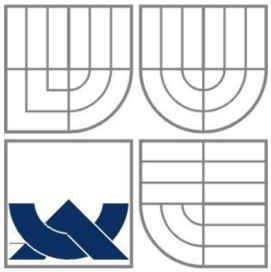
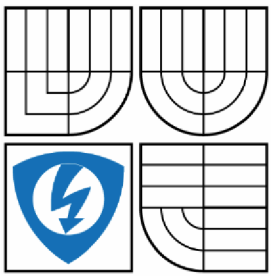


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

VÝVOJ GENERÁTORU KÓDU PLC PRO BLOKY FACEPLATE PRO S7-1200, S7-1500, S7-300/400

DEVELOPMENT OF A PLC CODE GENERATOR FOR FACEPLATE BLOCKS FOR S7-1200, S7-1500, S7-300/400

BAKALÁŘSK) PRÁCE
BACHELOR'S THESIS

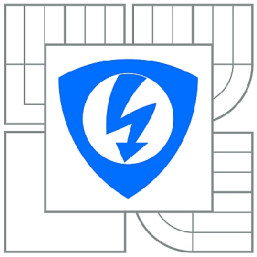
AUTOR PRÁCE
AUTHOR

PETR KŘÍŽ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JAN PÁSEK, CSc.

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav automatizace a měřicí techniky

Bakalářská práce

bakalářský studijní obor
Automatizační a měřicí technika

Student:	Petr Kříž	***ID:***	154781
Ročník:	3	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Vývoj generátoru kódu PLC pro bloky faceplate pro S7-1200, S7-1500, S7-300/400

POKYNY PRO VYPRACOVÁNÍ:

Jedná se o vytvoření nástroje, nejlépe v prostředí EXCEL, který v návaznosti na generátor tagů (stávající nástroj firmy COMPAS) umožní automaticky generovat programový kód pro jednotlivá el. zařízení v projektu. Cíle BP:

1. Proveďte analýzu stávajících objektů FACEPLATE.
2. Proveďte analýzu využitelných funkcí v prostředí TIA V12 ADVANCED a notace (zápisy) SCL.
3. Aplikujte generátor tagů.
4. Připravte návaznost na podklady z projektu elektro.
5. Vytvořte zadaný nástroj a proveďte jeho implementaci.
6. Otestujte zadaný nástroj pro jednotlivé platformy PLC

DOPORUČENÁ LITERATURA:

Firemní dokumentace COMPAS s.r.o.

Termín zadání:	9.2.2015	***Termín odevzdání:***	25.5.2015

Vedoucí práce: Ing. Jan Pásek, CSc.
Konzultanti bakalářské práce:

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

UPOZORNĚNÍ:

Abstrakt

Cílem této bakalářské práce bylo vytvoření nástroje pro generování PLC kódu v závislosti na zadaných datech. Generátor pracuje s programovými funkčními bloky, které jsou přidělené jednotlivým objektům faceplate z knihovny firmy COMPAS Automatizace, spol. s.r.o. Z vygenerovaného kódu se vytvoří zdrojový soubor, ze kterého vzniknou programové bloky v programu TIA Portal V13.

Klíčová slova

TIA Portal V13, faceplate, zdrojový soubor, import, PLC

Abstract

The main goal of this bachelor's thesis was creating a tool for generating PLC code according to entered data. Generator works with function blocks that are assigned to each faceplate object from faceplate library provided by COMPAS Automatizace, spol s.r.o. Source file is created from generated code and program blocks in program TIA Portal V13 are created from this source file.

Keywords

TIA Portal V13, faceplate, source file, import, PLC

Bibliografická citace:

KŘÍŽ, P. *Vývoj generátoru kódu PLC pro bloky faceplate pro S7-1200, S7-1500, S7-300/400*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2015. 52 s. Vedoucí bakalářské práce Ing. Jan Pásek, CSc..

Prohlášení

„Prohlašuji, že svou bakalářskou práci na téma Vývoj generátoru kódu PLC pro bloky faceplate pro S7-1200, S7-1500, S7-300/400 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: **22. května 2015** ……
podpis autora

Poděkování

Děkuji vedoucímu bakalářské práce panu Ing. Janu Páskovi, CSc. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce. Dále bych chtěl poděkovat odbornému konzultantovi, panu Ing. Zbyňku Bezchlebovi z firmy COMPAS Automatizace, spol. s.r.o. ze Žďáru nad Sázavou za velice cenné odborné rady, poskytnuté jak pomocí emailové korespondence, tak na osobních konzultacích.

V Brně dne: **22. května 2015** .....

podpis autora

Obsah

1 Úvod ... 9
1.1 Úpravy v zadání ... 9
1.1.1 Použitá verze software ... 9
1.2 Objekty faceplate ... 9
1.3 Cíle práce ... 9
2 Použitý software ... 12
2.1 TIA Portal V13 Update 6 ... 12
2.1.1 Popis TIA Portal V13 ... 12
2.1.2 STEP 7 Professional V13 ... 12
2.1.3 Programovací jazyk SCL ... 13
2.1.4 WinCC Advanced V13 Update 6 ... 14
2.1.5 SIMATIC S7-PLCSIM V13 ... 15
2.2 S7-PLCSIM V5.4 SP5 ... 15
3 Analýza stávajících objektů faceplate ... 16
3.1 Obecný popis funkčních bloků ... 16
3.2 Funkční blok CMPOP ... 16
3.3 Funkční blok LOGOP ... 17
3.4 Funkční blok AI ... 18
3.5 Funkční blok DI ... 19
3.6 Funkční blok DiagPB ... 19
3.7 Funkční blok DiagPN ... 20
3.8 Funkční blok PIDC ... 21
3.9 Funkční blok PIDS ... 23
3.10 Funkční blok AQM ... 25
3.11 Funkční blok DQM ... 27
3.12 Funkční blok DQV ... 28
3.13 Funkční blok AQV ... 29
3.14 Funkční blok PHASE ... 31
4 Import a export programových bloků ... 33
4.1 Export programových bloků do programu MS Excel ... 33
4.1.1 Programovací jazyk SCL ... 33
4.1.2 Programovací jazyky STL, LAD, FBD ... 34
4.1.3 Otevření souboru .scl nebo .awl v MS Excel ... 35

4.2 Export programových bloků z MS Excel do TIA Portal 35
4.3 Export a import tabulky symbolů 36
5 Návrh generátoru PLC kódu 38
5.1 List Projekční tabulka 38
5.2 List Tabulka symbolů 39
5.3 List AWL kód 40
5.4 List AWL zdroj 42
5.5 List Faceplate IO 43
6 Aplikace generátoru tagů 44
7 Test generátoru 45
7.1 PLC S7-300/400 45
7.2 PLC S7-1500 46
7.3 PLC S7-1200 47
8 Závěr 49

1 ÚVOD

1.1 Úpravy v zadání

K následujícím úpravám zadání a následného zpracování projektu jsem přistoupil po konzultaci s oficiálním konzultantem k této semestrální práci, panem inženýrem Zbyňkem Bezchlebou z firmy COMPAS Automatizace, spol. s.r.o. ze Žďáru nad Sázavou.

1.1.1 Použitá verze software

Ačkoliv je v oficiálním zadání uvedeno jako jeden z bodů analýza využitelných funkcí v prostředí TIA V12 Advanced, na konzultaci s panem inženýrem Bezchlebou mi bylo řečeno, ať projekt vypracovávám pro nejnovější dostupný software, v tomto případě pro programovací prostředí TIA Portal V13 Professional s nejnovějšími aktualizacemi. K tomu se váže i použití WinCC Advanced V13 (verze Professional nebyla v tomto případě nutnou) a pro praktické zkoušení programu také S7-PLCSIM V13 pro simulování programů založených na PLC SIMATIC S7-1500 a S7-PLCSIM V5.4 pro simulaci programů založených na PLC SIMATIC S7-300 a S7-400.

1.2 Objekty faceplate

Faceplate vznikají ve vizualizaci jako uskupení určitých zobrazovacích a kontrolních objektů. Ty se následně dají uložit do knihovny a dále se s nimi dá pracovat, jako s ostatními zobrazovacími objekty. Výhodou je také možnost používat již vytvořený objekt typu faceplate v jiných projektech. Faceplate v sobě může sdružovat informace o stavu, poruchách a řízení, které jsou vztažené ke kontrolovanému prvku. Uživatel vidí faceplate jako grafické objekty ve vizualizačním prostředí WinCC, pomocí nichž může sledovat nebo řídit jednotlivé prvky. To, že jsou veškeré zobrazovací a ovládací prvky sjednoceny do jednoho kompaktního rozhraní, umožňuje uživateli snadnější práci i snadnější úpravy, kdy úpravou vytvořeného faceplate ve speciálním faceplate editoru může lehce změnit všechny faceplate, které jsou v projektu použity. Editor faceplate je implementován ve vizualizačním prostředí WinCC V13 Advanced nebo Professional. [1]

1.3 Cíle práce

Hlavním cílem této bakalářské práce je vytvoření nástroje v prostředí Microsoft Excel, který slouží ke generování PLC kódu. Program je vytvořen za pomocí VBA skriptů, které jsou běžně dostupné a funkční v programech Microsoft Office, do nichž Excel patří. Program MS Excel byl zvolen hlavně z toho důvodu, že samotný generátor kódu funguje na bázi dat získaných z jedné nebo více tabulek. Samotný nástroj pak generuje zdrojový

soubor, který se následně implementuje v programovém prostředí TIA Portal V13 Professional.

Na základě dat, která programátor zadá do listu Projekční tabulka v samotném generátoru, se vytvoří zdrojový soubor, díky kterému vznikne kostra celého budoucího programu a samotnému programátorovi tak ušetří čas, který by jinak strávil zdlouhavým vytvářením této programové kostry. Mezi zadávaná data patří seznam označení jednotlivých technologických skupin, projekční značení jednotlivých prvků elektroinstalace, jako jsou motory, ventily a další, zkratka typu faceplate pro každý objekt elektroinstalace a poté vzor, poznámku a umístění bloku. Tato data by měl programátor získat z dokumentace a zadání elektroprojektu. Více o každé z těchto položek i o samotné projekční tabulce viz **List Projekční tabulka.**

Kostrou programu, který bude generátor PLC kódu vytvářet, se rozumí vytvoření bloků funkcí v prostředí TIA Portal. Pro každou technologickou skupinu se vytvoří jedna hlavní funkce s názvem skupiny. Každá technologická skupina může, dle zadání elektroprojektu, sloužit k ovládání velkých částí projektu, které v sobě obsahují samotné prvky elektroinstalace. Může se tak jednat o dopravník, na kterém je připojeno nějaké množství analogových a digitálních senzorů a motory, nebo se může jednat o kotel v teplárně, ve kterém se ovládá nějaké množství ventilů, a do programu se dostávají data z nainstalovaných senzorů. Každý jednotlivý prvek, jako je ventil, motor nebo analogový/digitální vstup má z projektové dokumentace přiřazeno projekční značení (například ventil V31 a digitální senzor D12 jsou součástí technologické skupiny L1). Aby byla práce v programu TIA Portal s bloky funkcí jednotlivých technologických skupin přehlednější, každá skupina se ještě dělí na podskupiny, které se starají o vstupy, výstupy, podmínky, ruční ovládání a automatické cykly. Až v těchto podskupinách se pak volají samotné prvky elektroinstalace.

Normálně by tak programátor musel v programu TIA Portal vytvořit pro každou technologickou skupinu blok funkce, v každém takto vytvořeném bloku by musel vytvořit dalších pět bloků, které by se staraly o vstupy, výstupy atd. a pak podle dokumentace elektroprojektu by musel do příslušného bloku funkce vkládat jednotlivě funkční bloky pro daný typ prvku elektroinstalace. Díky tomuto generátoru mu ke každému prvku stačí v projekční tabulce vyplnit pouze název technologické skupiny, do které patří, projekční značení daného prvku, příslušný faceplate k tomuto prvku a to, zda se má prvek volat v bloku, který se stará o vstupy, výstupy, ruční řízení atd. Generátor na základě těchto dat vytvoří kód, který se uloží jako zdrojový soubor, importuje se do TIA Portalu a z tohoto souboru pak vzniknou všechny potřebné bloky. Na obrázku níže je příklad kostry programu. Pokud by programátor pracoval ručně, musel by všechny bloky vytvořit v TIA Portalu. Pokud by použil generátor, stačilo by mu vyplnit jeden řádek projekční tabulky pro každý prvek elektroinstalace.

- **FCL1** - blok technologické skupiny, například dopravník
  - **FCL1_IN** - blok zajišťující práci se vstupy
    - **Senzory D1, D2,…** - prvky elektroinstalace s projekčním značením
  - **FCL1_COND** - blok zajišťující práci s podmínkami
    - **Jednotlivé prvky**
  - **FCL1_AUTO** - blok zajišťující automatické řízení
    - **Motor GP1,…**
  - **FCL1_MAN** - blok zajišťující ruční řízení
    - **Motor H2,…**
  - **FCL1_OUT** - blok zajišťující práci s výstupy
    - **Ventil V31, V32,…**

- **FCL2**
  - **…**
    - **…**

Obrázek 1.1 - příklad kostry projektu, jak by ji mohl vytvořit generátor

2 POUŽITÝ SOFTWARE

V této kapitole je uveden popis programů, které jsem při práci na této bakalářské práci využíval. Chybí zde pouze Microsoft Excel. [2]

2.1 TIA Portal V13 Update 6

Celým názvem Totally Integrated Automation Portal Version V13 Update 6, zkráceně TIA Portal V13, je softwarové prostředí firmy Siemens, s.r.o. umožňující vývoj uživatelských programů pro PLC a jiné připojené decentrální periferie. TIA Portal V13 dále umožňuje velmi rozsáhlé projektování vizualizací SCADA (zkratka ze „Supervisory Control And Data Acquisition“, volně přeloženo jako „Kontrolorské řízení a sběr dat“), vytváření projektů využívajících ovládací panely HMI, projektování síťových komponentů, uvádění pohonů do provozu a další činnosti související s průmyslovou automatizací. Všechny tyto prvky slučuje do jednoho přehledného vývojového prostředí.

