VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

OVLADAČE DO PLC TC700/FOXTROT PRO OCHRANY REF630/REF615

DRIVERS IN PLC TC700/FOXTROT FOR FEEDER PROTECTION REF630/REF615

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
Přemysl Till

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Radek Štohl, Ph.D.

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Automatizační a měřicí technika**
Ústav automatizace a měřicí techniky

Student: Přemysl Till ***ID:*** 192828
Ročník: 3 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Ovladače do PLC TC700/Foxtrot pro ochrany REF630/REF615

POKYNY PRO VYPRACOVÁNÍ:

1. Zpracujte rešerši komunikačního protokolu IEC 61850 pro silnoproudé ochrany REF630/REF615.
2. Stručně popište vývojové prostředí Mosaic.
3. Navrhněte a realizujte vertikální komunikaci mezi REF630/REF615 a TC700/Foxtrot.
4. Ověřte a zhodnoťte své řešení.

DOPORUČENÁ LITERATURA:

IEC 61850-7-420 — Communications systems for Distributed Energy Resources (DER) - Logical nodes.

Dle vlastního literárního průzkumu a doporučení vedoucího práce.

Termín zadání: 4.2.2019 ***Termín odevzdání:*** 20.5.2019

Vedoucí práce: Ing. Radek Štohl, Ph.D.
Konzultant: Ing. Martin Chládek

doc. Ing. Václav Jirsík, CSc.
předseda oborové rady

Abstrakt

Práce se zabývá komunikací v průmyslových systémech, konkrétně standardem IEC 61850. Teoretická část obsahuje rešerši informačních zdrojů zabývajících se IEC 61850 a stručné shrnutí datového modelu a modelu vertikální komunikace, definovaných tímto standardem.

V praktické části byla navržena a vytvořena knihovna, umožňující vertikální ovládání stanice splňující IEC 61850 automatem řady Foxtrot či TC700 od firmy Tecomat, k čemuž poskytuje funkční bloky pro vytvoření komunikačního kanálu, vyčtení a zápis dat. Funkčnost byla otestována na ovladačích ochran REF630 firmy ABB.

Klíčová slova

IEC 61850, MMS, programovatelné automaty, vertikální komunikace, komunikační protokoly

Abstract

This thesis deals with communication within industrial systems, specifically using standard IEC 61850. The theoretical part contains a list of useful resources detailing IEC 61850 and a brief summary of the data model and model of vertical communication defined by the standard.

In the practical part, a library for vertical communication between a Tecomat Foxtrot or TC700 PLC and an IEC 61850 conforming device was designed and created. It provides function blocks for establishing a communication channel, reading and writing data using the standard. Its functionality was tested using a feeder protection and control REF630 from ABB.

Keywords

IEC 61850, MMS, programmable logic controllers, vertical communication, communication protocols

Bibliografická citace:

TILL, Přemysl. *Ovladače do PLCTC700/Foxtrot pro ochrany REF630/REF615*. Brno, 2019. Dostupné také z: https://www.vutbr.cz/studenti/zav-prace/detail/119222. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Radek Štohl.

Prohlášení

„Prohlašuji, že svou diplomovou (bakalářskou) práci na téma Ovladače do PLC TC700/Foxtrot pro ochrany REF630/REF615 jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.”

V Brně dne: **18. května 2019**

……
podpis autora

Obsah

Seznam symbolů a zkratek

Zkratky[1]:

VUT	…	Vysoké učení technické v Brně
FEKT	…	Fakulta elektrotechniky a komunikačních technologií
FIT	…	Fakulta informatiky
IEC	…	International Electrotechnical Commission
ISO	…	International Organization for Standardization
OSI	…	Open Systems Interconnection
RFC	…	Request for Comments
MMS	…	Manufacturing Message Specification
GOOSE	…	Generic Object-Oriented Substation Events
SMV	…	Sampled Measured Values
ASN.1	…	Abstract Syntax Notation One
BER	…	Basic Encoding Rules
PDU	…	Protocol Data Unit
APDU	…	Application PDU
PPDU	…	Presentation PDU
SPDU	…	Session PDU
TCP	…	Transmission Control Protocol
COTP	…	Connection-Oriented Data Protocol
TPKT	…	ISO Transport Service on top of the TCP
IP	…	Internet Protocol
IED	…	Intelligent Electronic Device (fyzické zařízení)
LD	…	Logical Device (logické zařízení)
LN	…	Logical Node (logický uzel)

[1] Pojmy, u kterých nemá český překlad význam (názvy institucí, protokolů) jsou uvedeny pouze v anglickém jazyce, jinak je v závorce uveden český překlad. U zkratek používaných na jediném místě je vysvětlení uvedeno přímo na místě použití.

PLC	…	Programmable Logic Controller (programovatelný logický automat)
POU	…	Program Organization Unit (programová organizační jednotka)
FB	…	Function Block (funkční blok)
API	…	Application Programming Interface (aplikační rozhraní)
ST	…	Structured Text (strukturovaný text)

Použité programy[2]:

Mosaic 2019.1 SP1, dostupný z https://www.tecomat.cz/ke-stazeni/software/mosaic/

IEDexplorer 0.79, dostupný z https://sourceforge.net/projects/iedexplorer/

Wireshark 3.0.1, dostupný z https://www.wireshark.org/download.html

Hercules 3.2.8, dostupný z https://www.hw-group.com/cs/software/aplikace-hercules-setup

[2] Odkazy platné k 15.5.2019

Seznam obrázků

Seznam tabulek

1. ÚVOD

Průmyslová komunikace je stále významnější součástí technologických procesů. Často je důležitá spolupráce mezi systémy různých výrobců, k čemuž jsou klíčové standardizované komunikační protokoly. Takovým protokolem je právě IEC 61850, který byl vytvořen již na konci 20. století mezinárodní elektrotechnickou komisí (IEC) ve spolupráci s mnoha velkými výrobci průmyslových zařízení. Tento protokol definuje jednotné rozhraní a datový model pro komunikaci na aplikační úrovni, jazyk pro popis konfigurace systému a její export (SCL – Substation Configuration Language) a mapování komunikace na protokoly nižších úrovní (MMS, GOOSE, SVM).

1.1 Cíl práce

Cílem teoretické části této práce je vytvoření rešerše protokolu IEC 61850 (zejména částí důležitých pro vertikální komunikaci mezi stanicemi) a stručné shrnutí informací, které jsou nutné pro pochopení vertikální komunikace dle IEC 61850 a její implementaci.

Praktická část pak obsahuje návrh a vytvoření ovladačů pro automaty firmy TECOMAT (řady Foxtrot a TC700), umožňujících vertikální komunikaci se stanicemi splňujícími IEC 61850, pracujícími v režimu server. V rámci této práce jde konkrétně o ovládání (vyčtení a zápis dat) ochran REF615/630 od firmy ABB, nicméně implementace by teoreticky měla umožnit komunikaci s libovolným konformním zařízením.

Jelikož standard IEC 61850 definuje pro vertikální komunikaci mapování na protokol MMS v aplikační vrstvě ISO/OSI (L7), ale použité automaty nativně podporují pouze TCP/IP ve vrstvě transportní (L4), práce se dotýká i protokolů potřebných pro propojení ve vrstvách ostatních – tj, zejména protokolů MMS (ISO 9506) a TPKT/COTP (RFC 1006, ISO 8073).

2. PROTOKOL IEC 61850

Teoretická část této práce se věnuje zejména těm částem protokolu, které jsou důležité pro vertikální komunikaci s jiným zařízením. Pro komplexnější pohled na celý protokol doporučuji pročíst některý ze zdrojů, uvedených v následující podkapitole.

2.1 Rešerše zdrojů

MATOUŠEK, Petr. *Description of IEC 61850 Communication* [online]. FIT-TR-2018-01, Brno: Fakulta informačních technologií VUT v Brně, 2018. Dostupné z: http://www.fit.vutbr.cz/research/view_pub.php.cs?id=11832. [EN]

Velmi pěkný rozbor implementace protokolu IEC 61850. Popisuje jak mapování na protokol GOOSE pro horizontální komunikaci, tak MMS pro komunikaci vertikální. Stručně se dotýká i protokolů nižších vrstev, na kterých je komunikace postavená, jako například TPKT a COTP.

STODŮLKA, Ivo. *Model elektrické stanice s komunikačním protokolem IEC 61850* [online]. Brno, 2012. Dostupné z: https://www.vutbr.cz/studenti/zav-prace?zp_id=51986. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. [CZ]

Diplomová práce, zabývající se protokolem IEC 61850 a jeho použitím v ochranách REF 542plus a REF 615. Obsahuje podrobný popis protokolu, jím stanoveného datového modelu a konfiguračního jazyka SCL. Nezabývá se konkrétním mapováním na nosné protokoly.

LEDNICKÝ, Pavel. *Digitalizace rozvodny vysokého napětí při použití komunikačního standardu IEC61850* [online]. Brno, 2011. Dostupné z: https://www.vutbr.cz/studenti/zav-prace?zp_id=41195. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií. [CZ]

Tato práce se zabývá zejména praktickým využitím protokolu IEC 61850. Obsahuje stručný popis protokolu a různých komunikačních režimů, které definuje. Hlavní částí práce je pak srovnání komunikace pomocí IEC a přímého propojení vodiči zejména z hlediska časové odezvy a finanční výhodnosti. Krom toho obsahuje také popis programu PCM600, sloužícího pro nastavení IEC 61850 v ochranách REF.

IEC 61850: 2018 Series. TC 57 - Power systems management and associated information Exchange, 2018. Dostupné z: https://webstore.iec.ch/publication/6028.[EN]

Oficiální verze standardu od International Electrotechnical Commission. Obsahuje všechny části protokolu, ale zobrazení/stažení je zpoplatněné.

630 series IEC 1.3, IEC 61850 Communication Protocol Manual [online]. 2015. Dostupné z: https://new.abb.com/medium-voltage/distribution-automation/numerical-relays/feeder-protection-and-control/relion-for-medium-voltage/feeder-protection-and-control-ref630-iec. Manuál. [EN]

Manuál od firmy ABB, zabývající se implementací IEC 61850 v jejich ochranách. Obsahuje stručný popis relevantních částí standardu a následně seznam všech logických bloků, použitých v ochranách řady REF 630. Obdobné manuály jsou na stránkách ABB dostupné i pro ostatní řady výrobků podporující tento standard (REF 615, REF 650, …).

Protection and Control IED Manager PCM600 [online]. 2018. Dostupné z: https://library.e.abb.com/public/938d6c3e12fd44cf89082bdbb48a7b15/1MRS757866-C-15352.pdf. Manuál. [EN]

Manuál od firmy ABB, popisující prostředí PCM600, sloužící pro konfiguraci ochran REF včetně nastavení komunikace přes IEC 61850.

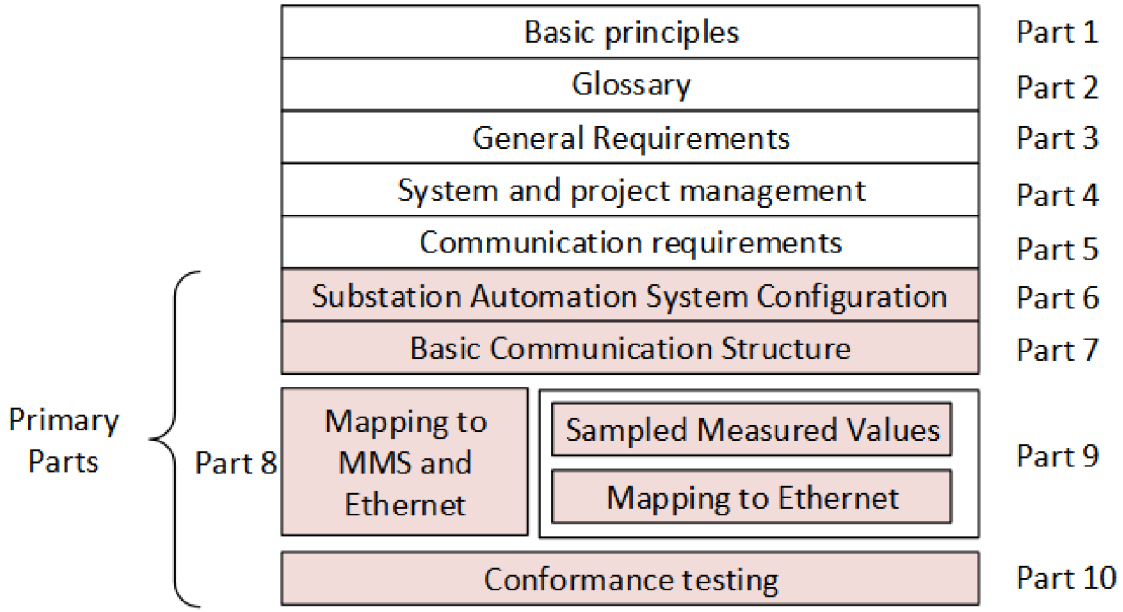
System Integration Specialists Company (SISCO). *Overview and Introduction to the Manufacturing Message Specification (MMS)* [online]. Sterling Heights, USA: 1995. Dostupné z http://www.sisconet.com/wp-content/uploads/2016/03/mmsovrlg.pdf. [EN]

Popis standardu MMS, který IEC 61850 používá pro přenos zpráv při vertikální komunikaci. Obsahuje shrnutí standardu a jeho podrobnější popis, představující jeho objektovou strukturu, použité datové typy, přístup k proměnným a možné akce. Nedotýká se mapování do nižších vrstev ISO/OSI modelu.

