastavením 5. bitu na logickou 1 je možné zaslat zprávu všem měničům na sériové sběrnici. Pro využití zrcadlového telegramu je nastaven 6. bit na logickou 1. Adresovaný měnič přijatou zprávu pošle nezměněnou zpět do řídicího systému. Zrcadlená zpráva je prospěšná pro otestování funkčnosti komunikace a případné hledání závady. Posledním 7. bitem lze nastavit telegram jako speciální. Takový telegram se používá pro speciální aplikace, které vyžadují odlišnou strukturu net charakters.



Obrázek 2.3: Struktura adresového bajtu ADR; zdroj [7]

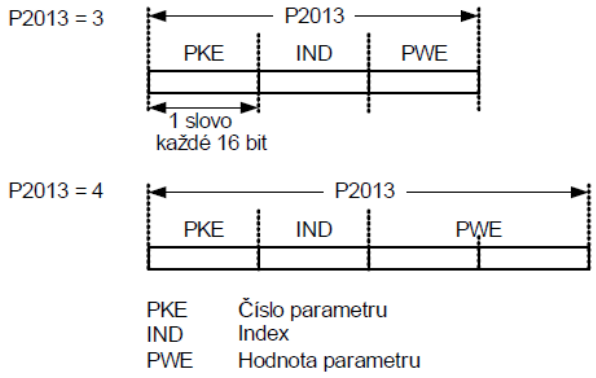
2.2.4 PKW část

Oblast PKW se uplatňuje pro parametrizaci, servis a diagnostiku měniče. Neobsahuje fyzické rozhraní, ale slouží k definici mechanismu, který zpracovává přenos parametrů mezi dvěma komunikačními partnery. Umožňuje tedy číst a zapisovat hodnoty parametru, obsahuje jejich definici a zajišťuje informaci o změně a použití parametru. PKW oblast se dále dělí na tři části PKE, IND a PWE. Detailní popis jednotlivých částí a jejich tvorba se nachází v dokumentu *Universal Serial Interface Protocol* [7, str. C - 10 až C - 24].

ID parametru (PKE) má vždy délku 1 word (2 byte) a využívá se pro identifikaci a uvedení požadavku/odpovědi pro zpracovaní parametru. Obsahuje číslo parametru.

Index (IND) je oblast o délce jednoho slova (word). Spousta parametrů měniče využívá pro detailnější nastavení indexy. Slouží tedy k zápisu nebo čtení textu, popisu parametru a hodnot, které jsou tvořeny polem.

Hodnota parametru (PWE) obsahuje samotnou hodnotu, text nebo popis parametru, který je požadován nebo přijat v odpovědi jako aktuální hodnota. Může mít délku 1 word, 2 wordy nebo variabilní délku a musí být předem definovaná v parametru měniče P2013 (viz Obrázek 2.4).

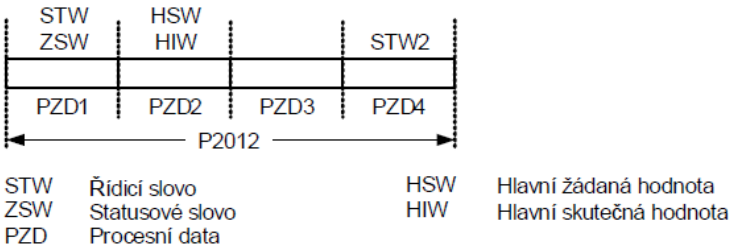


Obrázek 2.4: Struktura PKW o délce 3 a 4 word; zdroj [12]

2.2.5 PZD část

Procesní data, neboli PZD oblast slouží ke spojitému přenosu dat mezi PLC a měničem frekvence. Obsahuje signály potřebné pro automatizaci a může mít konstantní velikost 0 až 16 wordů. Tato délka musí být předem definovaná v parametru měniče P2012. Struktura je vždy stejná, liší se pouze počtem přenesených žádaných nebo skutečných hodnot. Tato část USS protokolu je stěžejní pro ovládání frekvenčního měniče a tedy i pro realizaci této bakalářské práce.

V závislosti na směru přenosu dat je řídicí slovo (Control word) nebo stavové slovo (Status word) vždy přenášené v PZD1. Hlavní žádaná hodnota (Main setpoint) nebo hlavní skutečná hodnota (Actual value) se vždy přenáší v PZD2. Je-li délka oblasti rovna 4, přenáší se další řídicí slovo jako čtvrté slovo v PZD4 (viz Obrázek 2.5). Další části PZD mohou obsahovat doplňkové hodnoty ovládání pro řízení vyšších řad frekvenčních měničů.



Obrázek 2.5: Struktura procesních dat PZD; zdroj [12]

Řídicí slovo slouží k ovládání pohonu a nastavuje se pomocí jednotlivých bitů wordu. Význam jednotlivých bitů je vysvětlen v tabulce 2.1. Stavové slovo informuje o stavu pohonu a může být použito pro diagnostiku. Smysl dílčích bitů stavu je vysvětlen v tabulce 2.2.

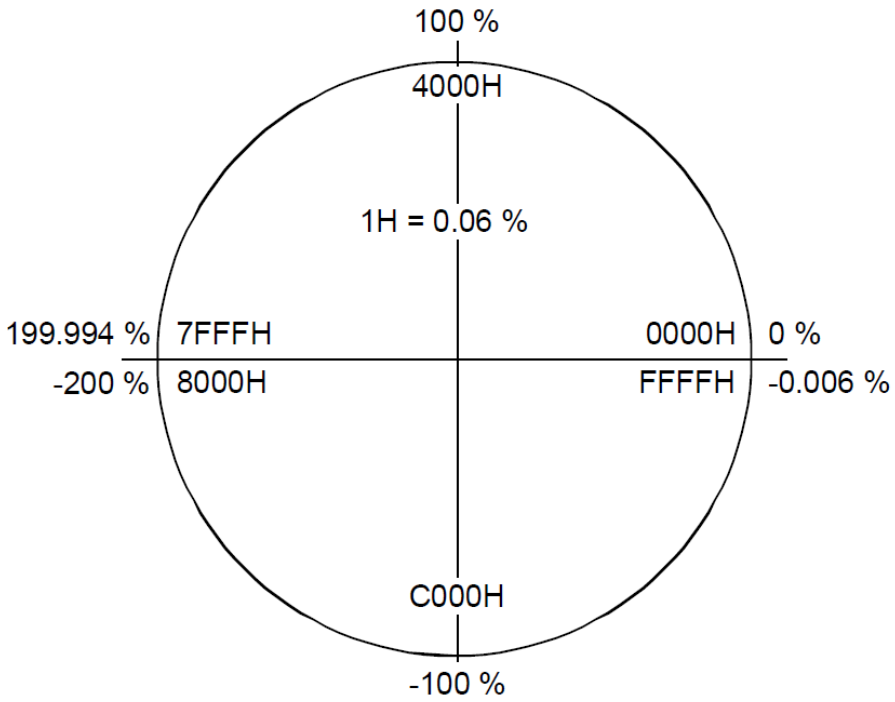
Tabulka 2.1: Řídicí slovo v bitovém formátu; zdroj [12]

Bit	Popis	Hodnota	
0	ON/OFF	0 NE	1 ANO
1	OFF2: Elektrické vypnutí	0 ANO	1 NE
2	OFF3: Rychlé vypnutí	0 ANO	1 NE
3	Povolení pulsu	0 NE	1 ANO
4	Zapnutí RFG	0 NE	1 ANO
5	Start RFG	0 NE	1 ANO
6	Zapnutí žádané hodnoty	0 NE	1 ANO
7	Nulování poruchy	0 NE	1 ANO
8	Krokování vpravo	0 NE	1 ANO
9	Krokování vlevo	0 NE	1 ANO
10	Požadavek řízení z řídicího systému	0 NE	1 ANO
11	Reverzace (změna chodu otáčení)	0 NE	1 ANO
12	Volný bit	-	-
13	Motorpotenciometr zvýšit	0 NE	1 ANO
14	Motorpotenciometr snížit	0 NE	1 ANO
15	Místní ovládání / Dálkové ovládání	0 NE	1 ANO

Tabulka 2.2: Stavové slovo v bitovém formátu; zdroj [12]

Bit	Popis	Hodnota	
0	Připraven k provozu	0 NE	1 ANO
1	Připraven k zapnutí	0 NE	1 ANO
2	Chod motoru	0 NE	1 ANO
3	Porucha	0 NE	1 ANO
4	OFF2	0 ANO	1 NE
5	OFF3	0 ANO	1 NE
6	Blokování zapnutí	0 NE	1 ANO
7	Výstraha	0 NE	1 ANO
8	Odchylka skutečné hodnoty otáček	0 ANO	1 NE
9	Požadavek řízení z řídicího systému	0 NE	1 ANO
10	f_act >= P1082 (f_max)	0 NE	1 ANO
11	Upozornění: Proudové omezení	0 ANO	1 NE
12	Brzda motoru aktivní	0 NE	1 ANO
13	Přetížení motoru	0 ANO	1 NE
14	Směr otáčení vpravo	0 NE	1 ANO
15	Přetížení měniče	0 ANO	1 NE

Hodnota části PZD2 udává žádanou výstupní frekvenci (popř. otáčky) pohonu vyjádřenou v procentech. Hodnota je vztažena k takzvanému referenčnímu kmitočtu. Referenční kmitočet musí být předem definován v parametru měniče P2000. Rozsah procent je -200% až +199,994%, takže je možné dosáhnout reverzace pohonu zadáním záporné hodnoty. Z obrázku 2.6 lze vidět, že každá procentuální hodnota odpovídá hodnotě v hexadecimálním tvaru. Proto je nutné provést přepočet pro kladný směr podle vzorce 2.2, který je potřebný pro správný chod protokolu. Procentuální poměr frekvencí vydělíme hodnotou odpovídající 1 hex, která souhlasí s hodnotou 0,006%. Pro záporné frekvence se pouze posuneme o dva kvadranty přičtením hodnoty 7FFF hex (viz vzorec 2.3). Vypočtená požadovaná rychlost je omezena v obou směrech otáčení motoru maximálním povoleným kmitočtem měniče. Tuto hodnotu je možno nastavit v parametru měniče P1082.



Obrázek 2.6: Rozsah hodnot z procent na HEX; zdroj [11]

$$\mathrm{PZD2}_{\text{kladná}}[\mathrm{HEX}] = \frac{f_{\text{žádaná}}[Hz]}{f_{\text{referenční}}[Hz]} \cdot \frac{100}{0{,}006} \tag{2.2}$$

$$\mathrm{PZD2}_{\text{záporná}}[\mathrm{HEX}] = \frac{f_{\text{žádaná}}[Hz]}{f_{\text{referenční}}[Hz]} \cdot \frac{100}{0{,}006} + 7\mathrm{FFF} \tag{2.3}$$

2.2.6 BCC byte

Kontrolní součet (BCC) se nachází na samotném konci telegramu a potvrzuje konec zprávy. Slouží k zabezpečení a k ověření správně přijatých nebo odeslaných dat. Má velikost 1 byte a jeho hodnotu vypočítáme součtem XOR všech předchozích bajtů obsažených v telegramu. Při přijímání zprávy se vypočte kontrolní součet a porovná se s přijatým kontrolním součtem. Jakmile se vypočtená hodnota neshoduje s přijatou, telegram se nevyhodnotí.