2.1.1 Popis TIA Portal V13

TIA Portal V13 velice usnadňuje práci na projektu, mimo jiné díky sjednocení a optimalizaci uživatelského prostředí. Jak jsem již zmiňoval dříve, v TIA Portalu je možné programovat PLC, vytvářet vizualizaci nebo ovládat pohony Siemens Sinamics. Všechny tyto pracovní nástroje jsou dostupné v rámci celého systému, což výrazně usnadňuje a urychluje instalaci a aktualizaci programu. TIA Portal nabízí jednotné prostředí pro všechny výše zmíněné aplikace, a proto nabízí jednoduchou orientaci v systému a snadné zaučení. Výhodou je také funkce „Drag and Drop“, která umožňuje snadné programování přetahováním a vkládáním různých funkcí nebo dokonce přetahováním prvků mezi různými editorovými okny. Inteligentní editory také nabízí kontextově citlivou nápovědu a možnosti výběru, navíc zabraňují vkládání chybných údajů při programování projektu.

2.1.2 STEP 7 Professional V13

STEP 7 Professional V13 (zkráceně STEP 7 V13) je vývojový software určený primárně pro programování PLC SIMATIC od firmy Siemens, s.r.o. Software STEP 7 V13 se liší podle několika verzí, kdy ta nejzákladnější a zároveň také nejlevnější verze Basic umožňuje uživateli programovat PLC SIMATIC S7-1200 a ta nejvyšší verze Professional umožňuje dále programovat i PLC SIMATIC řady S7-1500, S7-300 a S7-400 a řídicí systémy založené na PC, tzv. WinAC. STEP 7 V13 umožňuje jednotně a jednoduše nakonfigurovat potřebný hardware, stejně jako usnadňuje programování uživatelské aplikace. STEP 7 V13 umožňuje využívat při programování všechny programovací jazyky podle normy IEC 61131-3, tj. jazyky LAD („Ladder Diagram, LD), který odpovídá grafickému znázornění reléových schémat, FBD („Function Block Diagram“), který znázorňuje diagram logických funkčních bloků, STL, fungující jako seznam

instrukcí („Instruction List“, IL), SCL, psaný ve formě strukturovaného textu („Structured Text“, ST) a S7-Graph, pracující na principu sekvenčního vývojového diagramu („Sequential Function Diagram“, SFC).

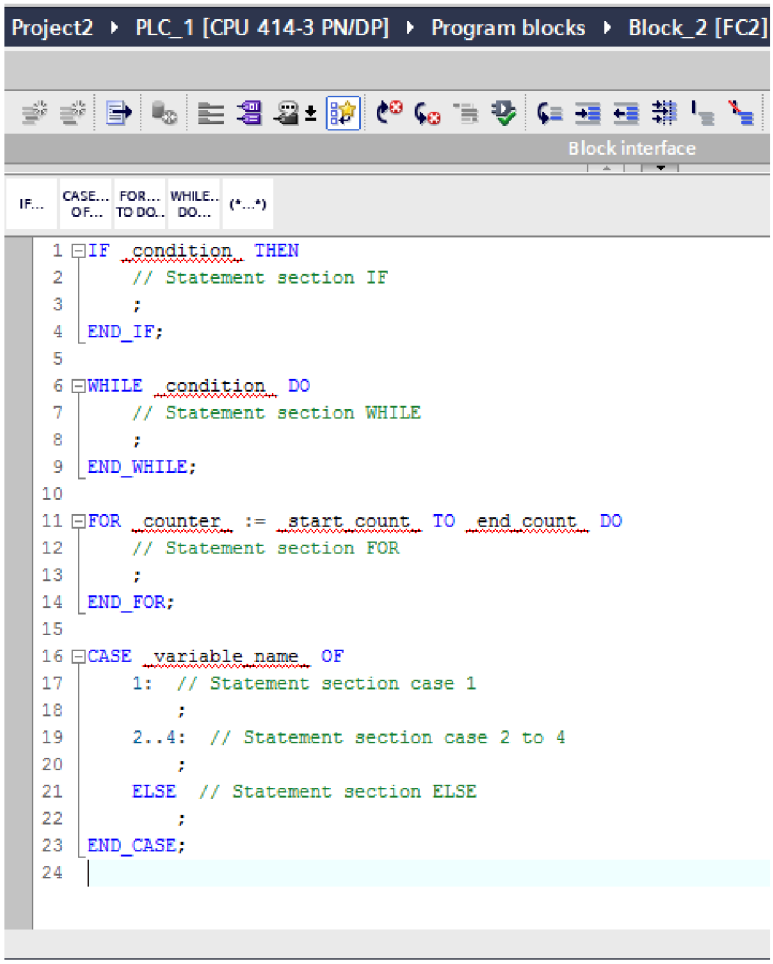
STEP 7 Professional V13 dále nabízí funkce sdíleného programování projektu včetně automatických synchronizací a kompletní nahrání pracovní stanice a periferií, včetně hardwarové konfigurace.

2.1.3 Programovací jazyk SCL

Programovací jazyk SCL (zkratka ze „Structured Control Language“) je textově založený programovací jazyk. Jeho struktura je založena jak na programovacím jazyku PASCAL, tak na standardních programovacích jazycích pro PLC. Hlavní výhodou jazyku SCL je to, že kromě prvků typických pro programovací jazyk vyšší úrovně využívá prvky typické pro programování PLC, jako je například práce se vstupy a výstupy, časovači, bitovou pamětí a podobné. Nabízí tedy programovací možnosti jazyků LAD a STL a doplňuje je o další užitečné prvky.

Programovací jazyk SCL je vhodný hlavně pro lidi, kteří již mají zkušenost s programováním v klasických programovacích jazycích, jako jsou například PASCAL, C nebo C++. SCL s nimi má totiž mnoho společného, co se struktury programu a instrukcí týče. Pro programování PLC se nejvíce hodí k tvorbě bloků, ve kterých se musí řešit složité podmínky, komunikace, analýza nebo cyklické běhy. Díky interaktivní nápovědě ve STEP 7 a díky funkci „Drag and Drop“ můžeme jednoduše přetáhnout potřebné instrukce ze široké nabídky, ať už se jedná o časovače, čítače, matematické operace nebo instrukce ovládající celý program. K nejčastěji používaným instrukcím v jazyce SCL patří podmínkové funkce, jako například *IF ... THEN* (pokud je splněna určitá podmínka, vykonají se vložené příkazy), *WHILE ... DO* (program se vykonává, dokud je splněna podmínka), *FOR ... TO ... DO* (program se vykonává, dokud hodnota zadaná v cyklu nedosáhne zadané konečné hodnoty). Jazyk SCL je také výhodný k programování projektů fungujících na bázi stavových automatů, k tomu se hodí instrukce *CASE ... OF* (program se vykonává podle hodnoty zadané veličiny). Každá z těchto instrukcí se musí zakončovat příkazem *END_instrukce* (například *END_IF, END_FOR* atd.). Výhodou funkce „Drag and Drop“ je také to, že každá takto natáhnutá instrukce se rovnou zapíše ve svém celém správném tvaru a programátorovi stačí jenom doplnit předem jasně vyznačené pole pro proměnné nebo hodnoty. V programovacím jazyku SCL je také velice snadné přiřazování hodnot proměnným. K přiřazení hodnoty stačí použít jednoduchý příkaz *název_proměnné := hodnota*.

Obrázek 2.1 - ukázka kódu v SCL, TIA Portal V13



2.1.4 WinCC Advanced V13 Update 6

WinCC Advanced V13 Update 6 (zkráceně WinCC V13) je další nezbytná součást vývojového prostředí TIA Portal V13, která uživateli umožňuje vytváření vizualizačního rozhraní v projektu. WinCC V13 se podobně jako STEP 7 V13 dodává v několika verzích. Nejzákladnější a nejlevnější verze Basic a pokročilejší verze Comfort nenabízí vytváření a práci s objekty faceplate. Pokročilejší verze Advanced a nejvyšší verze Professional již tuto nabídku mají. Verze WinCC Professional V13 má oproti verzi Advanced ještě několik velice zajímavých funkcí, v tomto projektu je však nepotřebuji, a proto mi stačí pouze verze WinCC Advanced V13.

WinCC V13 nabízí bohaté možnosti vytváření vizualizačního rozhraní, ať už se jedná o jednoduché ovládací panely nebo o rozsáhle aplikace SCADA (verze Professional). Velkou předností je propojení se STEP 7 V13 v prostředí TIA Portal V13. V tomto ohledu je asi největší výhodou to, že proměnné a datové bloky používané při programování PLC SIMATIC se dají automaticky používat i při vytváření vizualizačního rozhraní pro HMI. Další výhodou je to, že jak programování ve STEP 7 V13 tak i vytváření vizualizace

pomocí WinCC V13 probíhá v prostředí TIA Portal V13, takže stačí mít puštěný pouze tento program a přepínat mezi oběma aplikacemi ve vnitřním projektovém stromu. Díky integraci v TIA Portalu V13 také stačí pouze jednou nakonfigurovat komunikační proměnné a samotnou komunikační topologii, ta se později v rámci projektu používá všemi programy.

Prostředí WinCC V13 klade velký důraz na jednoduché a intuitivní ovládání, se kterým se projektant brzy naučí snadno pracovat. Velkou výhodou a zároveň prvkem urychlujícím celý návrh je důraz na použití metody „Drag and drop“ (volně přeloženo jako „přetáhni a pusť“), která dovoluje jak snadné přetahování a umisťování grafických a kontrolních prvků ze široké nabídky programu, tak například přetahování již vytvořených prvků mezi více operátorskými panely nebo mezi jednotlivými obrazovkami panelu.

2.1.5 SIMATIC S7-PLCSIM V13

Program SIMATIC S7-PLCSIM V13 (zkráceně PLCSIM V13) nám umožňuje spouštět vytvořené projekty v režimu simulace, takže není nutné fyzické připojení k projektovému PLC. Pomocí simulátoru PLCSIM V13 tak můžeme odlaďovat základní program, opravovat chyby před nahráním programu do reálného projektu, čímž se může předejít případným škodám na pracovních strojích a navíc odzkoušení programu pomocí simulace již během programování ušetří čas při konečném zprovozňování celého projektu. Program je navíc integrován v prostředí TIA Portal V13, a proto umožňuje snadné spouštění simulace programu naprogramovaného ve STEP 7 jedním tlačítkem. Ke spouštění vizualizace na PC slouží WinCC Runtime Advanced. V samotném programu ve STEP 7 pak můžeme přepnout do online režimu a sledovat, co program právě vykonává, stejně, jako bychom program pouštěli v normálním PLC. Stejně se dají také používat prvky Watch table a Force table.

Hlavní nevýhodou programu PLCSIM V13 je to, že dokáže simulovat pouze činnost PLC SIMATIC S7-1500. Pro použití pro PLC SIMATIC S7-1200 je zde možnost vytvořit nový projekt, kterému nadefinujeme v hardwarové konfiguraci PLC SIMATIC řady S7-1500 a potom z původního projektu s PLC S7-1200 postupně nakopírujeme jednotlivé programové bloky, datové bloky a tabulku symbolů. Výsledný nový projekt, který pracuje s PLC S7-1500 pak můžeme bez větších problému simulovat pomocí PLCSIM V13.

2.2 S7-PLCSIM V5.4 SP5

Program S7-PLCSIM V5.4 SP5 (zkráceně PLCSIM V5.4) je starší verzí programu SIMATIC S7-PLCSIM V13. Hlavním rozdílem je to, že PLCSIM V5.4 dokáže simulovat PLC SIMATIC pouze řady S7-300 a S7-400. Jeho pracovní prostředí má také odlišný design oproti PLCSIM V13, nabízí však téměř stejnou funkčnost. V této bakalářské práci tento program zmiňuji hlavně kvůli nutnosti testování projektů naprogramovaných pro PLC SIMATIC S7-300 a S7-400.

3 ANALÝZA STÁVAJÍCÍCH OBJEKTŮ FACEPLATE

Firma COMPAS Automatizace, spol. s.r.o. vyvinula vlastní knihovnu s objekty faceplate, se kterou budu v této bakalářské práci částečně pracovat. Ke každému typu faceplate je také přiřazen programový funkční blok, který vykonává potřebné funkce pro daný objekt faceplate a zajišťuje tak jeho správnou funkčnost a možnosti modifikace právě užívaného objektu. Tyto funkční bloky již byly vytvořeny firmou COMPAS Automatizace, spol. s.r.o., ve vygenerovaném kódu pak pouze volám tyto funkční bloky na základě údajů, které programátor vyplní do projekční tabulky. Tato data se získávají ze zadání a dokumentace aktuálně programovaného elektroprojektu. Všem prvkům elektroinstalace je tak přiřazen právě jeden druh objektu faceplate a tím pádem i jeden funkční blok. Každý z těchto funkčních bloků má vytvořené a pojmenované vstupy a výstupy. Funkční bloky mají takovou vnitřní strukturu, aby na základě signálů připojených na vstupy dokázaly nastavit potřebné vlastnosti a zajistit správné fungování přidělených objektů faceplate. V této kapitole bych chtěl stručně popsat každý z objektů faceplate, které se nachází v knihovně firmy COMPAS Automatizace, spol. s.r.o. [3]

3.1 Obecný popis funkčních bloků

Samotná dokumentace, která mi byla poskytnuta mým odborným konzultantem, panem inženýrem Zbyňkem Bezchlebou, je poměrně rozsáhlá a ke každému funkčnímu bloku (tj. ke každému typu faceplate objektu) je v ní mnoho parametrů. K vytvoření generátoru PLC kódu však potřebuji znát pouze vstupy a výstupy každého bloku, protože s vnitřní strukturou těchto bloků nijak nepracuji. Na vstupy se přivádí požadované signály tak, jak to vyplývá z daného elektroprojektu. Funkční blok ve své vnitřní struktuře potom může hodnoty dalších signálů, což se projeví na jednotlivých výstupech bloku. Níže uvedené tabulky ke každému bloku jsou přímo zkopírovány z oficiální dokumentace firmy COMPAS Automatizace, spol. s.r.o. Tabulky vstupů a výstupů ke každému funkčnímu bloku sem vkládám hlavně z toho důvodu, aby mohlo být patrné, jaký druh signálů se může na vstupy daného bloku připojit.

