České vysoké učení technické v Praze
Fakulta elektrotechnická

BAKALÁŘSKÁ PRÁCE

Sběr a sledování provozních dat

Praha, 2011 **Autor: Tomáš Kletečka**

Prohlášení

Prohlašuji, že jsem svou diplomovou (bakalářskou) práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne _____ _____

podpis

Poděkování

Děkuji především vedoucímu bakalářké práce ing. Pavlu Burgetovi, Ph.D. a kolegům z firmy Schneider Electric CZ s.r.o. za odborné vedení při realizaci této bakalářské práce. Děkuji firmě Schneider Electric CZ s.r.o. za poskytnutí prostředků nezbytných k vypracování této práce.

Abstrakt

Tato práce se zabývá návrhem a implementací systému pro sběr a analýzu provozních dat programovatelných automatů firmy Schneider Electric. Data jsou zpracovávána a uchovávána v PLC.Výsledkem je knihovna funkčních bloků pro vývojové prostředí CoDeSys V3, která usnadní programování následné konkrétní aplikace pro sběr a uchování dat v programovatelných automatech. Uživatel má možnost stahovat data do prostředí Microsoft Excel přes OPC server. Sledovaná data se zobrazují do přehledných tabulek. Z excelovského rozhraní lze modifikovat data uložená v registreh PLC a tím provádět nastavení bez potřeby programovacího prostředí a zdrojového kódu programu běžícího v automatu.

Abstract

The attached bachelor thesis deals with design and implementation of system for monitoring of process data of programmable controllers from Schneider Electric. Data are processed and stored in PLC. The result of this thesis is function block library for CoDeSys V3, which facilitates incoming programming of concrete application of process data monitoring. User is able to download data to Microsoft Excel enviroment over OPC server. Downloaded data are shown in transparent tables. It is possible to edit data stored in PLC from Microsoft Excel interface. It makes it possible to perform PLC settings without needs of programming enviroment and knowledge in program running in a controller.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Nedílnou součástí řídicích systémů je monitorování jejich stavu. Uživatel často požaduje statistické informace jako je počet motohodin, spotřebovaná energie nebo počet pracovních cyklů. Dále vyvstává potřeba archivace událostí, zejména chyb, vzniklých při provozu. Tato práce si klade za cíl nalézt pokud možno univerzální řešení pro sběr provozních dat a pro jejich předání uživateli v přehledné formě, pomocí běžně dostupných prostředků (např. pomocí aplikace Microsoft Excel).

Jednotlivá konkrétní řešení pro monitoring průmyslových zařízení jsou natolik specifická, že nelze vytvořit naprosto univerzální řešení pro všechny aplikace. Univerzalita zde popsaného řešení spočívá v usnadnění tvorby konkrétních aplikací definováním funkcí, které se často používají a aplikují. Následná konkrétní realizace již nemusí začít od definice základních často používaných funkcí, ale použitím zde popsané knihovny funkčních bloků. Další výhodou vytvoření jedné speciální knihovny pro monitoring, je snadné vyhledávání funkčních bloků. Uživatel nemusí hledat potřebné funkce či funkční bloky ve velkém množství knihoven, vše najde přehledně na jednom místě - v knihovně Monitoring.library.

Díky efektivnímu využití monitoringu lze šetřit náklady na provoz zařízení, minimalizovat dobu a náklady potřebné na udržbu či k případnému servisu. Snadné stažení a analýza sledovaných dat usnadní a urychlí diagnostiku vzniklých poruch. Dojde tak ke zkrácení doby potřebné k uvedení stroje opět do provozu.

Kapitola 2

Cíle

2.1 Popis problému

Práce se skládá ze dvou hlavních částí. První část zahrnuje vytvoření knihovny univerzálních funkčních bloků sloužících pro sběr a uchování dat v PLC. Tato část bude vytvořena v programovacím prostředí SoMachine. Jako produkt druhé části vznikne aplikace určená pro stažení, zobrazení a uložení dat z PLC do PC. Tato část bude zpracována v prostředí Microsoft Excel za použití VBA for Excel.

Data, která jsou určena k uchování či dalšímu zpracování jsou dvou typů - statistická data a události. Mezi statistická data patří například počet motohodin, počet pracovních cyklů, průměrný počet pracovních cyklů za hodinu apod. Data, výše nazvaná jako události, popisují stav systému v okamžiku, kdy dojde k nějaké definované situaci. Tato situace nemusí být vždy chyba, ale i jiná událost jako spuštění systému, dokončení pracovního cyklu, dosažení určitého počtu cyklů apod.

