TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Implementace komunikačního protokolu pro ovládání pohonů Siemens

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika
Studijní obor: 3906T001 – Mechatronika

Autor práce: **Bc. Radek Pažout**
Vedoucí práce: Ing. Martin Diblík, Ph.D.

Liberec 2015

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2014/2015**

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Radek Pažout**
Osobní číslo:	**M11000283**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Mechatronika**
Název tématu:	**Implementace komunikačního protokolu pro ovládání pohonů Siemens**
Zadávající katedra:	**Ústav mechatroniky a technické informatiky**

Zásady pro vypracování:

1. Seznamte se se strukturou a použitím sériového komunikačního protokolu pro ovládání elektrických pohonů Siemens.
2. Navrhněte a realizujte software, který umožní použití tohoto protokolu na PLC automatech B+R.
3. Ověřte funkčnost a použitelnost softwaru na vybraných elektrických pohonech Siemens.

Rozsah grafických prací: **dle potřeby dokumentace**

Rozsah pracovní zprávy: **40–50 stran**

Forma zpracování diplomové 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. New York : Springer, 2010. 390 s. ISBN 978-3-642-12014-5.**

[2] **SIEMENS A.G. Siemens Micromaster 440: Operating Instructions, Reference Manual. 2001. SIEMENS A.G. Universal Serial Interface Protocol USS: Specification. 09.94. 1994. E20125-D0001-S302-A1-7600.**

[3] **B+R Automatizace. Automation Training Materials, 2009.**

Vedoucí diplomové práce: **Ing. Martin Diblík, Ph.D.**
Ústav mechatroniky a technické informatiky

Konzultant diplomové práce: **Ing. Leoš Beran, Ph.D.**
Ústav mechatroniky a technické informatiky

Datum zadání diplomové práce: **10. října 2014**

Termín odevzdání diplomové práce: **15. května 2015**

prof. Ing. Václav Kopecký, CSc.
děkan

Technická univerzita v Liberci - 14 - L.S.

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci dne 10. října 2014

Prohlášení

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

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

Užiji-li diplomovou 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.

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

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

Datum: 15. 5. 2015

Podpis: 

Poděkování

Na tomto místě bych rád poděkoval Ing. Martinu Diblíkovi, Ph.D. za cenné připomínky, odborné rady, kterými přispěl k vypracování této práce a zapůjčení vybavení, díky kterému jsem mohl zrealizovat a otestovat praktickou část.

Abstrakt

Tato práce řeší implementaci komunikačního protokolu pro ovládání pohonů Siemens. Protože je USS protokol unikátní, není podporován konkurenčními řídicími systémy. Cílem díla je program, který umožní ovládání pohonů prostřednictvím sériové komunikace. Uživatel získá nástroj, který za něj vyřeší náležitosti spojené se stavbou protokolu, jeho odesíláním a přijímáním. Výsledný program obsahuje strukturovanou proměnnou, která přehledně zpřístupní data potřebná k přenosu informací. Zároveň je aplikace univerzální a lze díky ní řídit i nejsložitější frekvenční měniče. Přínosem práce je možnost užívání pohonů Siemens v dalších projektech, aniž bychom se museli omezovat pouze na jeden druh řídicího systému.

Klíčová slova:

USS protokol, frekvenční měniče Siemens, pohony Siemens, řídicí automat, PLC

Abstract

This thesis is for implementing the communication protocol for control Siemens drivers. USS protocol is unique, so other producers don’t support it. User program on control system for communication via USS protocol is the aim of thesis. User gets a tool for using USS protocol. Final program include a structured variable for exposing data from USS protocol. Application is in two variants. First is for Sinamic G110 frequency converter and second is for all Siemens drivers. The benefit of this thesis is in possibility to use Siemens actuators with other control systems.

Key words:

USS protocol, frequency converter Siemens, actuator Siemens, Siemens drives, PLC, control system

Obsah

Seznam obrázků

1 Úvod

Téma diplomové práce jsem si vybral zejména proto, že se týká programování řídicích automatů. V minulosti jsem měl možnost podílet se na několika zajímavých projektech spojených právě s tvorbou kódu pro PLC. Toto odvětví automatizace má nejen zásadní přínos pro průmyslovou výrobu, ale v dnešní době nás obklopuje skoro všude, třeba i v našich domácnostech.

USS protokol byl vyvinut firmou Siemens, a proto není součástí řídicích automatů jiných výrobců. Chceme-li přistupovat k řízení pohonů zmíněného výrobce přes sériovou komunikaci, nemáme v podstatě jinou možnost, než si USS protokol naprogramovat. Protože značná část výuky probíhá na řídicích automatech firmy B&R, vytvořil jsem zdrojový kód právě pro tyto zařízení.

Cílem práce je vytvořit a popsat aplikaci, která bude ovládat USS protokol tak, jakoby s pohonem komunikovalo PLC Siemens. S tím souvisí vytvoření uživatelského rozhraní, aby bylo možné data přehledně zpracovávat, případně dále využívat.

V teoretické části se budu věnovat USS protokolu, protože znalosti o něm byly zásadní pro celé řešení. V následující části uvedu základní informace o použitém hardwaru a softwaru. Programová část už bude zaměřena na samotnou realizaci a její popis.

2 USS protokol

Universal serial interface protocol (1) – Protokol univerzálního sériového rozhraní.

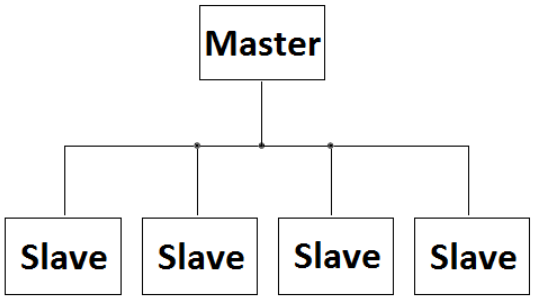
2.1 Specifikace protokolu

USS protokol je založený na principu MASTER / SLAVE pro komunikaci po sériové sběrnice.

Základní vlastnosti:

- jednoduché a spolehlivé rámce telegramu
- snadná implementace
- délka telegramu může být pevná, nebo proměnlivá
- technologie MASTER / SLAVE
- lze připojit až 32 zařízení na jednu sběrnici

Z výše uvedeného vyplívá, že můžeme na jednu sběrnici připojit 1x zařízení jako MASTER (např. PC, PLC) a 31x SLAVE (např. řídicí členy). MASTER přistupuje k SLAVE díky pevným adresám. Tato adresa je nastavena v každém zařízení a je součástí každého rámce protokolu (díky tomu se povel dostane k žádanému přístroji). SLAVE nikdy nemůže vysílat bez vyzvání, pouze odpovídá na řídicí povel od MASTERu. Komunikace je realizována v režimu polo-duplex.



Obrázek 1: Sériové rozhraní mezi SLAVE a vyšší úrovní řízení MASTER

V našem případě řídí komunikaci PLC a roli SLAVE představuje frekvenční měnič.

2.2 Přenos telegramu

Obecně platí, že je třeba rozlišovat mezi cyklickými a necyklickými přenosy telegramů. V oblasti technologií pohonů se používá pouze cyklický přenos telegramu. MASTER je zodpovědný za přenos dat, přičemž všechny SLAVE jsou určeny adresou a jeden po druhém komunikují v identických časových intervalech.

2.2.1 Přenos cyklického telegramu

Technologie pohonu vyžaduje definovanou dobu odezvy pro úkoly kontroly s otevřenou smyčkou. MASTER vysílá telegramy a čeká na telegram odezvy od SLAVE.

SLAVE musí poslat telegram s odpovědí, pokud:

- obdržel bezchybný telegram
- je to vyžadováno telegramem

SLAVE nemůže odeslat telegram, pokud:

- nebyly splněny předchozí podmínky
- byl v režimu odesílání dat

Z pohledu MASTERa byla komunikace úspěšná, obdržel-li telegram s odpovědí ve stanoveném čase. Při necyklickém přenosu telegramů může SLAVE sledovat selhání telegramu.

2.2.2 Přenos necyklického telegramu

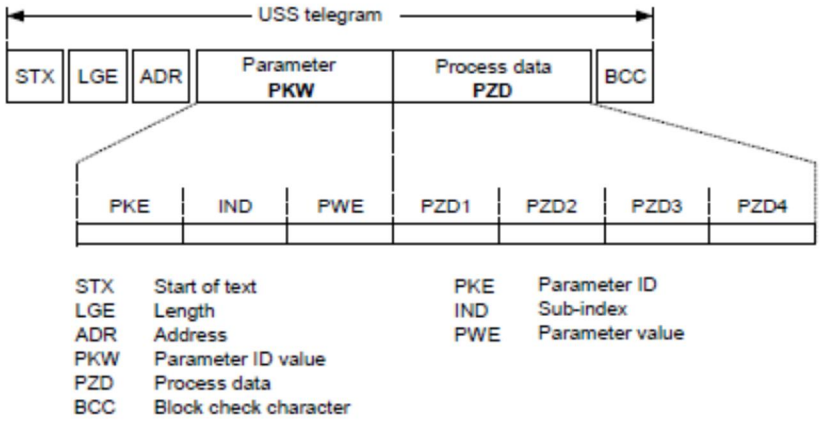
Cyklické a necyklické přenosy dat nelze užívat současně. V necyklickém provozu master posílá telegramy v nepravidelných intervalech. SLAVE reaguje na splnění podmínek stanovených pro cyklický přenos. Při necyklickém přenosu telegramů nemůže SLAVE sledovat selhání telegramu.

2.2.3 Režim vysílání - Broadcast

Tento režim slouží pro vyslání telegramu zároveň pro všechny SLAVE po sběrnici. Při přenosu MASTER nastaví vysílací bit (tzv. broadcast bit) na logickou úroveň 1. V tuto chvíli nehrají adresní hodnoty žádnou roli. SLAVE v tomto případě nevysílají odezvy po přijetí rámce.

2.3 Struktura telegramu

Každý telegram začíná startovním znakem STX, následuje specifikace délky LGE, adresa ADR, hodnoty pro parametrizaci PKW, hodnoty pro řízení PZD a celý protokol je ukončen kontrolním součtem BCC (viz Obrázek 2).



Obrázek 2: Struktura USS protokolu [4], s. 67

Znaky mimo PKW a PZD jsou data zapisována ve formátu bajt – byte (obsahuje 8 bitů, což odpovídá $0000\ 0000_{BIN}$). Parametrizace a řízení se zapisuje v tzv. slovech - WORD (16 bitů, $0000\ 0000\ 0000\ 0000_{BIN}$). Hodnoty protokolu se vysílají po bajtech, pro tvorbu slov tedy platí, že dřívější přijatý bajt je ten vyšší. Využívá se zápis v hexadecimálním tvaru, kde je zápis celého slova zjednodušen na 0000_{HEX}. Vytvořená aplikace tento přístup k datům zohledňuje. Pro představu rozdílů v zápisu uvedu příklad pro číslo např. 731:

- v desítkové soustavě 731
- v binární 10 1101 1011
    - pro zápis potřebujeme 2 bajty 1. vyšší 0000 0010 a 2. nižší 1101 1011
    - slovo je složeno z předchozích bajtů 0000 0010 1101 1011
- šestnáctková soustava (hexadecimální) zjednoduší zápis na 2DB (tedy 02DB)

2.4 STX

Startovní bajt STX vždy nabývá hodnoty 02_{HEX}. Slouží pro identifikaci začátku telegramu.

2.5 LGE

Hodnota v LGE udává délku přenášeného telegramu. V závislosti na konfiguraci protokolu můžeme definovat pevnou délku. V podstatě závisí na počtu přenášených PKW a PZD slov. Maximální povolená délka telegramu je 256 bajtů (odpovídá to maximální hodnotě, jakou lze zapsat do bajtu LGE). Skutečná délka celkového protokolu je o dva bajty delší, než je LGE. První dva bajty (STX a LGE) se totiž nepočítají. Z toho vyplývá, že LGE ukrývá počet bajtů ADR, PKW, PZD a BCC.

$$LGE = (PKW + PZD) * 2 + 2 \quad \{1 \leq LGE \leq 254\}$$

Násobení dvěma je proto, že udáme počet PKW a PZD slov a jedno slovo má 2 bajty. Maximální přípustný počet přenesených parametrů a řídicích povelů může být 252 bajtů.