3.2 Funkční blok CMPOP

Funkční blok FBHMI_CMPOP slouží ke sjednocení vizualizace porovnávacích funkcí <, >, <=, >=, využívá se pro vyhodnocování limit měřených veličin.

Tabulka 3.1 - vstupy a výstupy funkčního bloku CMPOP

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
PV	INPUT	REAL		0.0	Vstupní měřená hodnota

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
SP	INPUT	REAL		0.0	Hodnota parametru, se kterou se porovnává měřená hodnota PV.
HYST	INPUT	REAL		0.0	Hystereze.
TYP	INPUT	DWORD		DW#16#1	
Q	OUTPUT				Výstupní hodnota bloku. Výsledek porovnávací operace STAT.OP_RESULT.
Q_ERR	OUTPUT				Chyba vstupních hodnot.

3.3 Funkční blok LOGOP

Funkční blok FBHMI_LOGOP slouží ke sjednocení vizualizace logických funkcí OR, AND, NOR a NAND. Tento funkční blok se používá pro blokační podmínky nebo pro podmínky chodu libovolného zařízení. Tyto funkční bloky lze kaskádovitě spojovat, projektant by však měl dodržovat určité konvence pro pojmenovávání, aby byl projekt přehledný a na první pohled bylo jasné, které bloky jsou spolu spojeny.

Tabulka 3.2 – vstupy a výstupy funkčního bloku LOGOP

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
IN1..IN8	INPUT	BOOL		FALSE	Vstup 1..Vstup 8
USED1..USED8	INPUT	BOOL		TRUE	Vstup 1..Vstup 8 je použit
OP	INPUT	BYTE		B#16#1	Výsledná logická operace, která se provádí s IN1..IN8. 0=nic 1=OR 2=AND 3=NOR 4=NAND
BYPASS_PRE	INPUT	BOOL		FALSE	Bypass z předchozího bloku.
BYPASS_EN	INPUT	BOOL		TRUE	Povolení bypass log.operace.
Q	OUTPUT				Výstupní hodnota bloku.

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
					Výsledek logické operace STAT.OP_RESULT nebo přemostění
Q_ERR	OUTPUT				Chyba vstupních hodnot.
Q_BYPASS_SUM	OUTPUT				Přenos přemostění podmínek do dalšího bloku

3.4 Funkční blok AI

Funkční blok AI slouží ke sjednocení monitorování analogového signálu. Blok tak vlastně slouží k monitorování měřené analogové hodnoty. Tento blok se dá navíc kombinovat s porovnávacím blokem FBHMI_CMPOP.

Tabulka 3.3 – vstupy a výstupy funkčního bloku AI

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
I_PV	INPUT	REAL		0.0	Vstupní hodnota.
I_PER	INPUT	INT	-32767.. ..+32768	0	Vstupní signál ze vstupu AI.
CSF	INPUT	BOOL		FALSE	Externí porucha.
USER	INPUT	BOOL		FALSE	Uživatelské hlášení.
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch. Je shodné s FP_CMD.ERR_RES
OCC	INPUT	BOOL		FALSE	Alokace zařízení (pouze signalizace).
BYPASS	INPUT	BOOL		FALSE	Signalizace překlenutých blokačních podmínek v předchozím bloku.
TIME_LAG	INPUT	REAL	SAMPLE_T/2 ...	25.0	Časový filtr vstupního signálu I_PV [s].
TYP	INPUT	DWORD		DW#16#1	Typ vstupu
SAMPLE_T	INPUT	REAL		1.0	Délka cyklu programu [s].
Q	OUTPUT	REAL		0.0	Výstupní hodnota [].
Q_ERR	OUTPUT	BOOL		FALSE	Status – souhrnná porucha. Je shodné s STAT.
Q_RESET	OUTPUT	BOOL		FALSE	Proveden RESET poruch pomocí ACK nebo FP_CMD.ERR_RES

3.5 Funkční blok DI

Funkční blok FBHMI_DI slouží ke sjednocení monitorování digitálních vstupů. Blok také může generovat 2 druhy poruchových hlášení, kdy jedno závisí na externím vstupním signálu CSF, druhé závisí na hodnotě výstupního signálu Q (0 nebo 1).

Tabulka 3.4 – vstupy a výstupy funkčního bloku DI

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
I	INPUT	BOOL		FALSE	Vstupní signál.
CSF	INPUT	BOOL		FALSE	Externí porucha.
INH	INPUT	BOOL		FALSE	Potlačení poruchy.
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
OCC	INPUT	BOOL		FALSE	Alokace zařízení (pouze signalizace).
BYPASS	INPUT	BOOL		FALSE	Signalizace překlenutých blokačních podmínek v předchozím bloku.
TIME_MON	INPUT	REAL	0,1 ...	5,0	Časový filtr vstupního signálu I [s].
TIME_ERR	INPUT	REAL	0,0 ...	1,0	Časové zpoždění poruchy [s].
TYP	INPUT	DWORD		DW#16#1	Typ vstupu
SAMPLE_T	INPUT	REAL		1.0	Délka cyklu programu [s].
Q	OUTPUT	BOOL		FALSE	Výstupní hodnota.
Q_ERR	OUTPUT	BOOL		FALSE	Status – souhrnná porucha.
Q_RESET	OUTPUT	BOOL		FALSE	Proveden RESET poruch pomocí ACK nebo FP_CMD.ERR_RES

3.6 Funkční blok DiagPB

Blok FBHMI_DiagPB sjednocuje monitorování stavu decentralizovaných periferních jednotek připojených k CPU pomocí sítě Profibus DP. Tento funkční blok generuje poruchová hlášení na základě aktuálního stavu periferie. Ke své funkčnosti využívá funkci „Report Systém Errors“, která je součástí STEP 7. Tato funkce slouží k zobrazování diagnostických informací ve formě zpráv, které jsou automaticky převáděny do WinCC.

Tabulka 3.5 - vstupy a výstupy funkčního bloku DiagPB

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
DB_PBIO	INPUT	BLOCK_DB			Datový blok, který je výstupem „Report System Error“ pro Profibus DP.
PB_SYSTEM	INPUT	INT		0	Číslo Profibus DP systému (podle HW konfigurace)
PB_DEVICE	INPUT	INT		0	Adresa Profibus DP stanice.
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
SAMPLE_T	INPUT	REAL		1.0	Délka cyklu programu [s].
RETVAL	OUTPUT	WORD		0	Výstupní diagnostická hodnota: W#16#10 – chybné DB_PBIO W#16#21 – PB_SYSTEM neexistuje W#16#22 – PB_DEVICE neexistuje
Q_OK	OUTPUT	BOOL		FALSE	Status – sledovaná stanice je v pořádku. Je shodné s STAT.OK
Q_FAULTY	OUTPUT	BOOL		FALSE	Status – sledovaná stanice má poruchu. Je shodné s STAT.FAULTY
Q_FAILED	OUTPUT	BOOL		FALSE	Status – sledovaná stanice nekomunikuje.
Q_NOTCONFIG	OUTPUT	BOOL		FALSE	Status – sledovaná stanice není konfigurovaná.

3.7 Funkční blok DiagPN

Blok FBHMI_DiagPN sjednocuje monitorování stavu decentralizovaných periferních jednotek připojených k CPU pomocí sítě Profinet, jinak funguje stejně jako **Funkční blok DiagPB**.

Tabulka 3.6 – vstupy a výstupy funkčního bloku DiagPN

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
DB_PNIO	INPUT	BLOCK_DB			Datový blok, který je výstupem „Report System Error“ pro Profinet.

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
PN_SYSTEM	INPUT	INT		0	Číslo Profinet systému (podle HW konfigurace)
PN_DEVICE	INPUT	INT		0	Číslo zařízení na Profinetu (podle HW konfigurace).
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
SAMPLE_T	INPUT	REAL		1.0	Délka cyklu programu [s].
RETVAL	OUTPUT	WORD		0	Výstupní diagnostická hodnota: W#16#10 – chybné DB_PNIO W#16#20 – není žádný Profinet system W#16#21 – PN_SYSTEM neexistuje W#16#22 – PN_DEVICE neexistuje
Q_OK	OUTPUT	BOOL		FALSE	Status – sledovaná stanice je v pořádku
Q_FAULTY	OUTPUT	BOOL		FALSE	Status – sledovaná stanice má poruchu.
Q_FAILED	OUTPUT	BOOL		FALSE	Status – sledovaná stanice nekomunikuje.
Q_NOTCONFIG	OUTPUT	BOOL		FALSE	Status – sledovaná stanice není konfigurovaná.

3.8 Funkční blok PIDC

Funkční blok FBHMI_PIDC slouží ke sjednocení monitorování spojitého PID regulátoru. Pro vytvoření tohoto funkčního bloku se využívá knihovny ModularPID, která obsahuje specializované funkční bloky vhodné k nastavování a ovládání regulátorů. Tyto bloky jsou udělány tak, aby spolu snadno pracovaly.

Tabulka 3.7 – vstupy a výstupy funkčního bloku PIDC

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
PV	INPUT	REAL		0.0	Vstupní měřená hodnota.

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
DISV	INPUT	REAL		0.0	Poruchová veličina.
CSF	INPUT	BOOL		FALSE	Externí porucha.
COND_ON	INPUT	BOOL		TRUE	Podmínka zapnutí regulátoru.
EN_MAN	INPUT	BOOL		TRUE	Povolení přepnutí regulátoru do ručního režimu pomocí MAN nebo FP_CMD.MAN. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
MAN	INPUT	BOOL		FALSE	Příkaz - přepnutí do ručního režimu (pouze když MAN==1). Ruší (RESET) příznak STAT.AUT.
EN_AUT	INPUT	BOOL		TRUE	Povolení přepnutí regulátoru do automatického režimu pomocí AUT nebo FP_CMD.AUT. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
AUT	INPUT	BOOL		FALSE	Příkaz - přepnutí do automatického režimu (pouze když EN_AUT==1). Nastavuje (SET) příznak STAT.AUT.
AUT_RUN	INPUT	BOOL		FALSE	Příkaz - zapnutí==1 / vypnutí==0 regulátoru v automatickém režimu (pouze když STAT.AUT==1).
AUT_SP	INPUT	REAL	FP_PAR.SP_LMN ... FP_PAR.DP_HLM	0.0	Požadovaná hodnota vstupní veličiny PV v automatickém režimu.
AUT_LMN	INPUT	REAL	FP_PAR.LMN_LLM ... FP_PAR.LMN_HLM	0.0	Výstupní hodnota regulátoru Q_LMN v automatickém režimu při vypnutém regulátoru.
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
OCC	INPUT	BOOL		FALSE	Alokace zařízení (pouze signalizace)

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
BYPASS	INPUT	BOOL		FALSE	Signalizace překlenutých blokačních podmínek v předchozím bloku.
RESET	INPUT	BOOL		FALSE	Celkový reset regulátoru.
TYP	INPUT	DWORD		1	Typ vstupu
SAMPLE_PID	INPUT	REAL		2.0	Perioda volání PID [s].
SAMPLE_T	INPUT	REAL		0.1	Délka cyklu programu [s].
Q_LMN	OUTPUT	REAL	FP_PAR.LMN _LLM ... FP_PAR.LMN _HLM	0.0	Výstupní hodnota regulátoru [].
Q_ON	OUTPUT	BOOL		FALSE	Status - zapnutí regulátoru.
Q_ERR	OUTPUT	BOOL		FALSE	Status – souhrnná porucha.
Q_AUT	OUTPUT	BOOL		FALSE	Status – regulátor je v automatickém režimu.
Q_RESET	OUTPUT	BOOL		FALSE	Proveden RESET poruch pomocí ACK nebo FP_CMD.ACK

3.9 Funkční blok PIDS

Funkční blok FBHMI_PIDS slouží ke sjednocení monitorování krokového PID regulátoru. Jinak tento blok používá stejnou knihovnu ModularPID a má stejné možnosti kombinace, jako **Funkční blok PIDC**. Parametry těchto dvou bloků jsou téměř shodné.

Tabulka 3.8 – vstupy a výstupy funkčního bloku PIDS

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
PV	INPUT	REAL		0.0	Vstupní měřená hodnota.
DISV	INPUT	REAL		0.0	Poruchová veličina.
CSF	INPUT	BOOL		FALSE	Externí porucha.
COND_ON	INPUT	BOOL		TRUE	Podmínka zapnutí regulátoru.
EN_MAN	INPUT	BOOL		TRUE	Povolení přepnutí regulátoru do ručního režimu pomocí MAN nebo FP_CMD.MAN.

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
					Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
MAN	INPUT	BOOL		FALSE	Příkaz - přepnutí do ručního režimu (pouze když EN_MAN==1). Ruší (RESET) příznak STAT.AUT.
EN_AUT	INPUT	BOOL		TRUE	Povolení přepnutí regulátoru do automatického režimu pomocí AUT nebo FP_CMD.AUT. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
AUT	INPUT	BOOL		FALSE	Příkaz - přepnutí do automatického režimu (pouze když EN_AUT==1). Nastavuje (SET) příznak STAT.AUT.
AUT_RUN	INPUT	BOOL		FALSE	Příkaz - zapnutí==1 / vypnutí==0 regulátoru v automatickém režimu (pouze když STAT.AUT==1). Nastavuje STAT.CMD_ON.
AUT_SP	INPUT	REAL	FP_PAR.SP_LMN ... FP_PAR.DP_HLM	0.0	Požadovaná hodnota vstupní veličiny PV v automatickém režimu.
AUT_LMNUP	INPUT	BOOL		FALSE	Nastavení výstupní hodnoty regulátoru Q_LMNUP v automatickém režimu při vypnutém regulátoru.
AUT_LMNDN	INPUT	BOOL		FALSE	Nastavení výstupní hodnoty regulátoru Q_LMNDN v automatickém režimu při vypnutém regulátoru.