Použití statistik přináší tyto výhody:

- Alarmy - možnost upozornění při přiblížení ke kritickému stavu sledované veličiny. Například po uplynutí určitého počtu motohodin je uživatel upozorněn na potřebu pravidelného servisu.
- Uživatel získává přehled o vytížení a využití systému, který mu může posloužit k optimalizaci pracovního procesu.

Sledování událostí přináší značné usnadnění diagnostiky. Servisní technik získá přehled stavů systému, které usnadní diagnostiku a značně urychlí uvedení do provozu a zjednoduší odstranění případných problémů.

2.2 Požadavky na řešení

Na řešení jsou kladeny tyto požadavky:

- Nalezení vhodné struktury pro ukládání dat registrech PLC.
- Snadná modifikovatelnost monitorovacího programu v PLC i uživatelského rozhraní v MS Excel podle konkrétního řešení.
- Univerzální použití knihovny funkčních bloků na všech řídicích systémech určených pro platformu SoMachine - Modicon M238, Modicon M258, Modicon LMC058, Altivar - IMC.
- Uložená data jsou zachována i po restartu řídicího systému.
- Možnost zápisu do registrů v PLC z uživatelského rozhraní za účelem parametrizace zažízení.
- Data stažená z PLC jsou přehledně zobrazena v tabulkách.
- Intuitivní ovládání uživatelského rozhraní.
- Vytvoření dokumentace knihovny funkčních bloků a manuálu k jejímu použití spolu s uživatelským rozhraním.

Kapitola 3

Softwarové prostředky

3.1 CoDeSys

CoDeSys (Controller Development System) je obecný, na platformě nezávislý, vývojový nástroj pro průmyslovou automatizaci, který vyvinula firma 3S - Smart Software Solutions GmbH. V základu se skládá ze dvou částí: programovacího systému CoDeSys a runtime systému CoDeSys runtime.

Plní tyto hlavní úlohy:

- tvorba aplikací pro průmyslovou automatizaci,
- nahrávání a zpracovávání aplikací,
- ladění aplikací v libovolném jazyce dle IEC,
- komunikace s programovacím systémem,
- směřování zpráv v síti řídicího systému,
- obsluha vstupně/výstupních systému.

CoDeSys umožňuje programování ve všech pěti jazycích dle normy IEC 61131 - 3, jimiž jsou: instrukční list (IL), diagram funkčních bloků (FBD), strukturovaný text (ST), sekvenční funkční diagram (SFC) a žebříkový diagram (ladder diagram), označovaný také jako releové schéma.

Více než 250 OEM zákazníků a tisíce koncových uživatelů činí z CoDeSysu nejrozšířenější vývojový nástroj splňující normu IEC 61131 - 3. Stal se standardem v programování řídicích systému a PLC. [12]

3.2 SoMachine

SoMachine je sofwarový nástroj od firmy Schneider Electric určený pro návrh, programování, oživování a provozní správu strojů (PLC, HMI, pohony, frekvenční měniče). Vývojové prostředí SoMachine je založeno na výše popsaném CoDeSys V3. Vývojové prostředí SoMachine přináší oproti samotnému CoDeSysu mnohá rozšíření zejména z hlediska uživatelského pohodlí. SoMachine přináší například tyto výhody:

- rozšíření knihoven funkčních bloků,
- specializované OEM aplikační knihovny,
- snadné integrování externích zařízení,
- soubor funkcí pro monitorování a ladění stroje,
- možnost programovar PLC i HMI v jediné aplikaci.

V současné době přichází na trh nová verze SoMachine V3. Tato práce, jejíž součástí je aplikační knihovna pro sledování a zpracovávání provozních dat (Monitoring.library), je vypracována v SoMachine V2 RL3. Knihovna byla vytvořena ve starší verzi softwaru z toho důvodu, že v době započetí práce byl SoMachine V3 teprve ve vývoji.

3.3 OPC

OPC (OLE for process control) je soubor specifikací norem pro komunikaci s real-time řídicími systémy různých výrobců. První norma vznikla za spolupráce předních světových dodavatelů automatizace a společnosti Microsoft. Původně byla založena na technologiích OLE COM (component object model) a DCOM (distributed component object model) od Microsoftu. Specifikace definovala soubor objektů, rozhraní a metod pro usnadnění interoperability v aplikacích průmyslové automatizace a procesního řízení.