2.2 Stručný popis

IEC 61850 je mezinárodní standard, který vydala organizace International Electrotechnical Commission (IEC). Definuje protokoly pro komunikaci mezi inteligentními elektrickými zařízeními pomocí ethernetu (ISO/IEC 8802-3). Krom toho také definuje mapování na nosné protokoly, kterými jsou Manufacturing Message Specification (MMS) pro vertikální komunikaci, Generic Object Oriented Substation Event (GOOSE) pro komunikaci horizontální a Sampled Measured Values (SMV) pro přenos navzorkovaných hodnot s časovými značkami. Hlavním smyslem protokolu je usnadnění komunikace mezi zařízeními různých druhů a výrobců, zejména v oblasti energetiky.

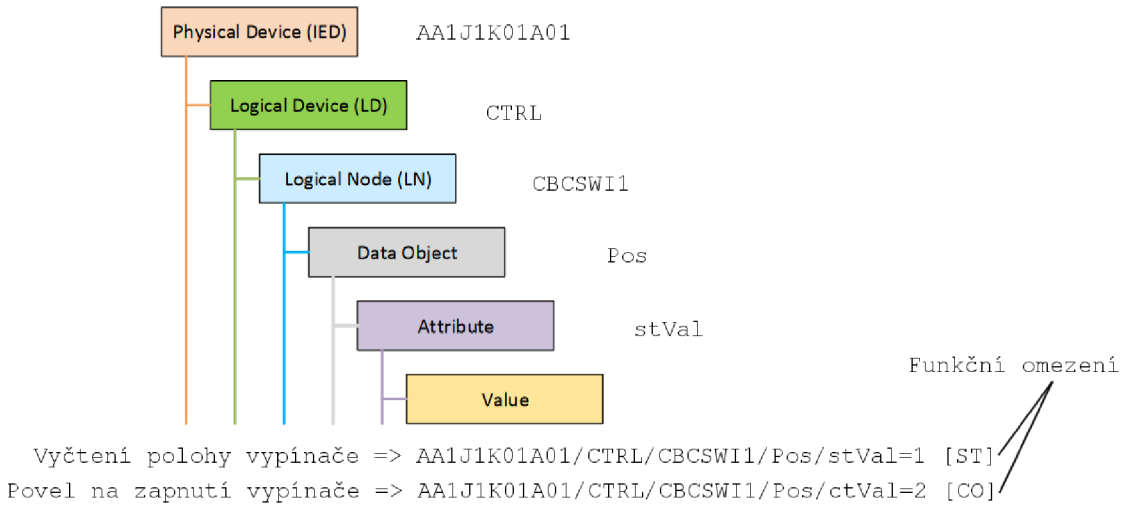
Samotný standard se skládá z několika částí, jak ukazuje Obr. 2-1. Pro účely této práce jsou nejdůležitější kapitoly 7 a 8, popisující strukturu komunikace a její mapování na MMS a vertikální komunikaci v režimu klient-server.



Obr. 2-1: Členění standardu IEC 61850 [5]

2.3 Datový model

IEC 61850 definuje objektově orientovaný model, který umožňuje přístup k datům nezávislý na konkrétním zařízení a implementaci. Data jsou ukládána ve stromové struktuře a pro adresaci jednotlivých položek používáme zřetězené názvy všech nadřazených objektů. Jak strukturu, tak postup sestavení jména zachycuje Obr. 2-2 (přesný význam jednotlivých úrovní vysvětlují následující podkapitoly).



Obr. 2-2: Adresace datových položek ([1] a [2], upraveno)

2.3.1 Logické uzly (LN)

Základním prvkem stromu jsou logické uzly (LN – Logical Node). Kapitoly IEC 61850-7-4, 61850-7-410 a 61850-7-420 definují přes sto různých druhů uzlů, které představují konkrétní zařízení jako vypínače, odpojovače nebo ochranné či měřicí

A	Automatizované řízení	**P**	Ochranné funkce
C	Dispečerské řízení	**Q**	Události související s kvalitou elektrické energie
D	Distribuované zdroje energie (IEC 61850-7-420)	**R**	Funkce související s ochranami
F	Funkční bloky	**S**	Dohled a monitoring
G	Všeobecné funkce	**T**	Přístrojové transformátory a senzory
H	Vodní elektrárny (IEC 61850-7-410)	**W**	Větrné elektrárny (IEC 61400-25)
I	Rozhraní a archivace	**X**	Rozvodny
K	Mechanická a neelektrická primární zařízení	**Y**	Výkonové transformátory a související funkce
L	Systémy logických uzlů	**Z**	Ostatní zařízení
M	Měření		

Obr. 2-3: Skupiny logických uzlů [2]

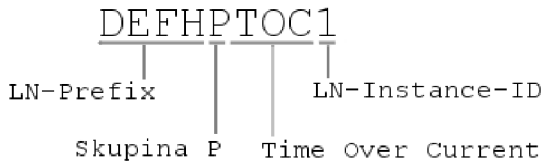
funkce. Uzly jsou členěny do několika typů, jak ukazuje Obr. 2-3.

Názvy všech logických uzlů se skládají ze tří částí. Kořenová část určuje, o jaké zařízení jde. Skládá se ze čtyř písmen velké abecedy, kde první určuje skupinu, do které uzel patří (dle Obr. 2-3), a zbývající tři reálné zařízení, které reprezentuje – např. XCBR – Circuit Breaker, PDIF – Differential Protection nebo RBRF – Breaker Failure.

Zbytek názvu jsou tzv. LN-Prefix (předpona) a LN-Instance-ID (přípona). Obě části jsou nepovinné a definované výrobcem, platí ale, že prefix se musí skládat

z písmen velké abecedy, sufix ze znaků 0-9 a součet jejich délek nesmí přesáhnout 12 znaků.

LN-Prefix slouží k bližšímu upřesnění funkce uzlu – vychází z jeho reálného použití. ID pak umožňuje rozlišení několika uzlů stejného typu umístěných v jednom logickém zařízení, kdy každé dostane unikátní číslo.



Obr. 2-4: Název logického uzlu (podle [2])

Na Obr. 2-4 je příklad názvu logického uzlu, který obsahuje všechny volitelné části. Jde o uzel skupiny P (Protection Functions – Ochranné funkce), konkrétně typu TOC, tj. Time Over Current (časová nadproudová ochrana). Prefix DEFH pak upřesňuje, že jde o směrovou zemní nadproudovou časově závislou ochranu – rychlý stupeň (což ale nevychází ze samotného standardu, ale z dokumentace od výrobce tohoto konkrétního zařízení).

2.3.2 Datové objekty a atributy, funkční omezení[3]

Každý logický uzel se skládá z několika datových objektů, které obsahují datové atributy – nejmenší funkční část zařízení (povely, parametry, logické stavy apod.).

Datové atributy spadají do různých tříd, určujících jejich funkci a datový typ – viz Tab. 2-1. Většina datových atributů má navíc definované také funkční omezení, které upřesňuje jeho charakter a poskytované služby – například stavová informace (ST), měřená informace (MX), řízení (CO) nebo popis (DC).

Zkratka	Význam
SPS	Jednobitový stav
DPS	Dvoubitový stav
INS	Celočíselný stav
SPC	Jednobitové řízení
DPC	Dvoubitové řízení
INV	Celočíselné řízení
MV	Měřená hodnota

[3] Datové objekty a atributy mají mnoho různých tříd a typů, které určují jejich charakter, formát dat a povolené operace. V této kapitole uvádím pouze vybrané, často používané třídy a jejich význam, pro hlubší přehled viz kapitola 3.4.4 v [2].

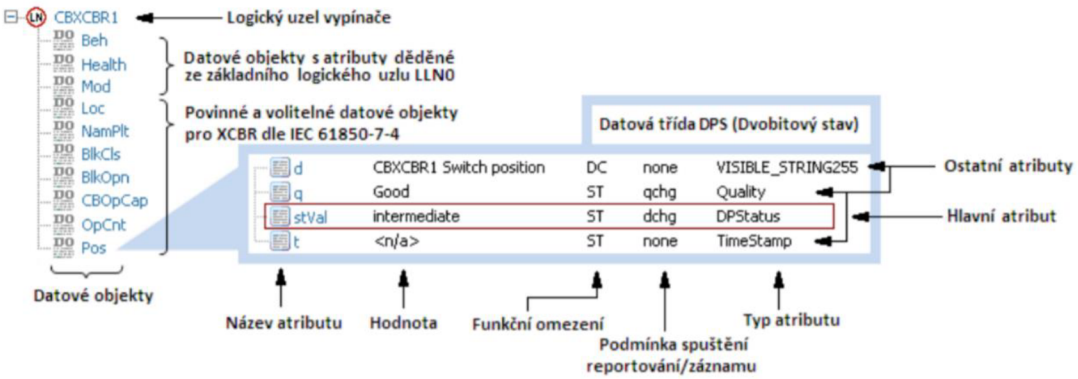
CMV	Komplexní měřená hodnota
DPL	Popisový štítek zařízení

Tab. 2-1: Časté typy datových atributů

Logický uzel XCBR			
Název datového objektu	**Obecná datová třída**	**Popis**	**●/○**
Společné informace o LN			
EEName	DPL	Popisový štítek externího vybavení *(Device nameplate)*	○
EEHealth	ENS	Fyzický stav externího vybavení *(External equipment health)*	○
LocKey	SPS	Místní nebo dálkové ovládání *(Local or remote key)*	○
Loc	SPS	Režim místního řízení *(Local control behaviour)*	●
OpnCnt	INS	Čítač počtu operací *(Operation counter)*	●
CBOpCap	ENS	Provozuschopnost vypínače *(Circuit breaker operating capability)*	○
POWCap	ENS	Možnost vypínaní v daném bodě vlny *(Point on wave switching capabality)*	○
MaxOpCap	INS	Provozuschopnost vypínače při akumulaci střadače *(Circuit breaker operating capabality when fully charged)*	○
Dsc	SPS	Nesouhlas *(Discrepancy)*	○
Ovládání			
LocSta	SPC	Oprávnění ke spínání na staniční úrovni *(Switch autority at station level)*	○
Pos	DPC	Přepnutí polohy*(Switch Position)*	●
BlkOpn	SPC	Blokování vypnutí *(Block opening)*	●
BlkClc	SPC	Blokování zapnutí *(Block close)*	●
ChaMotEna	SPC	Aktivace pohonu střadače *(Charger motor enabled)*	○
Nastavení			
CBTmms	ING	Vypínací čas vypínače *(Closing time of breaker)*	○
Měřené hodnoty			
SumSwARs	BCR	Suma vypnutých proudů s možností resetu *(Sum of Switched Amperes)*	○

Pozn.: ● povinný datový objekt, ○ volitelný (nepovinný) datový objekt

Obr. 2-5: Organizace dat v uzlu typu XCBR podle IEC 61850 [2]



Obr. 2-6: Organizace dat v uzlu XCBR v ochraně REF615 [2]

2.3.3 Logická zařízení (LD)

Logická zařízení sdružují dohromady logické uzly s podobnými rysy. Standard určuje, že každé LD musí obsahovat alespoň tři logické uzly, a to

- LPHD (Physical Device Information) – obecné informace (název, stav, …)
- LLN0 (Logical Node Zero) – obecné informace o jednotlivých uzlech, které dědí všechny ostatní LN v daném LD
- LN (Logical Node) – alespoň jeden uzel, reprezentující konkrétní funkce

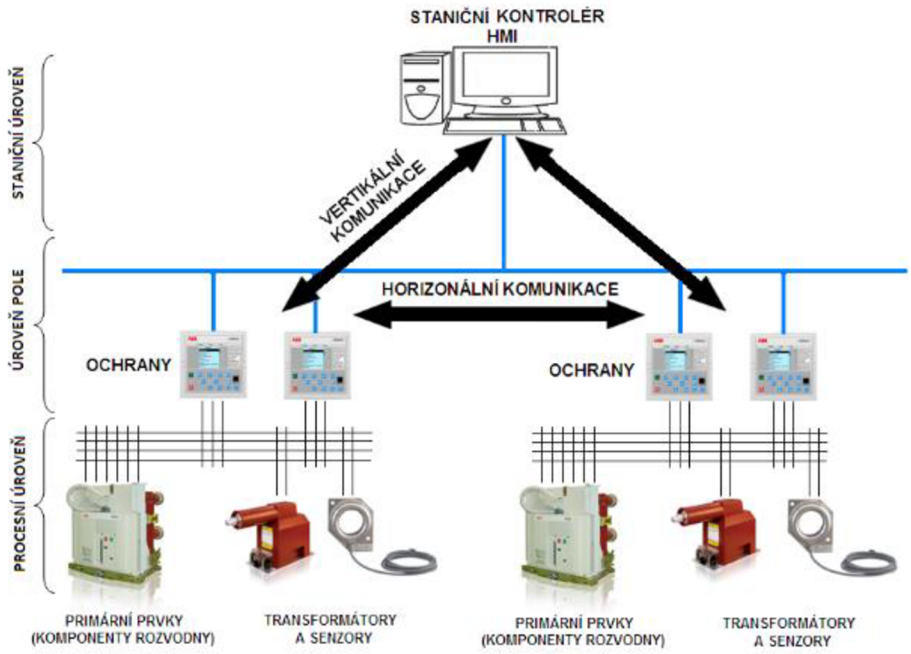
Standard nedefinuje konkrétní typy logických zařízení – jak jejich obsah, tak jmenné konvence jsou v kompetenci výrobce.