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
OCC	INPUT	BOOL		FALSE	Alokace zařízení (signalizace).
BYPASS	INPUT	BOOL		FALSE	Signalizace překlenutých blokačních podmínek v předchozím bloku.
RESET	INPUT	BOOL		FALSE	Celkový reset regulátoru.
TYP	INPUT	DWORD		1	Typ vstupu
SAMPLE_PID	INPUT	REAL		2.0	Perioda volání PID [s].
SAMPLE_T	INPUT	REAL		0.1	Délka cyklu programu [s].
Q_LMNUP	OUTPUT	BOOL		FALSE	Výstupní hodnota regulátoru UP
Q_LMNDN	OUTPUT	BOOL		FALSE	Výstupní hodnota regulátoru DOWN
Q_ON	OUTPUT	BOOL		FALSE	Status - zapnutí regulátoru
Q_ERR	OUTPUT	BOOL		FALSE	Status – souhrnná porucha.
Q_AUT	OUTPUT	BOOL		FALSE	Status – regulátor je v automatickém režimu.
Q_RESET	OUTPUT	BOOL		FALSE	Proveden RESET poruch

3.10Funkční blok AQM

Funkční blok FBHMI_AQM slouží k ovládání motorů s frekvenčním měničem. Ovládání motorů je možné pomocí několika výstupů a připojením libovolné kombinace snímačů.

Tabulka 3.9 – vstupy a výstupy funkčního bloku AQM

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
FB_SPD	INPUT	REAL		0.0	Zpětné hlášení o rychlosti motoru [].
FB_RUNA	INPUT	BOOL		FALSE	Zpětné hlášení o chodu motoru ve směru A.
FB_RUNB	INPUT	BOOL		FALSE	Zpětné hlášení o chodu motoru ve směru B.
FB_DRE	INPUT	BOOL		TRUE	Zpětné hlášení o poruše motoru (ochrana).

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
FB_SER	INPUT	BOOL		TRUE	Zpětné hlášení o odpojení servisního vypínače motoru.
FB_FMON	INPUT	BOOL		FALSE	Zpětné hlášení o zapnutém frekvenčním měniči.
CSF	INPUT	BOOL		FALSE	Externí porucha motoru.
COND_RUNA	INPUT	BOOL		TRUE	Podmínka zapnutí motoru ve směru A
COND_RUNB	INPUT	BOOL		TRUE	Podmínka zapnutí motoru ve směru B.
COND_FM	INPUT	BOOL		TRUE	Podmínka zapnutí frekvenčního měniče.
LOC	INPUT	BOOL		FALSE	Místní (lokální) ovládání motoru (1=místní, 0=dálkové). Přepíše se do statusu STAT.LOCAL. Místní ovládání má větší prioritu než dálkové.
EN_MAN	INPUT	BOOL		TRUE	Povolení přepnutí do ručního režimu pomocí MAN nebo FP_CMD.MAN. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
MAN	INPUT	BOOL		FALSE	Příkaz - přepnutí do ručního režimu (pouze když EN_MAN==1). Ruší (RESET) příznak STAT.AUT.
EN_AUT	INPUT	BOOL		TRUE	Povolení přepnutí do automatického režimu pomocí AUT nebo FP_CMD.AUT. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
AUT	INPUT	BOOL		FALSE	Příkaz - přepnutí do automatického režimu (pouze když EN_AUT==1).
AUT_RUNA	INPUT	BOOL		FALSE	Příkaz - zapnutí==1 / vypnutí==0 motoru ve směru A v automatickém režimu (pouze když STAT.AUT==1).

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
AUT_RUNB	INPUT	BOOL		FALSE	Příkaz - zapnutí==1 / vypnutí==0 motoru ve směru B v automatickém režimu (pouze když STAT.AUT==1).
AUT_SPD	INPUT	REAL		0.0	Požadovaná rychlost v automatickém režimu.
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
OCC	INPUT	BOOL		FALSE	Alokace ventilu (pouze signalizace).
BYPASS	INPUT	BOOL		FALSE	Signalizace překlenutých blokačních podmínek
TYP	INPUT	DWORD		0102	Typ motoru
SAMPLE_T	INPUT	REAL		1.0	Délka cyklu programu [s].
Q_SPD	OUTPUT	REAL		0.0	Výstup pro nastavení rychlosti motoru []
Q_CMD_A	OUTPUT	BOOL		FALSE	Výstup pro ovládání motoru ve směru A
Q_CMD_B	OUTPUT	BOOL		FALSE	Výstup pro ovládání motoru ve směru B
Q_CMD_FMON	OUTPUT	BOOL		FALSE	Výstup pro zapnutí frekvenčního měniče.
Q_ERR	OUTPUT	BOOL		FALSE	Status – souhrnná porucha motoru.
Q_AUT	OUTPUT	BOOL		FALSE	Status – motor je v automatickém režimu.
Q_RUNNINGA	OUTPUT	BOOL		FALSE	Status – motor běží ve směru A.
Q_RUNNINGB	OUTPUT	BOOL		FALSE	Status – motor běží ve směru B.
Q_STOPPED	OUTPUT	BOOL		FALSE	Status – motor stojí.
Q_RESET	OUTPUT	BOOL		FALSE	Proveden RESET poruch pomocí ACK nebo FP_CMD.ACK

3.11Funkční blok DQM

Funkční blok FBHMI_DQM slouží k ovládání motorů bez frekvenčního měniče. Jinak má blok stejné režimy ovládání a možnosti kombinace s blokem FBHMI_LOGOP (kromě kontroly frekvenčního měniče) jako **Funkční blok AQM**.

Vstupy a výstupy jsou téměř stejné, jako vstupy a výstupy bloku AQM, chybí zde pouze ty parametry, které se týkají ovládání rychlosti a frekvenčního měniče.

3.12Funkční blok DQV

Funkční blok FBHMI_DQV sjednocuje ovládání ventilů. Ventily můžeme ovládat pomocí několika ovládacích výstupů a připojením libovolné kombinace koncových snímačů. Stejně jako bloky pro ovládání motorů s a bez frekvenčního měniče, i zde máme tři režimy ovládání – lokální, manuální dálkový a automatický dálkový.

Tabulka 3.10 – vstupy a výstupy funkčního bloku DQV

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
FB_POS	INPUT	REAL	0...100	0.0	Aktuální analogová poloha ventilu [%]
FB_OPN	INPUT	BOOL		FALSE	Zpětné hlášení od koncového snímače ventil otevřen.
FB_CLS	INPUT	BOOL		FALSE	Zpětné hlášení od koncového snímače ventil zavřen.
CSF	INPUT	BOOL		FALSE	Externí porucha ventilu.
COND_OPN	INPUT	BOOL		TRUE	Podmínka otevření ventilu
COND_CLS	INPUT	BOOL		TRUE	Podmínka zavření ventilu
LOC	INPUT	BOOL		FALSE	Místní (lokální) ovládání ventilu (1=místní, 0=dálkové).
EN_MAN	INPUT	BOOL		TRUE	Povolení přepnutí do ručního režimu pomocí MAN nebo FP_CMD.MAN. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
MAN	INPUT	BOOL		FALSE	Příkaz - přepnutí do ručního režimu (pouze když EN_MAN==1).
EN_AUT	INPUT	BOOL		TRUE	Povolení přepnutí do automatického režimu pomocí AUT nebo FP_CMD.AUT. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
AUT	INPUT	BOOL		FALSE	Příkaz - přepnutí do automatického režimu (pouze když EN_AUT==1).
AUT_OPN	INPUT	BOOL		FALSE	Příkaz - otevření==1 / zavření==0 ventilu v automatickém režimu
AUT_CLS	INPUT	BOOL		FALSE	Příkaz - zavření==1 / otevření==0 ventilu v automatickém režimu
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
OCC	INPUT	BOOL		FALSE	Alokace ventilu (pouze signalizace).
BYPASS	INPUT	BOOL		FALSE	Signalizace překlenutých blokačních podmínek.
TYP	INPUT	DWORD		1110	Typ ventilu
SAMPLE_T	INPUT	REAL		1.0	Délka cyklu programu [s].
Q_CMD_OPN	OUTPUT	BOOL		FALSE	Výstup pro ovládání otvírání ventilu.
Q_CMD_CLS	OUTPUT	BOOL		FALSE	Výstup pro ovládání zavírání ventilu.
Q_ERR	OUTPUT	BOOL		FALSE	Status – souhrnná porucha ventilu.
Q_AUT	OUTPUT	BOOL		FALSE	Status – ventil je v automatickém režimu
Q_OPENED	OUTPUT	BOOL		FALSE	Status – poloha ventilu otevřeno.
Q_CLOSED	OUTPUT	BOOL		FALSE	Status – poloha ventilu zavřeno.
Q_RESET	OUTPUT	BOOL			Proveden RESET poruch pomocí ACK nebo FP_CMD.ACK

3.13Funkční blok AQV

Funkční blok FBHMI_AQV sjednocuje ovládání ventilů. Ventily můžeme ovládat pomocí analogového výstupu (na rozdíl od bloku FBHMI_DQV) a připojením libovolné kombinace koncových snímačů. Co se ovládání týče, chová se tento blok stejně, jako blok FBHMI_DQV, kombinace s výstupem bloku FBHMI_LOGOP.Q je možná přivedením tohoto signálu na vstup FBHMI_AQV.COND_OPN, pak se ve faceplate objeví tlačítko pro zobrazení podmínky otevření ventilu.

Tabulka 3.11 – vstupy a výstupy funkčního bloku AQV

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
FB_POS	INPUT	REAL	0...100	0.0	Aktuální analogová poloha ventilu [%]
FB_OPN	INPUT	BOOL		FALSE	Zpětné hlášení od koncového snímače ventil otevřen.
FB_CLS	INPUT	BOOL		FALSE	Zpětné hlášení od koncového snímače ventil zavřen.
CSF	INPUT	BOOL		FALSE	Externí porucha
COND_OPN	INPUT	BOOL		TRUE	Podmínka otevření ventilu.
LOC	INPUT	BOOL		FALSE	Místní (lokální) ovládání ventilu (1=místní, 0=dálkové).
EN_MAN	INPUT	BOOL		TRUE	Povolení přepnutí do ručního režimu pomocí MAN nebo FP_CMD.MAN. Jestliže EN_AUT==0 a současně EN_MAN==0, tak se nastaví automatický režim.
MAN	INPUT	BOOL		FALSE	Příkaz - přepnutí do ručního režimu (pouze když EN_MAN==1).
EN_AUT	INPUT	BOOL		TRUE	Povolení přepnutí do automatického režimu pomocí AUT nebo FP_CMD.AUT.
AUT	INPUT	BOOL		FALSE	Příkaz - přepnutí do automatického režimu (pouze když EN_AUT==1).
AUT_POS	INPUT	REAL	0...100%	0.0	Příkaz – požadovaná poloha otevření ventilu v automatickém režimu
ACK	INPUT	BOOL		FALSE	Reset (potvrzení) poruch.
OCC	INPUT	BOOL		FALSE	Alokace ventilu.
BYPASS	INPUT	BOOL		FALSE	Signalizace překlenutých blokačních podmínek.
TYP	INPUT	DWORD		1110	Typ ventilu
POS_CLS	INPUT	REAL	0..100	5.0	Poloha zavření ventilu [%].
POS_OPN	INPUT	REAL	0..100	95.0	Poloha otevření ventilu [%].

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
SAMPLE_T	INPUT	REAL		1.0	Délka cyklu programu [s].
Q_CMD_POS	OUTPUT	REAL	0...100	0.0	Výstup pro ovládání ventilu [%].
Q_PER	OUTPUT	INT	0..+32768	0	Výstupní signál pro AO. Výstupní hodnota Q_CMD_POS [0-100%]
Q_ERR	OUTPUT	BOOL		FALSE	Status – souhrnná porucha ventilu.
Q_AUT	OUTPUT	BOOL		FALSE	Status – motor je v automatickém režimu
Q_OPENED	OUTPUT	BOOL		FALSE	Status – poloha ventilu otevřeno.
Q_CLOSED	OUTPUT	BOOL		FALSE	Status – poloha ventilu zavřeno.
Q_RESET	OUTPUT	BOOL			Proveden RESET poruch pomocí ACK nebo FP_CMD.ACK

3.14Funkční blok PHASE

Funkční blok FBHMI_PHASE slouží k vytvoření a sjednocení stavové logiky fází.