Nejoblíbenější analogií pro vysvětlení potřeby originální specifikace pro přístup k datům jsou ovladače tiskáren v MS DOS a v MS Windows. Pod MS DOS museli vývojáři napsat pro každou aplikaci ovladače pro všechny tiskárny. Museli napsat zvlášť každý ovladač pro tiskárnu, kterou chtěli podporovat. Obdobně tomu bylo ve světě průmyslové automatizace, kde firma vyrábějící HMI musela ke každému svému produktu napsat ještě příslušný ovladač ke každému průmyslovému zařízení (zahrnující všechny výrobce PLC).

Windows vyřešil problém s tiskárnami začleněním podpory tiskáren do operačního systému. Nyní jeden ovladač k dané tiskárně obslouží všechny příslušné aplikace. Tyto ovladače byly napsány výrobci tiskáren, nikoli již vývojáři aplikací. Windows poskytl infrastrukturu, která umožňuje řešit problém s ovladači pro průmyslová zařízení stejně dobře jako v případě tiskáren. Přidáním specifikace OPC do technologie OLE v Microsoft Windows byla umožněna standardizace. Nyní výrobci průmyslových zařízení mohou napsat OPC DA servery a software (jako HMI) se stane OPC klientem.[11]

3.4 OFS - OPC Factory Server

OFS je produkt firmy Schneider Electric, který využívá výše zmíněný standard OPC. OFS vytvoří virtuální datový server, který naváže komunikaci s řídicími systémy Schneider Electric a následně předáva požadavaná data OPC klientům. OFS server poskytuje rozhraní mezi řídicím systémem a jednou nebo více klientských aplikací. Tyto aplikace slouží např. k zobrazení a editaci datových hodnot v cílovém zařízení, příp. k archivaci procesních dat. Schéma komunikace mezi PLC a klientskou aplikací popisuje obr. 3.1.

OFS je kompatibilní s verzemi OPC 1.0 a OPC 2.0.[5]

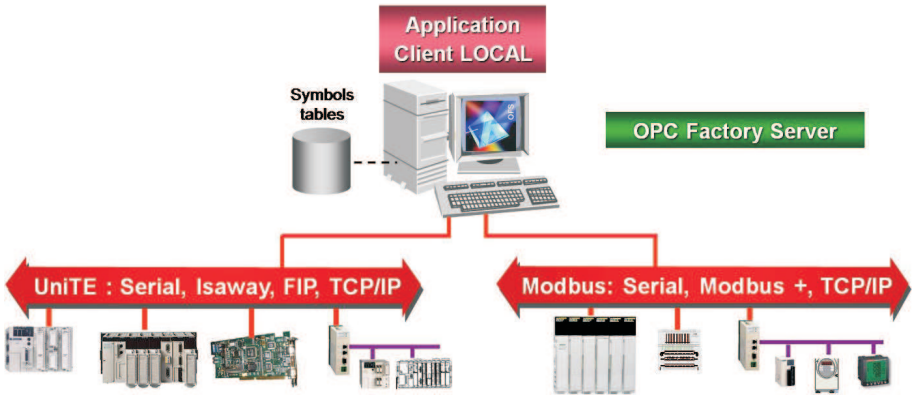
Roli OPC klienta může zastávat:

- Originálně dodávaný software, kde OFS hraje roli ovladače, který zaručuje komunikaci se všemi podporavanými zařízeními Schneider Electric.
- Vlastní vyvinutý software využívající buď rozhraní OLE Automation nebo OLE Custom.

Pro vytvoření vlastní klientské aplikace je nezbytná znalost jednoho z následujících jazyků:

- Microsoft Basic, verze 6.0 SP3 nebo vyšší,
- Microsoft Visual C++, verze 6.0 SP3 nebo vyšší,
- Microsoft VBA in Excel, verze 8.0 (Office 97) nebo vyšší,
- Microsoft Visual C#.

V této práci bylo k vytvoření klientské aplikace využito prostředí Microsoft Excel spolu s jazykem Microsoft VBA in Excel.



Obrázek 3.1: Schéma OPC komunikace mezi klientem, serverem a PLC

Poznámka: Obrázek obr. 3.1 byl přebrán z [5].

3.5 VBA

Visual Basic (VB) a Visual Basic for Applications (VBA) mají společný základ. Lze říci, že jádro jazyka Visual Basic 6.0 je součástí instalace Microsoft Office od verze 9.0 (komerční název Office 2000) a komunikace s ním probíhá přes moduly kódu a uživatelské formuláře, které se od formulářů Visual Basic liší určitými omezeními, avšak pro většinu aplikací postačí.