2.3.4 Fyzická zařízení (IED)

Fyzické zařízení reprezentuje jeden nezávislý fyzický objekt, dostupný pro vnější klienty přes Ethernet pomocí své IP adresy. Skládá se z libovolného počtu zařízení logických a jde o nejvyšší vrstvu datového modelu.

Každé fyzické zařízení je určené svým názvem. Ten se smí skládat z až deseti znaků a musí být v rámci systému unikátní. Konkrétní jmenné konvence jsou ale v kompetenci výrobce, obdobně jako u zařízení logických.

2.4 Vertikální komunikace



Obr. 2-7: Základní členění komunikace [2]

Vertikální komunikace dle IEC 61850 probíhá v režimu klient-server, kde klientem je nadřízená stanice (řídící PLC, HMI, operátorská stanice) a serverem stanice podřízená (např. ovladač ochran). Používá se zejména po vyčítání časově nekritických dat a předávání jednoduchých dat, zejména v rámci systémů SCADA (Supervisory Control And Data Acquisition). Servery umožňují vyčítání dat až pěti klientům zároveň, ale zapisovat povely smí kvůli bezpečnosti vždy pouze jeden.

Standard IEC 61850 definuje mapování vertikální komunikace na protokol MMS. Jde o protokol sedmé (aplikační) vrstvy ISO/OSI systému, jehož mapování do nižších vrstev zachycuje Obr. 2-8.

Layer	PDU	Protocols
Application (L7)	APDU	Manufacturing Message Specification (MMS): ISO 9506
		Association Control Service Element (ACSE): ISO/IEC 8650/X.227
Presentation (L6)	PPDU	OSI Connection Oriented Presentation ISO 8823/X.226
Session (L5)	SPDU	OSI Connection Oriented Session: ISO 8327/X.225
Transport (L4)	TPDU	Connection-Oriented Transport Protocol: ISO/IEC 8073/X.224
		ISO Transport over TCP (TPKT): RFC 1006
		Transmission Control Protocol (TCP): RFC 793
Network (L3)	NPDU	Internet Protocol (IP): RFC 791
Data Link (L2)	Data Frame	Ethernet: ISO/IEC 8802-3
Physical (L1)	Bits	

Obr. 2-8: Model ISO/OSI při vertikální komunikaci pomocí MMS [1]

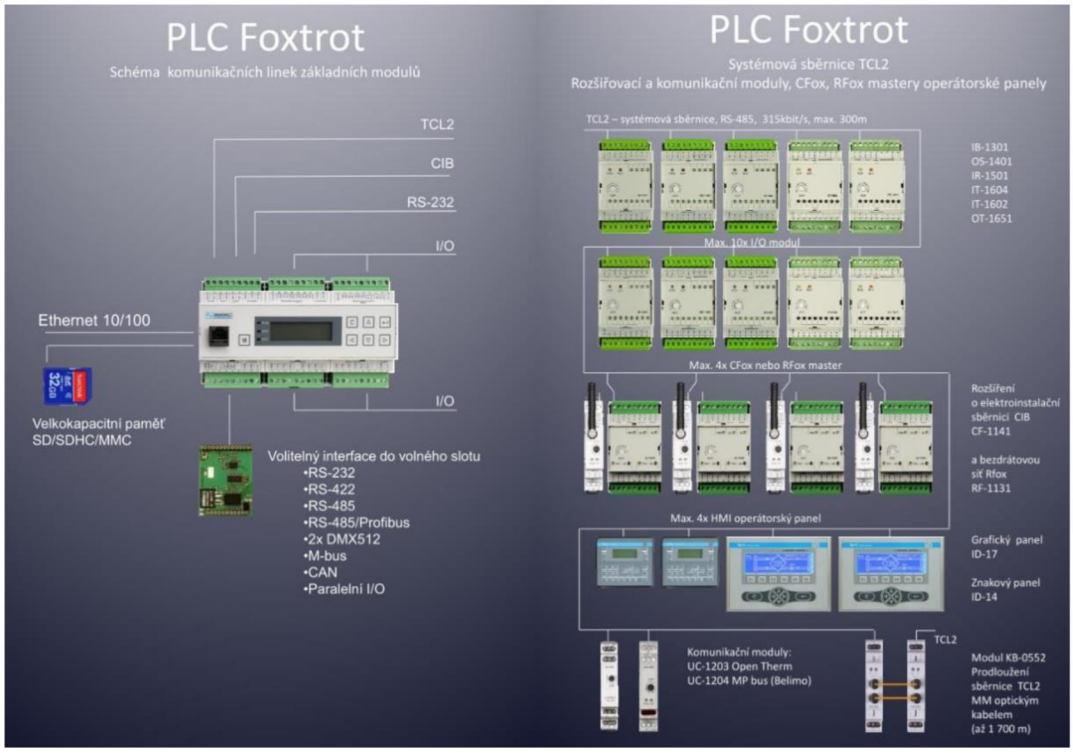
3. PLC TECOMAT

Tato kapitola obsahuje stručný popis použitých automatů a možností jejich programovacího prostředí. Předpokládá základní znalosti programování podle IEC 61131 a zejména jazyka ST (Structured Text) – nepopisuje proto základy jazyka jako cykly, podmínky, práci s proměnnými a podobně, ale zejména komunikační možnosti a rozšíření nad rámec standardu. Pro seznámení se základními pravidly a syntaxí ST viz například nápověda prostředí Mosaic, dostupná například z webových stránek firmy na adrese https://www.tecomat.cz/ke-stazeni/software/mosaic/.

3.1 Tecomat Foxtrot

Foxtrot je řada kompaktních modulárních PLC pro malé a střední aplikace od společnosti Teco a.s. V současnosti je na trhu více než deset různých centrálních modulů (značených CP-10xx) a desítky rozšiřujících modulů pro komunikaci, I/O, napájení a další. Kompletní specifikace programovatelných automatů této řady viz https://www.tecomat.cz/products/cat/cz/plc-tecomat-foxtrot-3/.

Jednou ze silných stránek systémů Foxtrot je komunikace. Centrální moduly obsahují vždy alespoň jeden 100Mb Ethernet a jeden sériový kanál, často více, s možností rozšíření řadou modulů. Programovací prostředí Mosaic navíc ke



Obr. 3-1: PLC Tecomat Foxtrot [8]

komunikačním linkám umožňuje velmi volný přístup – je možné ručně sestavovat zprávy v libovolném formátu, nebo použít některou ze zabudovaných knihoven pro protokoly jako TCP, ModbusRTU, IEC 60870-5, http a další.

3.2 Vývojové prostředí Mosaic

Mosaic je prostředí dodávané firmou Teco, která vyrábí PLC použitá v praktické části této práce. Toto prostředí mimo jiné umožňuje nastavení PLC od této firmy (I/O konfigurace, IP adresa, komunikační linky, …), vytváření rozsáhlých projektů, stažení a nahrání kódu z/do automatu, simulaci, real time debugging a sledování probíhající komunikace.

Hlavní funkcí tohoto prostředí je ale samotné vytváření programů. K tomu podporuje několik programovacích jazyků, běžných pro programovatelné automaty – jazyk reléových schémat (LD), jazyk funkčních bloků (FBD), strukturovaný text (ST), instrukční jazyk (IL) a grafické programování (CFC). Ne všechny jazyky jsou ale podporovány do stejné míry (např. některé funkce jsou dostupné pouze v IL, naopak knihovní bloky jsou často volatelné pouze z ST), navíc je dokumentace pro některé z nich často velmi nedostatečná (a během vytváření praktické části jsem narazil i na několik poměrně zásadních bugů, kdy chování dokumentaci přímo odporuje). Krom možností odpovídajících IEC 61131-3 umožňuje Mosaic použít i některá další rozšíření nad jeho rámec, jako jsou například ukazatele (popsané níže).

Krom základní funkcionality je s prostředím Mosaic dodáváno i velké množství knihoven, které doplňují pokročilejší a specializované funkce – například knihovna ComLib pro jednoduchou sériovou a Ethernetovou komunikaci, InternetLib s implementací protokolů http a SNMP, FileLib pro práci se soubory a desítky dalších. Přehled všech dodávaných knihoven a funkcí viz https://www.tecomat.cz/ke-stazeni/prirucky/prirucky-sw/mosaic-prehled-funkci/.

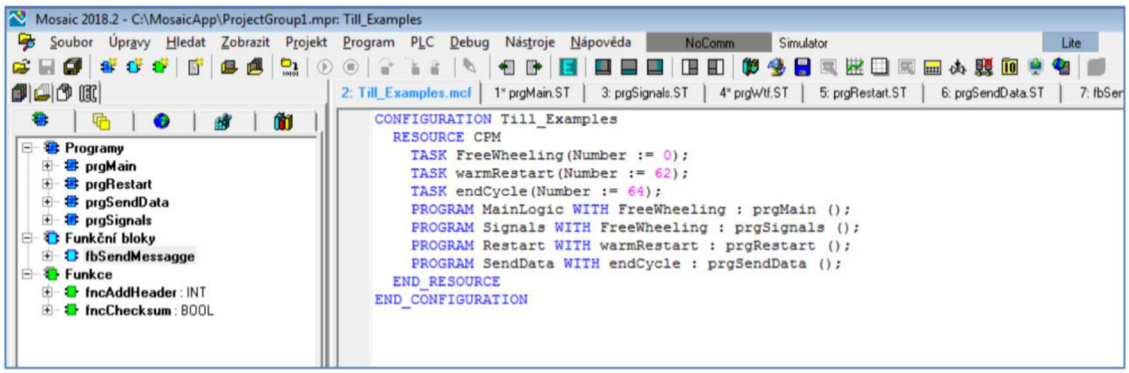
Vývojové prostředí lze stáhnout ve freeware verzi přímo ze stránek výrobce - https://www.tecomat.cz/ke-stazeni/software/mosaic/. Tato verze je ovšem značně omezená, a pro přístup k plné funkcionalitě je nutné zakoupit licenci a použít dodaný hardwarový klíč.

3.2.1 Struktura programu

Procesy v PLC Tecomat se vykonávají cyklicky, jak je pro PLC běžné. Nedochází ale k cyklickému volání jediného hlavního bloku, ale postupně všech podprogramů, které nejsou explicitně nastaveny jinak.

Základním prvkem jsou tzv. POU – Program Organization Units. Hlavním z nich je tzv. program. Jde o ucelený blok kódu s vlastními proměnnými, který je implicitně volaný v každém cyklu automatu (tzv. mód FreeWheeling). Je ale možné nastavit i jiné možnosti, jako je jednorázové volání při studeném/teplém startu, při každém n-tém cyklu, na začátku nebo konci hlavních cyklů nebo třeba v obsluze přerušení.

Zbývajícími dvěma druhy POU jsou funkce a funkční bloky. V obou příkladech jde o bloky volané explicitně programátorem z jiné části kódu, sloužícím zejména pro zlepšení čitelnosti rozdělením na menší, logické celky a omezení opakování kódu. Mohou mít vstupní, vstupně-výstupní a výstupní parametry i lokální proměnné. Liší se tím, že funkční blok je uložený v paměti (jako proměnná v nadřazené jednotce), a může tak mít trvalé i dočasné proměnné, zatímco funkce má proměnné pouze dočasné.



Obr. 3-2: Příklad konfigurace POU

3.2.2 Datové typy, ukazatele

Mosaic podporuje běžné datové typy, jako jsou

- Pravdivostní hodnoty typu BOOL (1 bit)
- Znaménkové celočíselné typy SIN, INT a DINT (8/16/32 bitů)
- Bezznaménkové varianty USINT, UINT, UDINT
- Varianty pro práci s raw pamětí BYTE, WORD, DWORD
- Desetinná čísla REAL a LREAL (32/64 bitů)
- Časové typy TIME, DATE, TIME_OF_DAY a DATE_AND_TIME
- Textové řetězce STRING

Všechny neinicializované proměnné jsou při vytvoření automaticky nulovány. Nulová hodnota absolutních časových typů (DATE_AND_TIME) je začátek epochy (obdobně jako v UNIXu), tj. 1.1.1970, 00:00:00. Neinicializované proměnné typu ukazatel (viz níže) nelze použít, vygenerují compile-time error – je do nich nutné nahrát adresu proměnné.