Tabulka 3.12 – vstupy a výstupy funkčního bloku PHASE

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
EN_MAN	INPUT	BOOL		TRUE	Povolení přepnutí fáze do ručního režimu
MAN	INPUT	BOOL		FALSE	Příkaz - přepnutí do ručního režimu
EN_AUT	INPUT	BOOL		TRUE	Povolení přepnutí fáze do automatického režimu
AUT	INPUT	BOOL		FALSE	Příkaz - přepnutí do auto režimu
START	INPUT	BOOL		FALSE	Příkaz – start fáze
PAUSE	INPUT	BOOL		FALSE	Příkaz – pozastavení fáze
HOLD	INPUT	BOOL		FALSE	Příkaz – přerušení fáze
COMPLETE	INPUT	BOOL		FALSE	Příkaz – dokončení fáze
STOP	INPUT	BOOL		FALSE	Příkaz – zastavení fáze
ABORT	INPUT	BOOL		FALSE	Příkaz – zrušení fáze
RESUME	INPUT	BOOL		FALSE	Příkaz – obnovení fáze
RESTART	INPUT	BOOL		FALSE	Příkaz – restartování fáze

Parametr	Typ	Datový typ	Rozsah hodnot	Init. Hodnota	Popis
RESET	INPUT	BOOL		FALSE	Příkaz – reset fáze.
EN_START	INPUT	BOOL		TRUE	Povolení startu fáze,
EN_PAUSE	INPUT	BOOL		TRUE	Povolení pozastavení fáze.
EN_HOLD	INPUT	BOOL		TRUE	Povolení přerušení fáze.
EN_COMPLETE	INPUT	BOOL		TRUE	Povolení dokončení fáze.
EN_STOP	INPUT	BOOL		TRUE	Povolení zastavení fáze.
EN_ABORT	INPUT	BOOL		TRUE	Povolení zrušení fáze.
EN_RESUME	INPUT	BOOL		TRUE	Povolení obnovení fáze.
EN_RESTART	INPUT	BOOL		TRUE	Povolení restartování fáze.
TO_RUNNING	INPUT	BOOL		FALSE	Signál pro přechod do RUNNING
TO_PAUSED	INPUT	BOOL		TRUE	Signál pro přechod do PAUSED
TO_HELD	INPUT	BOOL		TRUE	Signál pro přechod do HELD
TO_ABORTED	INPUT	BOOL		TRUE	Signál pro přechod do ABORTED
TO_STOPPED	INPUT	BOOL		TRUE	Signál pro přechod do STOPPED
TO_COMPLETED	INPUT	BOOL		TRUE	Signál pro přechod do COMPLETED
ACK	INPUT	BOOL		FALSE	Potvrzení poruch.
BYPASS	INPUT	BOOL		FALSE	Signalizace BYPASS.
C_NSTEP	INPUT	BOOL		FALSE	Podmínka pro příkaz NEXT_STEP
OREQ	INPUT	BOOL		FALSE	Signalizace operator request.
TYP	INPUT	DWORD		1	Typ vstupu
SAMPLE_T	INPUT	REAL		0.1	Délka cyklu programu [s].
Q_AUT	OUTPUT	BOOL		FALSE	Status – automatický režim
Q_MAN	OUTPUT	BOOL		FALSE	Status – fáze je v ručním režimu.
Q_ACT	OUTPUT	BOOL		FALSE	Status – fáze je aktivní.
Q_WORK	OUTPUT	BOOL		FALSE	Status – fáze je v chodu.
Q_RESET	OUTPUT	BOOL		FALSE	Proveden příkaz RESET pomocí RESET nebo FP_CMD.RESET.
Q_NEXTSTEP	OUTPUT	BOOL		FALSE	Proveden příkaz NEXT_STEP.
Q_FORCESTEP	OUTPUT	BOOL		FALSE	Proveden příkaz FORCE_STEP.
UMSG_STAT	STAT	BYTE		0	Stav pro uživatelská hlášení
UMSG	STAT	STRUCT			Signály uživatelských hlášení.

4 IMPORT A EXPORT PROGRAMOVÝCH BLOKŮ

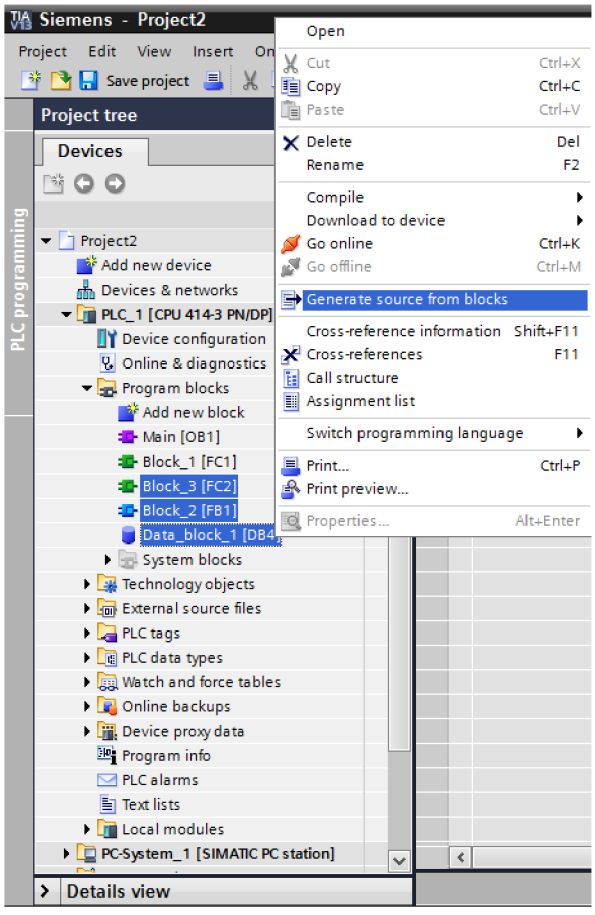
Pro generátor PLC kódu, vytvořený v této bakalářské práci, je jednou ze nejdůležitějších věcí možnost importu a exportu programových bloků do a z prostředí TIA Portal V13 za pomocí MS Excel.

4.1 Export programových bloků do programu MS Excel

Otevřeme projekt v TIA Portal V13 a přejdeme do zobrazení „Project View“. Rozklikneme položku „Program blocks“, ve které se nachází programové bloky. Možnost generování zdrojového kódu závisí na programovacím jazyku (o programovacích jazycích více v kapitole **STEP 7 Professional V13).**

4.1.1 Programovací jazyk SCL

Vybereme jeden nebo více bloků, které budeme chtít exportovat, klikneme na ně pravým tlačítkem a zvolíme možnost „Generate source from blocks“. Po zvolení této možnosti se objeví dialogové okno umožňující uložení souboru, který bude mít příponu .scl.

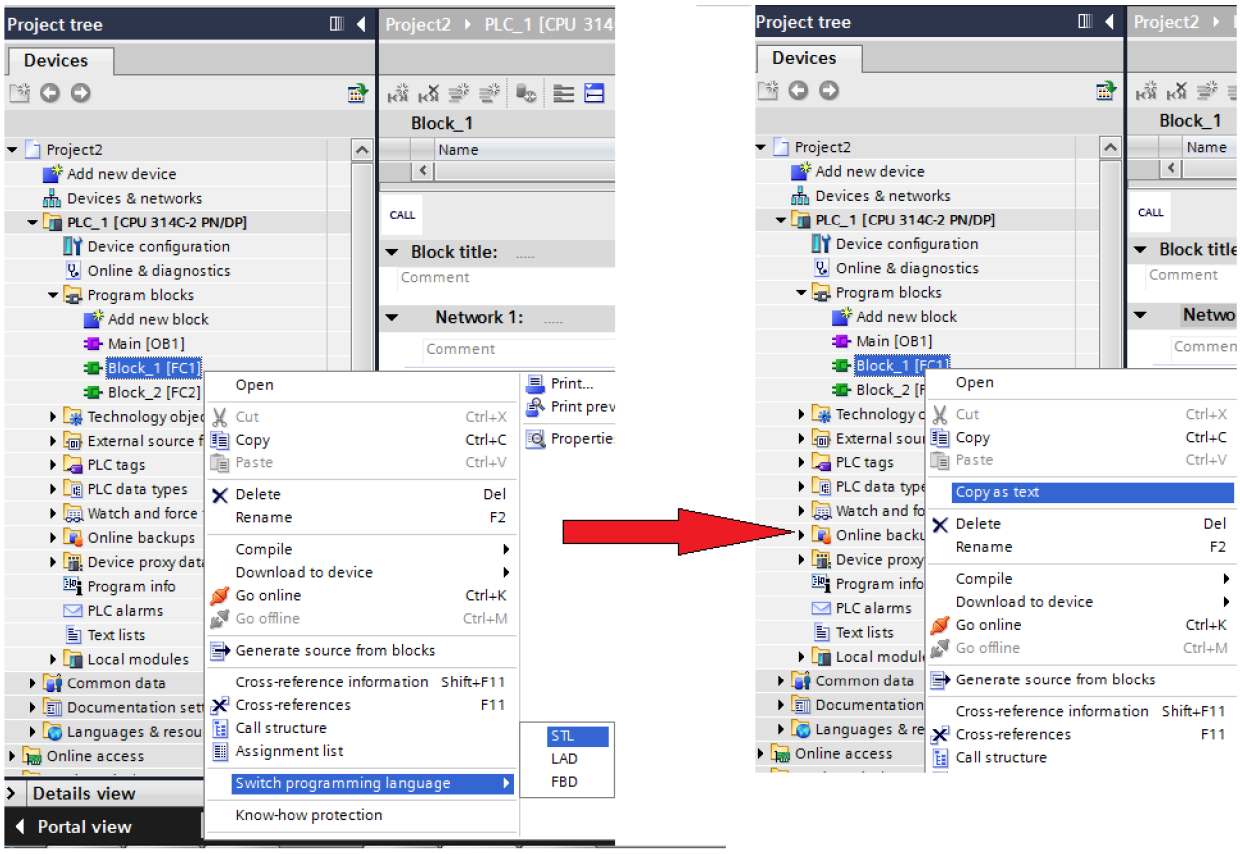


Obrázek 4.1– generování kódu bloků, jazyk SCL

4.1.2 Programovací jazyky STL, LAD, FBD

Druhým programovacím jazykem, ze kterého jde vytvářet zdrojový kód, je jazyk STL. Z grafických jazyků, jako je LAD, FBD a GRAPH nelze vytvářet zdrojový kód. U jazyků LAD a FBD je zde však možnost změny programovacího jazyka. Pokud budeme v blocích naprogramovaných v jazycích LAD nebo FBD používat pouze instrukce, které se dají převést do jazyka STL, kliknutím pravým tlačítkem na příslušný blok můžeme rozkliknout možnost „Switch programming language“ a zvolit STL. Změna programovacího jazyka z LAD nebo FBD do STL je však možná pouze u PLC SIMATIC řady S7-300 a S7-400, pro řady S7-1200 a S7-1500 tento postup není možné aplikovat.

Pokud máme blok psaný v programovacím jazyku STL, nemůžeme použít ani možnost „Generate source from blocks“, protože tato možnost se dá aplikovat pouze na programové bloky napsané v jazyce SCL, které nemají ochranu. Pro jazyk STL musíme pravým tlačítkem kliknout na programový blok (můžeme mít označený pouze jeden blok) a zvolíme možnost „Copy as text“. Tímto se nám uloží zdrojový kód bloku do schránky a my ho poté můžeme vložit rovnou do MS Excel nebo jiného textového editoru. Poté již ručně uložíme soubor jako soubor s příponou .awl.



Obrázek 4.2– generování kódu bloku, jazyky LAD, FBD a STL

4.1.3 Otevření souboru .scl nebo .awl v MS Excel

Soubory .scl a .awl jsou textové soubory, obsahující zdrojový kód programového bloku (bloků). Pokud chceme tento soubor otevřít v programu MS Excel 2010 nebo novějším, zvolíme v nabídce Soubor možnost Otevřít. Po otevření okna s možností souborů k otevření, v pravém spodním rohu musíme zvolit Všechny soubory a poté musíme najít soubor, který chceme otevřít. Tímto se nám otevře Průvodce importem textu. V tomto průvodci můžeme zvolit, zda se budou pole oddělovat zvláštním znakem (tabulátor, čárka, …) nebo budou mít všechna pole stejnou šířku (všechny sloupce v MS Excel budou mít pevnou šířku). Můžeme zde zvolit i formát dat v jednotlivých sloupcích (text, číslo, …), přitom v dolní části průvodce vidíme náhled dat, který se mění, pokud měníme přednastavené možnosti. Nakonec stačí zmáčknout tlačítko Dokončit a tímto se textový soubor naimportuje do programu MS Excel, kde s ním můžeme provádět další úpravy.

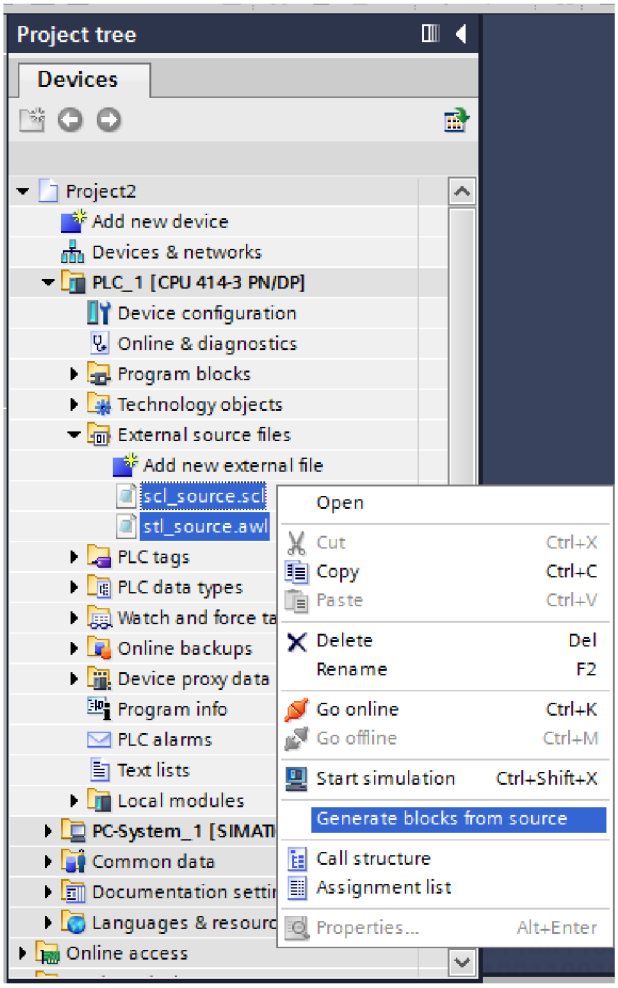
4.2 Export programových bloků z MS Excel do TIA Portal

Pokud v programu MS Excel vytvoříme nebo upravíme zdrojový kód pro programové bloky, musíme takto vytvořený soubor uložit. V nabídce Soubor klikneme na položku Uložit jako. Zde si zvolíme adresu, kam se má výsledný soubor uložit.