Pro ukázku si můžeme definovat 4 slova PKW a 2 slova PZD. Po dosazení do předešlé rovnice dopočítáme délku telegramu LGE.

$$LGE = (4 + 2) * 2 + 2 = 14\,dec = 0\mathrm{E}\,hex$$

Variabilní délka telegramu

Proměnná délka telegramu je závislá na konkrétním úkolu a vyžadovaných znacích v přenosu (MASTER → SLAVE). Pokud chceme tuto funkci využít, je třeba nastavit parametr P2013 ve frekvenčním měniči na hodnotu 127 (to umožní mít proměnlivou velikost části PKW, PZD je vždy konstantní a nastavuje se v parametru P2012).

Pevná délka telegramu

Pro přenos telegramů s předem definovanou pevnou délkou. Počet PKW slov je přesně určený při konfiguraci sběrnice systému. Pokud máme na sběrnici více zařízení, můžeme pro každý SLAVE nastavit různou délku LGE.

2.6 ADR

Jak jsem zmínil výše, k jednomu MASTERu můžeme připojit až 32 zařízení (0..31). Pro přiřazení adresy v bajtu ADR potřebujeme tedy jen 5 spodních bitů (XXX0 0000). Zbylé se používají na speciální nastavení, jako může být zrcadlení telegramu nebo vysílací bit (broadcast bit pro hromadné vyslání všem SLAVE zároveň).

2.7 PKW

Spolu s PZD nám zprostředkovávají přenos požadovaných dat. Obsažené informace se zapisují po slovech (wordech). Slouží k definici mechanizmu, který se stará o přenos parametru mezi dvěma komunikujícími zařízeními. Obsahuje tedy zápis a čtení hodnot parametrů, jejich definici a informaci o změně parametru. Všechny úlohy, které obsahuje PKW, slouží pro diagnostiku, servis a nastavení informací. Počet slov v PKW se nastavuje v parametru P2013. Počet slov PKW může být v rozsahu 0 až 127.

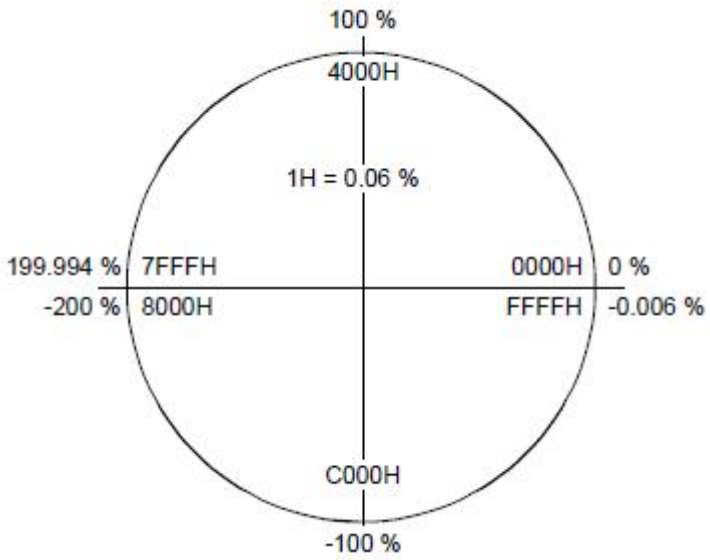
- PKE (ID parametru)
  - obsahuje číslo parametru
  - používá se k identifikaci úkolu a odezvy pro parametry

- IND (Index)
    - slouží k zapisování a čtení popisu parametru, textu a hodnot uložených v poli
- PWE
    - zapisuje/čte nám hodnotu parametru, text, nebo popis parametru
    - můžeme zde přetypovávat na double-word, pole (array)

2.8 PZD

Procesní data nám spolu s PKW slouží k přenosu požadovaných dat, nejsou však na sobě závislá. Obsažené informace se zapisují po slovech (wordech). Zahrnuje signály nutné pro automatizaci. Díky PZD můžeme vysílat z MASTERu řídicí slovo a dostávat skutečnou informaci o stavu SLAVE. Tento rámec dat je nutný pro realizaci této diplomové práce. Velikost PZD určíme parametrem P2012. Počet PZD slov může být v rozsahu 0 až 16, zvolená hodnota je již v protokolu neměnná.

- přenášíme zde řídicí a stavové slovo, požadovanou a skutečnou hodnotu
- PZD je při zpracovávání vždy nadřazena PKW
- i zde můžeme přetypovat na double-word, pole (array)
- v PZD1 jsou vždy uloženy pokyny k řízení a informace o stavu (zapnout, vypnout, reverzace, krokování, atd.)
- v PZD2 je vždy hodnota, která udává procentuální rychlost (kmitočet) a může dosahovat záporných hodnot (další přístup k reverzaci)



Obrázek 3: Rozsah hodnot pro zadávání kmitočtu [3], s 4-3

- na obr. vidíme, že lze zadávat kmitočet v % a dle odpovídající hodnoty v hexadecimálním tvaru přepočítat
- získáme rozsah -200% až +199,99% (8000_{HEX} až $7FFF_{HEX}$)
- 100% odpovídá hodnota 4000_{HEX}

$$f[Hz] = \frac{f(Hex)}{4000(Hex)} * P2000 = \frac{f(\%)}{100\%} * P2000$$

- musíme si však dát pozor, protože zadáváme pouze poměr maximální frekvence
- pokud by ve frekvenčním měniči byla tato hodnota uložena velmi malá, nedosáhli bychom nikdy velké rychlosti motoru
- je tedy dobré zvolit optimální hodnotu maximální frekvence (ta je uložena ve frekvenčním měniči v parametru P1082

- následují volitelné ovládání, informace o stavu v PZD3 až PZDn

2.9 BCC

Kontrolní součet nám sloučí pro ověření správnosti vyslaných/přijatých dat v protokolu. Pro sestavení kontrolního výpočtu se používá XOR (exklusive OR logického operátoru) všech předcházejících bajtů telegramu před BCC. Každé nové BCC_{NEW} nahrazuje původní BCC_{OLD}. Následující příklad naznačí, jak se tato operace provádí.

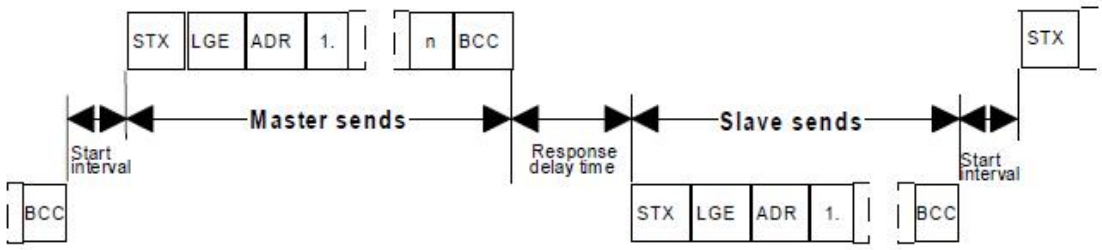
STX má vždy hodnotu 02_{HEX}. Řekněme, že délka LGE bude $14_{DEC} = 0E_{HEX}$ a adresa ADR $4_{DEC} = 04_{HEX}$. Začátek výpočtu kontrolního součtu BCC by byl následující.

BCC	=	0 0 0 0 0 0 0 0
BCC_{OLD}	=	0 0 0 0 0 0 0 0
	XOR	
STX	=	0 0 0 0 0 0 1 0
BCC_{NEW}	=	0 0 0 0 0 0 1 0
BCC_{OLD}	=	0 0 0 0 0 0 1 0
	XOR	
LGE	=	0 0 0 0 1 1 1 0
BCC_{NEW}	=	0 0 0 0 1 1 0 0
BCC_{OLD}	=	0 0 0 0 1 1 0 0
	XOR	
ADR	=	0 0 0 0 0 1 0 0
BCC_{NEW}	=	0 0 0 0 1 0 0 0

Takto by se postupovalo také s jednotlivými bajty z PKW a PZD, až získáme konečný kontrolní součet BCC. V aplikaci jsem napsal algoritmus, který přepočítává BCC automaticky.

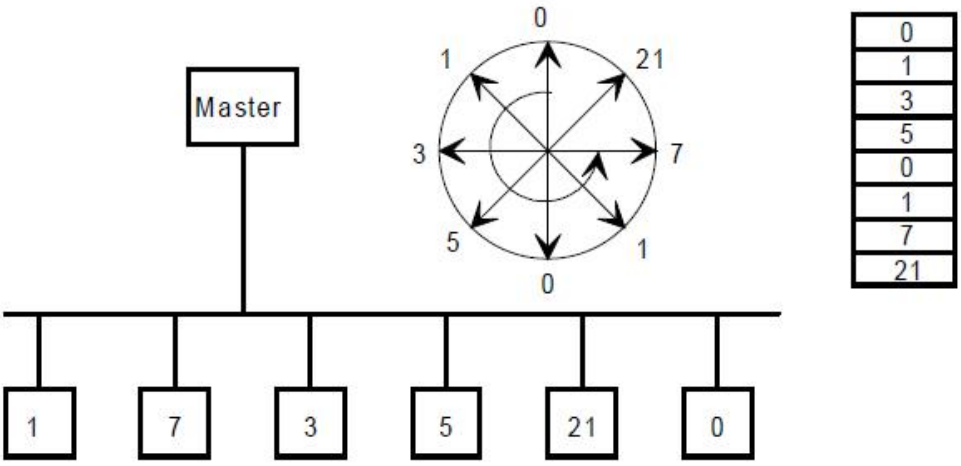
2.10 Postup při přenosu dat

MASTER implementuje cyklický přenos telegramu a postupně všechny SLAVE úkoluje. SLAVE po přijetí řídicího telegramu vyšle odpověď o stavu, případně o chybě při přenosu. MASTER čeká na odpověď, než vyšle další telegram. Toto vychází z popisu komunikace na úrovni MASTER / SLAVE.



Obrázek 4: Sekvence přenosu dat [1], s. A-8

Při komunikaci s pouze jedním zařízením MASTER vyšle telegram, SLAVE odpoví a poté MASTER opět vyšle telegram. Pokud na sběrnici připojíme 2 a více zařízení, zavádíme takzvaný kruhový seznam. Podle něj MASTER postupně obesílá zařízení a přijímá odpovědi. Odešle protokol prvnímu SLAVE, po přijetí odezvy dle seznamu provede stejný úkon s dalším SLAVE. Takto projde celý seznam a začne od začátku.



Obrázek 5: Zasílání telegramu na sběrnici (kruhový seznam) [1], s. A-6

Jak je vidět (viz Obrázek 5), seznam udává pořadí, v jakém budou SLAVE obesílány MASTERem. Pokud potřebujeme nějaký SLAVE řešit v rychlejším cyklu, můžeme ho do seznamu zařadit vícekrát (viz Obrázek 5 - zařízení s adresami 0 a 1 se budou vykonávat s 2x vyšší frekvencí).

2.10.1 Začátek intervalu

Počáteční bajt STX je sice vždy stejný, nezaručuje nám to ale správnou funkci a určení začátku telegramu. Může totiž dojít k situaci, kdy uvnitř telegramu budeme posílat hodnotu bajtu 02_{HEX}. Proto před STX bajtem předchází zpoždění startu o interval alespoň 2 bajtů. Pro např. přenosovou rychlost 9600bit/s je minimální velikost startovního intervalu 3,2ms a pro 19200bit/s je to 1,15ms. Pouze v této kombinaci jsou zařízení identifikovat platný USS protokol.