Oproti standardu IEC 61131-3 ale Mosaic obsahuje ještě jednu třídu proměnných – ukazatele. Ty jsou deklarované jako PTR_TO <TYPE> a chovají se velmi podobně, jako například ukazatele v jazycích rodiny C. Umožňují dereferenci (přístup k odkazovanému prvku) pomocí operátoru ^ a ukazatelovou aritmetiku. K dispozici jsou také pomocné funkce, zejména adr pro načtení adresy objektu, sizeof pro zjištění jeho velikosti a void pro přetypování na generický typ (umožňuje odkazovat proměnné ukazatelem jiného typu; používá se například při volání funkcí jako memcpy, které konkrétní typ nezajímá). Ukazatel je také možné naplnit přímo adresou v paměti (např. %MB100).

Ukazatele souvisí s další skupinou proměnných – polem. To je deklarované jako ARRAY [X..Y] OF <TYPE>, kde <TYPE> je typ proměnné a X, Y jsou indexy prvního a posledního prvku (není tedy nutné číslovat od nuly). K prvkům pole je možné přistupovat operátorem [] nebo pomocí ukazatele. Jednoduchý příklad použití viz Obr. 3-3: Příklad práce s polem a ukazatel.

```
PROGRAM prgMain
  VAR_INPUT
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR
     data : ARRAY [0..4] OF INT; // Inicializováno na [0,0,0,0,0]
     dataLong : ARRAY [5..8]OF DINT := [10, 11, 12, 13]; // Vlastní inicializace
     dataPtr : PTR_TO INT; // Ukazatel
  END_VAR
  VAR_TEMP
     i : INT; // Dočasná proměnná
  END_VAR

     dataPtr := ADR(data); // Získání adresy
     FOR i := 0 TO 4 DO
        dataPtr^ := i; // Zápis hodnoty
        dataPtr := dataPtr + 2 // Posun o 2 byty (velikost INT)
     END_FOR;

     dataPtr := ADR(dataLong);
     FOR i := 5 TO 7 DO
        dataPtr^ := dataPtr^ - 1; // Sníží hodnotu o 1
        dataPtr := dataPtr + SIZEOF(DINT); // Posun o 4 byty
     END_FOR;

     dataLong[8] := 14; // Přímá adresace v poli


END_PROGRAM
```

Obr. 3-3: Příklad práce s polem a ukazateli

3.2.3 Vlastní datové typy

Pro programování v Mosaicu je velmi užitečná možnost vytvářet vlastní datové typy. Ty lze použít jak pro jednoduché sdružení souvisejících dat do jedné struktury (například TEthStat, obsahující všechny informace o ethernetovém kanálu), tak pro

komplexní struktury jako pole napaječů. Krom toho umožňují vlastní typy také efektivní využití paměti (seřazení dat tak, aby využili zarovnání v paměti), rezervování prostoru pro další rozšíření (vytvořením skrytých dummy položek) a snadný přístup k položkám jak přes jejich jména, tak přes ukazatelovou aritmetiku.

Uživatelské typy podporují také další možnosti, jako type nesting (vytváření vnořených struktur), skrytí vybraných položek direktivou {HIDDEN} nebo sdílení dat mezi automaty direktivou {PLCNET}.

Pro příklad definice datových struktur viz Obr. 3-4: Uživatelské typy (na základě knihovny PlcNetBasic), který obsahuje krátký úryvek z definice struktur

```
_TPlcNetFrameStat : STRUCT                    (* struktura stavu rámce*)
  TrigPending          : bool;                  (* programově spuštěné přijímání dat probíhá*)
  LocsingComm          : bool;                  (* komunikace selhala nejméně 2 krát za sebou*)
  FreshData            : bool;                  (* proměnné rámce mají nová data (puls po dobu jedné programové smyčky)*)
  FailedGroups         : bool;                  (* inicializace některé skupiny rámce selhala*)
  // Kráceno
  FrameDataError       : bool;                  (* chyba inkonzistence dat rámce*)
  LastDataTimeStamp    : dt;                    (* datum a čas posledního čtení dat rámce*)
  dwReserve   (HIDDEN) : dword;                 (* rezerva*)
END_STRUCT;

_TPlcNetPlcStat : STRUCT                      (* struktura stavu automatu*)
  Frame1               : _TPlcNetFrameStat;     (* struktura stavu rámce 1*)
  Frame2               : _TPlcNetFrameStat;     (* struktura stavu rámce 2*)
  Frame3               : _TPlcNetFrameStat;     (* struktura stavu rámce 3*)
  Frame4               : _TPlcNetFrameStat;     (* struktura stavu rámce 4*)
  // Kráceno
  SentCommunicationsOvf: bool;                  (* počitadlo 'SentCommunications' přeteklo*)
  HasCovVariables      : bool;                  (* existuje nejméně jedna proměnná typu COV*)
  HasCovRejected       : bool;                  (* existuje nejméně jedna proměnná typu COV odmítnutá serverem*)
  FreshCovData         : bool;                  (* nejméně jedna proměnná typu COV má nová data (puls po dobu jedné programové smyčky)*)
  MissedCovBroadcast   : bool;                  (* jedno nebo více COV vysílání ze nebylo doručeno (puls po dobu jedné programové smyčky)*)
  FrameDataError       : bool;                  (* chyba inkonzistence dat nejméně jednoho rámce*)

  LastDataTimeStamp    : dt;                    (* datum a čas posledního čtení dat*)
  LastCovDataTimeStamp : dt;                    (* datum a čas posledního přijetí COV dat*)

  IndexOfPlc           : usint;                 (* index automatu*)
  byReserve            : byte;                  (* rezerva*)
  ServerTdiVersion     : word;                  (* verze TDI serveru automatu*)
  wReserve             : word;                  (* rezerva*)
  SentCommunications   : udint;                 (* celkový počet komunikací vyslaných klientem*)
  LostCommunications   : udint;                 (* počet nezodpovězených komunikací klienta*)
  LostConnections      : udint;                 (* počet ztracených spojení s automatem*)
  RecCovBroadcasts     : udint;                 (* počet přijatých COV vysílání*)
  dwReserve            : dword;                 (* rezerva*)
END_STRUCT;
```

Obr. 3-4: Uživatelské typy (na základě knihovny PlcNetBasic)

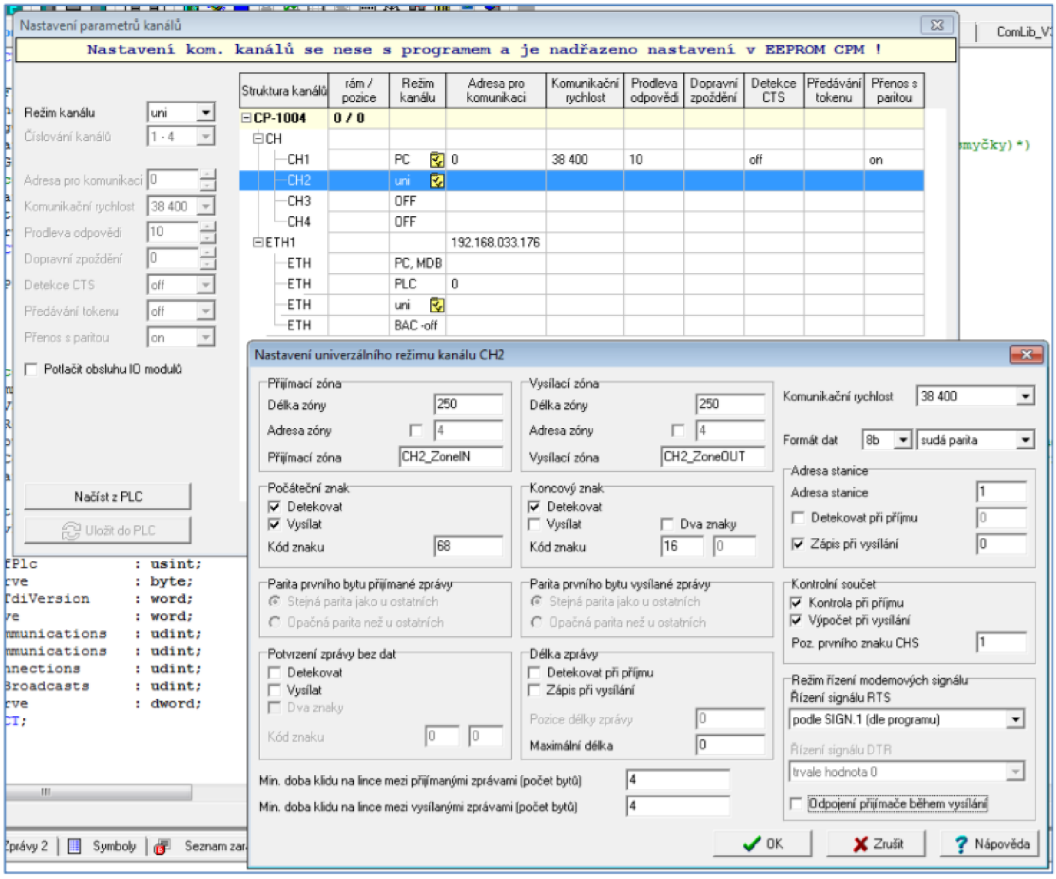
používaných knihovnou PlcNetBasic.

3.2.4 Nastavení komunikace

Jak bylo zmíněno v úvodní podkapitole, automaty Tecomat umožňují velkou volnost při nastavování komunikačních kanálů. K základnímu nastavení slouží dialogové okno Projekt/Manažer Projektu/Konfigurace HW. Zde je možné nastavit některý z předkonfigurovaných režimů, jako je komunikace s PC, jiným PLC, protokoly MPC a MDB pro sériové kanály nebo BACnet pro ethernet. Krom toho je možné zvolit režim uni, který poskytuje přímý přístup k datům a širokou škálu nastavení, jak ukazuje Obr. 3-5: Parametrizace sériového kanál.

Vzhledem k rozsáhlým možnostem a množství zabudovaných knihoven jsou PLC od firmy Tecomat velmi vhodné pro ovládání různých zařízení, komunikujících pomocí nekompatibilních protokolů. Umožňuje jak snadnou komunikaci pomocí běžných protokolů jako například MODBUS, tak implementaci funkcí pro komunikaci protokoly proprietárními či nepodporovanými, jako je DBnet firmy

AMIT automation nebo IEC 61850. K tomu je ale nutné vytvořit ovladače, které komunikaci překládají – jako je pro IEC 61850 cílem této práce.

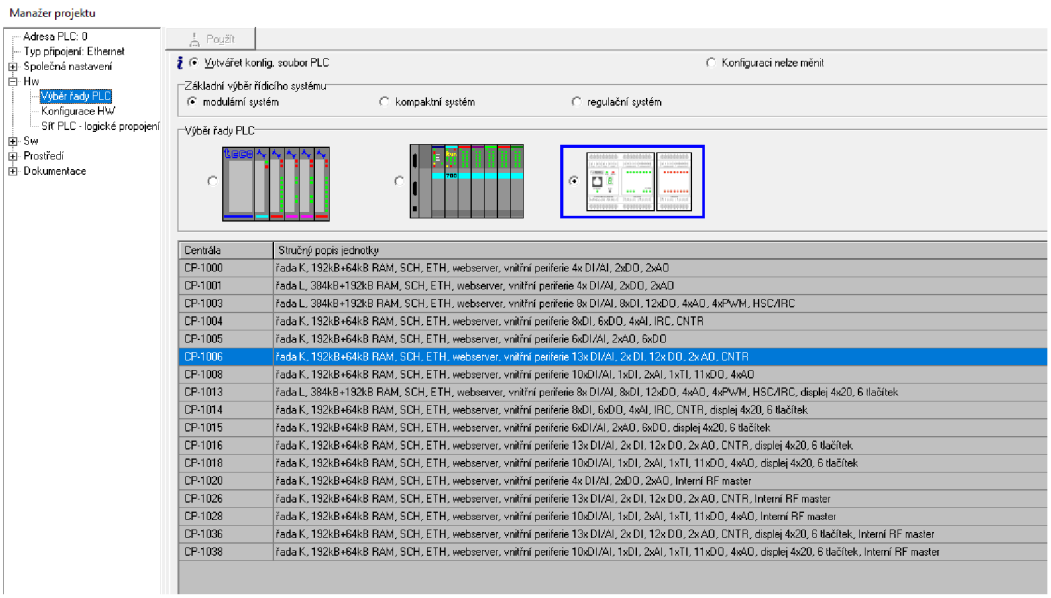


Obr. 3-5: Parametrizace sériového kanálu

3.2.5 Hardwarová konfigurace

Programy vytvořené v prostředí Mosaic jsou přenositelné mezi většinou PLC od firmy Teco (pokud jsou použité funkce v daném PLC podporované). Velmi podobné jsou zejména modulární řady Foxtrot a TC700, na které je tato práce cílená – přenos kódu mezi jednotlivými centrálními jednotkami těchto řad je poměrně bezproblémový.

Nicméně, projekty v Mosaicu jsou vázané na konkrétní typ zařízení a připojené moduly – pro přenos na jiný modul je nutné upravit hardwarovou konfiguraci. K tomu slouží menu Projekt/Manažer projektu, záložka Hw, jak ukazuje Obr. 3-6. V něm je nutné vybrat řadu a model centrální jednotky v záložce Výběr řady PLC, poté nakonfigurovat vstupy, výstupy a připojené moduly v záložce Konfigurace HW, a na závěr zkontrolovat komunikační nastavení (viz předchozí podkapitola, jsou přístupné proklikem z Konfigurace HW pomocí žluté ikonky napravo od nápisu CPU) – je možné, že při změně konfigurace se změní i ty, a je nutné je znovu nastavit.