Před přijmutím prvního bajtu telegramu je hodnota BCC rovna 0. Po přijetí prvního bajtu STX dochází k součtu XOR s BCC. Každým dalším přijatým bajtem dochází k součtu XOR předchozího BCC_{old} s nově přijatým bajtem a vniká BCC_{new}. Konečná hodnota kontrolního součtu je vypočtena po posledním přijatém net character bajtu. Pro bližší představu by začátek výpočtu vypadal následovně.

$$
\begin{array}{rcr}
\text{BCC} & = & 0000\ 0000 \\
 & & \downarrow \\
\text{BCC}_{\text{old}} & = & 0000\ 0000 \\
 & & \text{XOR} \\
\text{STX} & = & 0000\ 0010 \\
\hline
\text{BCC}_{\text{new}} & = & 0000\ 0010 \\
 & & \downarrow \\
\text{BCC}_{\text{old}} & = & 0000\ 0010 \\
 & & \text{XOR} \\
\text{LGE} & = & 1101\ 0110 \\
\hline
\text{BCC}_{\text{new}} & = & 1101\ 0100 \\
 & & \downarrow \\
 & & \ldots
\end{array}
$$

2.3 Druhy přenosu

Přenos telegramu se dělí na cyklický a necyklický. V oblasti řízení pohonů se používá především cyklický přenos. Řídicí systém je zodpovědný za celý průběh řízení přenosu a postupně komunikuje v daných časových intervalech se všemi adresovanými měniči frekvence.

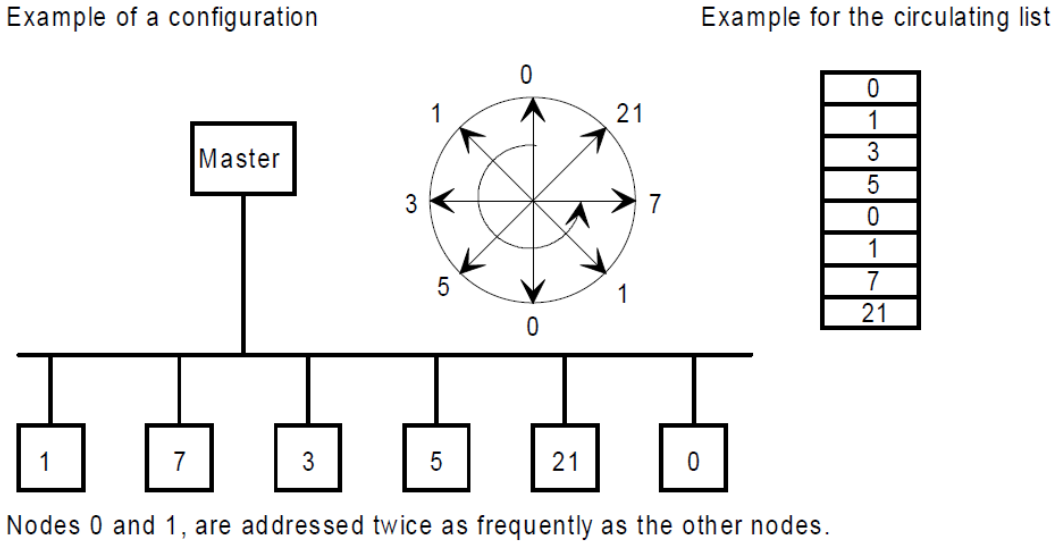
Při cyklickém přenosu telegramu řídicí systém průběžně vysílá telegramy jednotlivým měničům frekvence a čeká na odpověď v podobě telegramu. Měnič kmitočtu musí poslat telegram s odpovědí, pokud byl určený pro jeho adresu a obdržel telegram bez chyb. Naopak nesmí poslat odpovědět v případě nesplnění dvou předchozích podmínek nebo když obdrží telegram, který je určený všem jako broadcast. Pro řídicí systém byla komunikace úspěšná v případě, kdy obdrží telegram s odpovědí od žádané adresy ve stanoveném čase. Tímto lze snadno sledovat selhání přenosu telegramu.

Necyklický přenos dat nelze používat současně s cyklickým přenosem dat. Řídicí systém v režimu necyklického přenosu dat odesílá telegramy v nepravidelných intervalech. Měniče kmitočtu reagují stejně jako v cyklickém přenosu dat. V tomto režimu nelze sledovat selhání přenosu telegramu.

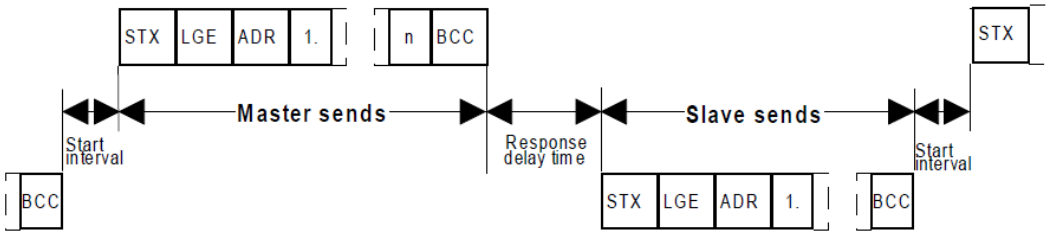
2.4 Postup přenosu dat

Jak bylo výše psáno, pro řízení pohonů v automatizovaných provozech se používá cyklický přenos. Při komunikaci pouze s jedním měničem frekvence dochází k neustálému odesílání telegramu a čekání na odpověď. Jestliže se na sběrnici nachází dva a více zařízení s požadavkem řízení, použijeme kruhový seznam adresovaných frekvenčních měničů (viz Obrázek 2.7). V tomto seznamu jsou uloženy adresy se kterými řídicí systém postupně v daném pořadí komunikuje. Pokud je požadavek k řízení pohonu v rychlejším cyklu, může se adresa tohoto frekvenčního měniče vyskytnout v seznamu několikrát. Tento případ vidíme na obrázku 2.7 s adresami 0 a 1.

Startovní znak STX sám o sobě nestačí k jasné identifikaci začátku telegramu, protože hodnota 02 hex se může vyskytovat i v jiných částech zprávy. Z tohoto důvodu dochází vždy ke zpoždění startu minimálně o délku dvou bajtů. Čas zpoždění závisí na přenosové rychlosti, takže například pro rychlost 19200bit/s je minimální čas 1,15ms. Zpoždění doby odezvy musí být minimálně stejně dlouhé jako zpoždění startu a maximálně 20ms. Pokud řídicí systém nepřijme odpověď v požadovaném čase, uloží se tato událost do chybového hlášení a pokračuje v komunikaci s dalším frekvenčním měničem. Přenos dat je tedy realizován v poloduplexním režimu (viz Obrázek 2.8).



Obrázek 2.7: Kruhový seznam adresovaných frekvenčních měničů; zdroj [7]



Obrázek 2.8: Sekvence přenosu dat; zdroj [7]

2.5 Monitorovací mechanizmy a reakce na chyby

Když je telegram přijat, musí se nejprve identifikovat správný začátek zprávy STX byte a poté se vyhodnotí délka LGE byte. Telegram je odmítnut, pokud informace o délce neodpovídá zvolené hodnotě pro pevnou nebo proměnnou délku telegramu. Jestliže se bajt ADR byte neshoduje s očekávanou adresou měniče (master) nebo nesouhlasí adresa měniče (slave), tak je telegram zamítnut. Kontrolní součet BCC byte se generuje během přijmu dat. Po přečtení celého telegramu se porovná vypočtená hodnota kontrolního součtu s přijatou a pokud se neshodují, tak se telegram nevyhodnocuje. Pokud se v žádném z přijatých znaků nevyskytuje chyba rámce nebo parity, tak se přijatý telegram vyhodnotí pro danou adresu. Souběžně s vyhodnocováním telegramu se musí sledovat časy doby zpoždění před přijetím telegramu a doba trvání odezvy příjmu telegramu. Softwarové rozhraní měniče může poskytovat informace o stavu komunikace a zaznamenaných chybách. Diagnostické informace by měly být zobrazeny na ovládacím panelu obsluhy měniče.

2.6 Příklad ovládacího telegramu

Zde je uveden příklad master telegramu, který lze použít pro počáteční testování komunikace.

STX = 02 hex

LGE = 6 bajtů → 06 hex
PKW oblast bude mít nulovou velikost.
PZD oblast bude mít velikost 2 word (4 byte).
Délka přenášené zprávy se vypočítá podle vzorce 2.1.

ADR = 00 hex
Oblast ADR je bez speciálního nastavení a obsahuje pouze adresu měniče.

PZD1 = 0000 0100 0111 1111 bin = 04 7F hex
Řídicí slovo (Control Word) pro uvedení pohonu do provozu se vytvoří dle tabulky 2.1, kde nastavíme jednotlivé bity.

PZD2 = 50% → 20 00 hex
Pro požadovanou frekvenci 25Hz při referenčním kmitočtu 50Hz je procentuální rychlost 50%.
Přepočet na hexadecimální tvar se provede dle vzorce 2.2.

BCC = 02 XOR 06 XOR 04 XOR 7F XOR 20 = 5F hex
Provádí se součet XOR všech nenulových částí telegramu.

Výsledný odesílaný ovládací telegram:

{02 06 00 04 7F 20 00 5F} hex

Možná varianta odpovědi přijatého telegramu:

{02 06 00 FB 34 20 00 EB} hex

3 Testovací konfigurace

Jako testovací konfigurace byl zvolen řídicí systém firmy B&R-Automation vybavený sériovou linkou RS232. Napájení PLC řeší kompaktní 24V spínaný zdroj. Následně byl vybrán jednoduchý frekvenční měnič od firmy Siemens s ovládacím panelem a přídavným panelem RS232.

3.1 Řídicí systém

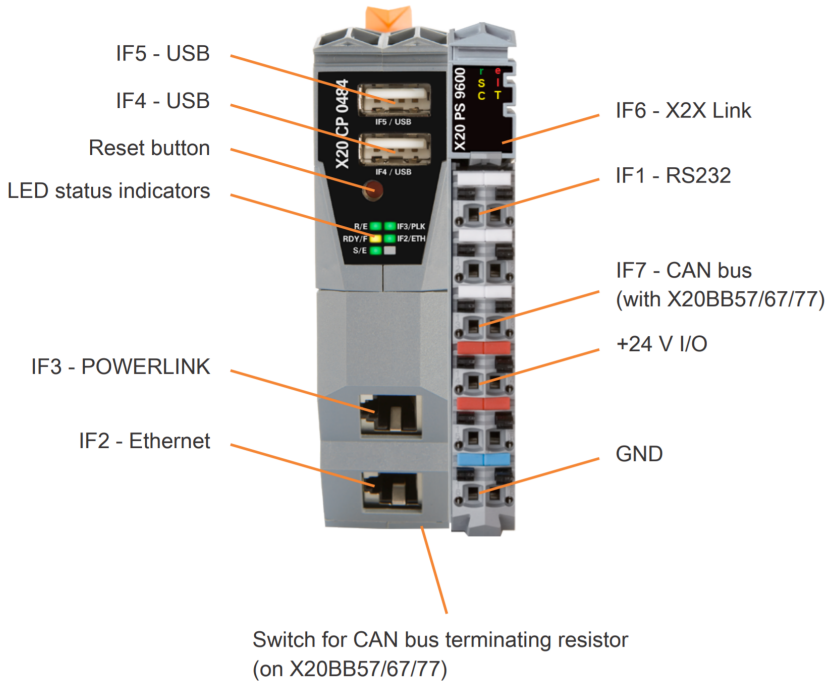
Všechny produktové řady PLC systémů společnosti B&R-Automation běží na společné softwarové platformě Automation Runtime. Jedná se o deterministický operační systém v reálném čase, který umožňuje vytvořit program nezávisle na hardwarové konfiguraci. Tato vlastnost poskytuje jednoduchý přechod na jinou produktovou řadu programovatelných automatů s tímto operačním systémem. Při změně konfigurace je tedy nutné změnit pouze hardwarovou konfiguraci v prostředí Automation Studio a provést nové sestavení programu.