Pokud mluvíme o VBA, nemluvíme o Excel VBA, Word VBA nebo Access VBA. Syntaxe jazyka je pro všechny aplikace stejná, liší se pouze její objektový model, se kterým pracujeme. [1]

Kapitola 4

Způsob uchování dat v PLC

4.1 Datové typy

Stejně jako při každém programování i v SoMachine, potažmo CoDeSys V3, jsou k dispozici různé datové typy proměnných, které lze rozdělit do tří základních kategorií - celočíselné, reálné (s plovoucí řádovou čárkou) a logické (booleovské). Pro efektivní využití strojového času a paměti je zapotřebí využívat správné datové typy. Popis některých datových typů, které jsou využívány v knihovně Monitoring, je uveden níže, viz tabulka 4.1.

Tabulka 4.1: Datové typy

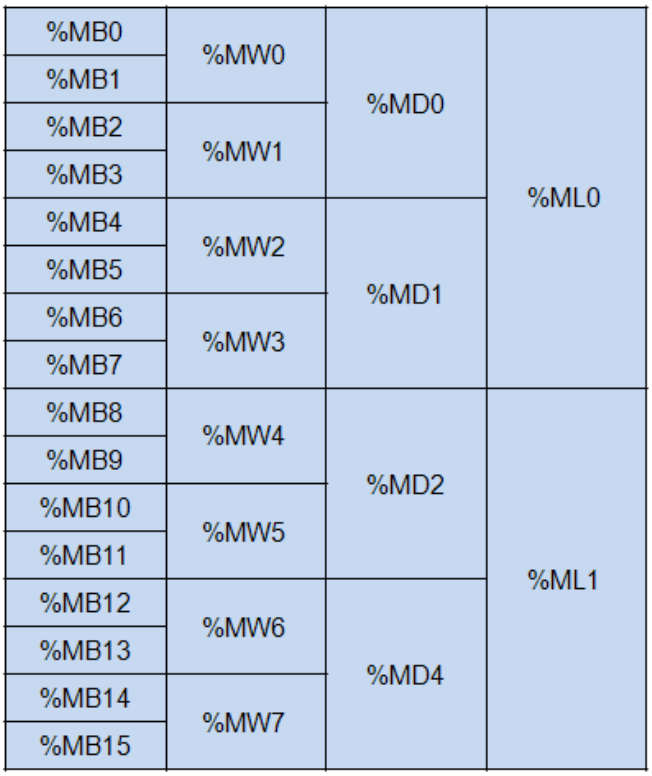
Typ	**Dolní limit**	**Horní limit**	**Velikost**
BYTE	0	255	8 bit
WORD	0	65535	16 bit
DWORD	0	4294967295	32 bit
INT	-32768	32767	16 bit
DINT	-2147483648	2147483647	32 bit
REAL	-1.175494351e-38	3.402823466e+38	32 bit
BOOL	0	1	1 bit

4.2 Paměťová struktura PLC

K určitému paměťovému místu lze přistupovat různými způsoby, které se liší velikostí adresovaného prostoru - 8 bitů (byte), 16bitů (word), 32 bitů (double word), případně 64 bitů (long word). Každá adresa se v souladu s normou IEC[1] skládá z prefixu %M, písmene značící velikost paměťového místa(B,W,D,L) a pořadového čísla. Například padesátý word v paměti se adresuje jako %MW49.

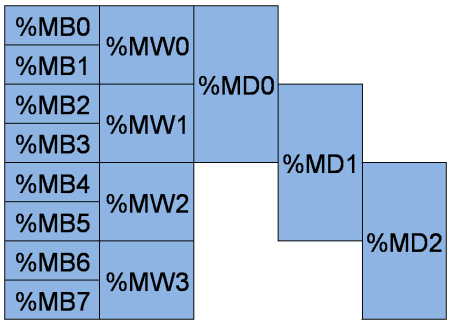
Způsob adresování vnitřní paměti PLC může být různý. Dokonce i někteří výrobci mohou u různých zařízení používat odlišné způsoby adresace paměťového místa. Pro případ automatů rodiny SoMachine firmy Schneider Electric je způsob uložení dat znázorněn na obr. 4.1.

Již při stahování dat z PLC pomocí OPC serveru nastane komplikace způsobená rozdílnou adresací. OPC Factory server používá jiné adresování než PLC.