Obr. 3-6: Hardwarová konfigurace

4. PROTOKOLY NIŽŠÍCH VRSTEV

Jak vyplývá z předchozích kapitol, pro vertikální komunikaci dle IEC 61850 je nutné implementovat protokol MMS. Jelikož jde o protokol aplikační (sedmé) vrstvy ISO/OSI, ale Tecomat nativně podporuje pouze TCP/IP ve vrstvě čtvrté, je nutné vytvořit jednoduchou implementací i pro protokoly ostatních vrstev. Ve většině z nich ale stačí triviální implementace, jelikož stačí vytvořit paket žádající o vytvoření komunikace, přijmout jeho potvrzení a doplnit všechny následující pakety o statickou hlavičku (data paket), jak popisují následující podkapitoly.

4.1 Transportní vrstva (L4)

Na čtvrté vrstvě používá implementace MMS protokol TPC/IP, který je v PLC Foxtrot/TC700 nativně podporován. Pro diskretizaci paketů ale používá ještě další dva protokoly – TPKT a COTP. Definují je standardy RFC-1006 a ISO 8073, rozlišující desítky druhů paketů. Nicméně, při komunikaci pomocí MMS jsou využity pouze tři z nich, a to

- Connection Request (CC) – navázání spojení v transportní vrstvě
- Connection Confirm (CR) – potvrzení spojení v transportní vrstvě
- Data (DT) – přenos zpráv z vyšších vrstev

4.2 Relační vrstva (L5)

Relační vrstva využívá OSI Connection Oriented Session, definovanou standardem ISO/IEC 8327/X.225. Ten definuje velké množství různých SPDU (session protocol data units), ze kterých ale MMS opět využívá pouze tři

- Connect (ID 13) – navázání spojení v relační vrstvě
- Accept (ID 14) – potvrzení spojení v relační vrstvě
- Give Tokens/Data Transfer (ID 1) – přenos zpráv z vyšších vrstev

4.3 Prezentační vrstva (L6)

Obdobně jako v předchozích vrstvách, používáme tři PPDU (presentation protocol data units) definovaných v OSI Connection Oriented Presentation, kterými jsou

- Connect Presentation – navázání spojení v prezentační vrstvě
- Connect Presentation Accept – potvrzení spojení v prezentační vrstvě
- CPC-type – přenos zpráv z vyšších vrstev

V rámci prezentační vrstvy je také definované kódování zbývajících dat. Standard určuje, že pro MMS komunikaci bude použito ASN.1 kódování dle pravidel BER – viz kapitola 4.5.

4.4 Aplikační vrstva (L7)

Protokol MMS je v této vrstvě zapouzdřen do APDU (application protocol data units) definovaných standardem Asssociation Control Service Element (ACSE). V této vrstvě použije pouze dva typy paketů, vyslané v úvodní části komunikace v rámci Connect a Acceptr PPDU.

- AARQ – navázání spojení
- AARE – potvrzení spojení

Další komunikace už ACSE nevyužívá a MMS pakety následují přímo za hlavičkou prezentační vrstvy.

Standard MMS pak definuje 14 různých PDU (protocol data units). Na ty jsou přímo mapované jednotlivé funkce protokolu IEC 61850. Běžně jsou využívané

- Confirmed-RequestPDU – např. funkce read, write, identify
- Confirmed-ResponsePDU – odpovědi na Confirmed-RequestPDU
- Confirmed-ErrorPDU – hlášení, že Confirmed-RequestPDU selhal
- RejectPDU – zpráva byla odmítnuta (většinou značí špatně formátovaný paket)
- UnconfirmedPDU – jednostranná zpráva, zejména zasílání reportů
- Initiate-RequestPDU – žádost o navázání spojení
- Initiate-ResponsePDU – potvrzení navázání spojení

4.5 ASN.1, kódování BER

Standard MMS také určuje, že přenášená data budou ukládána podle pravidel ASN.1, konkrétně BER (Basic Encoding Rules). Každá MMS zpráva je tak popsaná pomocí ASN.1 syntaxe, jak ukazuje Obr. 4-1. Kompletní definice MMS v ASN.1 syntaxi viz [13].

```
-------------------------------- READ --------------------------------

Read-Request ::= SEQUENCE
        {
        specificationWithResult         [0] IMPLICIT BOOLEAN DEFAULT FALSE,
        variableAccessSpecificatn       [1] VariableAccessSpecification
        }


Read-Response ::= SEQUENCE
        {
        variableAccessSpecificatn [0] VariableAccessSpecification OPTIONAL,
        listOfAccessResult        [1] IMPLICIT SEQUENCE OF AccessResult
        }

-------------------------------- WRITE --------------------------------


Write-Request ::= SEQUENCE
        {
        variableAccessSpecificatn       VariableAccessSpecification,
        listOfData                      [0] IMPLICIT SEQUENCE OF Data
        }

Write-Response ::= SEQUENCE OF CHOICE
        {
        failure         [0] IMPLICIT DataAccessError,
        success         [1] IMPLICIT NULL
        }
```

Obr. 4-1: ASN.1 [13]

Kódování BER pak určuje, jak konkrétně jsou jednotlivé typy ukládány – používá takzvaný TLV (type-length-value) formát, ve kterém je každý prvek zakódován pomocí tripletu (tří polí) – typ, délka a hodnota. Definuje různé globální datové typy (BOOLEAN, INTEGER, VISIBLE STRING, …), a umožňuje definovat vlastní, složité typy pomocí standardizované syntaxe, jak ukazuje Obr. 4-1.

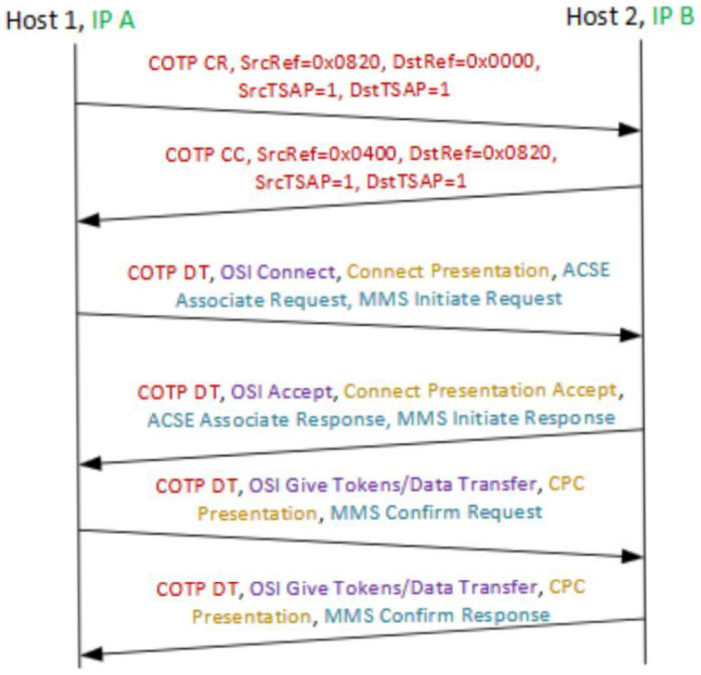
Pro komplexní realizaci IEC 61850 by bylo nutné vytvořit BER enkodér a dekodér pro vytváření a parsování zpráv, navíc schopný pracovat rekurzivně (jelikož datové typy často obsahují vložené typy, jednoduchým příkladem jsou například data typu struct), který není triviální (implementace IEC 61850 na jiných platformách většinou používají již existující implementace ASN.1, pro PLC Tecomat ale bohužel v současnosti nic takového neexistuje). Nicméně, vzhledem k tomu, že budeme používat pouze malou podmnožinu všech typů a paketů, implementace kompletní parser neobsahuje – s BER formátem se pracuje pouze na několika místech, a většina z nich je konstantní – proto bylo snazší implementovat pouze enkodér schopný přenášet jednoduché datové typy, a rekurzivní/složené typy nepoužívat (je možné je přenést po jednotlivých položkách).

4.6 Průběh komunikace

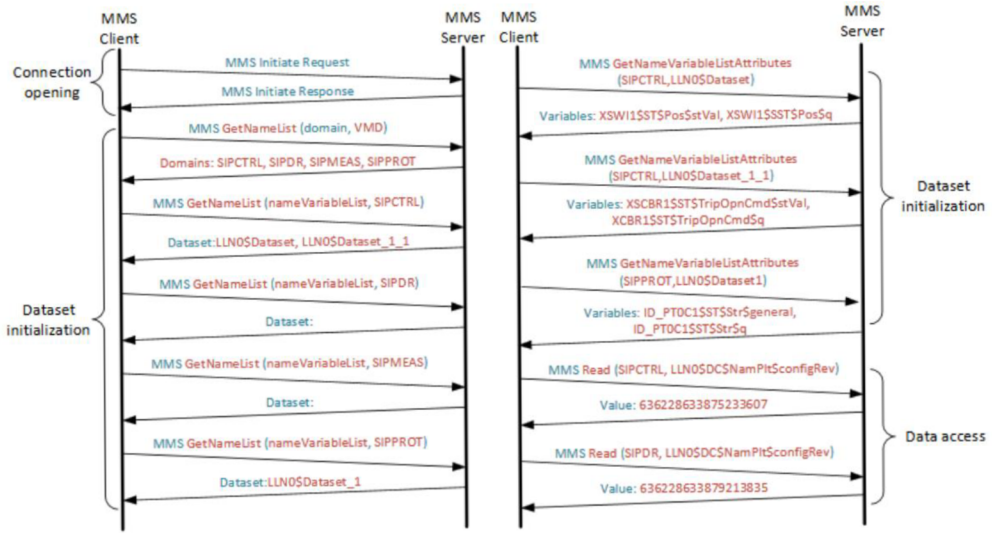
Vertikální komunikace dle IEC 61850 začíná navázáním spojení přes TCP/IP. Poté dojde k

- Navázání spojení v transportní vrstvě – výměna zpráv COTP CR a CC
- Vyslání paketu MMS Initiate-RequestPDU, který zároveň obsahuje pakety Connect v L5, L6 a L7
- Odpověď Initiate-ResponsePDU, která zároveň obsahuje odpovědi Accept v L5, L6 a L7
- Samotná komunikace, probíhající jako
    - Výměna paketů Confirmed-RequestPDU a Confirmed-ResponsePDU, případně Confirmed-ErrorPDU
    - Jednostranné zasílání reportů serverem pomocí Unconfirmed-PDU (například při použití reportu při změně proměnné)
- Ukončení komunikace přerušením spojení TCP/IP¨
    - Standard definuje i jednoduché pakety typu ConcludePDU, které požádají o nenásilné ukončení komunikace a uvolnění zdrojů k ní využitých. Nicméně, testované stanice tuto zprávu nevyžadovaly a ani při dlouhodobějším testování neměly viditelný problém s násilným ukončováním spojením přerušením TCP kanálu, zůstal jsem tedy u toho.

Výše zmíněný průběh ilustruje Obr. 4-2, který zachycuje výměnu inicializačních paketů a jednoduchý datový přenos. Obr. 4-3 pak ukazuje složitější komunikaci, kde po navázání komunikace pokračuje výměna paketů Confirmed-Request a Confirmed-Response.



Obr. 4-2: Navázání spojení [1]



Obr. 4-3: Ukázka výměny dat [1]

5. PRAKTICKÁ IMPLEMENTACE

Cílem práce bylo implementovat funkce pro vertikální komunikaci s podřízenou stanicí, která pracuje v režimu server. Celý protokol IEC 61850 je velmi rozsáhlý a pro dosažení tohoto cíle stačí implementovat pouze poměrně malou podmnožinu. Další zjednodušení vyplývá z toho, jakého charakteru je použitý hardware a zamýšlené využití, pro které některé funkce nedávají velký smysl. Tím se zabývá následující podkapitola.

5.1 Rozbor okolí systému

Velký vliv na výslednou implementaci má to, že předpokládáme nasazení PLC v automatizovaných průmyslových procesech, bez nutnosti (a často možnosti) zásahů lidského operátora. Z toho vyplývá mj. to, že celý systém by měl být znám již v době programování a nakonfigurován staticky. Z toho vyplývá, že funkce jako getName, getNameList nebo getVariableAccessAttributes nemají větší význam, jelikož jejich výsledky jsou programátorovi předem známy. To je navíc podporováno i tím, že použitá PLC neumožňují žádnou formu dynamické alokace, takže je velmi důležité, že již při psaní programu známe typ a velikost zasílaných objektů.

Krom toho nemají nasazená PLC ani žádnou nativní podporu pro interakci s lidským uživatelem. Nejpravděpodobnějším použitím je nasazení jako prostřední článek v systémech SCADA, kde PLC sbírá a zpracovává informace z podřízených ochran (a případně dalších druhů zařízení) a odesílá je do vyšších systémů, nejčastěji vizualizace. To opět podporuje závěr předchozího odstavce, že předávaná data by měla být známá již v době kompilace.