3.1.1 Programovatelný automat - B&R X20CP0484

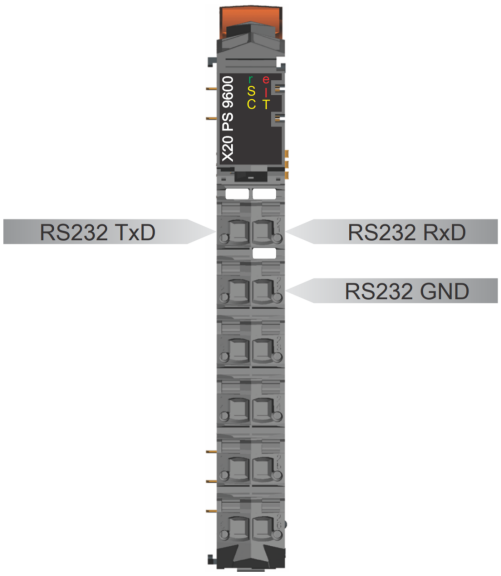
Pro testování v této bakalářské práce byl zvolen kompaktní model X20CP0484 z řady systémů X20 Compact-S. Tento model disponuje výkonným procesorem ARM Cortex A9 s frekvencí 667MHz, pamětí RAM o velikosti 256 MB a 2GB flesh pamětí pro nahrané programy. Na samotné řídicí jednotce se nachází dva USB porty, Ethernet 10/100Mb/s a POWERLINK (viz Obrázek 3.1). Pro diagnostiku jsou na předním straně stavové LED diody, které signalizují napájení, mód a funkci CPU, komunikaci POWERLINK a Ethernetu.

3.1.2 Napájecí modul - B&R X20PS9600

Do napájecího modulu X20PS9600 je přivedeno 24V stejnosměrných z externího zdroje napětí. Tento modul napájí samotný programovatelný automat a X2X linku, která distribuuje napájení a komunikaci mezi jednotlivými přídavnými moduly. Díky technologii X2X link se moduly nemusí mezi sebou propojovat vodiči. Na kartě se nachází svorkovnice pro připojení sběrnice CAN a RS232 (viz Obrázek 3.1). Zapojení komunikace po sériové lince RS232 je zobrazeno na obrázku 3.2. V horní části se opět nachází stavové LED diody s indikací stavu napájení, komunikaci RS232 a sběrnici CAN.



Obrázek 3.1: Automat X20CP0484 s napájecím modulem X20PS9600; zdroj [2]



Obrázek 3.2: Zapojení RS232 na napájecím modulu B&R X20PS9600; zdroj [3]

3.1.3 Sběrnice - B&R X20BB67

Do sběrnice X20BB67 je usazen programovatelný automat společně s napájecím modulem. Sběrnici je možné rozšířit o další pozice a slouží jako distributor napájení a komunikace pro přídavné karty pomocí technologie X2X link.

3.2 Frekvenční měnič

Představitelem jednoduchých, kompaktních a cenově dostupných měničů firmy Siemens je řada Sinamics G110. Jednofázový vstup je usměrněn na diodovém usměrňovači a následně v meziobvodu vyfiltrován od vyšších harmonických filtračním kondenzátorem. Třífázový výstup je řízen mikroprocesorem, který spíná bipolární tranzistory s izolovaným hradlem (IGBT). Výstupní frekvence může dosáhnout až 650Hz. Obsahuje základní funkce pro regulaci otáček asynchronních motorů pomocí skalárního řízení. Využívá se v rozmanitých průmyslových aplikacích s výkony 0,12 až 3kW. Vyrábí se ve verzi s analogovým řízením a univerzálním sériovým rozhraním USS využívající standard RS485. V současné době je tato řada nahrazena modernější vybavenější řadou Sinamics V20.

Obrázek 3.3: Siemens Sinamics G110; zdroj [4]

3.2.1 Ovládací panel

Jedná se o způsob manuálního zadávání hodnost přes panel s alfanumerickým displejem a tlačítky (viz Obrázek 3.4). Umožňuje snadný přístup k parametrům měniče a k jeho ovládání. Tento způsob však není příliš praktický a slouží k základnímu uvedení do provozu.

Obrázek 3.4: Siemens - Ovládací panel; zdroj [9]

3.2.2 Svorkovnice

Obsahuje analogové a digitální vstupy a výstupy, na které lze připojit externí ovládací prvky. Analogovým vstupem s připojeným potenciometrem nastavujeme žádané otáčky motoru. Digitální vstupy se využívají pro vyvolání předem nastavených požadovaných funkcí (např. reverzace, start/stop,...) pomocí sepnutí spínačů.

3.2.3 Sériová komunikace

K řízení a parametrizování měniče pomocí sériové komunikace je potřeba nadřazený řídicí systém s podporou sériové komunikace USS po sběrnici RS232 nebo RS485. Nejčastěji se jedná o PC nebo PLC. Verze měniče s analogovým řízením požaduje k sériové komunikaci přídavný panel s RS232, který je vidět na obrázku 3.5.

Obrázek 3.5: Siemens - Panel s RS232; zdroj [6]

3.3 Napájení

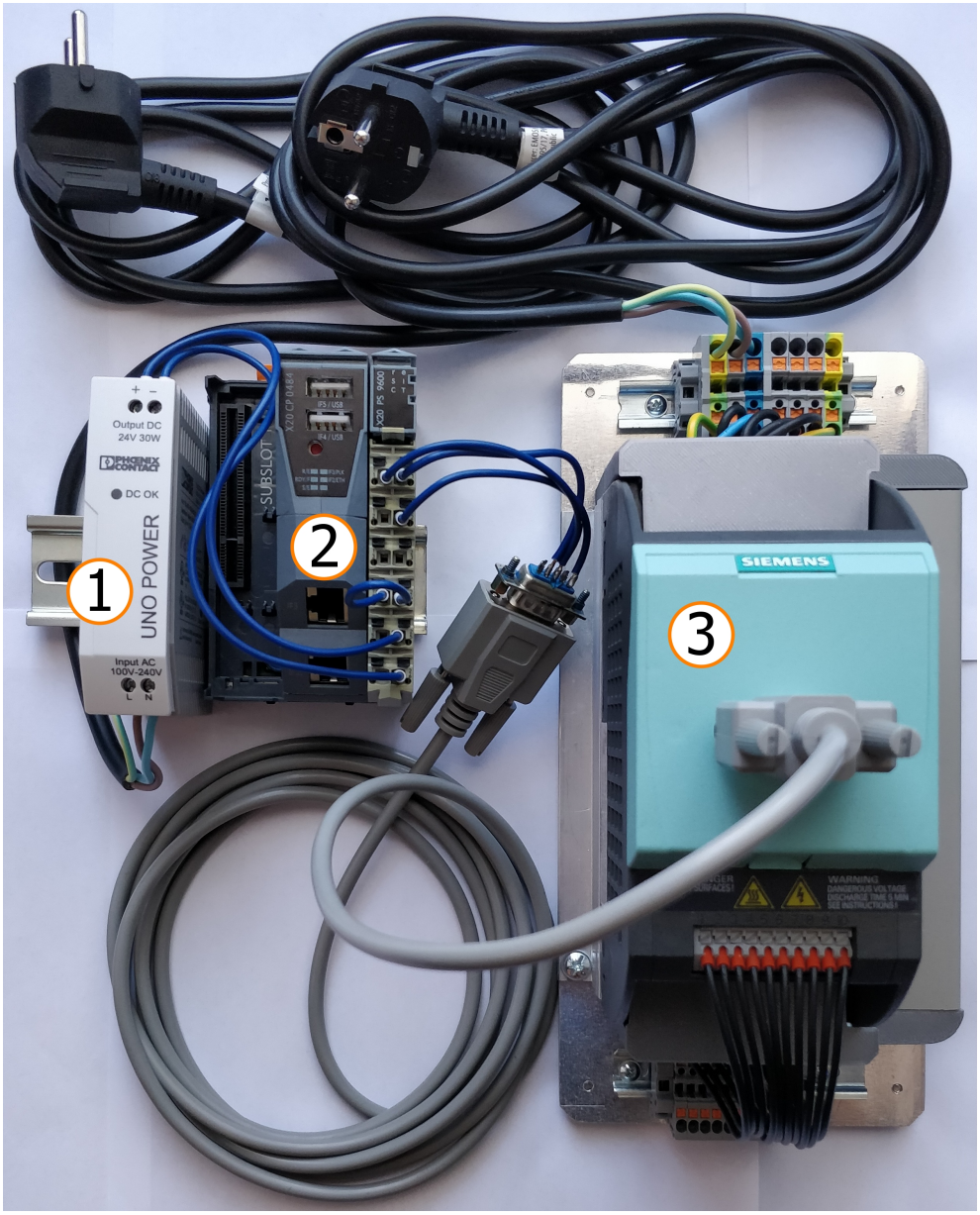
O napájení PLC se stará jednoduchý spínaný zdroj od firmy PHOENIX CONTACT. Jednofázový vstup zvládne s rozsahem napětí 85 až 264V na výstup přivést 24V stejnosměrných s maximálním proudem 1,25A. Zdroj díky vysokému stupni účinnosti a nízkým ztrátám při chodu na prázdno dosahuje vysoké energetické účinnosti. Konstrukce je uzpůsobena pro montáž na DIN lištu a vzhledem k malým rozměrům se jedná o ideální řešení v kompaktních rozvaděčích.

3.4 Převodník USB-RS232

Pro jednoduché počáteční testování USS protokolu, byl použitý PC k odesílání a přijímaní telegramů. Jelikož konektor RS232 již není běžnou součástí stolních ani přenosných počítačů, tak bylo nutné použít USB převodník. Jedná se o převodník firmy ASIX s produktovým označením UCAB232. Obsahuje dvě indikační diody, které signalizují příchozí a odchozí stav komunikace po sériové lince.

3.5 Zapojení

Testovací konfigurace je rozdělena na dvě části, aby byla lépe přenositelná a kompaktní. Frekvenční měnič a řídicí systém spojuje propojovací sériový kabel s konektory CAN9 (female).