2.10.2 Monitorovací mechanizmy a chybová hlášení

Při přijetí telegramu je nejprve identifikována kombinace startovního intervalu a STX, poté se hodnotí délka LGE. Telegram je odmítnut, pokud informace o délce neodpovídá zvolené hodnotě s pevnou délkou telegramu, nebo je-li neplatná pro proměnou délku. Časy jsou sledovány před přenosem i během něj. Současně s příjmem dat již zařízení počítá kontrolní součet BCC, který musí odpovídat hodnotě

v posledním bajtu protokolu (který to má být je známo z LGE). Nenastane-li chyba rámce nebo parity v žádném z přijatých znaků, může být vyhodnocena adresa zařízení. V případě, že číslo adresy ADR se neshoduje s adresou zařízení, je protokol odmítnut.

Sledování časů (viz Obrázek 4:):

- zpoždění doby odezvy (response delay time)
  - čas mezi posledním znakem vyslaného telegramu BCC a začátkem odezvy STX
  - tento čas hlídá MASTER
  - maximální povolený čas je 20ms
  - zároveň nesmí být kratší než počáteční interval
  - neodpoví-li SLAVE ve stanoveném čase, uloží se v MASTERu chybové hlášení o nedoručení telegramu a MASTER pokračuje v zasílání protokolu dalšímu SLAVE
  - toto chybové hlášení se vymaže až po přijetí odpovědi od SLAVE, protože zařízení není odstraněno z kruhového seznamu
- startovní interval
  - zpoždění startu o interval alespoň 2 bajtů před STX znakem
  - hlídá SLAVE pro přijetí protokolu
- doba chodu telegramu (telegram residual time)
  - sledování zbytkové doby přenosu telegramu je závislé na sjednané délce telegramu (LGE)
  - sleduje MASTER i SLAVE při přijímání rámce
  - u variabilní délky protokolu je z druhého bajtu zjištěna hodnota LGE a po doběhu přenosu je čas porovnán s touto hodnotou
  - při pevně dané délce LGE není třeba porovnávat, může se čas jen kontrolně měřit

2.10.3 Zpracování přijatého telegramu

Pouze protokol, který byl bezchybně přijat, je zpracováván. Chybné jsou pouze identifikovány.

Možné chyby:

- chyba parity
- chyba znaku rámce
- nesprávná délka LGE
- neodpovídající BCC
- překročení doby odezvy
- přerušení spojení

Pokud je chyba v protokolu, SLAVE neodešle odpověď!

Obecně se doporučuje zobrazovat přijatá data, aby mohl uživatel vyžít tyto informace k diagnostice potíží nebo měl přístup k datům.

Diagnostické parametry (v 16bitovém zobrazení):

1. verze hardwarového rozhraní
2. softwarové uvolnění rozhraní
3. rezervováno
4. 16 bitové slovo pro status chyby
    - 15-8 bit: rezervovány
    - 7 bit: MASTER: vypršení doby odezvy SLAVE: uplynutí doby pro monitorování telegramu
    - 6 bit: špatná délka telegramu LGE
    - 5 bit: zbytková doba chodu telegramu uplynula
    - 4 bit: nesprávný výsledek kontrolního součtu BCC
    - 3 bit: neidentifikovatelný počáteční znak (pokud není STX = 02_{HEX})
    - 2 bit: chyba parity
    - 1 bit: přetečení zásobníku (buffer)
    - 0 bit: chyba znaku rámce
5. Počet přijatých bezchybných telegramů během jedné minuty

6. Počet odmítnutých telegramů během jedné minuty
7. Čítač: bezchybných telegramů
8. Čítač: odmítnutých telegramů
9. Čítač: chyb znaku rámce
10. Čítač: chyb přetečení
11. Čítač: chyb parity
12. Čítač: nezjištěných počátečních znaků STX
13. Čítač: uplynutých zbytkových dob chodu telegramu
14. Čítač: chyb BCC
15. Čítač: nesprávných délek telegramu LGE
16. Čítač: uplynulých časů monitorování

Diagnostické parametry lze rozšiřovat podle potřeb uživatele.

3 Použitý hardware a software

3.1 Sinamics G110

Jedná se o nejmenší kompaktní měniče vhodné pro jednoduché pohony od firmy Siemens. Primárně se používají pro řízení třífázových asynchronních motorů. (2), (3), (4)

Měniče jsou řízeny mikroprocesorem a díky bipolárním tranzistorům s izolovaným hradlem (IGBT) jsou univerzální a spolehlivé. Tichý a rovnoměrný chod motoru zajišťuje pulzně-šířková modulace s přepínatelným spínacím kmitočtem.

Sinamics G110 v továrním nastavení je vhodný pro jednoduché aplikace řízení pohonu s jednoduchou U/f charakteristikou. Pomocí širokého spektra programovatelných parametrů lze přístroj adaptovat pro spoustu aplikací. Pro změnu parametrů můžeme využít univerzální sériové rozhraní USS nebo ovládací panel OP.

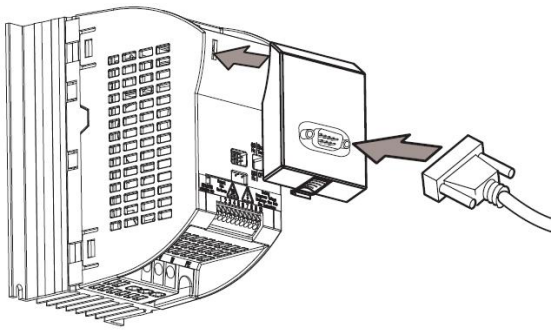
Na trhu je k dostání verze s analogovým řízením a s univerzálním sériovým rozhraním RS485. První případ jsem použil pro realizaci úlohy.

Obrázek 6: Frekvenční měnič Siemens Sinamics G110 [2], s. 29

3.1.1 Vlastnosti

Základní vlastnosti

- snadná instalace a uvedení do provozu
- možnost provozu na izolovaných sítích
- 1 digitální výstup – izolovaný optron
- 3 digitální vstupy (neizolované)
- 1 analogový vstup: 0-10 V (pouze pro analogovou verzi), lze využít jako čtvrtý digitální vstup
- informace o stavu a chybová hlášení prostřednictvím ovládacího panelu
- ovládacím panelem lze kopírovat nastavené parametry
- sériové komunikační rozhraní (pouze v USS verzi), nebo přídavným panelem s RS232



Obrázek 7: Přídavný operátorský panel (RS232) [6], s. 1

Výkonové charakteristiky

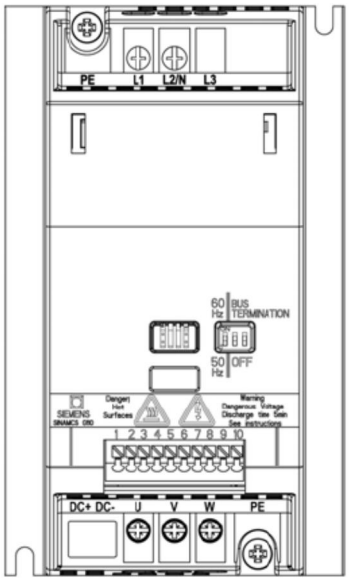
- rychlá odezva na řídicí signál
- možnost brzdění stejnosměrným proudem
- funkce motorpotenciometru
- akcelerační / decelerační časy s programovatelným vyhlazováním
- mimořádně rychlý start

3.1.2 Vstupy a výstupy frekvenčního měniče Sinamics G110

G110 disponuje řídicí, napájecí a vstupně/výstupní svorkovnicí:

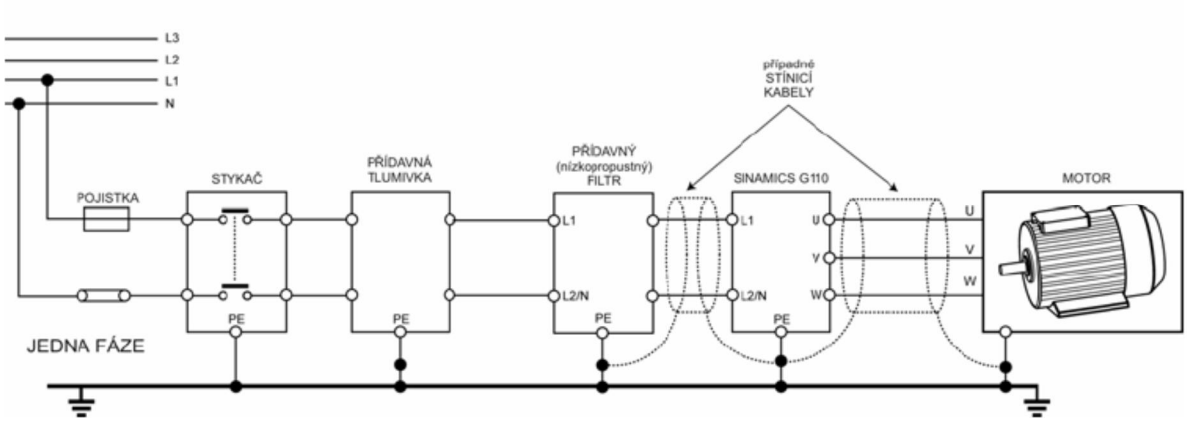
- napájecí svorky (L1, L2/N a PE)
- síťové napájení 230V, 50Hz, vstupní proud 10A
- DIP přepínač pro změnu napájecí sítě mezi standardními 50Hz a 60Hz

- vstupně/výstupní svorkovnice je složena z 10pozic (viz Obrázek 10)
- svorky pro připojení motoru (U, V, W)



Obrázek 8: Svorkovnice měniče kmitočtu Sinamics G110 [2], s. 25

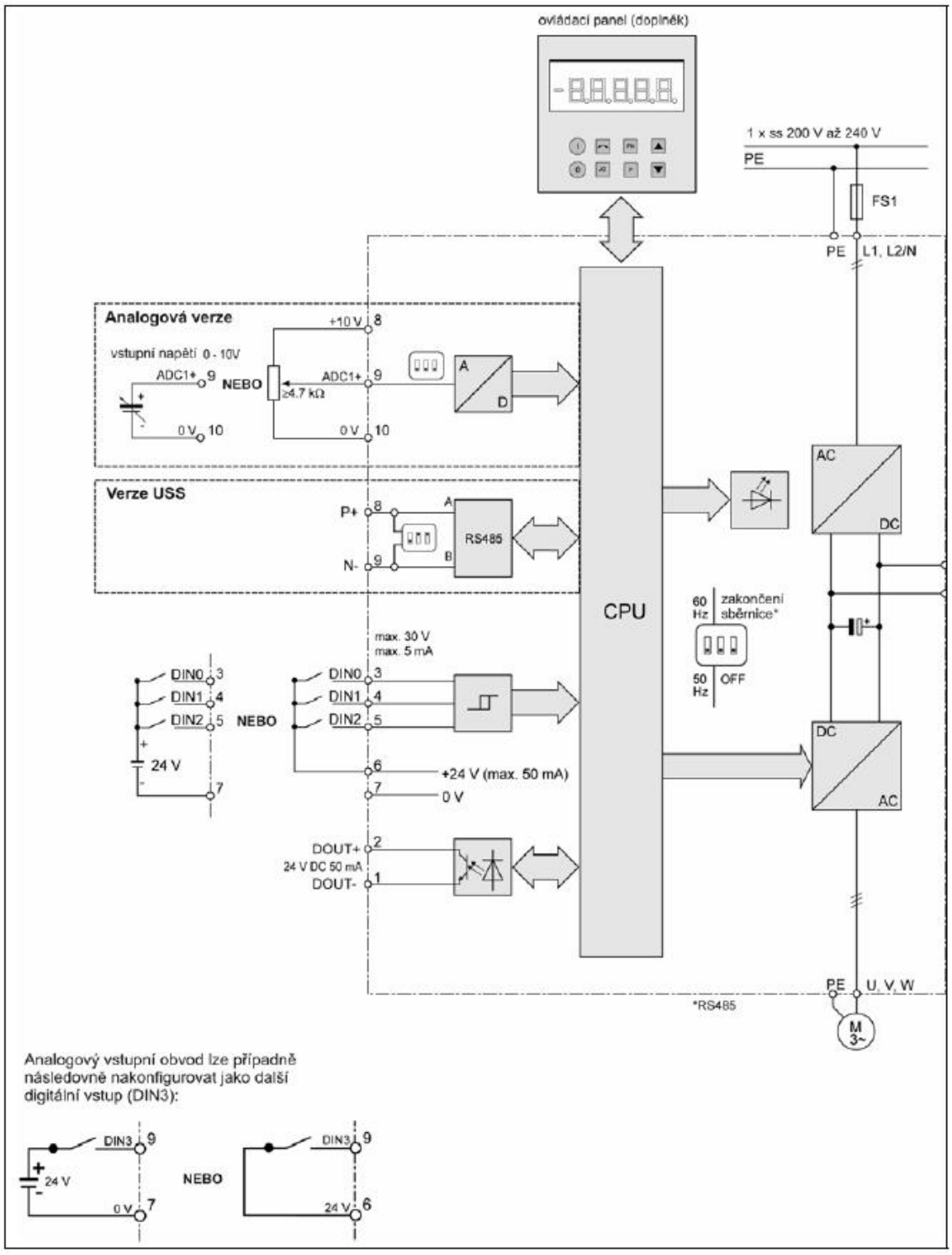
Na následujícím obrázku (viz Obrázek 9) vidíme způsob připojení třífázového motoru prostřednictvím jednofázového frekvenčního měniče.