Na základě těchto informací budou důležité zejména funkce

- Connect – propojí všechny vrstvy ISO/OSI, umožní zasílání ostatních zpráv (zahrnuje výměnu COTP CR a CC a poté MMS Initiate Request a Response – viz kapitola 4.6)
- Read, Write – odešlou MMS Confirmed Request, aktivují čekání na Response
    - Jako jeden z parametrů berou název atributu (viz kap. 2.3),
- Response Handler – zpracovává zprávy typu Confirmed Response, Reject a Error, data uloží podle metainformací nastavených při volání Read/Write

Tyto základní funkce umožňují kompletní ovládání podřízené stanice, když známe její datový strom. Případné rozšíření by bylo možné přidáním zpracování UnconfirmedPDU do handleru, což by umožnilo využít změnové reporty a

zefektivnit komunikaci (nebylo by nutné cyklické vyčítání), problémem je ale přiřazení obdržených dat do odpovídající proměnné (na rozdíl od modelu request-response předem nevíme, kam data ukládat).

5.2 Získání dat, postup práce

Pro vytvoření programu bylo nutné seznámit se se standardem, vytvořit zkušební pakety, otestovat jejich funkčnost a na závěr implementovat funkce, které pakety v odpovídajícím formátu vytvoří. K tomu jsem postupoval v několika krocích

- Seznámení se s formátem – zejména díky [1] a [2], přečtením některých částí standardu.
- Zachycení vzorových paketů – využil jsem testovací programy IEDexplorer a ITT600 SA Explorer, které umožňují simulovat klienta IEC 61850 na PC. Toto PC jsem pak propojil se serverem REF630, zahájil komunikaci a pomocí programu Wireshark zachytil pakety pro analýzu.
- Na základě předchozích dvou bodů jsem vytvořil několik zkušebních paketů. V PC jsem pak pomocí programu Hercules vytvořil TCP client, který jsem opět propojil s REF630 a zjišťoval, jestli na dané pakety dostanu očekávanou odpověď.
- Po otestování paketů různých formátů jsem vytvořil funkční bloky pro PLC Tecomat, které odpovídající pakety generovaly. Tyto bloky jsem nahrál do automatu Foxtrot CP-1006, který jsem opět propojil s PC, kde jsem v programu Hercules vytvořil TCP server a testoval, jak PLC komunikuje a reaguje na různé zprávy.
- Na závěr jsem propojil Foxtrot a REF630 a sledoval, jestli komunikace probíhá dle očekávání. Zasílané pakety jsem také zkontroloval Wiresharkem, abych měl jistotu, že odpovídají formátu MMS.

Popsaný postup jsem opakoval postupně pro pakety Initiate, Read i Write.

5.3 Základní struktura programu

Program se skládá ze čtyř hlavních funkčních bloků. Jsou jimi

- fbConnect – naváže spojení s cílovou stanicí, připraví komunikaci
- fbRead – vyšle žádost o čtení, připraví přijetí odpovědi
- fbWrite – vyšle zápis dat, připraví přijetí potvrzení
- fbResponseHandler – při každém zavolání projde přijaté zprávy, zpracuje případné Error-PDU a pokud je nastaveno čekání na odpověď a přišla odpovídající zpráva (sedí typ a invokeID – viz níže), zpracuje ji

K tomu obsahuje ještě několik pomocných funkcí pro extrakci opakujícího se kódu

- fncCreateHeader – zkontroluje stav stanice, na začátek dodaného pole zapíše hlavičky všech nosných protokolů od TPKT po MMS
- fncInsertName – na dodanou adresu vloží jméno proměnné dle IEC 61850 v TLV formátu, jak jej definuje ASN.1 / BER

Všechny funkce berou jako jeden z parametrů datový typ IEC_station, který v sobě uchovává všechna metadata týkající se dané stanice, jako je její stav, IP adresa, číslo současné zprávy, zachycené chyby a jiné. Detailněji budou popsány v následujících podkapitolách.

5.4 Datový typ IEC_station

Pro uchovávání informací o stanici slouží struktura IEC_station. Její obsah ukazuje Obr. 5-1. Následující podkapitoly popisují jednotlivé složky.

```
IEC_station : STRUCT
   name     : STRING;
   ethAdr   : TRemoteEthAdr;
   state    : IEC_state := off;
   errors   : IEC_errors;
   invokeID : USINT;

   lastARC {HIDDEN} : BOOL;
   dstRefU {HIDDEN} : USINT;
   dstRefL {HIDDEN} : USINT;
END_STRUCT;
```

Obr. 5-1: Struktura IEC_station

5.4.1 Položka name, typ STRING

Obsahuje jméno stanice dle IEC 61850, tj. jméno fyzického zařízení. Využívá se při sestavování zpráv read a write k vytvoření jména proměnné (viz popis datového modelu).

Použitý datový typ STRING umožňuje ukládání textových informací o libovolné délce. Na jednotlivé znaky je možné odkazovat ukazatelem typu PTR_TO USINT a iterovat pomocí ukazatelové aritmetiky – toho využívám k přenosu textových informací do polí USINT, které používají komunikační kanály. K zjištění délky řetězce slouží funkce LEN.

Při zapisování řetězců existuje několik řídících znaků ($, ‘), které nelze napsat přímo – je nutné před ně přidat tzv. escape character, kterým je v Mosaicu znak $.

5.4.2 Položka ethAdr, typ TRemoteEthAdr

Typ TRemoteEthAdr, definovaný v knihovně ComLib, který má položka ethAdr, slouží pro uložení vzdálené IP adresy. Má tři části

- remoteIP – typ ARRAY [0..3] OF USINT, udává adresu vzdálené stanice (vychází z topologie sítě)
- remotePort – typ UINT, udává číslo vzdáleného portu (pro MMS port 102)
- localPort – typ UINT, udává číslo místního portu (pro MMS port 102)

Jak je z výše uvedeného zjevné, programátor by měl měnit pouze hodnotu remoteIP, ostatní jsou napevno nastaveny na 102.

5.4.3 Položka state, typ IEC_state

IEC_state je výčtový typ (enum), definovaný pro potřeby této práce. Představuje řídící proměnnou stavového automatu, který určuje stav stanice. Může nabývat hodnot

- error – kritická chyba, ze které se nelze vzpamatovat. Dojde k uzavření komunikačního kanálu a nedojde k žádným dalším pokusům se stanicí komunikovat.
- reset – žádost o restartování komunikace, tj. vynulování metadat, uzavření kanálu a opětovné připojení. Používá se pro řešení nekritických chyb.
- off – stanice je vypnutá, metadata prázdná, kanál zavřený. Výchozí stav, ze kterého dochází k zahájení komunikace (voláním fbConnect).

- tcpReq – byl odeslán požadavek k otevření TCP kanálu, čeká se na odpověď
- tcpEstab – TCP kanál je otevřen
- cotpReq – byla odeslána zpráva COTP CR, čeká se na potvrzení
- cotpEstab – byla přijata zpráva COTP CC, L4 je propojena
- mmsReq – byla odeslána zpráva MMS Initiate-RequestPDU, čeká se na potvrzení

- mmsReady – byla přijata zpráva MMS Initiate-ResponsePDU, je navázáno spojení. Jde o závěrečný stav fbConnect, značí jeho úspěšné zakončení. Zároveň jde o výchozí stav pro volání fbRead a fbWrite.
- readReq – byla vyslána žádost o vyčtení dat, čeká se na data
- writeReq – byla vyslána data k zápisu, čeká se na potvrzení zápisu

Tato položka je používána zejména uvnitř vytvořených funkcí, k jejím úpravám vnějším kódem by mělo docházet pouze výjimečně (například pro násilné vypnutí stanice nebo přechod do chybového stavu, řešení timeoutů). Ke zpracování žádostí o vypnutí při stavech reset a error dojde při nejbližším volání kteréhokoliv z funkčních bloků.

5.4.4 Položka errors, typ IEC_errors

Položka errors slouží k zaznamenání chyb, které v průběhu komunikace nastaly. IEC_errors není výčtový typ, ale agregát, aby nedocházelo k přepsání starších chyb novějšími jiného typu – jediný způsob, jak chybu vynulovat, je explicitním nastavením odpovídající položky na hodnotu FALSE. Pomocí přetypování na UINT je také možné kontrolovat bezchybovost porovnáním s 0 a vynulovat celé pole zápisem 0. Definici struktury IEC_errors zachycuje Obr. 5-2.

```
IEC_errors : STRUCT
   connErr     : BOOL;
   connTout    : BOOL;
   readErr     : BOOL;
   readTout    : BOOL;
   writeErr    : BOOL;
   writeTout   : BOOL;
   mmsErr      : BOOL;
   mmsReject   : BOOL;

   unknownMes       : BOOL;
   dummy10 {HIDDEN} : BOOL;
   dummy11 {HIDDEN} : BOOL;
   dummy12 {HIDDEN} : BOOL;
   dummy13 {HIDDEN} : BOOL;
   dummy14 {HIDDEN} : BOOL;
   dummy15 {HIDDEN} : BOOL;
   dummy16 {HIDDEN} : BOOL;
END_STRUCT;
```

Obr. 5-2: Struktura IEC_errors

Význam jednotlivých položek je

- connErr – došlo k chybě při navazování spojení (na žádost o propojení v některé z vrstev přišla záporná odpověď nebo odpověď v neočekávaném formátu. Pokud nedošlo k přepsání stavu, bližší informace může poskytnout proměnná state, která zachytí, ve které z vrstev k chybě došlo.)
- connTout – při čekání na spojení nastal timeout
- readErr – místo vyčtených dat přišel ErrorPDU
- readTout – při čekání na čtení nastal timeout
- writeErr – místo potvrzení zápisu přišel ErrorPDU

- writeTout – při čekání na zápis nastal timeout
- mmsErr – přišel paket ErrorPDU (server nemůže odpovědět – např. problém na straně serveru, dotaz na neexistující proměnnou, …)
- mmsReject – přišel paket RejectPDU (pravděpodobně odpověď na paket ve špatném formátu)
- unknownMes – došla zpráva, jejíž hlavička neodpovídá MMS formátu
- dummyXY – jde o skryté položky, které slouží k doplnění délky na 16 bitů (2B). Zaručují zarovnání paměti, umožňují pracovat se strukturou přetypováním na UINT a je možné je využít pro potenciální rozšíření, kdyby bylo vhodné přidat další typy chyb.

5.4.5 Položka InvokeID, číslování paketů

Jelikož pro většinu komunikace využíváme dvojici zpráv ConfirmedRequest – ConfirmedResponse, musíme být schopní přiřadit ke každé žádosti odpovídající odpověď. K tomu protokol MMS využívá tzv. InvokeID, což je celé číslo, které je umístěné na začátku každého Confirmed paketu a musí být shodné pro žádost a odpověď. V implementaci mu odpovídá položka InvokeID struktury IEC_station, která je typu USINT (bytové bezznaménkové číslo) a udává číslo žádosti. Platí pro ni, že

- Při vyslání žádosti se InvokeID nemění, hodnota odpovídá číslu právě vyslané zprávy
- Při zachycení odpovědi či chyby dojde ke kontrole, že její InvokeID odpovídá očekávanému. Pokud ano, dojde ke zpracování a inkrementaci o jedna, takže je připraveno pro odeslání zprávy nové.
- Při dosažení ID 255 dojde k přetečení a počítá se opět od nuly

Zobrazení InvokeID má význam zejména pro debugging a analýzu komunikace, a vnější kód by neměl mít důvod jej měnit.

5.4.6 Interní položky struktury IEC_station

Zbývající položky struktury IEC_station jsou skryté, jelikož jde o vnitřní implementační proměnné a pro vnější kód nejen nemá smysl s nimi manipulovat, ale jejich úpravy by mohly způsobit fatální chyby.

Proměnná lastARC slouží pro práci s ethernetovým kanálem. Kvůli nepříliš důsledné dokumentaci a zvláštnímu chování bloku fbReceiveFrom z knihovny ComLib je použita manuální práce s komunikačním kanálem. K tomu je nutné sledovat změny bitu ARC, který alternuje pokaždé, když přijde nová zpráva. Aby bylo

možné změnu poznat, je nutné ukládat si poslední hodnotu, což zajišťuje právě proměnná lastARC.

Proměnné dstRefU a dstRefL slouží k uložení horního a spodního bytu pole dstRef v protokolu COTP, které určuje adresu vzdálené stanice. Slouží právě pro identifikaci stanice v rámci tohoto protokolu. K nastavení těchto proměnných dochází automaticky při přijetí zprávy COTP CC.

5.4.7 Proměnná IEC_readPtr

Mezi skryté položky IEC_station by v ideálním případě patřil ještě ukazatel readPtr, sloužící k přenosu adresy k uložení vyčtených dat mezi fbRead a fbResponseHandler. Bohužel, Mosaic neumožnuje přidání typů PTR_TO * mezi prvky struktury, a proto tento přístup není možný. IEC_readPtr je proto definován jako skrytá globální proměnná – toto řešení je dostatečné pro vyčítání jedné stanice, ale představuje problém pro čitelnost a rozšiřitelnost – momentálně není možné mít vyslaných několik žádostí o čtení do různých stanic současně. To ve většině případu není problém, jelikož zejména modely Foxtrot mají často jediný ethernetový kanál a stejně tak souběžné vyčítání neumožňují, ale pravděpodobně by bylo lepší tuto část upravit tak, aby souběžnost umožňovala.