Obrázek 3.6: Zapojení testovací konfigurace

Legenda

1. Napájení
2. Řídicí systém
3. Frekvenční měnič

4 Programové vybavení

4.1 Automation Studio

Automation Studio je vývojové prostředí projektů, které podporuje operační systém Microsoft Windows. Používá se především pro produkty automatizace společnosti Bernecker & Rainer, mezi které patří odvětví řízení, HMI vizualizace, řízení pohonů a komunikace. Všechny tyto odvětví lze konfigurovat v jednom prostředí v jediném projektu. Uživatelé si mohou vybrat ze široké škály programovacích jazyků, diagnostických nástrojů a editorů.

4.1.1 Programovací jazyky

Vývojové prostředí podporuje základní programovací režimy standardu normy IEC 61131-3. Tato norma zahrnuje dva textové a tři grafické programovací jazyky. Vývojové prostředí navíc umožňuje využít programovací jazyky ANSI C nebo C++, grafický programovací jazyk Continuous Function Chart (CFC) a programovací jazyk Automation Basic (AB) vyvinutý společností B&R-Automation.

Program se dělí na tři části. Inicializační část, která se provede pouze při prvním spuštění programu nebo při restartu. Cyklická část se spustí po dokončení inicializace a vykonává se cyklicky po definovaném časovém intervalu. Ukončovací část se vyvolá pouze při odstraňování programu.

4.1.2 Datové typy

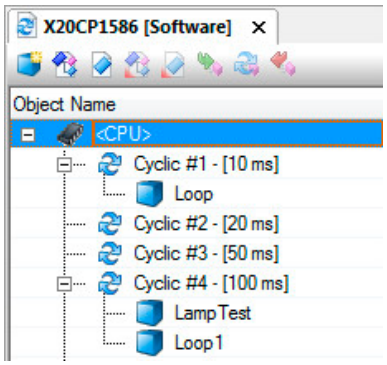
Datové typy určují vlastnosti proměnných, které definují rozsah a přesnost hodnot nebo možné operace. V Automation Studio si uživatel může definovat vlastní datové typy. Složením více datových typů vznikne datová struktura umožňující přehlednost proměnných.

4.1.3 Knihovny

Standardní knihovny v prostředí Automation Studio dodávají uživatelům přístup k již vytvořeným funkcím, které usnadňují tvorbu programu. Obsahují široké spektrum funkcí od jednoduchých logických operací až po složité řídicí algoritmy. Uživatel má možnost si definovat knihovnu do které si vytváří vlastní funkční bloky.

4.1.4 Cyklické třídy úloh

Každému programu je přiřazena určitá doba provádění nebo třída úloh v konfiguraci softwaru, které jsou identifikovány jako úlohy. Tyto úlohy se vykonávají periodicky v čase definovaném třídou úloh postupně za sebou. Třída úloh poskytuje až osm konfigurovatelných tříd, které pomáhají optimalizovat program pro konkrétní požadavky. Každá třída se vykonává se stejnou dobu cyklu, prioritou a tolerancí. Priorita je určena číslem třídy úkolů (viz Obrázek 4.1). Čím nižší je číslo, tím vyšší je priorita třídy. Doba provedení úlohy není ovlivněna přesunutím do jiné třídy. Změna třídy mění pouze počet volání úlohy za danou periodu.



Obrázek 4.1: Cyklické třídy úloh; zdroj [15]

4.1.5 Diagnostické nástroje

Mezi nejužitečnější funkce Automation Studia patří simulace, která umožňuje rychlé a snadné testování programů. Simulace probíhá na virtualizovaným prostředí Automation Runtime, který neprobíhá v reálném čase, ale odpovídá funkčností všech ostatních cílových systémů.

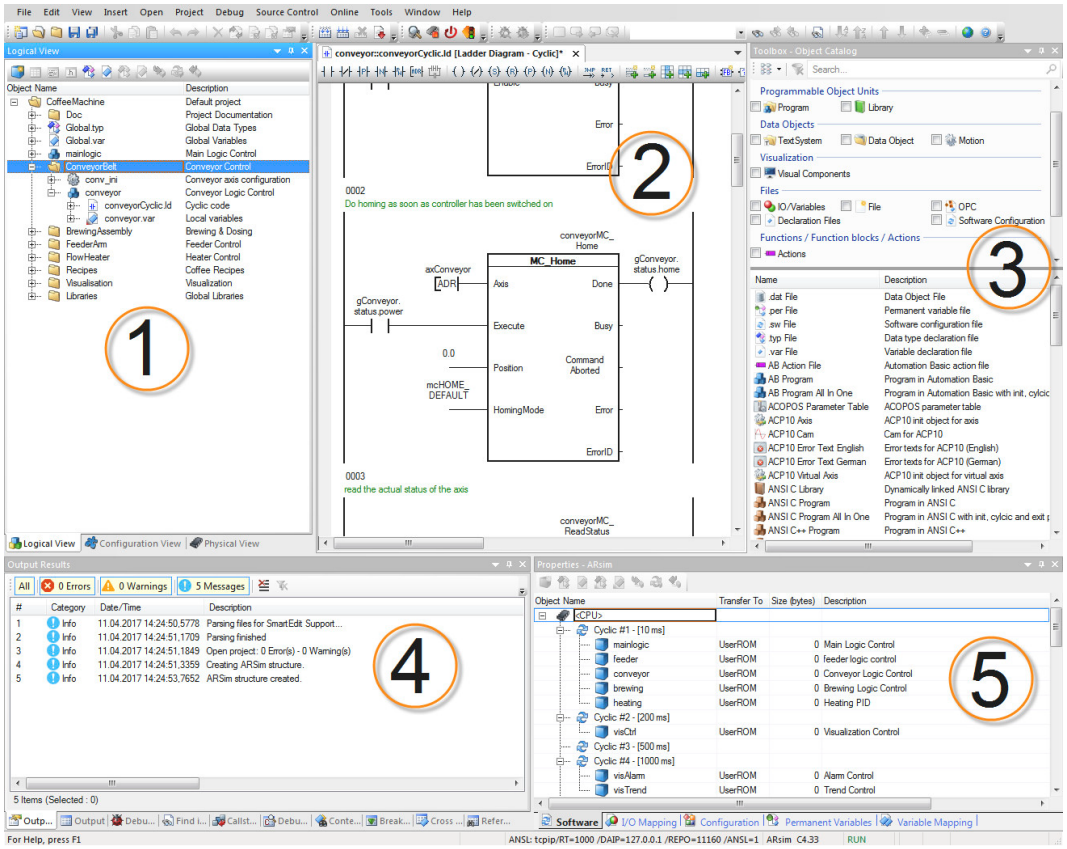
Dalším užitečným diagnostickým nástrojem je režim Watch a Monitor. Tyto režimy zobrazuji aktuální hodnotu vybraných proměnných, jejich typ a umístění v projektu a paměti. V editačním okně umožňují změnu hodnoty za běhu systému.

Pro zaznamenání rychlých změn hodnot proměnných je nutné použít nástroj Trace. Zaznamenané hodnoty zobrazí v grafu, kde na ose x je čas a na ose y je hodnota proměnné. Analyzováním průběhu lze detekovat chyby a optimalizovat procesy v programu.

Debugger je důležitý nástroj pro programátory, který usnadňuje hledání chyb ve zdrojovém kódu. Při překladu projektu informuje o nalezených chybách v okně zpráv, kde se zobrazí konkrétní řádek s chybou a popis. Umožňuje krokování programu po jednotlivých řádcích s paralelním monitorováním hodnot.

4.1.6 Struktura prostředí

Prostředí se skládá z několika částí. V horní části se nachází panel s nástroji a obsluhou prostředí. Na levé straně je umístěn průzkumník projektu, který se používá pro správu a editaci objektů v projektu. Uprostřed obrazovky se nachází samotný pracovní prostor například pro editor programu. V pravé části je panel s objekty pomocí kterého se do projektu přidávají funkce programu, softwarové objekty nebo hardwarové moduly. Okno zpráv zaujímá pozici v levé dolní části, kde se zobrazují informace o stavu sestavování projektu. Možnosti konfigurace pro aktuálně vybraný objekt nebo hardwarový modul se zobrazuje vpravo dole. Dolní okraj Automation Studia informuje o stavu a typu komunikace s daným typem procesoru.



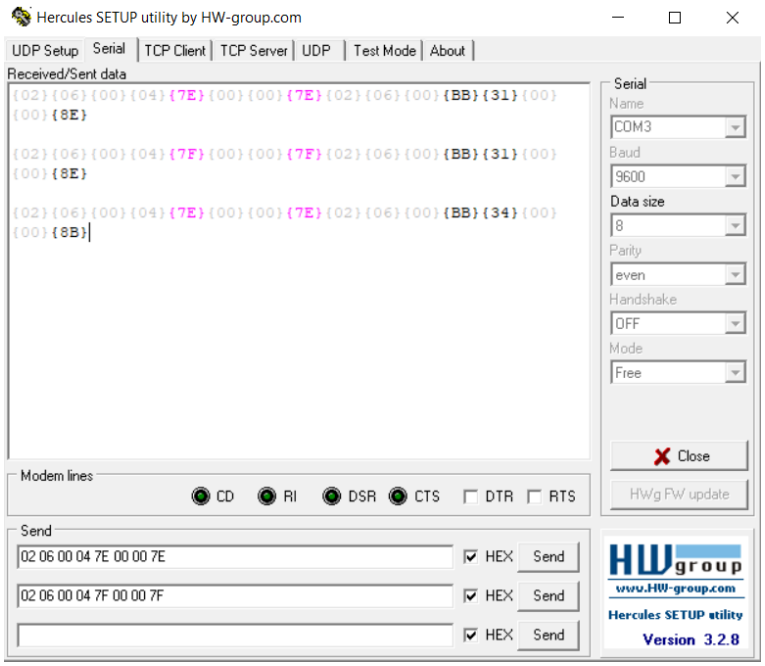
Obrázek 4.2: Struktura prostředí Automation Studio; zdroj [15]

Legenda

1. Průzkumník projektu
2. Pracovní prostor
3. Panel s objekty (Toolbox)
4. Okno zpráv
5. Okno vlastností

4.2 Hercules SETUP

Tato jednoduchá bezplatná aplikace od společnosti HW group poskytuje spoustu funkcí, mezi které patří i terminál pro obsluhu sériového portu s rozhraním RS232 nebo RS485. Tato funkce sloužila k testování komunikace s frekvenčním měničem. Ten byl připojen s PC pomocí převodníku USB-RS232 k měniči s panelem RS232. Rozhraní aplikace a výsledek prvního testování komunikace USS protokolu je zobrazen na obrázku 4.3.



Obrázek 4.3: Hercules SETUP; zdroj [1]

5 Tvorba programu

Po úspěšném testování ručně vytvořeného telegramu pomocí aplikace Hercules SETUP bylo zapotřebí vhodně navrhnout datovou strukturu a program v prostředí Automation Studio. Pro snadné a rychlé naprogramování byl zvolen programovací jazyk strukturovaného textu (ST). Celý vývoj programu probíhal na testovací konfiguraci popsané v kapitole 3.

Nejprve bylo nutné navrhnout část programu, která bude obstarávat samotnou tvorbu odesílaného telegramu. Pro získání telegramu s odpovědí na odeslaný telegram byla vytvořena část komunikace, která obsluhuje sériový port a samotné posílání a přijímání telegramu. Zpracování přijatého telegramu zajišťuje samostatná část programu. Poslední částí návrhu byl program, který bude mít na starost řízení všech těchto částí programu a obsluhu komunikace s více měniči.