Obrázek 9: Připojení sítě motoru [2], s. 26

Blokové schéma frekvenčního měniče (viz Obrázek 10) zobrazuje princip jeho funkce. Převod z jednofázového napájení na třífázové řízení pohonu je zajištěn usměrňovačem střídačem. Na levé straně jsou zakresleny vstupní a výstupní svorky. Operátorský panel OP jsem používal k nastavování parametrů ve frekvenčním měniči,

pro přenos USS protokolu jsem OP nahrazoval komunikačním panelem (viz Obrázek 7), který disponuje sériovým rozhraním RS232.



Obrázek 10: Blokové schéma měniče [2], s. 32

3.1.3 Parametry

Parametrů, které můžeme měnit (zvláště v expertním módu) je opravdu mnoho. Lze měnit hodnoty jmenovitých a maximálních proudů, napětí, výkonů, kmitočtů, účiníku motoru a jeho skluzu, rozběhové a doběhové doby, až po tepelné časové konstanty, ovládání vstupů a výstupů, atd. Pokud má uživatel dostatečné znalosti a k realizaci jeho aplikace potřebuje přesné nastavení frekvenčního měniče, nabízí mu už ten nejjednodušší široké pole možností. Abych zde nemusel vypisovat vše, rád bych odkázal na seznam parametrů v příloze (viz ADůležité parametry), případně na dokumentaci příslušného frekvenčního měniče.

3.2 Řídicí systém PLC

Řídicí systém představuje při komunikaci roli MASTERa. Pro realizaci jsem použil PLC firmy B&R, protože jsou na Technické univerzitě v Liberci využívány a vytvořená aplikace bude moct být využívána bez sebemenších komplikací. Protože firma B&R zaručuje při záměně řídicích systémů kompatibilitu programu, není tedy podstatné, pro jaké PLC byl program vytvořen. Při použití odlišného PLC totiž stačí jednoduše změnit hardwarovou konfiguraci a program je možné užívat.

3.2.1 X20 CP 1484-1

Jedná se o konkrétní typ řídicího systému (5), který mi byl zapůjčen. V této kapitole se pouze okrajově zmíním o základních parametrech a vlastnostech tohoto zařízení. PLC bylo potřeba kvůli testování programu a možnosti komunikace s frekvenčním měničem.

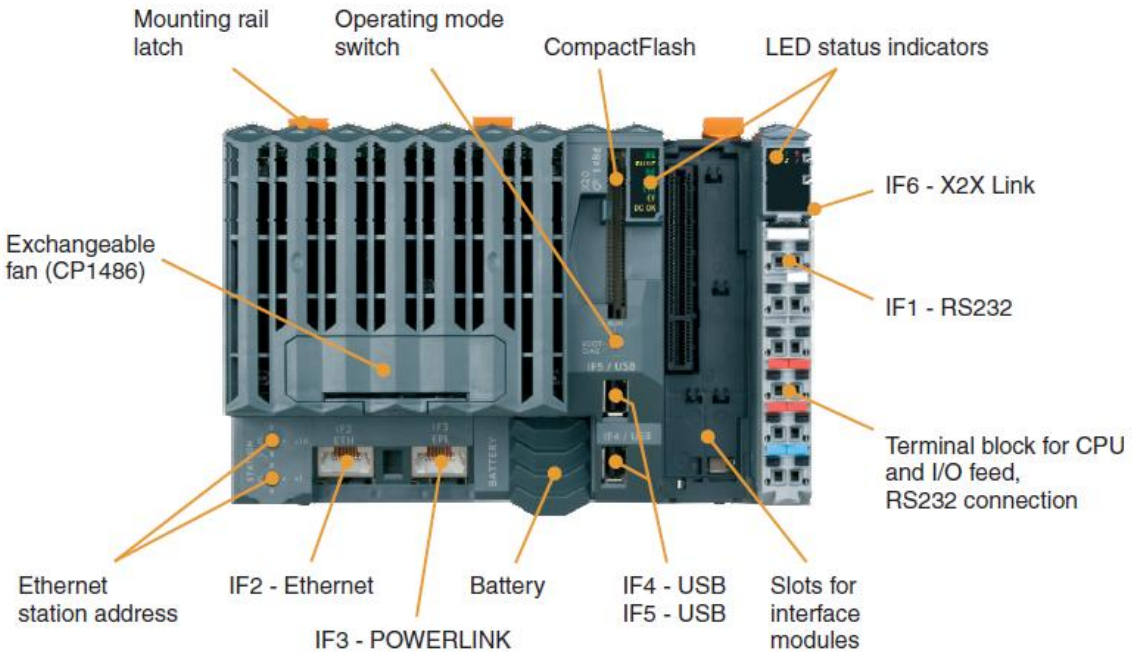
Obrázek 11: PLC X20 CP 1484-1 [5], s. 177

Disponuje kompaktními rozměry 150x99x85mm (šířka x výška x hloubka).

X20 CPU je vybaveno:

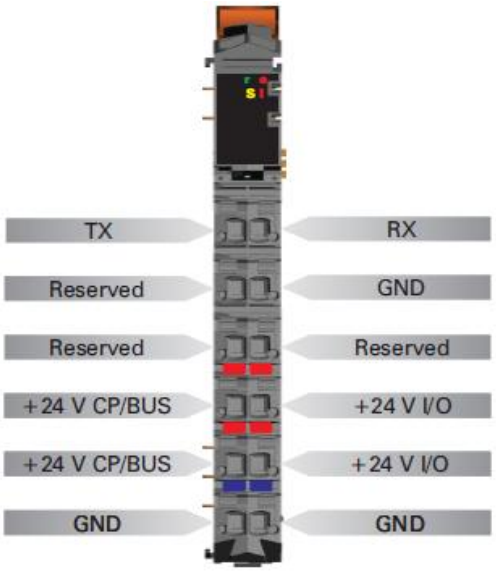
- procesorem Celeron 266 compatible
- 32 MB DRAM, 1 MB SRAM
- výměnnou kompaktní pamětí flash
- 1x slotem pro doplňující moduly (lze připojit karty se vstupními nebo výstupními signály, digitální i analogové)
- 2x USB
- 1x RS232
- 1x Ethernet 10/100 Base-T
- 1x POWERLINK V1/V2
- záložní baterií pro uchovávání dat při odpojeném napájení

Napájení PLC je zajištěno externím zdrojem napětí +24V. Pro uplatnění diagnostiky je systém zpředu vybaven stavovými LED, které přehledně signalizují funkci CPU, stavy, přehřátí, komunikaci po ethernetu a powerlinku, přítomnost kompaktní flash paměti a přítomnost napájení.



Obrázek 12: Konektivita X20 CP 1484-1 [5], s. 186

Na integrované kartě nalezneme připojení pro napájení a komunikaci po sériové lince RS232, která je nepostradatelná pro realizaci aplikace. Tato karta opět obsahuje LED indikátor stavů (informuje nás např. o chodu, aktivitě RS232, atd.).



Obrázek 13: Integrovaná karta [5], s. 188, 190

Tyto řídicí systémy mají mnoho výhod a předností, jako např. vlastní linku X2X (díky ní jsou všechny moduly napájeny a komunikují s PLC, aniž by se museli propojovat vodiči), možnost připojení dalších vstupních a výstupních modulů, přenos dat i na větší vzdálenosti, propojení několika řídicích systémů apod. Více funkcí a vlastností, než které byly výše popsány, nebylo zapotřebí.

3.3 Sériové komunikační rozhraní

3.3.1 Rozhraní RS 232

Sériové rozhraní pro komunikaci osobních počítačů a další elektroniky (řídicí automaty, frekvenční měniče, atd.). Díky RS232 lze propojit dvě zařízení, která mohou sériově komunikovat. Datové bity jsou vysílány nebo přijímány v sérii.

V průmyslu nejčastěji slouží pro připojování PC k řídicím systémům a komunikaci, kdy není třeba přenášet veliký objem dat v krátkém čase, jako je tomu v našem případě.

Jedná se o asynchronním typ komunikace, která vždy začíná sestupnou hranou (tzv. synchronizační hranou). Dále následuje Start bit, Datové slovo, Paritní bit a jeden nebo více Stop bitů. Je důležité, aby obě komunikující zařízení měla shodně nastavenou přenosovou rychlosti, počet datových bitů, typ parity (sudá, lichá) a počet Stop bitů.

3.3.2 RS 485

Stejně jako u RS 232 se jedná o sériovou komunikaci, která je převážně využívána v průmyslu.

Hlavní rozdíly oproti RS 232:

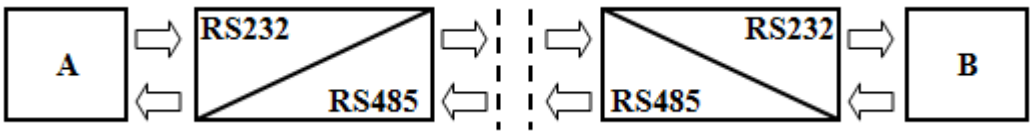
- Symetrické zapojení nezávislé na společné zemi
- Jiná definice napěťových úrovní
- Lze vytvořit síť složenou až z 32 zařízení
- Vzdálenost přenosu (až 1200m)
- Symetrickému zapojení nezávislé na společné zemi

Pro komunikaci využíváme dva vodiče, které se značí A (-) a B (+). Písmenné označení může být u různých výrobců jiné. Rozeznat se dají v klidovém stavu, kdy na vodiči A by mělo být nižší napětí než na vodiči B. Na rozdíl od RS 232 nemá RS 485 standardizovaný konektor. Používají se různé druhy konektorů, když narazíme na shodné, pravděpodobně budou mít jiné zapojení (přiřazení pinů).

3.3.3 Převodník RS 232 - RS 485

RS485 a RS232 můžeme obousměrně kombinovat, jen musíme zajistit převedení signálu pomocí převodníku úrovně.

V praxi se můžeme setkat s dalšími případy, kdy je těchto převodníků zapotřebí. Například můžeme mít dvě zařízení, která obě využívají komunikačního protokolu RS 232, ale vzdáleny budou např. 200m. Lze tedy u jednoho zařízení instalovat převodník na RS 485 a před cílovým zařízením druhý zapojený tak, že nám signál opět vrátí na RS 232 (viz Obrázek 14).



Obrázek 14: Využití převodníků SR232 a RS485 (prodloužení vzdálenosti)

3.3.4 Převodník USB – RS 232 a USB – RS 485

Tyto převodníky slouží pro komunikaci mezi počítačem a dalším zařízením. Použil jsem je pro testování komunikace frekvenčního měniče, abych měl jistotu, zda správně skládám data do USS protokolu a zda byla vůbec informace přijata a následně vrácen aktuální stav.