Pokud chceme, aby byly programové bloky napsané v jazyce SCL, vybereme v poli Uložit jako typ možnost Formátovaný text (oddělený mezerami). Soubor pojmenujeme a na konec názvu musíme přidat příponu .scl, Poté klikneme na Uložit.

V případě, že chceme programové bloky v jazyce STL, postupujeme stejně jako v předchozím případě jen s tím rozdílem, že na konec názvu přidáme koncovku .awl.

Pro nahrávání programových bloků z textových zdrojových souborů do TIA Portal V13 musíme v zobrazení „Project View“ otevřít složku „External source files“. Zde zvolíme možnost „Add new external file“, objeví se dialogové okno s možností výběru souborů. V tomto okně označíme naše požadované textové soubory s koncovkou .scl nebo .awl a klikneme na tlačítko „Open“. Takto se nám soubory zobrazí v nabídce „External source files“. Pokud chceme ze zdrojových kódů vytvořit programové bloky, označíme příslušné externí zdrojové soubory, klikneme pravým tlačítkem a zvolíme možnost „Generate blocks from source“. Objeví se dialogové okno oznamující, že některé bloky mohou být přepsány (pokud již v projektu máme blok, který má shodný název, jako blok, který chceme teprve vytvořit ze zdrojového souboru, data existujícího bloku v souboru se přepíšou). Po potvrzení se v položce „Program blocks“ objeví nové vytvořené bloky.

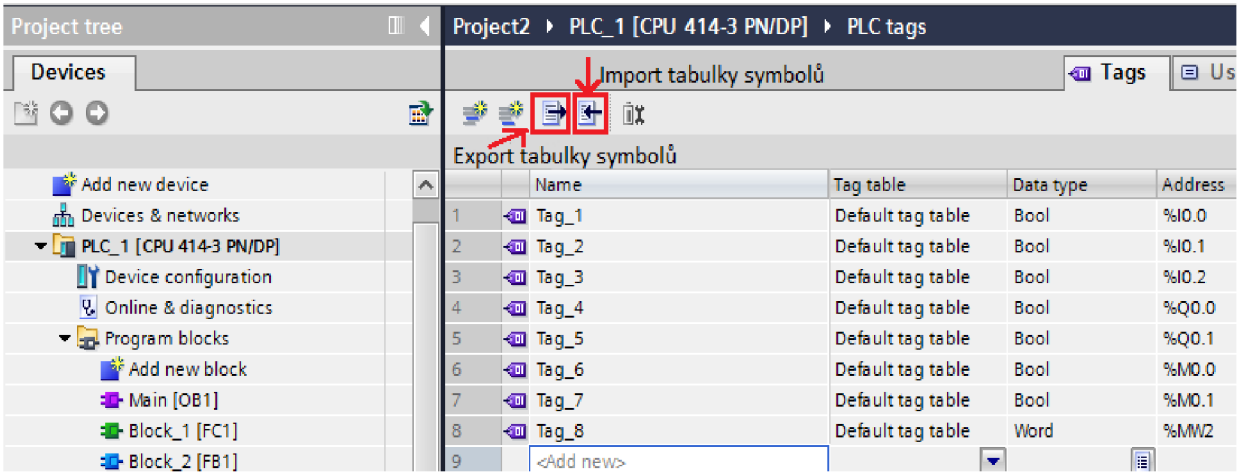


Obrázek 4.3- generování bloků z externího zdrojového souboru

4.3 Export a import tabulky symbolů

Pokud chceme exportovat tabulku symbolů z TIA Portal V13, otevřeme v „Project tree“ záložku „PLC tags“ a dále otevřeme buď tabulku, kterou chceme exportovat nebo otevřeme všechny symboly (možnost „Show all tags“). V horní liště se nachází tlačítko „Export“. Po kliknutí na toto tlačítko se otevře dialogové okno, ve kterém zadáme adresu, kam chceme tabulku exportovat a výběr, zda chceme exportovat symboly nebo i konstanty. Tabulka se ukládá ve formátu .xlsx (tvar pro MS Excel 2007 a novější).

Pokud chceme tabulku symbolů importovat do TIA Portal V13, zvolíme „Show all tags“ a v horní liště klikneme na tlačítko „Import“. Objeví se dialogové okno, do kterého napíšeme adresu souboru s tabulkou symbolů.



Obrázek 4.4 - import a export tabulky symbolů

	A	B	C	D	E	F	G
1	Name	Path	Data Type	Logical Address	Comment	Hmi Visible	Hmi Accessible
2	Tag_1	Default tag table	Bool	%I0.0		True	True
3	Tag_2	Default tag table	Bool	%I0.1		True	True
4	Tag_3	Default tag table	Bool	%I0.2		True	True
5	Tag_4	Default tag table	Bool	%Q0.0		True	True
6	Tag_5	Default tag table	Bool	%Q0.1		True	True
7	Tag_6	Default tag table	Bool	%M0.0		True	True
8	Tag_7	Default tag table	Bool	%M0.1		True	True
9	Tag_8	Default tag table	Word	%MW2		True	True

Obrázek 4.5 - tabulka symbolů v MS Excel 2010

5 NÁVRH GENERÁTORU PLC KÓDU

Generátor PLC kódu pro bloky faceplate je nástroj vytvořený v programu Microsoft Excel. Nástroj na základě dat zadaných do listu Projekční tabulka vygeneruje kód pro jednotlivé programové bloky. Tento kód se generuje do listu AWL kód, v tomto listu je také možnost vytvořený kód uložit jako zdrojový soubor .awl. Ten se později může importovat do programového prostředí TIA Portal (viz **Import a export programových bloků**). Do generátoru je také možno importovat tabulku symbolů z projektu v TIA Portalu, konkrétně na list Tabulka symbolů. List AWL zdroj zase umožňuje vložit .awl soubor, který slouží jako vzor pro tvorbu bloků. Posledním listem je Faceplate IO. Tento list slouží pouze jako seznam názvů vstupů a výstupů jednotlivých funkčních bloků pro objekty faceplate. V této kapitole se pak detailněji rozepíšu o každém listu.

Samotný nástroj je vytvořen pomocí VBA skriptů (zkratka Visual Basic for Applications). Jedná se o implementaci programovacího jazyka Visual Basic od firmy Microsoft. VBA je součástí většiny programů z řady Microsoft Office, umožňuje například vytváření maker pro práci v programech Microsoft Word nebo Microsoft Excel. Tyto programy v sobě mají zabudován VBA editor, ve kterém pak můžeme všechna požadovaná makra snadno naprogramovat. Pokud však v dokumentu vytvořeném pomocí MS Word nebo v sešitu vytvořeném pomocí MS Excel používáme makra, musíme konečný soubor uložit jako Dokument s podporou maker resp. Sešit s podporou maker. [4][5]

5.1 List Projekční tabulka

Toto je hlavní část celého generátoru. Zde do jednotlivých polí programátor zadává data, ze kterých se tvoří výsledný .awl soubor. Každý řádek této tabulky určuje, v jakém místě a v jaké technologické skupině se bude požadovaný faceplate vyskytovat.

Sloupec Skupina obsahuje název technologické skupiny. Názvy těchto technologických skupin jsou dány na základě zadání daného projektu. Projektant nemusí vždy vypisovat všechny prvky dané skupiny pod sebe, generátor je schopen najít a rozpoznat všechny unikátní názvy skupin.

Ve sloupci Číslo se zadává, jaké číslo bloku funkce v programu TIA Portal by zadaný prvek měl mít. Bohužel TIA Portal neumožňuje, na rozdíl od starších STEP7 programů, v externím zdrojovém souboru určit číslo vytvořeného bloku. Proto po domluvě s odborným konzultantem Ing. Zbyňkem Bezchlebou bylo rozhodnuto, že pro samotnou generaci kódu nebude toto pole využíváno.

Dalším sloupcem je Projekční značení. Jedná se o označení daného technologického prvku tak, jak se s ním pracuje v projektové dokumentaci (například ventil V31).

Jako další se zvolí Typ faceplate. Do tohoto sloupce se zadávají zkratky jednotlivých typů faceplate (viz **Analýza stávajících objektů faceplate**) podle toho, jestli chce programátor vytvořit například ventil, motor a další.

Sloupec Vzor přímo souvisí s listem AWL zdroj, proto jeho význam popíšu právě při popisu tohoto listu. Zadává se zde projekční značení vzorového bloku.

Do sloupce Poznámka může programátor vložit jakýkoliv komentář, který se vygeneruje v programu TIA Portal jako komentář u daného bloku. Může zde například popsat konkrétní funkci daného faceplate nebo cokoliv jinak důležitého.

Posledním sloupcem je Umístění bloku. Do tohoto sloupce může programátor zadat buď ručně, nebo pomocí výběru, který se nalézá vedle hlavičky tohoto sloupce, kde se má daný technologický objekt vyskytovat. Každá technologická skupina má totiž vytvořené funkce, které obstarávají vstupy (IN), podmínky (COND), automatické řízení (AUTO), ruční řízení (MAN) a výstupy (OUT). Hodnotou v tomto sloupci se vybírá, do které z těchto kategorií daný prvek patří.

Poté, co projektant vyplní do tabulky všechny údaje, může klinout na tlačítko Generovat a na základě zadaných dat se v listu AWL kód vygenerují data. Pokud chce obsah tabulky smazat, klikne na tlačítko Smazat tabulku.

	A	B	C	D	E	F	G	H
1	Skupina	Číslo	Projekční značení	Typ faceplate	Vzor	Poznámka	Umístění bloku	IN COND AUTO MAN OUT
2	L1		V31	AI	V36	komentar1	IN	
3	L1		V32	DQM	V36	komentar2	COND	
4	L2		H236	CMPOP		komentar3	MAN	
5	D01		GP	DQV	GP5	komentar4	OUT	
6	L111		DV2	AQV		komentar5	OUT	
7	L1		H2	AI	GP5	komentar6	IN	
8	L2		HD5	LOGOP		komentar7	AUTO	
9								
10								

Generovat

Smazat tabulku

Obrázek 5.1 - vyplněná projekční tabulka

5.2 List Tabulka symbolů

Tento list umožňuje importovat tabulku symbolů vytvořenou v programu TIA Portal (export tabulky symbolů z TIA Portal viz **Export a import tabulky symbolů**). Import se provede kliknutím na tlačítko Importovat tabulku. Objeví se okno dialogu otevření souboru, programátor si najde tabulku symbolů, kterou chce vložit do generátoru a klikne na Otevřít. Obsah tabulky symbolů se překopíruje do tohoto listu, původní soubor s tabulkou symbolů zůstane nezměněn i při jakýchkoliv manipulacích s daty v tabulce, které programátor může provést v tomto listu. Smazat importovaná data může pomocí tlačítka Smazat tabulku. Pokud se do tabulky přidají nová data nebo se tabulka vytvoří nová zcela od základu, je zde možnost uložení tabulky ve vhodném formátu pomocí tlačítka Uložit tabulku, stačí pouze vybrat umístění, kam se má tabulka uložit.

Tabulka symbolů se využije pro generaci kódu k přiřazování signálů na vstupy a výstupy použitého funkčního bloku objektu faceplate. Každý z těchto funkčních bloků má definované vstupy a výstupy (viz **Analýza stávajících objektů faceplate**), které mají určité jméno. Pokud se v tabulce symbolů vyskytuje název, který v sobě obsahuje název

požadované skupiny, projekční značení a název vstupu resp. výstupu, tento název a tím pádem i daný signál se přiřadí na správný vstup nebo výstup daného bloku.

Příklad: pokud volám v technologické skupině L1 prvek s projekčním značením V31 a vím, že je to faceplate objekt ventil, budu v tabulce symbolů hledat všechny názvy, které obsahují L1_V31_název_vstupu_výstupu (konkrétně pokud generátor najde v tabulce symbolů signál *I_L1_V31_CSF*, přiřadí tento signál na vstup s názvem *CSF* funkčního bloku objektu faceplate).

Do budoucna by šlo přiřazování signálů na vstupy a výstupy vylepšit, aby nemusely názvy signálů mít striktně danou formu, protože ve výsledném projektu nebudou všechny názvy sjednocené. To je však kvůli omezeným možnostem programování pomocí VBA pokročilá funkčnost.

	Name	Path	Data Type	Logical Address	Comment	Hmi Visible	Hmi Accessible		Smazat tabulku		Importovat tabulku
2	I_L1_GP5_pot	Tag table_1	Bool	%I0.0		PRAVDA	PRAVDA				
3	I_L1_GP5_INH	Tag table_1	Bool	%I0.1		PRAVDA	PRAVDA				
4	I_L1_GP5_ACK	Tag table_1	Bool	%I0.2		PRAVDA	PRAVDA				
5	I_L1_GP5_Uziv	Tag table_1	Bool	%I0.3		PRAVDA	PRAVDA				
6	I_L1_GP5_Potvr	Tag table_1	Bool	%I0.4		PRAVDA	PRAVDA				
7	Q_L1_GP5_chyb	Tag table_1	Bool	%Q0.0		PRAVDA	PRAVDA				
8	vystup_2	Tag table_1	Bool	%Q0.1		PRAVDA	PRAVDA				
9	vystup_3	Tag table_1	Bool	%Q0.2		PRAVDA	PRAVDA				
10	vystup_4	Tag table_1	Bool	%Q0.3		PRAVDA	PRAVDA				
11	pamet_1	Tag table_1	Bool	%M0.0		PRAVDA	PRAVDA				
12	pamet_2	Tag table_1	Bool	%M0.1		PRAVDA	PRAVDA				
13	pamet_3	Tag table_1	Bool	%M0.2		PRAVDA	PRAVDA				
14	pamet_4	Tag table_1	Bool	%M0.3		PRAVDA	PRAVDA				
15	pamet_5	Tag table_1	Bool	%M0.4		PRAVDA	PRAVDA				
16	I_PV	Tag table_1	Real	%MD2		PRAVDA	PRAVDA				
17											
18											

Obrázek 5.2 - ukázka importované tabulky symbolů

5.3 List AWL kód

Do tohoto listu se generuje kód pro vytvoření zdrojového souboru pro TIA Portal. Samotná generace probíhá tak, že se nejprve udělá seznam všech unikátních názvů technologických skupin, jak jsou zadány ve sloupci Skupina na listu Projekční tabulka. Poté generátor vybere první skupinu a vytvoří pro ni kód bloku funkce *FCnázevskupiny* (například FCL1 pro skupinu L1). V každé této hlavní skupinové funkci pak dojde k volání nižších funkcí, které obstarávají umístění daného bloku (například pro funkci FCL1 se vytvoří volání funkcí FCL1_IN, FCL1_COND, FCL1_AUTO atd.). Tím je ukončena tvorba hlavní skupinové funkce.