Dílčí programy neboli funkční bloky představují určitý proces s jednoduchým úkolem. V následujícím textu jsou pojmy funkce popřípadě funkční blok používány pro pojmenování určité části kódu v rámci programu. Nepředstavují zde funkce ani funkční bloky ve smyslu normy IEC61131. Vytváření funkčního bloku je inspirováno principem stavových automatů a je realizován pomocí instrukce *case*. Datové struktury proměnných slouží k uložení hodnot, které jsou využity jako vstupní nebo výstupní data pro funkční bloky. Takováto struktura funkčních bloků a datových struktur napomáhá k jednodušší správě kódu, rozšiřitelnosti a případné implementaci do jiných projektů.

Výsledný program pro ovládání frekvenčních měničů Siemens pomocí USS protokolu je vytvořen zcela univerzálně. Podmínka pro využití programu je řídicí systém se sériovým portem od firmy B&R-Automation a frekvenční měnič s podporou Siemens USS protokol. Program splňuje všechny náležitosti, které USS protokol vyžaduje k ovládání. Jelikož parametrizační část telegramu není vyžadovaná k ovládání, tak se její tvorba nezahrnuje do programu. Datové struktury jsou připravené pro případnou budoucí implementaci funkčního bloku, který bude řešit PKW část telegramu.

5.1 Struktura programu

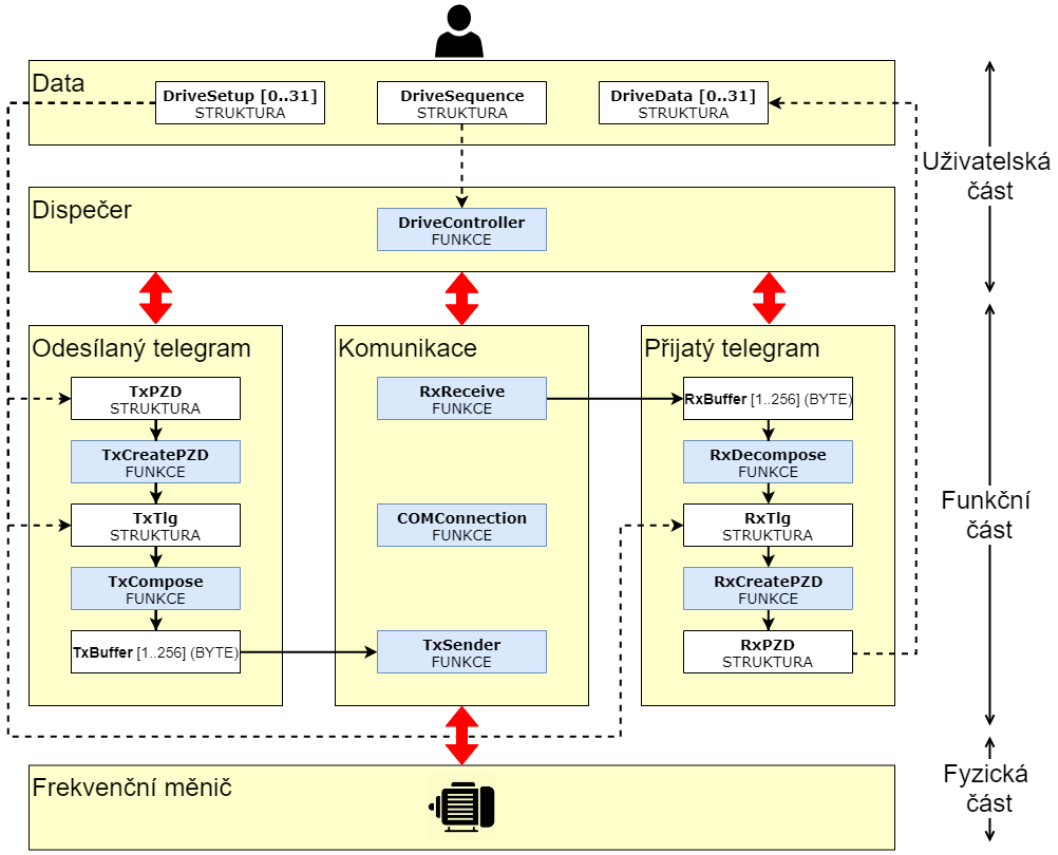
Navržená struktura programu se dělí do dvou částí, uživatelské a funkční. Nadřazená uživatelská část ovládá celý proces probíhající ve funkční části a poskytuje veškeré potřebné vstupní data, která jsou předem definovaná uživatelem. Současně slouží i k uchovávání výstupních dat z funkční části. Funkční část se skládá z tří úloh obsahující funkční bloky pro sestavení, odeslání, přijetí a následné zpracování telegramu. Fyzická část představuje samotný frekvenční měnič s připojeným elektrickým motorem, který komunikuje s funkční částí. Pro rychlé uvedení měniče do provozu je zapotřebí nastavit potřebné štítkové hodnoty motoru, minimální a maximální kmitočet, doba rozběhu a doběhu motoru a režim řízení. Postup nastavení popisuje dokument *SINAMICS G110*: Návod k obsluze - stručný [14, str. 15 až 17].

Datové bloky a funkční bloky jsou vytvořeny pomocí strukturovaných proměnných typů, které obsahují jednoduché proměnné nebo další strukturu proměnných. Například funkční blok se skládá ze struktury vstupních příkazů ***Cmd***, struktury výstupních informací o stavu ***Status*** a struktury interních proměnných ***Ints*** nebo funkcí ***Fcn***.

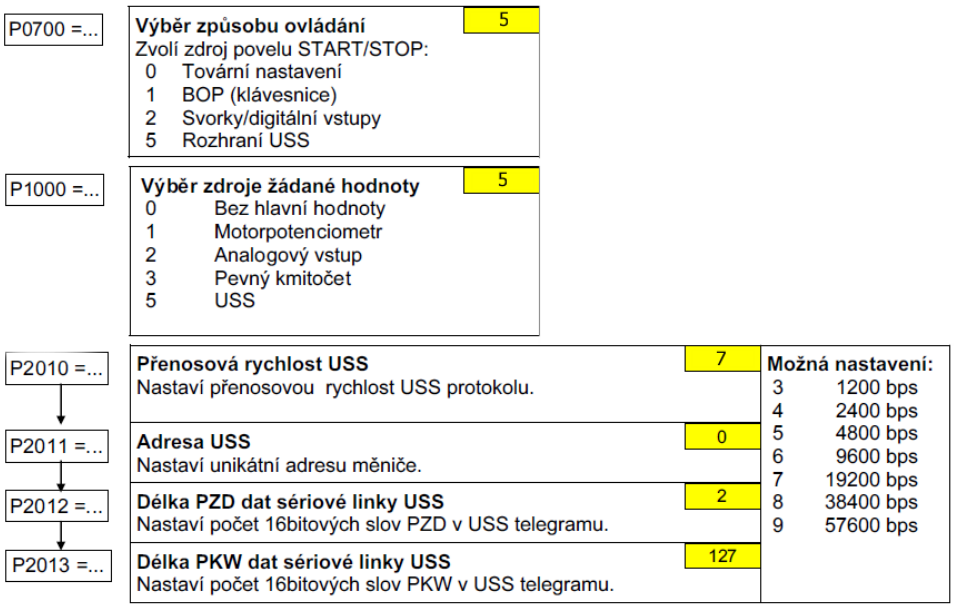
Celou strukturu zobrazuje obrázek 5.1, kde červené šipky znázorňují vazbu mezi jednotlivými části. Přerušovaná černá čára představuje tok vstupních a výstupních uživatelských dat a plná černá čára znázorňuje posloupnost jednotlivých úkonů ve funkční části.

5.2 Fyzická část

Před spuštěním programu se musí správně nakonfigurovat frekvenční měnič pro komunikaci pomocí USS protokolu. Na frekvenčním měniči Siemens G110 se nastavuje celkem šest parametrů, které se musí shodovat s parametry v programu (***DriveSetup***). Testování a vývoj programu probíhal s nastavením popsané na obrázku 5.2. Na levé straně se nachází nastavovaný parametr a zadaná hodnota je zobrazena v rámečku se žlutým pozadím.



Obrázek 5.1: Struktura programu



Obrázek 5.2: Parametry měniče pro nastavení USS komunikace; zdroj [14]

5.3 Funkční část

Hlavní náplň programu tvoří funkční část, která se dělí do tří úloh. Úlohy obsahují funkční bloky, které strukturují jednotlivé procesy pro tvorbu odesílaného telegramu, komunikaci a čtení dat z přijatého telegramu.

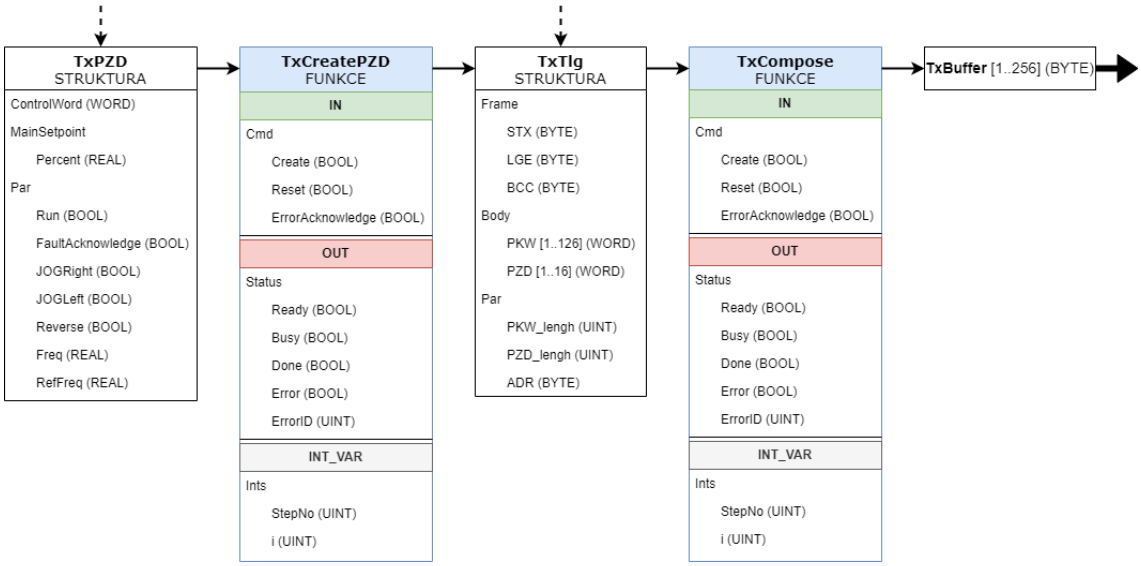
5.3.1 Odesílaný telegram

Pro vytvoření ovládacího telegramu je nutné z uživatelských dat znát potřebné hodnoty parametrů. Mezi tyto parametry patří délka PKW a PZD, adresa měniče, referenční kmitočet, část řídicího slova PZD a požadovaná frekvence. Hodnoty parametrů se překopírují ze struktury ***DriveSetup*** do datových struktur ***TxPZD*** a ***TxTlg***. Struktura vytváření odesílaného telegramu je znázorněna na obrázku 5.3.