3.4 Automatizační software - Automation studio

Automation Studio (6) je prostředí sloužící k tvorbě automatizačních aplikací (řídicí, vizualizační i komunikační části). Jedná se o vývojové prostředí podporující operační systémy Microsoft Windows, které poskytuje všechny nástroje potřebné pro vývoj včetně integrovaného návrhu grafického uživatelského rozhraní (vizualizace). Za zmínění stojí také velmi přehledná nápověda.

Automatizační projekty často začínají velmi jednoduchým řídicím úkolem a později se vyvíjejí do komplexnějších řešení, které potřebují více úkonů (vizualizace, řízení, komunikace a další). V případě zařízení B&R jsou všechny vytvořené projekty kompatibilní pro různé typy řídicích systémů, protože procesory obsahují shodný operační systém. Pokud vyměníme typ řídicího systému, tak musíme pouze změnit hardwarovou konfiguraci.

Programování ve vývojovém prostředí Automation Studio odpovídá standardu IEC 61131–3 a můžeme ho provádět ve všech standardních jazycích.

Programovací jazyky (lze kombinovat některé jazyky v rámci jednoho projektu):

- Strukturovaný text (ST – Structured text)
  - textový programovací jazyk používající strukturované angličtiny, svou syntaxí podobný jazyku pascal
  - je definovaný standardem IEC 61131-3
- Funkční bloky (FBD)
  - graficky orientovaný programovací jazyk (IEC 61131-3)
  - využívá předdefinované funkční bloky, které se pouze graficky pospojují.
  - lze si vytvořit vlastní knihovnu, ve které si definujeme funkčními bloky
- Kontaktní schémata (LD – Ladder diagram)
  - grafický orientovaný programovací jazyk (IEC 61131-3)
  - program se podobá elektronickému schématu, kde jednotlivé součástky zastupují jednoduché funkce
- Instrukční list (IL – instruction list)

    - textový programovací jazyk se sekvenčním zápisem jednoduchých instrukcí
    - co se týká míry abstrakce, jedná se o jazyk nižší úrovně než jazyk C (IEC 61131-3)
- Sekvenční strukturované diagramy (SFC – Sequential fiction chart)
    - z větší části graficky orientovaný programovací jazyk s prvky textového jazyka (IEC 61131-3)
    - používaný k popisu časových posloupností akcí uvnitř programu
- Spojité strukturované diagramy (CFC – Continuous fiction chart)
    - graficky orientovaný programovací jazyk, který oproti SFC umožňuje nepřetržité vykonávání programu ve smyčkách pomocí zpětných vazeb
- Automation basic (AB)
    - programovací jazyk vyvinutý speciálně pro B&R
    - umožňuje využívat všechny dostupné programovatelné
- ANSI C

Automation studio má v sobě zakomponované knihovny, které obsahují široké spektrum standardních funkčních bloků pro různé druhy operací (od jednoduchých logických a matematických operací až ke komunikačním protokolům a složitým řídicím algoritmům). Pomocí správce knihoven lze vytvářet a spravovat vlastní funkční bloky.

Řídicí aplikaci můžeme rozdělit do více úloh, které odpovídají procesům z pohledu operačního systému a které mají určité vlastnosti, označované jako třídy úloh. U jednotlivých tříd úloh definujeme prioritu vykonávání, dobu komunikačního cyklu a jazyk, ve kterém jsou úlohy dané třídy psány.

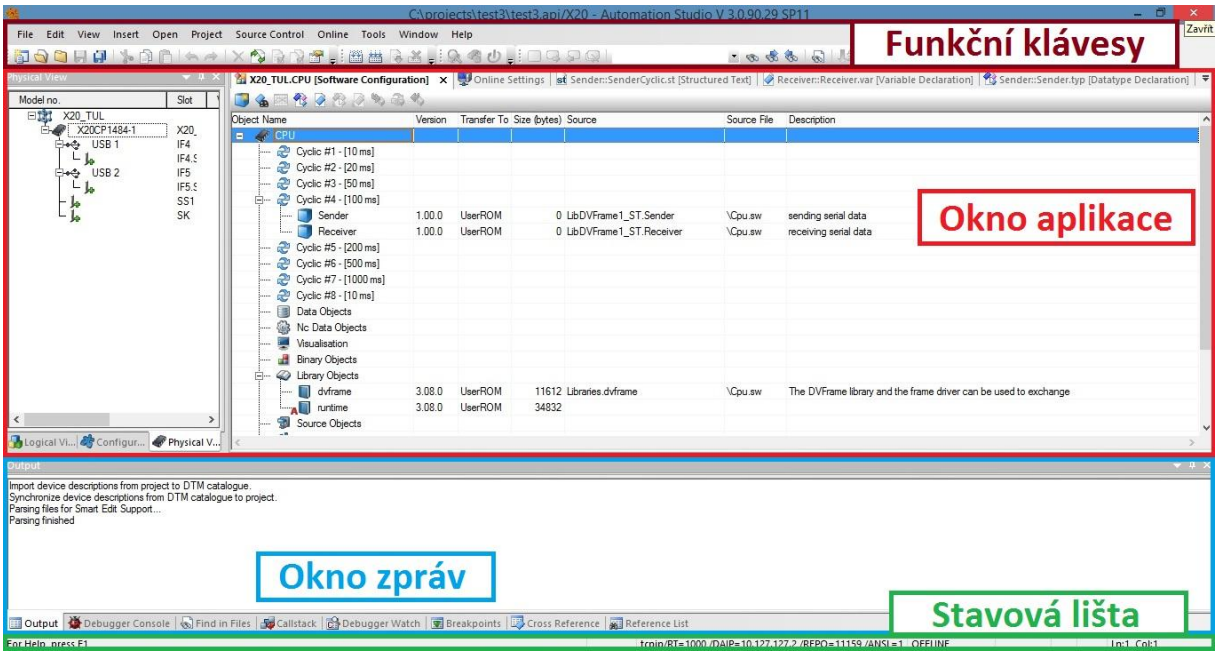
Program každé úlohy se dělí na inicializační část, která se provede jen po restartu (novém nahrání kódu) a na cyklickou část, která se opakuje po definovaném časovém intervalu. Jednotlivé úlohy řeší určitou část aplikace a lze je psát v libovolném jazyce, který je optimální pro danou problematiku.

3.4.1 Struktura Automation studia

Při otevření nového projektu se spustí průvodce vytvořením projektu. Po volbě umístění a názvu projektu je potřeba zvolit používaný model PLC (v našem případě se jedná o X20 CP 1484-1).

Automation studio se skládá z několika částí **(**viz Obrázek 15):

- funkčních kláves (otevření a uložení projektu, překlad programu, nahrání do řídicího systému, zapnutí stavu online a restart řídicího systému)
- okna aplikace (hardwarová konfigurace a objekty jednotlivých komponent)
- okna zpráv (informuje o chybách při tvorbě SW)
- stavové lišty (zobrazuje typ procesoru, typ komunikace a její stav)



Obrázek 15: Struktura Automation studia

Okno aplikace (levá část):

Projekt je rozdělen do 3 možných pohledů ve formě záložek:

- *Logical View* – V tomto pohledu můžeme spravovat zdrojové kódy programů v různých programovacích jazycích, knihovny a grafické návrhy aplikace. Prvky v *logical view* jsou hardwarově nezávislé a přenositelné do jiných projektů a hardwarových sestav.

- *Configuration View* – Pohled zahrnuje několik souborů s nastavením mapování vstupů a výstupů z PLC. Slouží především ke správě hardwarově závislých konfigurací.
- *Physical View* – Pohled umístěný pod třetí záložkou se užívá ke správě hardwarových modulů a jejich zobrazení ve stromové struktuře.

Okno aplikace (levá část):

Pravá strana se automaticky přizpůsobí požadavkům levé strany. To znamená, že ji využíváme k vytváření zdrojového kódu, nastavování (např. klikneme-li na prvek v hardwarové konfiguraci, zobrazí se nám odlišné položky v pravé části (např. pro procesor je to software, parametry, komunikace a výpis hlášení z procesoru, ale pro jiné komponenty se to liší)). V jednotlivých položkách potom můžeme vkládat automatizační úkoly a systémové objekty, definovat potřebné parametry, nastavit parametry komunikace atd.

3.4.2 Monitorování, vyhodnocování a ladění

Automation Studio poskytuje řadu možností k monitorování, vyhodnocování a ladění. Používá k tomu Online Monitor procesních proměnných, Watch, Tracer, Line Coverage, Debugger a System log book .

Online Monitor

V aplikačním okně ukazuje u jednotlivých objektů projektu jejich umístění (systém, projekt), umístění (user Rom, system Rom) a stav (run, use). Dále umožňuje sledování hodnot proměnných přímo v kódu aplikace (popřípadě i jejich změnu ve vybraných jazycích).

Line Coverage

Pomocí barevného označení řádků (částí) v kódu programu ukazuje, jaká část aplikace se vykonává.

Watch

Umožňuje zobrazovat hodnoty vybraných proměnných ve vybraném kódu (tzn. binárně, dekadicky atd.) v editačním okně a měnit jejich hodnotu.

Tracer

Tato komponenta graficky zaznamenává změnu hodnot proměnných v časovém intervalu. Tento displej dovoluje naměřená data dále zpracovávat (nalezení maximální/minimální hodnoty, změřit čas mezi dvěma provedenými stavy atd.).

Debugger

Překladač kontroluje správnost zdrojového kódu a informuje o chybách, které vypíše v okně zpráv a zobrazí nám její popis i pozici. Dále umožňuje krokování programu, vstupováním do procedur i provedení pouze jednoho cyklu.

System log book

Informuje o fatálních a kritických chybách vzniklých za běhu aplikace, dále o stavech procesu a komunikace. Vyskytující se hlášení se zobrazují ve výstupním okně, kde je zobrazen jejich kód, popis a čas vzniku.

4 Realizace programu

Program jsem vytvářel pro řídicí automat od B&R (typ X20 CP 1484-1) v softwaru s názvem Automation Studio. Hotový program však lze použít bez potíží v jakémkoliv řídicím systému od tohoto výrobce (nutná je pouze hardwarové konfigurace). Pro testování jsem měl zapůjčený také frekvenční měnič Siemens Sinamics G110.

Aby mohla být výsledná aplikace skutečně univerzální pro jakýkoliv frekvenční měnič Siemens, ale zároveň vzniklo řešení konkrétního řízení, rozhodl jsem se na základě konzultací pro dvojí přístup posílání USS protokolu. V univerzálním řešení zpřístupňuji rozčleněný protokol a uživatel si zvolí, jaká data chce posílat (slova PKW a PZD). Program již za něj vyřeší všechny ostatní náležitosti, které USS protokol vyžaduje. Pokud ale uživatel upřednostní jednoduché řízení přímo frekvenčního měniče Sinamics G110, tak bude moci volit z konkrétních ovládacích povelů a bude mít možnost si zobrazit informace o stavu pohonu.

4.1 Nastavení komunikace

V první řadě je nutné synchronizovat komunikační nastavení. Detailní popis parametrů pro frekvenční měnič jsem uvedl v příloze (viz A Důležité parametry), proto si nyní dovolím uvést jen krátký seznam.

- P0719
  - nastavíme hodnotu 55
  - ovládání i zdroj žádané hodnoty řídí USS protokol
  - lze také nahradit nastavením P0700 = 5 a P1000 = 5
- P2010
  - nastavíme hodnotu 7 (odpovídá 19200 baud)
  - rychlost přenosu dat sériové komunikace
- P2011
  - nastavíme na libovolnou nevyužitou hodnotu [0..31]
  - adresa měniče

- P2012
  - nastavíme na libovolnou hodnotu [0..15]
  - počet procesních dat PZD
- P2013
  - nastavíme na hodnotu [0,3,4, nebo 127]
    - pro přímé řízení G110 zvolíme hodnotu 0
    - pro univerzální program na skladbu USS protokolu doporučuji nastavit hodnotu 127, získáme tím proměnnou délku
  - délka části PKW

Pokud se uživatel rozhodne nějaké hodnoty zaměnit, musí tak učinit nejen ve frekvenčním měniči, ale také v PLC. U P2010 bychom změnu provedli nastavení PLC. Parametry P2012 a P2013 mohou být omezeny možnostmi měniče.