5.5 fbConnect, navázání spojení

Funkční blok fbConnect slouží k otevření komunikačního kanálu. Jeho jediným parametrem, pojmenovaným station, je struktura typu IEC_station, která obsahuje data o cílové vzdálené stanici. Celý blok tvoří jediný příkaz SWITCH, který na základě stavu stanice určí, jaký krok provést.

Postupným voláním tohoto bloku dochází k

- Vyřešení chybových stavů (uzavření kanálu, vynulování dat; přepnutí na stav off pro řešitelné chyby (reset))
- Navázání TCP spojení se zadanou adresou
- Odeslání zprávy COTP CR (žádost o navázání spojení v L4)
    - Jelikož zpráva CR je při každém volání stejná, je trvale uložena v globální paměti (ideální by byla paměť globální konstantní, z té ale bohužel MOSAIC neumožňuje načítání zpráv)
- Zpracování COTP CC (potvrzení spojení), zaznamenání případných chyb
    - Probíhá kontrola délky zprávy a typu paketu (0xd0)
- Odeslání zprávy MMS Initiate-Request (žádost o spojení v L5, L6, L7)
    - Obdobně jako u COTP je zpráva uložena a vyčítána z globální paměti
- Zpracování MMS Initiate-Response, zaznamenání případných chyb

    - Kontrolují se typy. Ignorují se dohodnuté parametry jako maximální zanoření, počet souběžných volání aj., jelikož vzhledem k použitému formátu komunikace nemají vliv.
- Přepnutí do režimu mmsReady, kdy další volání fbConnect nemají vliv a je možné volat fbRead a fbWrite

V případě, že při některém kroku dojde k chybě, je zapsáno do odpovídající položky IEC_station.errors a navazování spojení je ukončeno (přepnutím do stavu reset). Pomocí globálních konstant je také možné nastavit délku timeoutů, po kterou se bude čekat na dokončení připojení.

5.6 Hlavička paketů, fncCreateHeader

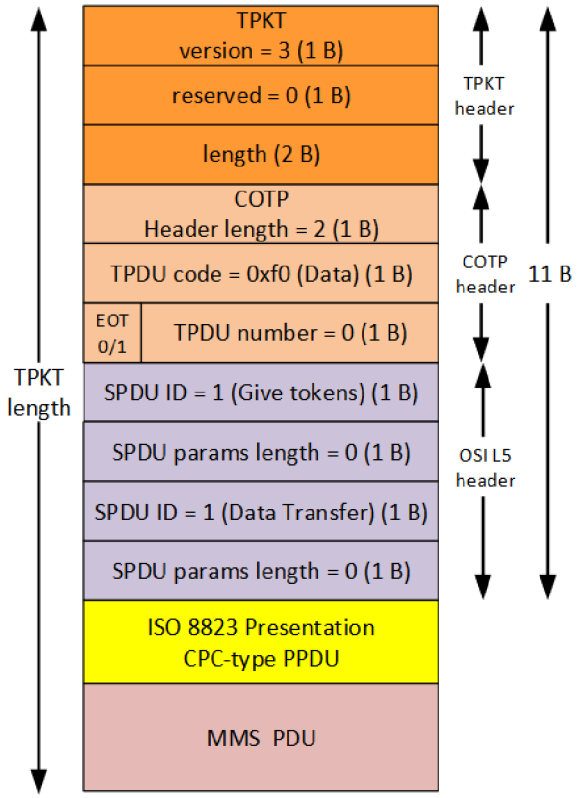
Jelikož hlavičky nosných protokolů jsou neměnné nezávisle na konkrétní IEC 61850 funkci, kterou přenáší (mění se pouze délka), je vhodné vytvořit jednu funkci, která hlavičku vytvoří a kterou volají ostatní bloky. Tou je právě fncCreateHeader, jejíž interface zachycuje **Chyba! Nenalezen zdroj odkazů.**.

```
FUNCTION fncCreateHeader : BOOL (* Prepends L4, L5, L6 and L7 headers *)
  VAR_INPUT
     writePtr : PTR_TO USINT; (* Pointer to the beggining of the message buffer *)
     payloadLength : USINT;   (* Number of bytes following the header *)
  END_VAR
  VAR_IN_OUT
     station : IEC_station;   (* Target IEC 61850 station *)
  END_VAR
  VAR
  END_VAR
  VAR_TEMP
     tmp : USINT;
  END_VAR
```

Obr. 5-3: Rozhraní fncCreateHeader

Význam celé hlavičky viz přiložený program – u každého pole je komentář, který popisuje, co je na dané adrese uložené. Vytvořená hlavička se skládá z vyplněných hlaviček všech nižších vrstev, jak zachycuje **Chyba! Nenalezen zdroj odkazů.**. Délka TPKT je určena sečtením délky hlavičky a parametru payloadLength, většina ostatních polí je statická. Z MMS PDU je předvyplněn kód PDU (0xa0 – Confirmed Request), délka MMS zprávy (na základě payloadLength) a InvokeID podle station.invokeID.

Za takto hlavičkou vytvořenou touto funkcí následují pole Service Type, Service Length a Service Data. Ta už se odvíjí od konkrétní IEC 61850 funkce, kterou daný paket realizuje – z implementovaných funkcí jde o 0xa4 (Read) a 0xa5 (Write). Jejich obsah je pak tvořený polem variableAccessSpecification, ve kterém je zakódované jméno proměnné (viz následující podkapitola), a v případě funkce write ještě daty k zapsání.



Obr. 5-5: Složení hlavičky [1]

5.7 Sestavení adresy, fncInsertName

```
FUNCTION fncInsertName : BOOL (* Writes variableAccessSpecification *)
  VAR_INPUT
     writePtr : PTR_TO USINT; (* Address to write at *)
     domainID : STRING; (* Logical Device name *)
     itemID   : STRING; (* Path from LN to leaf, formatted layer1$layer2$...$layerN *)
  END_VAR
  VAR_IN_OUT
     station : IEC_station;
  END_VAR
  VAR
     nameLength : USINT;
     readPtr : PTR_TO USINT;
  END_VAR
  VAR_TEMP
     tmp : USINT;
  END_VAR
```

Obr. 5-4: Rozhraní fncInsertName

Jak funkce read, tak funkce write potřebují znát adresu dle IEC 61850, kde je proměnná v cílové paměti uložena. Jelikož je kód v obou případech stejný, je extrahován do vlastní funkce – fncInsertName. Ta přijímá parametry adresu, kam jméno zapsat, stanici, ke které proměnná patří a části jména domainID a itemID, jak ukazuje Obr. 5-4.

Na základě těchto údajů sestaví BER TLV triplet variableAccessSpecification, který v MMS zprávě představuje IEC 61850 název proměnné. Ta se skládá ze dvou částí (domainID, itemID), které jsou společně uloženy v typu listOfVariables.

domainID je TLV typu VISIBLE STRING (řetězec ASCII znaků) a je vytvořen spojením názvu fyzického zařízení (IED), získaného z proměnné station, a názvu logického zařízení, předaného parametrem domainID.

itemID je také VISIBLE STRING, představuje zbytek názvu proměnné. Jak určuje datový model IEC 61850 (viz kapitola 2.3), název získáme spojením jmen všech nadřazených uzlů v datovém stromě. Ty představuje právě proměnná itemID, ve které jsou uloženy názvy jednotlivých uzlů, spojené znakem $ (pozor, jelikož Mosaic $ používá jako řídicí znak, je nutné při vytváření literálů psát $$, kde první $ slouží jako escape character).

Oba prvky jméno jsou pak spojeny do jednoho bloku a uloženy do bufferu. Pro přesný formát viz přiložený kód, kde je význam jednotlivých prvků okomentovaný.

5.8 fbRead

Funkční blok fbRead slouží k vyčtení jedné proměnné z cílové stanice. Předpokladem úspěšného vyčtení je, že stanice je ve stavu mmsReady (proběhl kompletní fbConnect, nečeká se na čtení ani zápis). Pokud je stav reset nebo error, dojde k uzavření komunikace a opuštění bloku, volání s jakýmkoliv jiným stavem nemá efekt (dojde k okamžitému opuštění).

```
FUNCTION_BLOCK fbRead  (* Requests read via IEC 61850 *)
  VAR_INPUT
     domainID : STRING; (* Logical Device name *)
     itemID   : STRING; (* Path from LN to leaf, formatted layer1$layer2$...$layerN *)
     saveAt   : PTR_TO BYTE; (* Address to save receieved data *)
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR_IN_OUT
     station : IEC_station; (* Station to read from *)
  END_VAR
  VAR
     data : ARRAY[0..200] of USINT;
     writePtr  : PTR_TO USINT;
     dataBlock : fbSendTo;
  END_VAR
  VAR_TEMP
     tmp : USINT;
     nameLength : USINT;
  END_VAR
```

Obr. 5-6: Rozhraní fbRead

Funkce přijímá několik parametrů, jak ukazuje Obr. 5-6. Adresa vyčítaného prvku vychází z datového modelu IEC 61850 (viz teoretická část), kde název IED

udává station.name, název LD parametr domainID a parametr itemID obsahuje zbytek cesty k proměnné – viz předchozí podkapitola.

Volající by měl zajistit, že paměť odkazovaná saveAt je dostatečně velká, aby pojmula typ proměnné, kterou chceme vyčíst (jinak dojde přepsání jiných dat). Samotný typ není nutné specifikovat, je jednoznačně určen přijatou odpovědí (obsahuje pole type).

Při volání fbRead dojde ke

- Kontrole položky station.state
- Sestavení zprávy pro vyčtení proměnné
- Odeslání zprávy (pomocí bloku fbSendTo z ComLib)
- Nastavení station.state := readReq; IEC_readPtr := saveAt;

5.9 fbWrite

```
FUNCTION_BLOCK fbWrite (* Writes data via IEC 61850 *)
  VAR_INPUT
     domainID    : STRING; (* Logical Device name *)
     itemID      : STRING; (* Path from LN to leaf, formatted layer1$layer2$...$layerN *)
     sourcePtr   : PTR_TO USINT; (* Address containing the data to be written *)
     length      : USINT; (* Number of bytes to be written *)
     dataType    : IEC_types; (* IEC 61850 type of the target variable *)
  END_VAR
  VAR_OUTPUT
  END_VAR
  VAR_IN_OUT
     station : IEC_station;  (* Station to  *)
  END_VAR
  VAR
     data : ARRAY[0..255] of USINT;
     writePtr : PTR_TO USINT;
     nameLength : USINT;
     payloadLength : USINT;
     dataBlock : fbSendTo;
  END_VAR
  VAR_TEMP
     tmpPtr : PTR_TO USINT;
     i    : USINT;
     tmp : USINT;
  END_VAR
```

Obr. 5-7: Rozhraní fbWrite

Blok fbWrite je velmi podobný fbRead, popsanému v předchozí kapitole. Na začátku bloku probíhá kontrola stavu stanice obdobně jako u čtení. I vstupní parametry jsou podobné, ale oproti fbRead obsahuje několik parametrů navíc. SourcePtr je obdobou saveAt, určuje adresu v paměti PLC, kde se nachází data k odeslání. Parametry length a dataType pak určují typ proměnné (podle IEC 61850, je odeslán do cílové stanice) a délku dat. Pro předávání typů je vytvořen zvláštní výčtový typ (enum) IEC_types. Ten obsahuje seznam podporovaných typů dle IEC 61850 (jde o všechny jednoduché typy, různé stringy a bit field, podporovaná není struct a obecná array (je možné je přenést, ale není implementován kód na jejich parsování)), přičemž každý typ má definovanou underlying hodnotu odpovídající

kódu daného typu v ASN.1. Díky tomu je možné při zapisování zprávy pouze zkopírovat hodnotu.

Opět platí, že volající kód by měl zajistit, že adresa sourcePtr odkazuje proměnnou odpovídající délky – jinak budou odeslána náhodná data, uložená za danou adresou.

Při vykonávání fbWrite tedy dojde ke

- Kontrole položky station.state
- Sestavení zprávy pro zápis proměnné
- Odeslání zprávy (pomocí bloku fbSendTo z ComLib)
- Nastavení station.state := writeReq;

5.10 **fbResponseHandler**

Poslední z definovaných bloků je fbResponseHandler. Ten se stará o zpracování příchozích zpráv. V současnosti zpracovává pakety Confirmed Response, zaznamenává Error a Reject a ignoruje pakety Confirmed Request a Unconfirmed. Zachycení zprávy, která neodpovídá formátu MMS, je logováno do station.errors.unknownMes.

Na začátku bloku probíhá kontrola station.state a ošetření chybových stavů jako u ostatních bloků. Poté se porovná hodnota bitu ARC (alternující bit, měnící se kdykoliv dojde zpráva) ethernetového kanálu s hodnotou station.lastARC – pokud se rovnají, nepřišla od posledního volání žádná zpráva a blok se ukončí, jinak dojde k aktualizaci lastARC a pokračuje se ke zpracování zprávy.