Dále se vytvoří kód pro každou funkci zajišťující umístění bloku. Pokud v projekční tabulce nebyl do dané funkce umístěn žádný prvek, vytvoří se funkce prázdná, do které později v TIA Portalu může projektant dodělat potřebné úpravy, pokud nějaké budou. Jinak generátor projde celou projekční tabulku a vždy pro danou skupinu a umisťovací funkci vezme všechny technologické prvky, které se v ní mají vyskytovat. Na základě údajů ze sloupce Typ faceplate v listu Projekční tabulka pak vygeneruje volání funkčního bloku příslušného objektu faceplate. Pokud jsou navíc v tabulce symbolů na listu Tabulka

symbolů vyhovující názvy signálů, dojde k přiřazení těchto signálů na požadované vstupy a výstupy daného funkčního bloku. Ke každému volanému funkčnímu bloku objektu faceplate se také automaticky vytvoří instanční datový blok. Tímto je ukončena generace příslušného umisťovacího bloku.

Výše uvedený postup se opakuje pro všechny umisťovací bloky. Když je vytvořen kód pro poslední blok v dané technologické skupině (například blok FCL1_OUT), ze seznamu technologických skupin se vezme další skupina v pořadí a celý postup se opakuje. Po vytvoření všech požadovaných bloků poslední technologické skupiny dojde k vytvoření hlavního organizačního bloku Main (OB1). Tento blok se cyklicky volá v PLC a tím v podstatě umožňuje fungování programu nahraného v PLC. V organizačním bloku vygenerovaném pomocí generátoru PLC kódu pak dojde pouze k volání jednotlivých hlavních skupinových funkcí.

Příklad struktury vytvořeného programu:

Main (OB1)

Volání FCL1 (hlavní skupinová funkce)
Volání FCL1_IN (umisťovací funkce)
Volání FBHMI_DQV (funkční blok pro faceplate ventil)
Volání FCL1_COND (obdobně pro zbytek umisťovacích funkcí)

Volání FCL2 (další hlavní skupinová funkce)
Volání FCL2_IN…

Poté, co je kód vygenerován, může se na něj programátor podívat a provést jakékoliv úpravy. Uložit kód může pomocí tlačítka Uložit jako awl. Objeví se dialogové okno pro uložení souboru, ve kterém si vybere umístění a název vytvořeného zdrojového .awl souboru. Tento soubor pak již snadno může vložit do programu TIA Portal. Pokud si přeje vytvořený kód z listu smazat, stačí kliknout na tlačítko Smazat kód a potvrdit smazání.

	A	B	C	D	E	F	G	H	I	J	K
1									Smazat kód		Uložit jako awl
2	FUNCTION FCL1 : Void										
3	{ S7_Optimized_Access := 'FALSE' }										
4	VERSION : 0.1										
5											
6	BEGIN										
7	NETWORK										
8	TITLE =										
9		CALL "FCL1_IN";									
10		NOP 0;									
11	NETWORK										
12		CALL "FCL1_COND";									
13		NOP 0;									
14	NETWORK										
15		CALL "FCL1_AUTO";									

Obrázek 5.3 - ukázka vygenerovaného kódu

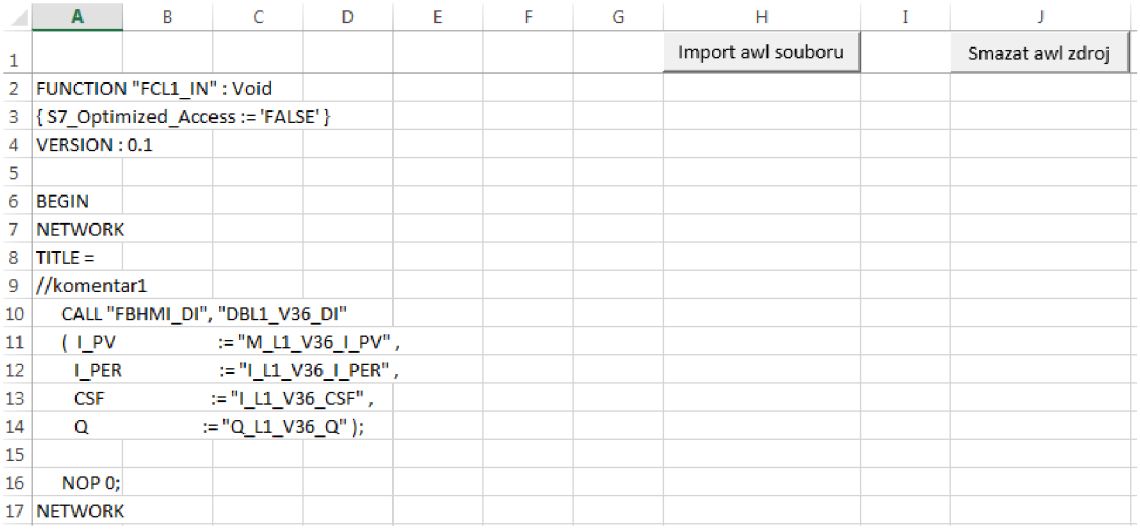
5.4 List AWL zdroj

Tento list umožňuje import .awl souboru, ve kterém je ukázka zapojení vzorového bloku. Programátor vytvoří v TIA Portalu zapojení jednoho technologického prvku, například ventilu. Na vstupy a výstupy přiřadí požadované signály a kód tohoto bloku pak exportuje do samostatného .awl souboru. Kliknutím na tlačítko Import awl souboru se otevře dialogové okno otevření souboru, vybere se požadovaný zdrojový soubor .awl a kliknutím na Otevřít se importuje do tohoto listu. Pokud chce obsah listu smazat, stačí kliknout na tlačítko Smazat awl zdroj (smaže se pouze překopírovaný kód, samotný soubor se zdrojovým kódem zůstane netknutý).

Tento list slouží jako zdrojový kód pro vytvoření volání funkčního bloku objektu faceplate, pokud byl v listu Projekční tabulka vyplněn údaj do sloupce Vzor. Program prohledá kód importovaný ze vzorového .awl souboru a pokud narazí na projekční značení, které je uvedeno právě jako hodnota Vzor v projekční tabulce, začne kopírovat kód tohoto bloku na příslušné místo volání bloku na listu AWL kód. Nevytvoří však pouhou kopii vzorového kódu, generátor změní všechny údaje, ve kterých se vyskytuje technologická skupina a projekční značení vzoru za název technologické skupiny a projekčního značení z projekční tabulky. Jelikož zdroj vzorového kódu byl vytvořen po připojení signálů na vstupy a výstupy, generátor dokáže tyto signály přejmenovat, aby odpovídaly danému prvku projekční tabulky (právě změnou názvu technologické skupiny a projekčního značení). Poté generátor zanalyzuje, zda je takto přejmenovaný signál na listu Tabulka symbolů a pokud ano, překopíruje způsob zapojení ze vzoru do nově vytvořeného bloku.

Příklad použití: programátor v TIA Portalu vytvoří zapojení jednoho funkčního bloku pro objekt faceplate ventil, vzorová technologická skupina bude H52, projekční značení Y14. Tento blok pak uloží jako vzorový .awl soubor. Generátor pak přejmenuje vytvořené volání FB pro ventil například na skupinu L1, projekční značení V31. Navíc pokud byl ve vzoru zapojen na vstup EN_MAN signál *I_H52_Y14_rucni*, generátor dokáže tento signál přejmenovat na *I_L1_V31_rucni* a pokud tento název najde v tabulce symbolů, přiřadí ho na vstup EN_MAN bloku L1_V31.

Použití vzoru má velkou výhodu, pokud je v projektu velké množství téměř stejných prvků. Programátor tak může 1 tento prvek naprogramovat a například další desítky až stovky se stejným zapojením vytvořit díky generátoru.



Obrázek 5.4 - ukázka kódu pro zapojení vzoru

5.5 List Faceplate IO

Na tomto listu je pouze seznam všech vstupů a výstupů jednotlivých funkčních bloků objektů faceplate. Seznam názvů vstupů a výstupů byl potřebný k tomu, aby se tyto názvy mohly porovnávat s názvy symbolů na listu Tabulka symbolů. Během generace kódu se tak ke každému volanému faceplate bloku přiřadí i patřičné názvy vstupů a výstupů a díky tomu může dojít k určení, zda jsou v tabulce symbolů vhodné názvy k připojení signálů.

6 APLIKACE GENERÁTORU TAGŮ

Nástroj firmy COMPAS Automatizace, spol. s.r.o. pomáhá při tvorbě automatizačních projektů, ve kterých se používají objekty z knihovny faceplate, která byla vymyšlena právě firmou COMPAS Automatizace, spol. s.r.o. Generátor tagů funguje tak, že z vloženého zdrojového souboru, který je napsán v programovacím jazyce STL (jedná se tak o zdrojový soubor .awl) dokáže vytvořit v prostředí Microsoft Excel tabulkový mezisoubor, se kterým se pak dále pracuje.

Generátor PLC kódu pro bloky faceplate, nástroj, který je cílem této bakalářské práce, pak má za cíl vytvořit kostru automatizačního projektu a tím podstatně urychlit tvorbu samotného projektu. Generátor PLC kódu pracuje na tom principu, že data zadaná do projekční tabulky převede na zdrojový soubor programovacího jazyka STL (soubor .awl) a ten se následně vloží do programového prostředí TIA Portal. Tím je položen základ celému projektu, programátor pak již musí zbytek cílového programu dodělat ručně.

Právě v okamžik, kdy se dodělá celý projekt, který byl podstatnou měrou vytvořen na základě dat vytvořených pomocí generátoru PLC kódu pro bloky faceplate, programátor ho převede do jednoho komplexního .awl souboru. Tento soubor se pak použije jako vstupní zdrojový soubor pro nástroj generátor tagů. Po vytvoření mezisouboru v generátoru tagů (tabulka programu MS Excel) se pak pomocí údajů právě v této tabulce může vytvořit základ vizualizace projektu v programu WinCC.

Jak pro stávající nástroj firmy COMPAS Automatizace, spol. s.r.o. generátor tagů, tak pro nástroj vytvořený v této bakalářské práci generátor PLC kódu pro bloky faceplate, je velice důležitou vlastností schopnost importu tabulky symbolů. Pro obě tyto aplikace je práce s tabulkou symbolů velmi důležitá, oba nástroje také musí mít importovanou buď stejnou, nebo pokud možno co nejaktuálnější verzi tabulky symbolů (pokud se například některé symboly přidávaly až v průběhu tvorby programu).

7 TEST GENERÁTORU

V této kapitole popíšu výsledky testování kódu vytvořeného na základě údajů, které jsem vložil do projekční tabulky. Pro tvorbu kódu jsem do tabulky zadával takové údaje, abych otestoval pokud možno všechny funkce generátoru a také abych vyzkoušel, jak si poradí s různými případy zadaných dat. Proto jsem například v projekční tabulce vytvořil několik technologických skupin, kdy jsem nedával všechny prvky jedné technologické skupiny pod sebe, ale skupiny vytvářel střídavě (příklad: první prvek je z technologické skupiny L1, druhý ze skupiny L2, třetí opět ze skupiny L1, čtvrtý ze skupiny L111 apod.). Dále jsem vytvořil takové názvy skupin, abych otestoval, zda generátor opravdu pracuje jen s celými názvy skupin nebo mu stačí jen část názvu (příklad: mám skupinu L1 a chci se ujistit, že například se skupinou L111, která v sobě obsahuje i název první skupiny L1, nebudou při generování kódu žádné problémy).

Další věcí byla zkouška importování tabulky symbolů a zdrojového .awl souboru, který slouží jako vzor pro tvorbu několika prvků. V tabulce symbolů se vyskytují pouze některé správné symbolické názvy, otestuje se tak správné a nesprávné přiřazování signálů na vstupy a výstupy. Ve zdrojovém souboru pro generaci kódu podle vzoru jsem vyzkoušel variantu s tím, že jsem v TIA Portalu vytvořil dvě vzorové zapojení různých prvků a z kódu obou dvou zapojení jsem vytvořil jeden vzorový .awl zdrojový soubor. Tím jsem odzkoušel, zda je nutné mít v daný okamžik pouze jeden vzor, podle kterého tvořit kód, nebo jich můžu mít libovolné množství.