4.2 Strukturovaná proměnná

Než jsem začal s tvorbou samotného programu, musel jsem si předem stanovit interface, díky kterému bude následný uživatel přistupovat k USS protokolu. Cílem práce nebylo vytvořit uživatelské prostředí ve vizualizaci, protože tento program bude primárně sloužit pro doplňování kódu o další funkce a využívání hotového řízení frekvenčních měničů přes USS protokol. Musel jsem tedy přehledně a jednoduše zpřístupnit všechna potřebná data. Jako nejlepší řešení se jevilo vytvořit strukturovanou datovou proměnnou, kterou uživatel najednou zobrazí v monitorovacím módu Automation Studia. Tento mód slouží pro zobrazení hodnot a stavu proměnných, zároveň je zde také můžeme měnit.

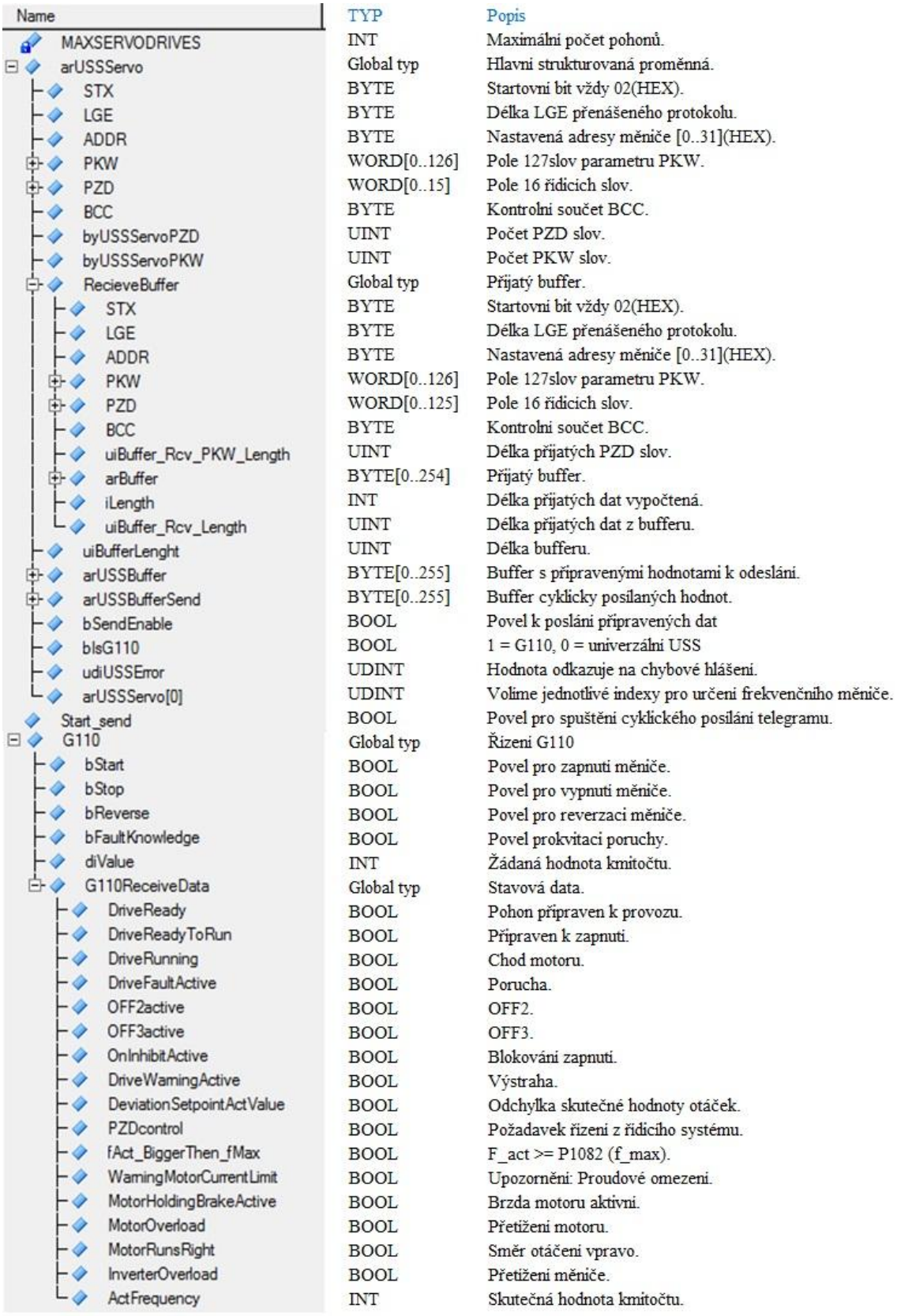
Proměnné strukturovaných typů mají (na rozdíl od jednoduchých proměnných) vnitřní strukturu. Vnitřní komponenty už mohou být jednoduchého typu nebo mohou být opět členěny do dalších komponent. Uživatel tedy v monitorovacím módu zobrazí proměnnou, kterou může rozvětvit na další vnitřní proměnné.

Program jsem sice vytvářel za účelem zprovoznění posílání a přijímání USS protokolu, musel jsem ale uvažovat také situaci, že bude program použit pro řízení více frekvenčních měničů na jedné sběrnici (po sériově sběrnici RS485 totiž může

komunikovat až 32 zařízení). Celou strukturovanou proměnnou jsem tedy rozšířil jako instanční. V podstatě se naše strukturovaná proměnná rozšířila o index, který udává předem zvolený počet frekvenčních měničů. Program cyklicky prochází onu strukturovanou proměnnou, jen v každém kroku mění její index. Hodnota indexu odpovídá jednomu frekvenčnímu měniči. Jakmile program zpracuje cykly pro všechny frekvenční měniče, začne zpracovávat data opět od prvního.

Než tedy program nahrajeme do řídicího automatu, musíme nastavit počet frekvenčních měničů, kolik jich budeme využívat. V globálních proměnnách si najdeme konstantu s názvem MAXSERVODRIVES a tento počet do ní uložíme. Pokud se počet měničů změní, není problém to do konstanty zapsat. Zbylé proměnné už lze měnit za běhu cyklického programu.

Na následujícím obrázku (viz Obrázek 16) jsou vypsány proměnné, které se budou uživateli hodit pro odesílání a kontrolu přenášených dat. Při psaní kódu jsem jich samozřejmě potřeboval více, ale mezivýpočtové a indexové není třeba vypisovat. Abych mohl přehledně prezentovat rozvětvení strukturované proměnné, pořídil jsem print-screen zobrazení v monitoru při on-line ladění a doplnil k němu informace. Všechny typy a proměnné v následujícím seznamu jsou globální.



Name	TYP	Popis
MAXSERVODRIVES	INT	Maximální počet pohonů.
arUSSServo	Global typ	Hlavni strukturovaná proměnná.
STX	BYTE	Startovni bit vždy 02(HEX).
LGE	BYTE	Délka LGE přenášeného protokolu.
ADDR	BYTE	Nastavená adresy měniče [0..31](HEX).
PKW	WORD[0..126]	Pole 127slov parametru PKW.
PZD	WORD[0..15]	Pole 16 řidicich slov.
BCC	BYTE	Kontrolni součet BCC.
byUSSServoPZD	UINT	Počet PZD slov.
byUSSServoPKW	UINT	Počet PKW slov.
RecieveBuffer	Global typ	Přijatý buffer.
STX	BYTE	Startovni bit vždy 02(HEX).
LGE	BYTE	Délka LGE přenášeného protokolu.
ADDR	BYTE	Nastavená adresy měniče [0..31](HEX).
PKW	WORD[0..126]	Pole 127slov parametru PKW.
PZD	WORD[0..125]	Pole 16 řidicich slov.
BCC	BYTE	Kontrolni součet BCC.
uiBuffer_Rcv_PKW_Length	UINT	Délka přijatých PZD slov.
arBuffer	BYTE[0..254]	Přijatý buffer.
iLength	INT	Délka přijatých dat vypočtená.
uiBuffer_Rcv_Length	UINT	Délka přijatých dat z bufferu.
uiBufferLenght	UINT	Délka bufferu.
arUSSBuffer	BYTE[0..255]	Buffer s připravenými hodnotami k odeslání.
arUSSBufferSend	BYTE[0..255]	Buffer cyklicky posilaných hodnot.
bSendEnable	BOOL	Povel k posláni připravených dat
bIsG110	BOOL	1 = G110, 0 = univerzální USS
udiUSSError	UDINT	Hodnota odkazuje na chybové hlášeni.
arUSSServo[0]	UDINT	Volime jednotlivé indexy pro určeni frekvenčniho měniče.
Start_send	BOOL	Povel pro spuštěni cyklického posiláni telegramu.
G110	Global typ	Řizeni G110
bStart	BOOL	Povel pro zapnuti měniče.
bStop	BOOL	Povel pro vypnuti měniče.
bReverse	BOOL	Povel pro reverzaci měniče.
bFaultKnowledge	BOOL	Povel prokvitaci poruchy.
diValue	INT	Žádaná hodnota kmitočtu.
G110ReceiveData	Global typ	Stavová data.
DriveReady	BOOL	Pohon připraven k provozu.
DriveReadyToRun	BOOL	Připraven k zapnuti.
DriveRunning	BOOL	Chod motoru.
DriveFaultActive	BOOL	Porucha.
OFF2active	BOOL	OFF2.
OFF3active	BOOL	OFF3.
OnInhibitActive	BOOL	Blokování zapnuti.
DriveWarningActive	BOOL	Výstraha.
DeviationSetpointActValue	BOOL	Odchylka skutečné hodnoty otáček.
PZDcontrol	BOOL	Požadavek řizeni z řidicího systému.
fAct_BiggerThen_fMax	BOOL	F_act >= P1082 (f_max).
WarningMotorCurrentLimit	BOOL	Upozorněni: Proudové omezeni.
MotorHoldingBrakeActive	BOOL	Brzda motoru aktivni.
MotorOverload	BOOL	Přetiženi motoru.
MotorRunsRight	BOOL	Směr otáčeni vpravo.
InverterOverload	BOOL	Přetiženi měniče.
ActFrequency	INT	Skutečná hodnota kmitočtu.

Obrázek 16: Strukturovaná proměnná s popisem funkce

4.3 Odeslání a přijetí telegramu

Posílání a přijímání dat po sériovém rozhraní patří mezi běžné úkony. Firma B&R má proto knihovnu DVFrame, která má v sobě připravený kód pro odeslání balíčků dat s názvem Sender a pro jejich přijetí Receiver. Programový kód pro stavbu USS protokolu jsem tedy psal do příslušných částí komunikační přípravy.

4.4 Popis programu

Pro lepší přehlednost jsem nakreslil stavový diagram (viz Obrázek 17), který umožní lépe pochopit následující text. Jednotlivé proměnné jsou popsány výše (viz Obrázek 16), budu tedy převážně používat pouze jejich názvy.

Pokud jsme si zvolili počet frekvenčních měničů a zadali tuto hodnotu do MAXSERVODRIVES (neučiníme-li tak, defaultně bude nastaven jeden měnič), můžeme program nahrát do PLC. V Automation Studiu spustíme funkci Monitor, sloužící pro on-line sledování a změnu hodnot. Do Monitoru přidáme naši hlavní strukturovanou proměnnou arUSSServo, proměnnou Start_send a G110.

Nejprve zvolíme, zda chceme řídit pouze Sinamics G110 (blsG110 = 1), nebo potřebujeme přenášet univerzální protokol do jakéhokoliv typu měniče Siemens (dlsG110 = 0).