Dalším krokem je, že odpovídá formát zprávy – konkrétně se kontrolují nultý (TPKT version 0x03), pátý (COTP data 0xf0), sedmý a deváty (SPDU give tokens/data 0x01) a jedenácty (PPDU fully encoded data: 0x61) byte. Pokud některá z hlaviček nesedí, zaznamená se přijetí neznámé zprávy a ukončí vykonávání bloku, jinak následuje zpracování MMS zprávy.

Zpracování MMS zpráv je rozděleno podle typu PDU

- 0xa0 Confirmed-RequestPDU – je ignorováno (PLC pracuje v režimu klient, na požadavky nemá důvod odpovídat)
- 0xa1 Confirmed-ResponsePDU – přejde se ke zpracování
- 0xa2 ErrorPDU – zaznamená se chyba
- 0xa3 UnconfirmedPDU – je ignorováno (není implementován kód pro zpracování)
- 0xa4 RejectPDU – zaznamená se odmítnutí. Pravděpodobně důsledek programátorské chyby nebo komunikačního šumu, RejectPDU je odpověď na neplatný MMS paket

Nakonec dochází ke zpracování samotného paketu. Nejprve se zkontroluje, že invokeID odpovídá očekávanému, a pokud ano, provede se činnost odpovídající typu

služby, tj. pro read (type 0xa4) rozparsování zprávy a zápis dat na adresu IEC_readPtr, pro write (type 0xa5) potvrzení zápisu.

5.11 Globální proměnné, timeouts

Krom výše popsaných bloků obsahuje implementace ještě několik globálních konstant, které slouží k parametrizaci funkcí. Jsou jimi

- IEC_chanCode – kód kanálu, který se bude používat pro komunikaci
- IEC_timeout_connect – délka timeoutu pro fbConnect, určuje maximální čas mezi prvním pokusem o propojení TCP (stav off) a kompletním navázáním spojení (stav mmsReady). K realizaci timeoutu je použit časovač TON, který do parametru PT (preset) načítá tuto proměnnou. Pokud timeout vyprší, je nastavena odpovídající položka station.errors a stanice přejde do stavu error – další postup je na vnějším kódu (nemožnost navázat spojení je kritická chyba)
- IEC_timeout_read, IEC_timeout_write – maximální doba mezi voláním fbRead (respektive fbWrite) a obdržením odpovídající odpovědi uvnitř fbResponseHandler. Timeout je realizován pomocí TON ve fbResponseHandler, obdobně jako u IEC_timeout_connect. Pokud vyprší, je nastavena odpovídající chyba, ale na rozdíl od timeoutu připojení přejde stanice do stavu reset – komunikace může dále pokračovat (chyba není kritická, je možné, že selhává vyčítání jedné proměnné z mnoha). Další práce s timeoutem je ale opět na vnějším kódu.

5.12 Práce s knihovnou

Předchozí kapitoly popsaly jednotlivé části vytvořené knihovny. Nyní ještě nastíním, jak by zhruba mohl vypadat vnější kód, který by ji využíval. Velmi jednoduchý příklad je také v dodaném kódu, v jednotce prgMain.

Na začátku je vhodné nastavit používané proměnné. Nejprve je nutné vyčíslit globální konstanty – IEC_channelCode a IEC_timeout_*. Channel code bude ve většině případů ETH1_uni0 (zabudovaná konstanta Mosaicu, rozbalí se na číslo odpovídající uni módu prvního kanálu), nastavení timeoutů je zcela v kompetenci programátora. Dalším krokem je pak příprava proměnných – jeden z možných postupů je vytvořit pole struktur IEC_station (o délce odpovídající počtu vyčítaných stanic), a všechny je inicializovat ve funkčním bloku volaném při startu PLC (konkrétně je nutné nastavit jméno a IP adresu, ostatní položky se nastavují automaticky).

Po inicializaci proměnných je možné přejít k samotnému volání. Jednou z možností je periodicky zapisovat/vyčítat všechna data, která nás zajímají

(přidávaná např. pomocí FIFO struktury nebo bitového pole, určujícího, která data posílat). K tomu je vhodné vytvořit samostatný organizační blok (typu program), volaný v módu FreeWheeling. Ten by měl mít přístup k proměnným IEC_station a jednomu od každého z bloků fbConnect, fbRead, fbWrite a fbResponseHandler. Jelikož je podporováno vyčítání pouze jedné stanice zároveň (všechny volání používají stejný kanál), je nutné vytvořit vnější cyklus, určený pro přepínání mezi stanicemi. V rámci každé stanice pak dojde k volání fbConnect a fbResponseHandler (není nutné je ošetřovat podmínkou, pokud nejsou aktivní, jejich volání nemá efekt) a postupnému zpracování všech žádostí, které jsou mířeny na danou stanici (postupným voláním fbRead a fbWrite, kdykoliv je ve stanice ve stavu mmsReady). Jakmile jsou všechna data zpracována, je možné přejít na další stanici.

Při ukončování komunikace či přepínání stanic je vhodné nejprve stanici převést do stavu reset (případně error) a zavolat některý z bloků, což vynutí okamžité uzavření TCP kanálu – jinak nebude možné komunikovat s jinou stanicí, dokud nevyprší její timeout a neuzavře se kanál díky tomu.

6. ZÁVĚR

V rámci teoretické části práce bylo vytvořeno stručné shrnutí datového modelu a vertikální komunikace dle IEC 61850, a sestaven seznam zdrojů vhodných pro detailnější seznámení s tímto standardem. Dále pak obsahuje krátký popis automatů řady Foxtrot a programování v prostředí Mosaic v takové míře, aby čtenáře se základními znalostmi programování a jazyka ST seznámilo s jejich použitím (a umožnilo pochopení kódu v praktické části). Na závěr obsahuje ještě krátké shrnutí paketů nosných protokolů v jednotlivých vrstvách ISO/OSI, které jsou při vertikální komunikaci pomocí standardu IEC 61850/MMS využity.

V praktické části byla vytvořena jednoduchá knihovna, která umožňuje vertikální řízení stanice IEC 61850 v režimu server. K tomu byly vytvořeny funkční bloky fbConnect, fbRead, fbWrite a fbResponseHandler, které umožňují připojení ke stanici, vyčítání a zápis dat a zpracování odpovědí. Krom toho je součástí implementace také několik datových typů a pomocných funkcí, které pomáhají k lepší práci s knihovnou.

Knihovna umožňuje kompletní řízení stanice (zápis a vyčtení libovolných dat), nicméně v některých případech nepodporuje nejefektivnější možný přístup. Současný stav implementace podporuje odesílání i čtení jednoduchých datových typů a různých druhů řetězců, bohužel ale nedokáže přenášet proměnné složených typů struct a array, ke kterým by bylo nutné vytvořit kompletní dekodér ASN.1 pro rekurzivní zpracování jejich členů. Nicméně, takové typy lze jednoduše přenést postupným vyčítáním jednotlivých položek (za cenu pomalejšího zpracování, jelikož pro každou položku je potřeba nová zpráva). Také nejsou podporovány pakety typu UnconfirmedPDU, používané pro přenášení reportů (periodické nebo změnové jednostranné přenosy předem definovaných dat) – je nutné o vyčítání dat vždy žádat (jelikož na straně PLC neodpovídá datový model IEC 61850, bylo by nutné vytvořit způsob, jak převádět IEC 61850 jména na proměnné v PLC – samotné zpracování paketu není problém).

Druhou nevýhodou použité implementace je serializace volání – není umožněno ani vyslání několika souběžných požadavků na stejnou stanici, ani komunikace s několika stanicemi zároveň. To je způsobeno několika problémy – zaprvé, v současnosti se informace mezi voláním a odpovědí přenáší pomocí jednoduchých proměnných (invokeID, state, IEC_readPtr). Pro realizaci paralelních spojení by ale bylo nutné nějakým způsobem provázat metadata s odpovědí – buď použitím funkčních bloků, kde by každý funkční blok odpovídal jednomu spojení (a data si uchovával uvnitř – měl by na starosti jak odeslání, tak přijetí zprávy), nebo vytvořením mapové struktury, kde by klíče tvořily InvokeID a hodnoty odpovídající metadata (a na základě přijaté odpovědi by fbResponseHandler určil, ke které

žásdosti náleží). Obě možnosti by ale výrazně komplikovaly implementaci i práci s knihovnou, a měly by vliv pouze na efektivitu, ne na rozsah poskytovaných služeb.

Souběžnou komunikaci s několika stanicemi pak komplikují zejména limitace prostředí Mosaic – nemožnost zahrnout proměnné typu PTR_TO mezi prvky struktury (tj. není možné přiřadit každé stanici vlastní readPtr) a problémy s blokem fbRecieveFrom z knihovny ComLib, kvůli kterým musela být použita přímá práce s ethernetovým kanálem (pomocí jeho kontrolních proměnných a datových bufferů), která se poměrně těžko parametrizuje. Použití jiného kanálu než ETH1_uni0 proto vyžaduje i úpravy uvnitř bloků a použití několika kanálu zároveň není bez zásadních změn možné vůbec.

Vytvořené řešení bylo otestované s použitím PLC Foxtrot CP-1006 a ovladače ochran REF630, mezi kterými probíhal přenos dat bez problému. Pakety read a write generované v PLC byly také zkontrolovány pomocí disektoru programu Wireshark – jejich formát odpovídá MMS zprávám. Pro jiné druhy IEC 61850 program testovaný není, ale vzhledem k tomu, že nepoužívá žádné postupy specifické pro REF630, by měl být kompatibilní s libovolným IEC 61850 serverem.

Literatura

[1] MATOUŠEK, Petr. *Description of IEC 61850 Communication* [online]. FIT-TR-2018-01, Brno: FIT VUT, 2018 [cit. 2018-12-31]. Dostupné z: http://www.fit.vutbr.cz/research/view_pub.php.cs?id=11832

[2] STODŮLKA, Ivo. *Model elektrické stanice s komunikačním protokolem IEC 61850* [online]. Brno: FEKT VUT, 2012 [cit. 2018-12-31]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace?zp_id=51986

[3] LEDNICKÝ, Pavel. *Digitalizace rozvodny vysokého napětí při použití komunikačního standardu IEC61850* [online]. Brno: FEKT VUT, 2011 [cit. 2018-12-31]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace?zp_id=41195

[4] *630 series IEC 1.3, IEC 61850 Communication Protocol Manual* [online]. 2015 [cit. 2018-12-31]. Dostupné z: https://library.e.abb.com/public/09d948a4eeb57dd7c1257dc7004cade3/RE_630_iec61850prot_756793_ENd.pdf

[5] *615 series IEC 1.3, IEC 61850 Communication Protocol Manual* [online]. 2015 [cit. 2018-12-31]. Dostupné z: https://library.e.abb.com/public/63c98d28e525f279c1257b2f0054c237/RE_615_IEC61850eng_756475_ENg.pdf

[6] *Protection and Control IED Manager PCM600* [online]. 2018 [cit. 2018-12-31]. Dostupné z: https://library.e.abb.com/public/938d6c3e12fd44cf89082bdbb48a7b15/1MRS757866-C-15352.pdf

[7] System Integration Specialists Company (SISCO). *Overview and Introduction to the Manufacturing Message Specification (MMS)* [online]. Sterling Heights, USA: 1995 [cit. 2018-12-31]. Dostupné z http://www.sisconet.com/wp-content/uploads/2016/03/mmsovrlg.pdf

[8] TECO, a.s. *PLC Tecomat Foxtrot* [online]. Webový katalog [cit. 2018-12-31]. Dostupné z https://www.tecomat.cz/products/cat/cz/plc-tecomat-foxtrot-3/

[9] TECO, a.s. *Příručky SW* [online]. Dokumentace k prostředí Mosaic [cit. 2018-12-31]. Dostupné z: https://www.tecomat.cz/ke-stazeni/prirucky/prirucky-sw/

[10] KOTÁL, Tomáš. *Moderní komunikace protokolem IEC61850*. Praha, 2009. Diplomová práce. České vysoké učení v Praze, Fakulta elektrotechnická.

[11] *IEC 61850 Communication Networks and Systems in Substations Part 7-1: Principles and models*. International Electrotechnical Commission, 2003.

[12] *IEC 61850 Communication Networks and Systems in Substations Part 8-1: Mappings to MMS (ISO/IEC9506-1 and ISO/IEC 9506-2)*. International Electrotechnical Commission, 2003.

[13] SISCO. *MMS Abstract Syntax* [online]. [cit. 2019-05-13]. Dostupné z: http://www.sisconet.com/wp-content/uploads/2016/03/mms_abstract_syntax.txt

[14] ITU-T Recommendation. *X.690* [online]. Geneva, Switzerland, 2002 [cit. 2019-05-15]. Dostupné z: https://www.itu.int/ITU-T/studygroups/com17/languages/X.690-0207.pdf

Seznam příloh

Příloha 1 - Zdrojový kód na přiloženém CD

Zdrojový kód programu je uložen na přiloženém CD. K otevření použijte program Mosaic verze 2019.1 SP1, dostupný z webu např. na adrese https://www.tecomat.cz/ke-stazeni/software/mosaic/ [platí k 13.5.2019]. Pro aktivaci všech funkcí je nutné použít hardwarový klíč, otevření projektu i nahrání do PLC by ale mělo být možné i v Lite verzi.