Obrázek 4.1: Paměťová struktura - PLC pro SoMachine

[1]International Electrotechnical Commission



Obrázek 4.2: Paměťová struktura - OPC

Organizace paměti OFS je uvedena na obrázku 4.2. Rozdíl oproti PLC je v adresování 32bitových proměnných (double word). Např. adrese %MD50 v PLC odpovídá na straně OFS adresa %MD100, tedy dvojnásobně vysoké číslo. Při psaní funkcí ve VBA, které slouží pro čtení z PLC, bylo tedy zapotřebí všechny 32bitové adresy, definované v PLC, vynásobit dvěma.

4.3 Proměnné typu retain a persistent

Každé proměnné lze přiřadit atribut retain nebo retain persistent. Tyto proměnné se poté uloží na speciální místo paměti, které je zálohované baterií pro případ výpadku napájení, a získají vlastnosti popsané níže.

Retain

- Zachovává hodnotu po normálním vypnutí a zapnutí PLC.
- Zachovává hodnotu i po nekontrolovaném vypnutí PLC.
- Zachovává hodnotu po provedení příkazu Reset warm application.

Retain persistent

- Zachovává hodnotu po nahrání programu.
- Pokud dojde při přehrání programu k nekonzistenci proměnné, je uživatel vyzván k provedení inicializace.
- Zachovává hodnotu po provedení příkazu Reset warm application.
- Zachovává hodnotu po provedení příkazu Reset cold application.

Pokud je některá lokální proměnná funkčního bloku deklarována jako VAR RETAIN, celá instance funkčního bloku je uložena v retain oblasti.

Protože sledovaná data musí být robustně uložena a přečkat restart systému, jsou všechny důležité proměnné funkčních bloků knihovny Monitoring deklarovány jako VAR RETAIN.

4.4 Struktura pro ukládání událostí

Pro usnadnění manipulace s parametry jednotlivých událostí, byla vytvořena struktura strRecData, která obsahuje parametry 1 až 9 v řadě za sebou. Prvních šest proměnných ve struktuře je typu WORD a další tři typu INT. Každá událost tedy bude vždy shrnuta do jedné struktury o devíti prvcích.

Každému záznamu vzniklé události odpovídá jedna časová značka, která se ukládá do proměnné typu DATE_AND_TIME (DT), která vyjadřuje čas ve vteřinách od 1.1.1970. Typ DATE_AND_TIME má velikost 32 bitů.

Protože počet uchovávaných událostí byl stanoven na dvacet, je výhodné vytvořit pole struktur strRecData a pole časových značek o dvaceti prvcích, což značně ulehčí cyklický přístup k datům.

4.5 Adresy určené k následnému stahování dat

V souladu s následným stahováním dat z PLC je nutné data ukládat na předem specifikovaná místa. V tomto řešení se počítá se třemi bloky pro záznam událostí, šedesáti statistickými proměnnými a jednou proměnnou pro nastavování systémového času PLC.

V prostředí SoMachine se proměnné typu pole struktur nebo pole proměnných typu double word musí adresovat jako %ML. Při načítání prvků těchto polí se ale přistupuje pomocí %MD nebo %MW v souvislosti na tom, zda čteme parametr události, který má velikost 16 bitů, nebo časovou značku, která je typu DT (32 bitů).

Tabulka 4.2 zobrazuje paměťová místa, na která je nutné lokalizovat proměnné určené k následnému stahování. Ke každé proměnné je uveden název listu v excelovské aplikaci, v němž se stažená data zobrazí. Pro lepší představu o rozmístění proměnných je na obrázku 4.3 uvedena mapa paměti PLC.

Při stahování dat do prostředí MS Excel je k proměnným přistupováno jednotlivě. Jsou vybírány jednotlivé prvky polí dle jejich adresy. Všechny načítané hodnoty, kromě časových položek typu DT, jsou kvalifikovány jako 16bitové číslo (word). Položky polí, která obsahují časové hodnoty, jsou načítány jako 32bitov8 čísla (double word). Tato čísla uvádí počet vteřin od 1. 1. 1970.