4.4.1 Řízení G110 (blsG110 = 1)

- nastavíme adresu měniče do ADDR
- zadáme počet PKW slov do byUSSServoPKW = 0
- zadáme počet PZD slov do byUSSServoPZD = 2
  - MUSÍ tomu odpovídat P2012 v měniči
- program průběžně přepočítává zadané hodnoty a skládá telegram do bufferu arUSSBuffer, který je pouze připraven pro odeslání
  - startovní bit STX = 2 (vždy)
  - dopočítá délku LGE
  - spočítá kontrolní součet BCC

- pro odeslání těchto dat do bufferu arUSSBufferSend, který už slouží pro odeslání, musíme nastavit proměnnou bSendEnable = 1
- data se začnou cyklicky posílat až v momentu, kdy nastavíme povel pro spuštění přenosu dat Start_send
- cyklicky se posílají data z arUSSBufferSend, dokud nezopakujeme krok s nahráním arUSSBuffer do arUSSBufferSend, kdy se odesílaná data obnoví
- načasování, kdy lze data přehrát funguje automaticky, aby nebyl přerušen příjem dat
- pokud budeme chtít zadat povel pro zapnutí pohonu, nastavíme bStart= 1
- data nahrajeme do arUSSBufferSend povelem bSendEnable = 1
- tento povel nám naplní první řídicí slovo PZF hodnotou $047F_{HEX}$
- motor se spustí (hned za návodem je rozepsané složení PZD1 pro G110)
- pro vypnutí zadáme bStop = 1
- opět potvrdíme bSendEnable = 1
- tento povel nám naplní první řídicí slovo PZF hodnotou $047E_{HEX}$
- pohon se vypne
- stejně to funguje pro bReverse a bFaultKnowledge
- pro změnu kmitočtu zadáme do rValue procentuelní hodnotu
- rozsah pro řízení kmitočtu je 199,994% až -199,994%
- bStart = 1
- bSendEnable = 1
- přijatá data jsou nejprve zkontrolována
    - přišlo správné STX?
    - pokud ne, tak se do udiUSSError zapíše 10001 a cyklus se zastaví
    - přišlo správné LGE?
    - pokud ne, tak se do udiUSSError zapíše 10002 a cyklus se zastaví
    - přišlo správné BCC?
    - pokud ne, tak se do udiUSSError zapíše 10003 a cyklus se zastaví
- proběhl-li přenos správně, přijatá data se nám přehrají z arBuffer do ReceiveBuffer a USS protokol se nám rozloží zpět do přijatých STX, LGE, ADDR, PKW, PZD a BCC

- všechny proměnné v arUSSServo a G110 nám slouží pro diagnostiku přijatých a odeslaných protokolů
- ještě doporučuji měnit zobrazení hodnoty v monitoru v hexadecimálním tvaru u příslušných proměnných (zpřehlední to kontrolu dat)

Řídicí slovo 1 Rozepsáno po jednotlivých bitech.

Bit 00	ON / OFF	0 = NE	1 = ANO
Bit 01	OFF2: elektrické vypnutí	0 = ANO	1 = NE
Bit 02	OFF3: rychlé vypnutí	0 = ANO	1 = NE
Bit 03	Povolení pulsu	0 = NE	1 = ANO
Bit 04	Zapnutí RFG	0 = NE	1 = ANO
Bit 05	Start RFG	0 = NE	1 = ANO
Bit 06	Zapnutí žádané hodnoty	0 = NE	1 = ANO
Bit 07	Nulování poruchy	0 = NE	1 = ANO
Bit 08	Krokování vpravo	0 = NE	1 = ANO
Bit 09	Krokování vlevo	0 = NE	1 = ANO
Bit 10	Požadavek na řízení z řídicího systému	0 = NE	1 = ANO
Bit 11	Reverzace (změna chodu otáčení)	0 = NE	1 = ANO
Bit 12	Volný bit		
Bit 13	Motorpotenciometr zvýšit	0 = NE	1 = ANO
Bit 14	Motorpotenciometr snížit	0 = NE	1 = ANO
Bit 15	Místní ovládání / Dálkové ovládání	0 = NE	1 = ANO

Stavové slovo 1 Rozepsáno po jednotlivých bitech.

Bit 00	Připraven k provozu	0 = NE	1 = ANO
Bit 01	Připraven k zapnutí	0 = NE	1 = ANO
Bit 02	Chod motoru	0 = NE	1 = ANO
Bit 03	Porucha	0 = NE	1 = ANO
Bit 04	OFF2	0 = ANO	1 = NE
Bit 05	OFF3	0 = ANO	1 = NE

Bit 06	Blokování zapnutí	0 = NE	1 = ANO
Bit 07	Výstraha	0 = NE	1 = ANO
Bit 08	Odchylka skutečné hodnoty otáček	0 = ANO	1 = NE
Bit 09	Požadavek řízení z řídicího systému	0 = NE	1 = ANO
Bit 10	F_act >= P1082 (f_max)	0 = NE	1 = ANO
Bit 11	Upozornění: Proudové omezení	0 = ANO	1 = NE
Bit 12	Brzda motoru aktivní	0 = NE	1 = ANO
Bit 13	Přetížení motoru	0 = ANO	1 = NE
Bit 14	Směr otáčení vpravo	0 = NE	1 = ANO
Bit 15	Přetížení měniče	0 = ANO	1 = NE

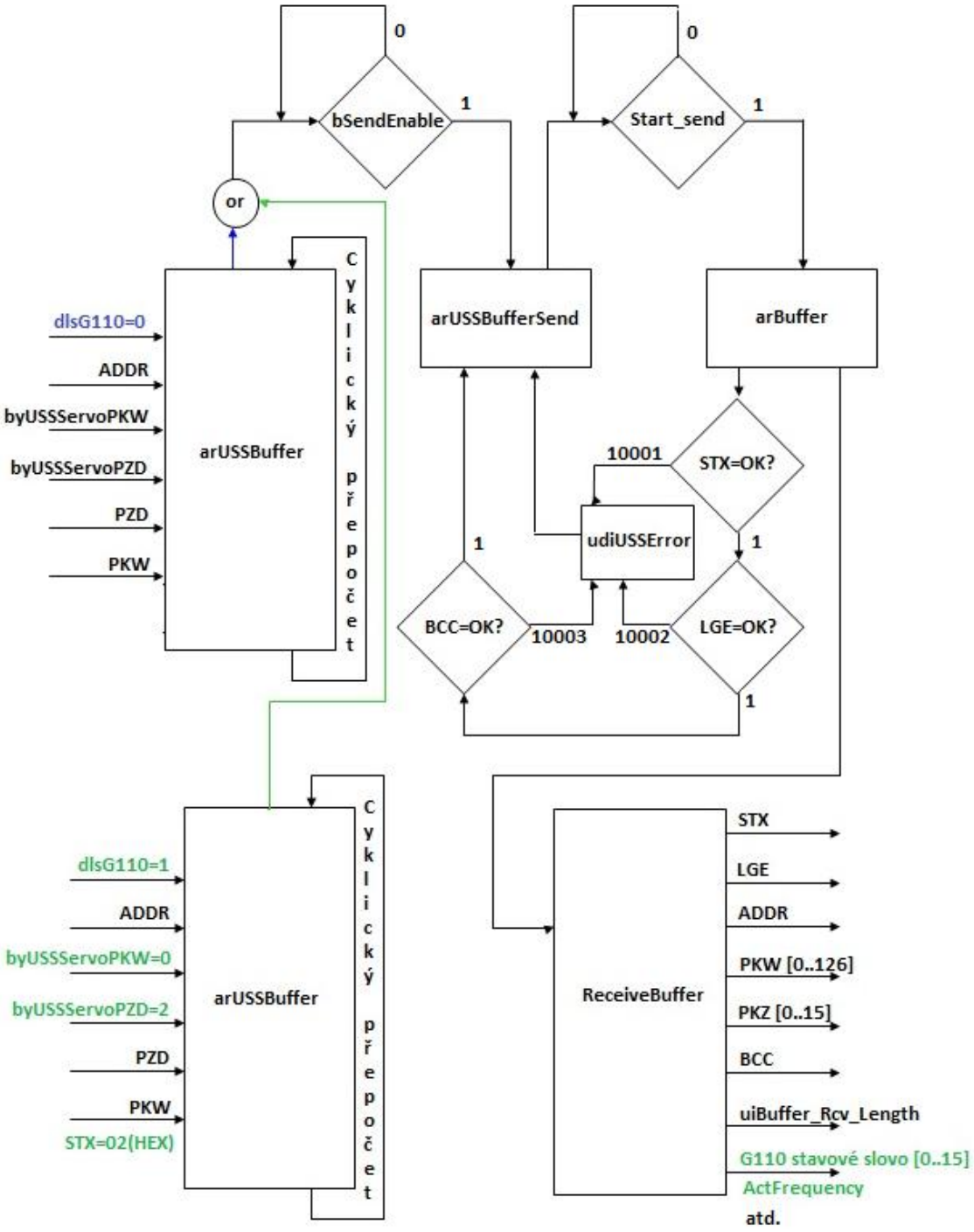
Chybová hlášení zobrazena v udiUSSError

Hodnota	**Popis**	**Hodnota**	**Popis**
10001	špatný start bit STX	8210	analýza poškozené vyrov.pam.
10002	špatná délka telegramu LGE	8251	rámec nebyl otevřen
10003	chybný kontrolní součet BCC	8252	rámec nebyl nalezen
60	nebyla přijata zpráva	8253	chyba syntaxe
8071	není volná vyrovnávací paměť	8254	mnoho rámců otevř. najednou
8072	chyba vyrovnávací paměti	8255	zastaralý operační systém
8073	neplatný řídicí kód	8256	parita, dolní a horní bit
8078	čekání na uvolnění vyrov.pam.	8257	příkaz nepodpor. ovladačem
8079	telegram s vadným charakterem	8259	není k dispoz. paměť, OS,…

4.4.2 Řízení univerzálního USS protokolu (blsG110 = 0)

- nastavíme adresu měniče do ADDR
- zadáme počet PKW slov do byUSSServoPKW = 127 (doporučuji)
- zadáme počet PZD slov do byUSSServoPZD
  - MUSÍ tomu odpovídat P2012 v měniči
- vyplníme data do PKW
- vyplníme data do PZD

- program průběžně přepočítává zadané hodnoty a skládá telegram do bufferu arUSSBuffer, který je pouze připraven pro odeslání
    - startovní bit STX = 2 (vždy)
    - dopočítá délku LGE
    - spočítá kontrolní součet BCC
- pro odeslání těchto dat do bufferu arUSSBufferSend, který už slouží pro odeslání, musíme nastavit proměnnou bSendEnable = 1
- data se začnou cyklicky posílat až v momentu, kdy nastavíme povel pro spuštění přenosu dat Start_send
- cyklicky se posílají data z arUSSBufferSend, dokud nezopakujeme krok s nahráním arUSSBuffer do arUSSBufferSend, kdy se odesílaná data obnoví
- načasování, kdy lze data přehrát funguje automaticky, aby nebyl přerušen příjem dat
- přijatá data jsou nejprve zkontrolována
    - přišlo správné STX?
    - pokud ne, tak se do udiUSSError zapíše 10001 a cyklus se zastaví
    - přišlo správné LGE?
    - pokud ne, tak se do udiUSSError zapíše 10002 a cyklus se zastaví
    - přišlo správné BCC?
    - pokud ne, tak se do udiUSSError zapíše 10003 a cyklus se zastaví
- proběhl-li přenos správně, přijatá data se nám přehrají z arBuffer do ReceiveBuffer a USS protokol se nám rozloží zpět do přijatých STX, LGE, ADDR, PKW, PZD a BCC
- všechny proměnné v arUSSServo nám slouží pro diagnostiku přijatých a odeslaných protokolů
- ještě doporučuji měnit zobrazení hodnoty v monitoru v hexadecimálním tvaru u příslušných proměnných (zpřehlední to kontrolu dat)



Obrázek 17: Stavový diagram

5 Závěr

Výstupem práce je program vytvořený ve vývojovém prostředí Automation Studio a tato technické zpráva, ve které jsem zpracoval informace k teoretické i praktické části diplomové práce.