Funkční blok ***TxCreatePZD*** vytvoří PZD část telegramu o délce 2 word. Provede složení řídicího slova (PZD1), kde význam každého bitu je vysvětlen v tabulce 2.1. Přepočet žádané hodnoty (PZD2) je proveden pomocí vzorců, které jsou popsány na straně 17. Výstupní hodnoty se ukládají do proměnné PZD struktury ***TxTlg***.

Struktura ***TxTlg*** obsahuje všechny odesílané části USS protokolu. Ačkoliv se PKW část v tomto programu neřeší, tak struktura je na tuto část připravena a umožňuje manuální zadání této části.

Složení výsledného telegramu provádí funkční blok ***TxCompose***. Z parametrů ***TxTlg*** vypočítá délku odesílaného telegramu LGE podle vzorce na straně 13 a z dat vypočítá kontrolní součet BCC byte. Následně vytvoří výsledný odesílaný telegram do proměnné ***TxBuffer***.



Obrázek 5.3: Struktura odesílaného telegramu

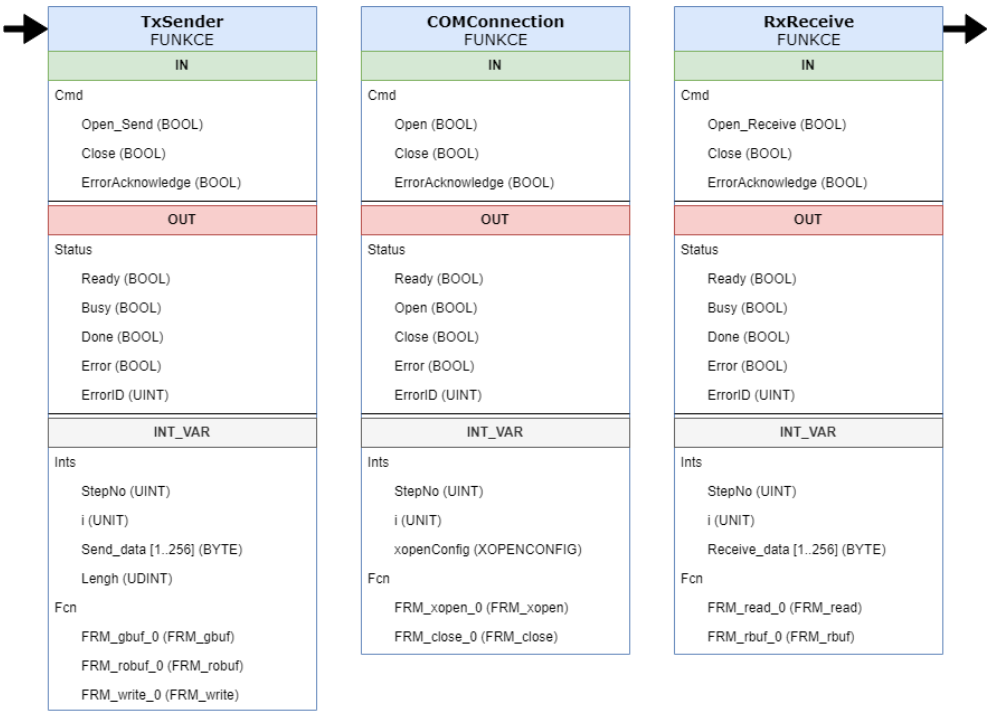
5.3.2 Komunikace s měničem

Pro komunikaci s externími zařízeními poskytuje Automation Studio knihovnu *DVFrame*. Tato knihovna umožňuje výměnu dat po sériové lince s požadovaným nastavením rozhraní, přenosové rychlosti, parity, počtem datových bitů atd. V nápovědě Automation Studia je připravený vzorový program k odesílání a přijímaní dat po sériové lince. Z toho příkladu vycházely funkční bloky ***TxSender***, ***RxReceive*** a ***COMConnection***, které využívají funkce z dané knihovny. Struktura všech tří funkčních bloků je zobrazena na obrázku 5.4.

Otevírání a zavírání sériového portu obstarává funkční blok ***COMConnection***. Využívá k tomu funkce ***FRM_xopen*** a ***FRM_close***. Konfigurace sériového portu RS232 je nastavena s přenosovou rychlostí 19200bit/s se sudou paritou a 8 datovými bity.

K odeslání telegramu slouží funkční blok ***TxSender***. Jakmile je otevřený sériový port, tak funkce ***FRM_gbuf*** nakopíruje odesílaný data z ***TxBuffer*** do vymezený vyrovnávací paměti a následně se pošlou funkcí ***FRM_write***. V případě neúspěšného pokusu zápisu dat se zavolá funkce ***FRM_robuf***, která uvolní vyrovnávací paměť.

Přijímaní telegramu provádí funkční blok ***RxReceive***, který přijme data ze sériové linky funkcí ***FRM_read***. Po úspěšném přijetí dat je překopíruje z vyrovnávací paměti do proměnný ***RxBuffer*** a funkce ***FRM_rbuf*** uvolní vyrovnávací paměť.



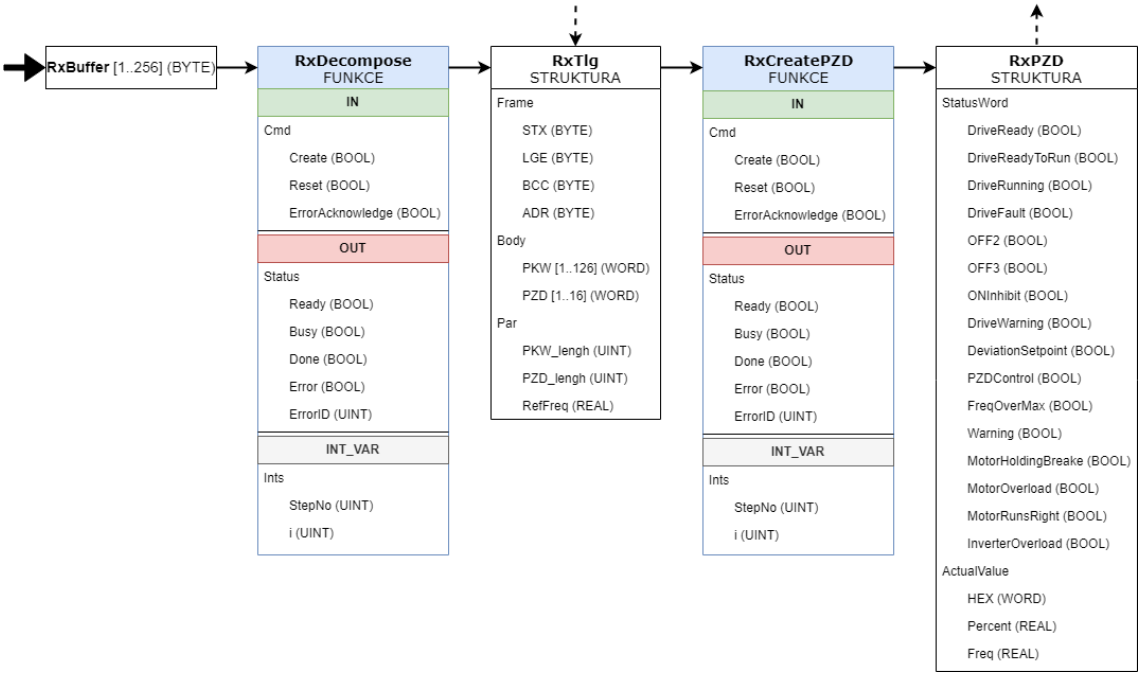
Obrázek 5.4: Struktura komunikace s měničem

5.3.3 Přijatý telegram

Struktura zpracování přijatého telegramu (viz Obrázek 5.5) je opačný proces zpracování odesílaného telegramu. Pro správnou funkčnost je nutné znát parametry přijaté délky PKW a PZD části a hodnotu referenčního kmitočtu. Jelikož délka přijatého telegramu musí být stejná jako délka odesílaného telegramu, tak jsou hodnoty těchto parametrů stejné pro celou funkční část programu.

Přijatý ***RxBuffer*** je pomocí funkčního bloku ***RxDecompose*** rozložen na jednotlivé části USS telegramu. V průběhu vytváření datové struktury ***RxTlg*** se kontroluje správná délka přijatého telegramu a adresa s odesílaným telegramem. Nakonec se vypočítá kontrolní součet a porovná se v přijatým BCC. Pokud některá kontrola vyhodnotí nesrovnalost, tak funkční blok zastaví vykonávaný proces a nahlásí chybu se stavovým kódem.

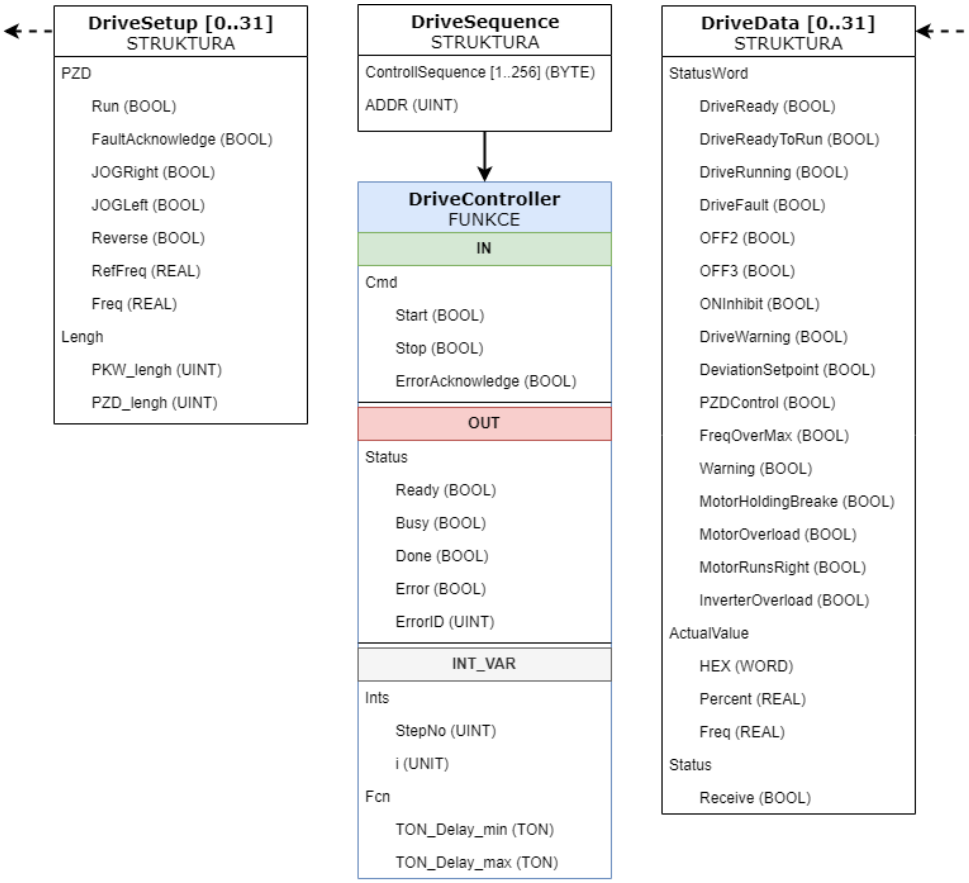
Poslední funkční blok ***RxCreatePZD*** přečte z PZD části telegramu stavové slovo a každou hodnotu bitu přiřadí k danému stavu. Skutečná hodnota rychlosti se uloží do třech různých tvarů. Přepočet je proveden obdobně jako u odesílané části telegramu. Hexadecimální tvar se nejprve přepočte na procenta a následně na hodnotu skutečné frekvence. Všechny tyto informace jsou uloženy ve struktuře ***RxPZD*** a dále překopírovány do struktury uživatelské části ***DriveData***.