Po této přípravě jsem nechal vygenerovat zdrojový kód, uložil ho jako .awl soubor a následně vložil do TIA Portalu, kde jsem z něj nechal vygenerovat bloky. Ke každé řadě testovaného PLC jsem ještě vložil .awl zdrojový soubor, který v sobě měl kód pro vytvoření potřebných funkčních bloků pro všechny objekty faceplate (FBHMI_AI, FBHMI_AQM, …). Tento zdrojový soubor mi byl poskytnut firmou COMPAS Automatizace, spol. s.r.o. Vytvořené bloky jsem pak potřeboval k otestování správné funkčnosti volání jednotlivých funkčních bloků v kódu vytvořeném pomocí generátoru. Do každého projektu s PLC jsem také vložil stejnou tabulku symbolů, aby mohlo správně proběhnout přiřazení některých signálů na vstupy nebo výstupy funkčních bloků.

7.1 PLC S7-300/400

PLC řady Simatic S7-300 a S7-400 jsou, alespoň co se týče programové stránky, téměř totožná. Program jsem však zkoušel vložit a zkompilovat pro obě tato PLC, výsledek však byl stejný. Vzhledem k tomu, že práce v TIA Portalu s PLC řady S7-300 a S7-400 umožňuje nejlepší možnosti provádění různých změn (změna programovacích jazyků), rozhodl jsem se celý zdrojový kód, vytvořený generátorem, přizpůsobit právě na PLC těchto dvou řad. To se projevilo hlavně tím, že datový přístup k blokům je v kódu generovaném pomocí generátoru nastaven jako standardní, nikoliv jako optimalizovaný (S7_Optimized_Access := False), neboť ten není možné pro PLC řady S7-300 a S7-400

použít. Rozdíl mezi standardním přístupem a optimalizovaným je ten, že bloky se standardním přístupem mají absolutní přístup ke všem adresám a proměnným, kdežto optimalizovaný přístup pracuje se symbolickými názvy. Standardní přístup tak nabízí větší flexibilitu v přístupových mechanismech, kdežto optimalizovaný přístup je zase bezpečnější a vzhledem k optimalizaci bloků pro dané PLC odkáže dosahovat mnohem větších rychlostí.

Po importu zdrojového souboru do TIA Portal a následném vytvoření programových bloků na základě tohoto kódu se musely všechny takto vytvořené bloky zkompilovat. Důvodem je to, že bloky vytvořené pouze na základě externího zdrojového souboru hlásily chybu při volání dalších takto vytvořených bloků (příklad: ve vytvořených bloku FCL1 je volání dalšího vytvořeného bloku FCL1_IN, ale před kompilací program neregistruje, že blok FCL1_IN byl vytvořen a proto zahlásí chybu, po kompilaci už je vše v pořádku). Po této dodatečné kompilaci už bylo blokové volání v pořádku, nebyl ani problém s voláním v programu již existujících funkčních bloků jednotlivých faceplate objektů. Ke každému volanému funkčnímu bloku se navíc vytvořil i instanční datový blok. Vzhledem k tomu, že kód byl psaný primárně pro odzkoušení na PLC řady S7-300 a S7-400, existuje zde dokonce možnost přepnout z textového programovacího jazyka STL (AWL) do grafického jazyka LAD (viz **Programovací jazyky STL, LAD, FBD**).

Jediný problém byl s vytvořením hlavního programového bloku Main (OB1). Blok se sice ze zdrojového souboru vytvořil, je u něj však ten problém s kompilací. Blok Main (OB1) v tomto případě vytvoří chybovou hlášku s tím, že jeho interface musí být větší než 20 bytů, čehož kvůli generování tohoto bloku z externího zdrojového souboru není dosaženo. Zde se nabízí pouze jediné řešení, a to zkopírovat obsah tohoto bloku (buďto do jiného prázdného bloku nebo do jakéhokoliv textového editoru), blok smazat a ručně vytvořit blok Main (OB1) nový. Do takto vytvořeného prázdného bloku pak vložíme kód z předchozího nefunkčního bloku, zkompilujeme a vše již funguje tak, jak má.

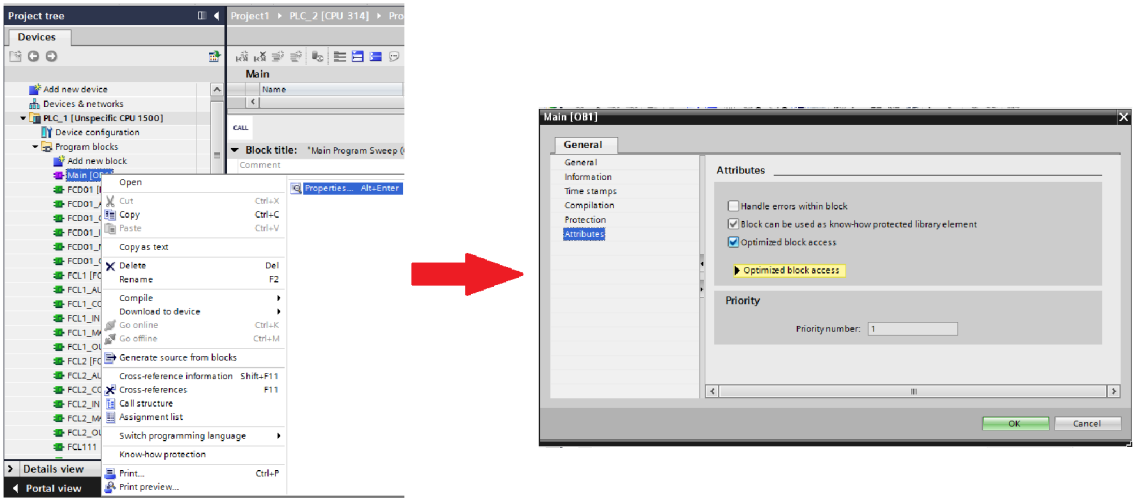
7.2 PLC S7-1500

Hlavním rozdílem mezi PLC Simatic S7-1500 a S7-300/400 je z programovací stránky to, že v PLC S7-1500 nemáme možnost měnit programovací jazyk. Pokud tedy vytvoříme jakýkoliv blok například v grafickém jazyku LAD, později již tuto volbu nemůžeme změnit. Na druhou stranu PLC této řady umožňují optimalizovaný přístup k programovým blokům. V generátoru je však nastavena tvorba bloků se standardním přístupem, tyto bloky ale můžeme v PLC S7-1500 normálně používat. Do budoucna by bylo možné do generátoru přidat možnost volby optimalizovaného nebo standardního přístupu v závislosti na tom, jaké PLC bude v projektu použito.

Stejně jako při použití PLC řady S7-300/400, i zde je nutné po vygenerování programových bloků z externího zdrojového souboru všechny tyto bloky znovu zkompilovat, abychom se vyhnuli chybám vzniklým kvůli volání jednotlivých bloků.

Při vytvoření hlavního programového bloku Main (OB1) však nastane stejný problém, jako nastal u PLC řady S7-300/400. Situaci můžeme vyřešit stejně jako v předchozím

případě ručním vytvořením tohoto bloku a následným zkopírováním obsahu. Je zde však i druhá možnost. Jelikož PLC řady S7-1500 umožňují u bloků nastavit optimalizovaný přístup i po jejich vytvoření, můžeme tak učinit právě u bloku Main (OB1). Nastavení provedeme tak, klikneme pravým tlačítkem na blok v sekci *Program blocks*, tím se nám otevře menu s různými možnostmi. Klikneme na záložku *Properties*, zde v levém sloupci otevřeme okno *Attributes* a zaškrtneme položku *Optimized block access*. Po následné kompilaci již blok Main (OB1) funguje správně. Problémem tohoto řešení je, že se nedoporučuje mít v projektu bloky s různým druhem přístupu. Možností by bylo nastavit optimalizovaný přístup u všech vytvořených bloků, bylo by to však časově náročné, protože přístup by se musel nastavovat u každého bloku zvlášť, není zde možnost hromadné změny vlastností. Řešením by bylo v budoucí verzi generátoru dát programátorovi možnost vybrat si typ přístupu (viz výše).



Obrázek 7.1 - nastavení optimalizovaného přístupu v bloku Main (OB1)

7.3 PLC S7-1200

PLC Simatic řady S7-1200 nemají žádnou možnost programování v jazyce STL (zdrojový soubor .awl). Z tohoto důvodu nebyla možná implementace zdrojového kódu vytvořeného pomocí tohoto generátoru. Jedinou možnou alternativou by bylo v některé budoucí verzi generátor upravit, že by si programátor mohl zvolit, zda chce vygenerovat kód v programovacím jazyce STL, tudíž tak, jak generátor funguje v této verzi, nebo jestli chce vygenerovat zdrojový soubor v jazyce SCL. S programovacím jazykem SCL totiž dokáže PLC řady S7-1200 pracovat normálně. Ačkoliv jsou oba tyto programovací jazyky textového charakteru, je mezi nimi hodně velký rozdíl, hlavně ve struktuře vytvořeného kódu. Pokud by tedy v gene átoru v budoucnosti přibyla možnost generovat zdrojové soubory typu .scl, generátor pro tuto variantu by se musel naprogramovat v podstatě celý od začátku znovu. Navíc zdrojový soubor, který obsahuje kód k funkčním blokům pro objekty faceplate dodaný firmou COMPAS Automatizace, spol. s.r.o., je také

napsán v programovacím jazyku STL (jedná se o .awl zdrojový soubor), pokud firma nemá stejný soubor v programovacím jazyku SCL, museli by ho pravděpodobně vytvořit úplně celý od základu.

Dalším problémem s generováním .scl zdrojového souboru pomocí generátoru by byla jeho následná implementace v nástroji firmy COMPAS Automatizace, spol. s.r.o. Generátor tagů. Ten totiž potřebuje jako vstupní soubor zdrojový soubor .awl, aby z něj mohl později vygenerovat příslušné věci.

8 ZÁVĚR

Hlavním cílem této bakalářské práce bylo vytvoření nástroje, který by sloužil k usnadnění prvotní fáze při vytváření automatizačních projektů v programovacím prostředí TIA Portal V13. Jelikož právě tyto počáteční fáze vývoje mohou obsahovat zdlouhavé a repetitivní kroky, které jsou však v základu poměrně jednoduché, generátor PLC kódu pro bloky faceplate by měl celý proces podstatně urychlit.

Součástí programu je poměrně přehledné a jednoduché ovládací rozhraní. Programátor, který bude generátor používat, tak může vcelku snadno do generátoru naimportovat veškeré potřebné soubory, ať už se jedná o tabulku symbolů využívaných v projektu, nebo zdrojový soubor se vzorovým zapojením funkčního bloku objektu faceplate. Pak již stačí vyplnit jednoduchou projekční tabulku a generátor sám vytvoří zdrojový kód, který je pak snadné importovat do programovacího prostředí TIA Portal. Výhodou programu generátoru je to, že k jeho fungování není potřeba jakákoliv instalace dodatečných programů. Veškerá generace kódu probíhá pomocí skriptů a maker VBA, která jsou běžnou součástí produktů Microsoft Office. Programátorovi tak stačí mít nainstalován pouze program Microsoft Excel.

Nevýhodou generátoru PLC kódu v jeho stávající podobě je to, že nedokáže generovat zdrojový kód pro PLC Siemens Simatic řady S7-1200. Jelikož tato konkrétní řada PLC nepodporuje práci v programovacím jazyku STL, na kterém je celý generátor založen, v některé budoucí verzi by se musela přidat možnost výběru, zda má být výsledný zdrojový kód generovaný v programovacím jazyce STL nebo v programovacím jazyce SCL, který je vhodný i pro PLC S7-1200. Dalším problémem je omezená možnost programového přiřazování symbolů ze symbolické tabulky na vstupy a výstupy volaného funkčního bloku. Tento problém vychází z toho, že ne všechny symboly, které je potřeba na daný vstup nebo výstup připojit, mají ustálené značení, a proto je z programátorského hlediska obtížné najít jakýkoliv vzorec, podle kterého by se symboly připojovaly. Podobný problém by mohl nastat také při vytváření kódu na základě vzorového zdrojového souboru. Pokud by byly napojeny symboly, které neobsahují název technologické skupiny nebo projekčního značení, generátor ve stávající verzi nezkopíruje tato zapojení do výsledného zdrojového souboru.

Výsledky testování generátoru pro zbylé PLC, konkrétně řady S7-300/400 a S7-1500 pak proběhly úspěšně (viz **Test generátoru**). Na závěr by se tak dalo říct, že pokud je projekt plánovaný pro PLC jedné z těchto řad a je tak velkého rozsahu, že ruční programování základní kostry programu by trvalo podstatně delší dobu, než jen napsat potřebná data do projekční tabulky a následně vygenerovat ze zdrojového souboru programové bloky, použití generátoru PLC kódu pro bloky faceplate může být přínosné.

Literatura

[1] SIEMENS, s.r.o.: *Configuration Instruction for Generating Faceplates* [online]. ©2013, první vydání 2013-03-14 [cit. 2014-11-29]. Dostupné z http://support.automation.siemens.com/WW/llisapi.dll?func=cslib.csinfo&lang=en&objid=68014632&caller=view

[2] SIEMENS, s.r.o.: *STEP 7 Professional V13.0, System Manual* [online]. ©2014, aktualizováno 2014-03-10 [cit. 2014-12-13]. Dostupné z http://support.automation.siemens.com/WW/llisapi.dll?func=cslib.csinfo&lang=en&objid=89515142&caller=view

[3] Firemní dokumentace. PROKOP, Antonín. COMPAS Automatizace, spol. s.r.o., 2011.

[4] Excel Easy. Excel Easy. [online]. © 2010-2015 [cit. 2015-04-18]. Dostupné z: http://www.excel-easy.com/

[5] Ken Carney. Excel VBA Programming. Home and Learn. [online]. [2013] [cit. 2015-04-03]. Dostupné z: http://www.homeandlearn.org/

Seznam příloh

Příloha 1. DVD-ROM

Příloha 1

- Elektronická verze bakalářské práce ve formátu PDF
- Nástroj Generátor kódu PLC pro bloky faceplate ve formě sešitu Microsoft Excel s podporou maker .xlsm
- Verze nástroje s předvyplněnými daty jako zkušební příklad