Program dokáže komunikovat s frekvenčními měniči firmy Siemens prostřednictvím USS protokolu, jak bylo zadáno. Zprostředkovává uživateli přehledný přístup k datům, ze kterých se skládá komunikační rámec protokolu. Aby nebyla omezena možnost volby hardwaru, což také dalo vzniknout této práci, lze použít zdrojový kód pro různé řídicí automaty. Použití jiného PLC, než od firmy B&R, by s sebou samozřejmě neslo nutné úpravy. Zůstaneme-li u řídicích automatů B&R, můžeme volit ze všech typů, pouze snadno změníme hardwarovou konfiguraci. Tato univerzálnost souvisí také s volbou frekvenčních měničů. Zde sice musíme zachovat volbu z produktů firmy Siemens kvůli USS protokolu, avšak jejich sortiment je veliký. Díky programu můžeme řídit všechny typy měničů od nejjednodušších až po ty nejsofistikovanější.

V první teoretické části se zabývám složením USS protokolu (viz 2 USS protokol). Snažil jsem se zde shrnout všechny důležité poznatky, které souvisejí s tímto protokolem.

Druhá kapitola (viz 3 Použitý hardware a software) obsahuje stručné informace o hardwaru a softwaru, který byl při realizaci aplikace využíván. Najdeme zde popis frekvenčního měniče Sinamics G110, řídicího systému X20 CP 1484-1, krátkou kapitolu zahrnující sériovou komunikaci RS232,(RS485) a vývojového prostředí Automation Studio.

Ve třetí části jsem zaznamenal postupný vývoj aplikace. Tato kapitola (viz 4 Realizace programu) zároveň může sloužit jako návod k užívání. Doporučuji se věnovat části s popsanou stavbou strukturované proměnné, protože zahrnuje také nejdůležitější nastavení parametrů frekvenčního měniče, aby mohl být protokol správně přenášen.

I když jsem už měl zkušenosti s programováním PLC před touto diplomovou prací, nečerpal jsem jen z vlastních zkušeností. Téma stavby komunikačního protokolu pro mě bylo úplně nové, mohu tedy říci, že pro mě byla práce přínosná.

S ohledem na zadání mohu říci, že bylo úspěšně splněno.

Seznam použité literatury

1. **MÖLLER-NEHRING, Walter a BOHRER, Wolfgang.** *Universal Systém Interface Protocol USS® Protocol.* 09.94. Erlangen : Siemens, 1994. stránky 0-77. Sv. I. E20125-D0001-S302-A1-7600.

2. **SIMENS AG.** *SINAMICS G110 Návod k obsluze.* 1.0. Erlangen : Siemens, 4/2003. stránky 0-86. 6SL3298-0AA11-0BP0.

3. **SIEMENS AG.** *SIMOVERT MASTERDRIVES MotionControl.* AF. Erlangen : Siemens, 1/2002. stránky 0-1442. 6SE7087-6QX50.

4. **SIEMENS AG.** *SINAMICS G110: Seznam parametrů.* 1.0. Erlangen : Siemens, 4/2003. stránky 0-84. Sv. I. 6SL3298-0BA11-0BP0.

5. **BERNECKER & RAINER.** *X20 Systém: User'sManual.* 2.10. místo neznámé : B&R, 5/2009. stránky 0-1202. Sv. I, Dostupné z: http://www.br-automation.com/en/products/control-systems/x20-systém/documentation/max20-eng/#downloads.

6. **Bernecker & Rainer.** *Working with Automation Studio.* 1.2.0.17. a : B&R, 2012. stránky 0-44. Sv. I. 0.

A Důležité parametry

Obnovení továrního nastavení

Protože mohly být parametry frekvenčního měniče hodně pozměněny, může být před jeho použitím vhodné uvedení do továrního nastavení. Slouží k nastavení všech parametrů na výchozí hodnoty. Provede se následujícím nastavením:

P0010 nastavit na hodnotu 30
P0970 nastavit na hodnotu 1

Proces obnovení nastavení trvá cca 10 sekund.

Důležité parametry

P0003	Přístupová práva	Definuje úroveň přístupových práv ke skupinám parametrů. Ve 4 úrovních můžeme omezit anebo zpřístupnit jednotlivé parametry od základních až po expertní.
P0010	Volba stavu měniče	Filtruje parametry tak, že jsou vybrány pouze ty, které se vztahují ke zvolené funkční skupině. Nastavením hodnoty 0 získáme připravený frekvenční měnič k chodu.
r0052	Stavové slovo 1	Zobrazení prvního aktivního stavového slova měniče v bitovém formátu. Slovo můžeme použít pro diagnostiku stavu frekvenčního měniče.
r0054	Řídicí slovo 1	Zobrazení prvního řídicího slova měniče v bitovém formátu. Slovo můžeme použít pro indikaci, který příkaz je právě aktivní.
P0700	Způsob ovládání měniče	Volba místa, ze kterého je měnič ovládán.

		Např. USS protokolem, nebo operátorským panelem. P0700 a P1000 můžeme zastoupit nadřazeným parametrem P0719. 0 = tovární nastavení měniče 1 = OP ovládací panel (klávesnice) 2 = svorkovnice 5 = USS
P0719	Současný výběr způsobu ovládání a zdroje žádané hodnoty	Má vyšší prioritu než P0700 a P1000. Díky němu můžeme zvolit ovládání a zdroj žádané hodnoty na USS protokol (případně jiné) a ještě díky dvěma indexům rozlišit dálkový a místní přístup k ovládání. Volba obsahuje mnoho parametrů, nejpodstatnější pro náš případ je hodnota 55 = ovládání i zdroj žádané hodnoty je USS.
P0970	Tovární nastavení parametrů	Obnoví výchozí nastavení všech parametrů.
P1000	Výběr zdroje žádané hodnoty	Zde můžeme definovat zdroj žádané hodnoty. P1000 a P0700 můžeme zastoupit nadřazeným parametrem P0719. 0 = bez hlavní hodnoty 1 = motorpotenciometr 2 = analogový vstup 3 = pevný kmitočet 5 = USS
P1082	Maximální kmitočet	Nastavení maximálního kmitočtu v [HZ], na kterém může motor pracovat bez ohledu na žádanou hodnotu kmitočtu. Zvolená hodnota platí pro oba směry otáčení. Tento parametr spolu s P2000 úzce souvisí s částí programu pro řízení kmitočtu frekvenčního měniče G110.

P2000	Referenční kmitočet	Parametr P2000 představuje vztažnou hodnotu kmitočtu pro hodnoty kmitočtu zobrazené/přenášené jako procento hexadecimální hodnoty. Tento parametr spolu s P1082 úzce souvisí s částí programu pro řízení kmitočtu frekvenčního měniče G110.
P2010	Rychlost přenosu dat sériové komunikace USS	Nastavení rychlosti přenosu dat pro sériovou komunikaci USS. 3 = 1200 baud 4 = 2400 baud 5 = 4800 baud 6 = 9600 baud 7 = 19200 baud 8 = 38400 baud 9 = 57600 baud
P2011	Adresa měniče na sériové lince USS	Nastavení unikátní adresy pro měniče. Přes sériové rozhraní lze řídit připojit až 31 měničů (0..31 adres) a řídit je protokolem sériové sběrnice USS.
P2012	Délka procesních dat PZD sériové linky USS	Obsahem parametru je počet 16bitových slov části PZD v telegramu přenášeného po sériové lince USS. U frekvenčního měniče máme možnost volit mezi 0 až 4 slovy PZD (více měnič neumí). Pokud bychom použili frekvenční měnič vyšší řady, můžeme zadat až 16 slov PZD.
P2013	Délka části PKW sériové linky USS	Obsahem parametru je počet 16bitových slov části PKW v telegramu přenášeného po sériové lince USS. Část PKW se může měnit. Podle daného požadavku lze parametrizovat tříslovné nebo čtyřslovné

		proměnné délky slov. PKW část telegramu slouží ke změně a čtení hodnot parametrů měniče. 0 = PKW není přenášeno 3 = 3 slova 4 = 4 slova 127 = proměnná délka části PKW Pokud zvolíme hodnotu parametru P2013 127, tak se délka slova upravuje automaticky podle požadavku.
r2018	Přijatá data PZD ze sériové linky USS	Zobrazení procesních dat přijatých ze sériové linky USS. Výběrem indexu (0..3) můžeme volit mezi 4 posledními protokoly.
r2024	Počet bezchybných telegramů sériové linky USS	Zobrazení počtu bezchybně přijatých telegramů přenášených po sériové lince USS.
r2025	Počet odmítnutých telegramů sériové linky USS	Zobrazení počtu přijatých telegramů přenášených po sériové lince USS, které byly z důvodu chyby odmítnuty.
r2026	Počet chybných znaků v telegramu sériové linky USS	Zobrazení počtu chybně přijatých znaků v telegramu přenášeném po sériové lince USS.
r2027	Počet telegramů sériové linky USS s přetečením	Zobrazení počtu telegramů přenášených po sériové lince USS, které nebyly v daném časovém úseku přeneseny celé.
r2028	Počet telegramů sériové linky USS s paritní chybou	Zobrazení počtu telegramů přenášených po sériové lince USS, které obsahovaly chybný paritní bit.

r2029	Počet telegramů sériové linky USS bez start signálu	Zobrazení počtu telegramů přenášených po sériové lince USS, u kterých nebyl rozpoznán startovací puls.
r2030	Počet telegramů sériové linky USS s BCC chybou	Zobrazení počtu telegramů přenášených po sériové lince USS, které obsahovaly chybný kontrolní součet.
r2031	Počet telegramů sériové linky USS s chybnou délkou	Zobrazení počtu telegramů přenášených po sériové lince USS, u nichž skutečná délka neodpovídá délce očekávané.

B Nastavení parametrů ve frekvenčním měniči

Budeme-li chtít změnit hodnotu parametrů ve frekvenčním měniči, můžeme to provést využitím vytvořené aplikace, nebo díky OP ovládacímu panelu. Ten může vypadat u různých frekvenčních měničů různě, uvedu příklad ovládacího panelu OP přímo ze Sinamics G110.

Obrázek 18: OP Ovládací panel [2], s. 37

Popis jednotlivých tlačítek OP:

	Indikace stavu	LCD displej zobrazuje nastavení, které měnič aktuálně používá
	Zapnutí frekvenčního měniče	Stisknutím tlačítka zapneme měnič. Tlačítko je však aktivní pouze při nastavení parametrů P0700 = 1 a P0719 = 10..15.
	Vypnutí frekvenčního měniče	OFF1 – jedním stiskem plynule zastavíme motor podle nastavené doběhové rampy. Tlačítko je však aktivní pouze při nastavení parametrů P0700 = 1 a P0719 = 10..15. OFF2 – dvojí stisknutí tlačítka (případně dlouhé přidržení) způsobí zastavení

		motoru s volným doběhem. Vždy dostupná funkce.
	Změna směru otáčení motoru	Tzv. reverzace. Tlačítko je však aktivní pouze při nastavení parametrů P0700 = 1 a P0719 = 10..15.
	Funkční tlačítko	Díky tlačítku můžeme zobrazovat dalších informací. Při poruše lze krátkým stisknutím kvitovat poruchový stav.
	Přístup k parametrům	Tlačítko slouží k volbě parametru. Zobrazíme díky němu aktuální hodnotu v parametru a případně potvrdíme volbu nové hodnoty.
	Zvýšit hodnotu	Tlačítko slouží ke zvýšení zobrazené hodnoty, nebo k pohybu v seznamu parametrů směrem nahoru.
	Snížit hodnotu	Tlačítko slouží ke snížení zobrazené hodnoty, nebo k pohybu v seznamu parametrů směrem dolu.
	Krokování	Po stisknutí tohoto tlačítka při stojícím pohonu se motor začne pohybovat v závislosti na nastavených parametrech. Při uvolnění tlačítka se motor zastaví. Při běžícím motoru nemá tlačítko žádný vliv.

Obsah přiloženého CD

- text diplomové práce
  - diplomova_prace_2015_radek_pazout.pdf
  - diplomova_prace_2015_radek_pazout.docx
  - kopie_zadani_diplomova_prace_2015_radek_pazout.pdf
- zdrojový kód programu
  - složka projektu s názvem USS_protocol
  - USS_protocol.rar