Obrázek 5.5: Struktura přijatého telegramu

5.4 Uživatelská část

Uživatelská část programu obsahuje strukturu uživatelských dat a funkční blok, který ovládá celý proces vytváření telegramu, komunikace a zpracování přijatého telegramu. Strukturovaná proměnná uživatelských dat je vytvořena jako instanční. Index strukturovaný proměnný představuje adresu frekvenčního měniče. Struktura této části programu je zobrazena na obrázku 5.6.



Obrázek 5.6: Struktura uživatelské části

5.4.1 Vstupní a výstupní data

Datová struktura v uživatelské části je navržena pro komunikaci s až 32 adresami frekvenčních měničů. Obsluha všech požadovaných adres je zajištěna kruhovým seznamem ve struktuře ***DriveSequence***. Tato struktura obsahuje pole byte ***ControllSequence*** o délce 255, které představuje kruhový seznam. Do toho seznamu uživatel zadá adresy měničů s kterými chce komunikovat. Zkrácení cyklu lze provést zadáním hodnoty 255 do pole. Jakmile dispečer (***DriveController***) načte tuto hodnotu z pole, tak začne vykonávat seznam adres od začátku.

Výstupní data ze struktury ***DriveSetup*** jsou nastavitelná pro každou adresu zvlášť pomocí indexu. Obsahuje parametry referenčního kmitočtu a délky PKW a PZD části, které jsou nakonfigurovány na adresovaném frekvenčním měniči. Mezi další parametry patří požadovaná frekvence a část řídicího slova. Tyto hodnoty může uživatel měnit za chodu programu.

Vstupní data se ukládají pro každou adresu (index) zvlášť do struktury ***DriveData***. Tato struktura je kopii ***RxPZD*** s jednou přidanou proměnnou ***Receive*** typu Bool, která informuje o stavu úspěšného přijetí telegramu. Z těchto dat uživatel vyčte informace o stavu měniče a skutečnou hodnotu rychlosti.

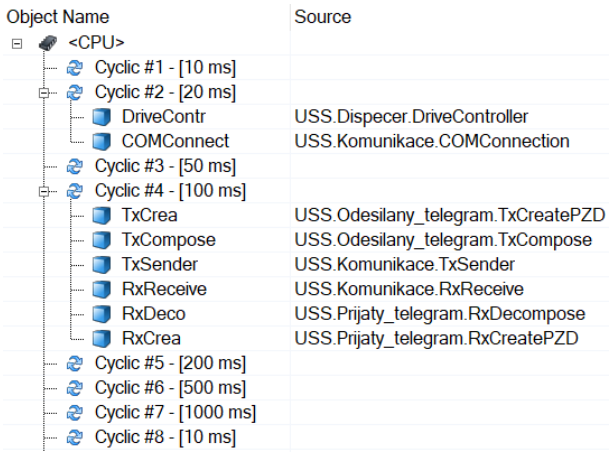
5.4.2 Dispečer

Dispečer neboli funkční blok ***DriveController*** je řízený kruhovým seznamem adres ve struktuře ***DriveSequence***. Postupně volá funkční bloky z funkční části a čeká na jejich stav. Po připraveném telegramu k odeslání zkontroluje zda je otevřený sériový port a telegram odešle. Po odeslání telegramu se spustí časovač se zpožděným sepnutím ***TON_Delay_min***, který zajistí zpoždění doby odezvy před přijetím telegramu. Při přijímání telegramu se spustí časovač ***TON_Delay_max***. Jakmile se do požadované doby nepřijmou data, adresa se přeskočí a dále se neřeší. Správně přijatá data se rozloží do jednotlivých datových struktur a výsledná data ***RxPZD*** se nakopírují do ***DriveData***. Zastavením vykonávaní funkce dispečera se zavře sériový port komunikace a všechny funkční bloky z funkční části se restartuji do výchozího nastavení.

5.5 Optimalizace programu

Pro optimální chod programu bylo nutné správně zařadit funkční bloky do cyklických tříd úloh (viz Obrázek 5.7). Dispečer hlídá status funkčního bloku ve funkční části a podle stavu o dokončení spouští vykonávání dalšího funkčního bloku. Proto musí být zařazen do rychlejší cyklické třídy s periodou 20ms než ostatní funkční bloky. Aby byl včas zajištěn otevřený sériový port, tak je funkční blok ***COMConnection*** ve stejné cyklické třídě jako dispečer.

Všechny ostatní funkční bloky jsou ve stejné cyklické třídě s periodou 100ms. Tato perioda je dostatečná pro funkční bloky ***TxSender*** a ***RxReceive***, které komunikují s pevně nastavenou přenosovou rychlostí 19200bit/s. Doporučené rychlosti přenosu pro USS protokol jsou zobrazeny tučným písmem v tabulce 5.1.



Obrázek 5.7: Příklad přiřazení programů do cyklických tříd úloh

Tabulka 5.1: Podporované přenosové rychlosti USS protokolu

300	bit/s
600	bit/s
1200	bit/s
2400	bit/s
4800	bit/s
9600	**bit/s**
19200	**bit/s**
38400	**bit/s**
57600	bit/s
76800	bit/s
93750	bit/s
115200	bit/s
187500	**bit/s**

5.6 Chybová hlášení a stavové kódy

Každý funkční blok obsahuje proměnnou typu bool s názvem ***Error*** ve stavové informaci. V případě že vyhodnotí chyba, tak informuje uživatele nastavenou hodnotou na logickou 1. Hodnota v proměnné ***ErrorID*** udává stavový kód, pomocí kterého lze zjistit bližší popis chyby. Tabulka se stavovými kódy obsaženými v programu je součástí přílohy C. Jakmile se proces zastaví kvůli vzniklé chybě, tak uživatel musí chybu potvrdit příkazem ***Erroracknowledge*** pro znovuzprovoznění funkčního bloku. Chybové hlášení sloužilo také jako záchytné body při testování a optimalizaci programu.

6 Zhodnocení programu

Dle zadání bylo úkolem vytvořit program v prostředí Automation Studio komunikující po sériové lince prostřednictvím Siemens USS protokolu. Velké množství času bylo stráveno studiem a testováním principu komunikace a struktury USS telegramu. Návrh struktury programu a jednotlivých strukturovaných proměnných probíhal průběžně při řešení jednotlivých úloh.

Konečná struktura programu se skládá z jednotlivých funkčních bloků, které je možné využít samostatně. Datové struktury proměnných poskytují snadný přístup k informacím, které vystupují z funkčních bloků. Tyto datové struktury umožňují vložení vlastních hodnot, které použije následující funkční blok. Struktura programu je připravena pro komunikaci s až 32 frekvenčními měniči pomocí cyklického kruhového seznamu adres.

Pro optimalizaci programu a zachycení chyb bylo důležité zavést chybová hlášení. Stavový kód, který popisoval danou chybu, pomáhal k rychlejšímu nalezení problému.

Přestože bylo zavedeno chybové hlášení, tak největší problém při optimalizaci programu byla část komunikace s frekvenčním měničem a následným zpracováním přijatých dat. Pro správný chod komunikace bylo zapotřebí správně nastavit čas zpoždění doby odezvy pro danou přenosovou rychlost sériové linky. Jakmile se nastavil vyhovující čas, tak přijatý telegram byl zpracován včas a bezchybně.

Program byl testován pouze s jedním poskytnutým frekvenčním měničem Siemens G110. Přestože je program připraven k ovládání více měničů, tak by bylo potřeba před skutečným použitím tuto vlastnost otestovat.

Testování a ladění programu probíhalo s nastavenou přenosovou rychlostí komunikace 19200bit/s. Jiné přenosové rychlosti nebyly v rámci bakalářské práce testovány. V případě ovládání frekvenčních měničů s nejrychlejší možnou odezvou komunikace by bylo nezbytné vhodně zařadit program do cyklických tříd úloh a nastavit vyhovující přenosovou rychlost komunikace. Tato optimalizace rychlosti by zabrala více času pro testování spolehlivosti programu.

7 Závěr

Cílem této bakalářské práce bylo prostudovat strukturu a způsob použití komunikačního protokolu USS firmy Siemens pro ovládání frekvenčních měničů, seznámit se s programovatelnými automaty a vývojovým prostředím Automation Studio firmy B&R-Automation. Následně měla být navržena vhodná datová struktura a program, který bude zajišťovat ovládání jednoduchých frekvenčních měničů Siemens pomocí USS protokolu. Nakonec měl být program otestován na vybraném typu frekvenčního měniče. Všechny body zadání byly splněny.

Teoretická část bakalářské práce popisuje podstatu komunikace po sériové lince prostřednictvím USS protokolu. Je zde také vysvětlena struktura přenášeného telegramu a princip vytváření ovládacího telegramu.

Následně je popsána testovací konfigurace, která se skládá z řídicího systému firmy B&R-Automation a jednoduchého frekvenčního měniče Siemens G110. K počátečnímu testování telegramu byl využit USB-RS232 převodník společně s programem Hercules Setup. Následná kapitola popisuje vývojové prostředí Automation Studio a jeho hlavní části, které byli potřebné k tvorbě programu.

V rámci praktické části bakalářské práce je popsána vytvořená struktura a program, který slouží k ovládání frekvenčního měniče. Program zajišťuje celý proces vytvoření odesílaného telegramu, komunikaci s měničem a zpracování přijatého telegramu. Uživateli umožňuje ovládat až 32 frekvenčních měničů. Každý měnič lze ovládat s jinou požadovanou rychlostí a sledovat jejich aktuální stav. Jelikož princip komunikace zůstává stejný, tak základní způsob ovládání by se neměl lišit od jiných řad měničů. Takže by mělo být možné použít tento program pro ovládání i jiných frekvenčních měničů Siemens (řada V20, popřípadě starší typy řady Micromaster).

Do budoucna se nabízí přidání možnosti čtení potřebných parametrů měniče potřebné pro ovládání. To by vyžadovalo rozšíření o další funkční bloky a strukturované proměnné. Také by bylo vhodné přidat možnosti změny konfigurace komunikace pomocí jednoduché struktury proměnných.

Použitá literatura

[1] *Aplikace Hercules SETUP*. 2019. URL: https://www.hw-group.com/cs/software/aplikace-hercules-setup (cit. 20. 04. 2019).

[2] *Datasheet X20CP04xx*. 1.02. Eggelsberg: B&R Industrial Automation GmbH, 2017. ISBN: BRP4440000000000000512505. URL: https://www.br-automation.com/downloads_br_productcatalogue/BRP4440000000000000512505/X20CP04xx-ENG_V1.02.pdf (cit. 20. 04. 2019).

[3] *Datasheet X20PS9600*. 1.04. Eggelsberg: B&R Industrial Automation GmbH, 2018. ISBN: BRP4440000000000000573591. URL: https://www.br-automation.com/downloads_br_productcatalogue/BRP4440000000000000573591/X20PS9600-ENG_V1.04.pdf (cit. 20. 04. 2019).

[4] *Frekvenční měnič Siemens SINAMICS G110*. 2019. URL: https://asset.conrad.com/media10/isa/160267/c1/-/cs/198079_BB_00_FB/Frekven\%C4\%8Dn\%C3\%AD+m\%C4\%9Bni\%C4\%8D+Siemens+SINAMICS+G110+\%286SL3211-0AB11-2BA1\%29\%2C+1f\%C3\%A1zov\%C3\%BD.jpg?align=center\&x=1000\&ex=1000\&y=1000\&ey=1000 (cit. 20. 04. 2019).

[5] *Frekvenční měniče a vše o nich*. 2019. URL: https://www.elektromotory.cz/frekvencni-menice-proc-a-jak (cit. 20. 04. 2019).

[6] *Montážní sada - Panel+propoj.kabel na PC*. 2015. URL: www.jork.shop/priloha.php?skk=9356 (cit. 20. 04. 2019).

[7] Walter Walter Möller-Nehring a Wolfgang Bohrer. *Universal Serial Interface Protocol. Specification USS® Protocol*. Germany, 1994. ISBN: E20125-D0001-S302-A1-7600.

[8] Vladimír Mýlek. *Průmyslové elektrické pohony - motory, frekvenční a stejnosměrné měniče*. 2015. URL: https://w5.siemens.com/web/cz/cz/corporate/portal/home/produkty_a_sluzby/IADT/tia_na_dosah/Documents/2015_unor/Prumyslove\%20elektricke\%20pohony\%20-\%20male\%20menice.pdf (cit. 20. 04. 2019).

[9] *Ovládací panel - Pro všechny Sinamics G110/G120*. 2015. URL: http://www.jork.shop/priloha.php?skk=16276 (cit. 20. 04. 2019).

[10] *Převodník USB - RS232*. 2008. URL: http://asix.cz/usb_ucab232.htm (cit. 20. 04. 2019).

[11] *SIMOVERT MASTERDRIVES. Motion Control Compendium*. 09/2008. Germany: Siemens AG, 2008. ISBN: 6SE7087-6QX70.

[12] *SINAMICS G110. Seznam parametrů*. 1.0. Germany: Siemens AG, 2003. ISBN: 6SL3298-0BA11-0BP0.

[13] *SINAMICS G110. Návod k obsluze*. 04/2003. Germany: Siemens AG, 2003. ISBN: 6SL3298-0AA11-0BP0.

[14] *SINAMICS G110. Návod k obsluze - stručný*. 11/04. Germany: Siemens AG, 2004. URL: http://www1.siemens.cz/ad/current/content/data_files/technika_pohonu/menice/stridave_menice/nizkonapetove_menice/sinamics_g110/_manualy/opic_sinamics_g110_11-2004_cz.pdf (cit. 20. 04. 2019).

[15] *TM210. Working with Automation Studio*. V2.4.0.1. Eggelsberg: B&R Industrial Automation, 2018. ISBN: TM210TRE.444-ENG.

[16] *TM213. Automation Runtime*. V2.2.0.0. Eggelsberg: B&R Industrial Automation, 2018. ISBN: TM213TRE.444-ENG.

[17] *TM223. Automation Studio diagnostics*. V2.3.0.1. Eggelsberg: B&R Industrial Automation, 2018. ISBN: TM223TRE.444-ENG.

[18] *TM230. Structured Software Development*. V1.2.0.1. Eggelsberg: B&R Industrial Automation, 2013. ISBN: TM230TRE.00-ENG.

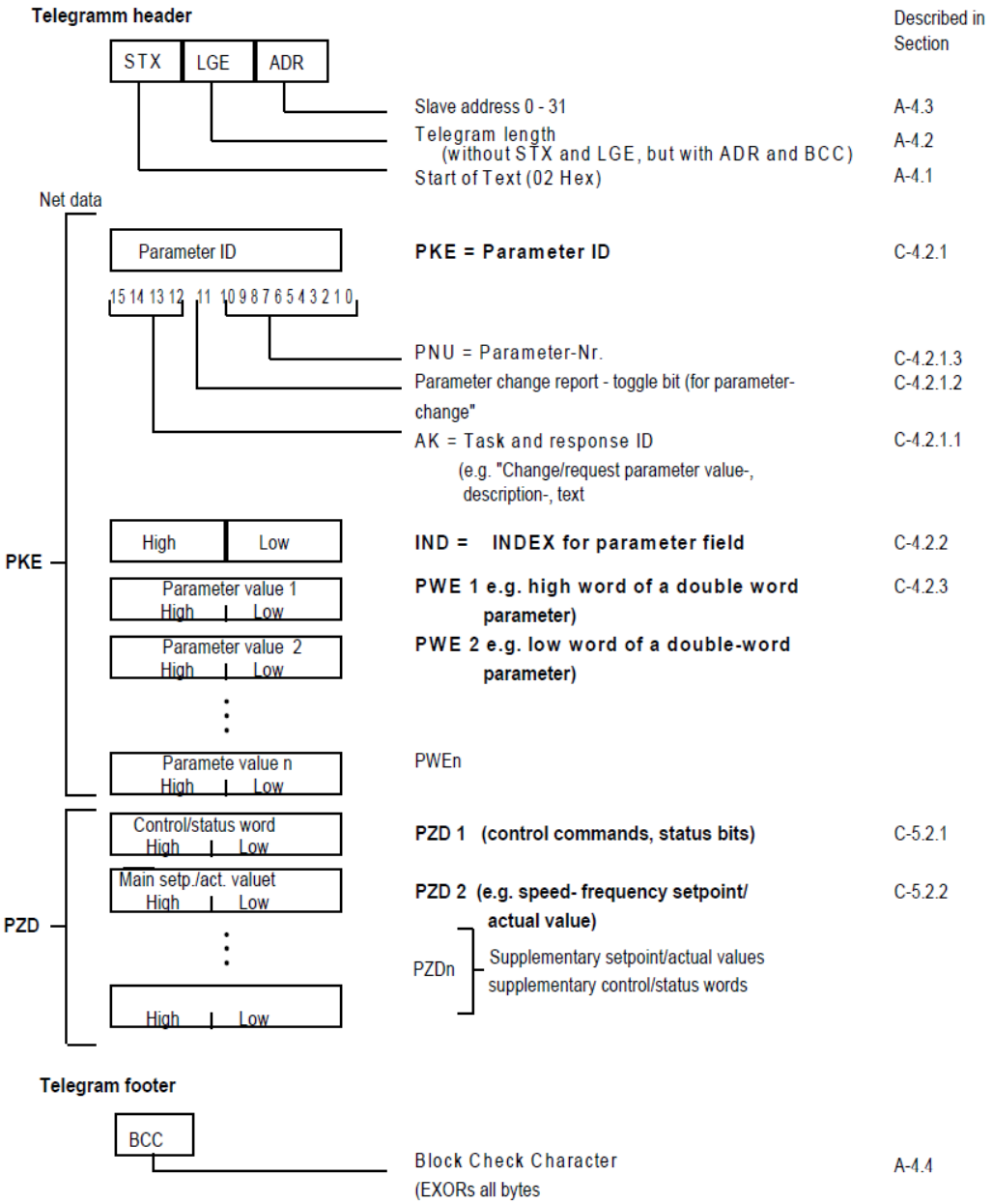
[19] *TM246. Structured Text*. V1.0.8.1. Eggelsberg: B&R Industrial Automation, 2015. ISBN: TM246TRE.00-ENG.

[20] *TM250. Memory management and data processing*. V2.2.0.2. Eggelsberg: B&R Industrial Automation, 2016. ISBN: TM250TRE.425-ENG.

A Přiložený soubor s programem

- Bakalarska_prace_2019_Daniel_Matocha.pdf – tento dokument v elektronické formě
- Soubor USS_Protocol.zip – obsahuje projekt se zdrojovým kódem ve formě Automation Studio v jazyce ST

B Struktura USS telegramu



Obrázek B.1: Podrobná struktura USS telegramu; zdroj [7]

C Stavové kódy programu

TxCreatePZD

ID	Popis
100	ControlWord není v rozsahu 16#0000 až 16#FFFF
101	Referenční frekvence není v rozsahu 1 až 650 Hz
102	Požadovaná frekvence není v rozsahu -200% až 199,994% referenční frekvence
103	Procenta požadované frekvence nejsou v rozsahu -200% až 199,994%

TxCompose

ID	Popis
200	STX není roven 16#02
201	PKW_lengh není v rozsahu konstatních 0, 3, 4 nebo variabilních 127 slov
202	PZD_lengh není v rozsahu 0 až 16 slov
203	LGE není v rozsahu 16#03 až 16#FE
204	ADR není v rozsahu 16#00 až 16#1F
205	PKW_lengh není v rozsahu konstatních 0, 3, 4 nebo variabilních 127 slov
206	PKW není v rozsahu 16#0000 až 16#FFFF
207	PZD_lengh není v rozsahu 0 až 16 slov
208	PKW_lengh není v rozsahu konstatních 0, 3, 4 nebo variabilních 127 slov
209	PZD není v rozsahu 16#0000 až 16#FFFF
210	PZD není v rozsahu 16#0000 až 16#FFFF
211	PZD_lengh je příliš dlouhý (dosaženo maximální počtu net charakters [126])
212	PKW není v rozsahu 16#0000 až 16#FFFF
213	PZD není v rozsahu 16#0000 až 16#FFFF
214	BCC není v rozsahu 16#00 až 16#FF
215	BCC není v rozsahu 16#00 až 16#FF

TxSender + RxReceive + COM (DvFrame)

ID	Popis
60	Stavové kódy jsou převzaty z knihovny DVFrame, bližší popis v Automation Help
8071	
8072	
8073	
8078	
8079	
8251	
8252	
8253	
8254	
8255	
8256	
8257	
8258	
600	Nebyla přijata žádná zpráva

RxDecompose

ID	Popis
300	STX není roven 16#02
301	LGE není v rozsahu 16#03 až 16#FE
302	LGE není stejná odeslaná a přijatá délka
303	ADR není v rozsahu 16#00 až 16#1F
304	PKW_lengh není v rozsahu konstatních 0, 3, 4 nebo variabilních 127 slov
305	PZD_lengh není v rozsahu 0 až 16 slov
306	PKW_lengh není v rozsahu konstatních 0, 3, 4 nebo variabilních 127 slov
307	PKW_lengh není v rozsahu konstatních 0, 3, 4 nebo variabilních 127 slov
308	BCC vypočtené není rovno s přijatým
309	BCC vypočtené není rovno s přijatým
310	ADR není stejná odelaná a přijatá

RxCreatePZD

ID	Popis
400	Referenční frekvence není v rozsahu 1 až 650 Hz
401	PZD není v rozsahu 16#0000 až 16#FFFF

Drive

ID	Popis
500	TxCreatePZD - nevykazuje žádný status
501	TxCompose - nevykazuje žádný status
502	TxSender - nevykazuje žádný status
503	RxReceive - nevykazuje žádný status
505	RxDecompose - nevykazuje žádný status
506	RxCreatePZD - nevykazuje žádný status
507	FRM_read se neprovedl správně