Tabulka 4.2: Adresy určené ke stahování dat

Místo v paměti	**Typ**	**List MS Excel**
%ML250	ARRAY [1..20] of strRecData	Event1
%ML295	ARRAY [1..20] of DT	Event1
%ML305	ARRAY [1..20] of strRecData	Event2
%ML350	ARRAY [1..20] of DT	Event2
%ML360	ARRAY [1..20] of strRecData	Event3
%ML405	ARRAY [1..20] of DT	Event3
%MW1720 - %MW1739	WORD nebo INT	Statistics1
%MW1740 - %MW1759	WORD nebo INT	Statistics2
%MW1760 - %MW1779	WORD nebo INT	Statistics3
%MW992	DWORD	Slouží k resetování bloků příkazem z PC.
%MD993	DWORD	Slouží k resetování bloků příkazem z PC.
%MD994	DWORD	Slouží k resetování bloků příkazem z PC.
%MD995	DWORD	Slouží k nastavení nového času PLC z PC.
%MD998	DWORD	Povoluje zobrazení listů Events1 a Statistics1.
%MD999	DWORD	Povoluje zobrazení listů Events2 a Statistics2.
%MD498	DWORD	Obsahuje novou hodnotu pro nastaveni času PLC.

%ML248	%MW992	%MW993	%MW994	%MW995	
%ML249	%MW996	%MW997	%MW998	%MW999	
%ML250	%MW1000	%MW1001	%MW1002	%MW1003	Events1
. . .	. .	. .	. .	. .	
%ML294	%MW1176	%MW1177	%MW1178	%MW1179	
%ML295	%MW1180	%MW1181	%MW1182	%MW1183	Events1 time
. . .	. .	. .	. .	. .	
%ML304	%MW1216	%MW1217	%MW1218	%MW1219	
%ML305	%MW1220	%MW1221	%MW1222	%MW1223	Events2
. . .	. .	. .	. .	. .	
%ML349	%MW1396	%MW1397	%MW1398	%MW1399	
%ML350	%MW1400	%MW1401	%MW1402	%MW1403	Events2 time
. . .	. .	. .	. .	. .	
%ML359	%MW1436	%MW1437	%MW1438	%MW1439	
%ML360	%MW1440	%MW1441	%MW1442	%MW1443	Events3
. . .	. .	. .	. .	. .	
%ML404	%MW1616	%MW1617	%MW1618	%MW1619	
%ML405	%MW1620	%MW1621	%MW1622	%MW1623	Events3 time
. . .	. .	. .	. .	. .	
%ML414	%MW1656	%MW1657	%MW1658	%MW1659	
.	. .	. .	. .	. .	
%ML430	%MW1720	%MW1721	%MW1722	%MW1723	Statistics1
%ML431	%MW1724	%MW1725	%MW1726	%MW1727	
%ML432	%MW1728	%MW1729	%MW1730	%MW1731	
%ML433	%MW1732	%MW1733	%MW1734	%MW1735	
%ML434	%MW1736	%MW1737	%MW1738	%MW1739	
%ML435	%MW1740	%MW1741	%MW1742	%MW1743	Statistics2
%ML436	%MW1744	%MW1745	%MW1746	%MW1747	
%ML437	%MW1748	%MW1749	%MW1750	%MW1751	
%ML438	%MW1752	%MW1753	%MW1754	%MW1755	
%ML439	%MW1756	%MW1757	%MW1758	%MW1759	
%ML440	%MW1760	%MW1761	%MW1762	%MW1763	Statistics3
%ML441	%MW1764	%MW1765	%MW1766	%MW1767	
%ML442	%MW1768	%MW1769	%MW1770	%MW1771	
%ML443	%MW1772	%MW1773	%MW1774	%MW1775	
%ML444	%MW1776	%MW1777	%MW1778	%MW1779	

Obrázek 4.3: Mapa paměťových míst PLC určených ke čtení nebo zápisu

Kapitola 5

Knihovna Monitoring

Jako jeden z hlavních výsledků této práce vznikla knihovna funkčních bloků pro CoDeSys V3 nazvaná Monitoring. Tato knihovna obsahuje specifikaci sedmi funkčních bloků a jedné datové struktury. Knihovna v sobě nese informaci o knihovnách použitých pro její vlastní tvorbu, po instalaci a přidání knihovny do projektu v prostředí SoMachine se tedy přidají i další závislé knihovny[1].

Při tvorbě knihovny byl kladen důraz na co největší obecnost a komplexnost. Z těchto důvodů byly do knihovny zahrnuty i funkční bloky, jako je čítač či počítání maxima a minima. Tyto funkce nejsou nikterak složité, ale jednotné uložení v jedné knihovně s ostatními bloky, stejně jako jednotné označení vstupů a výstupů, zvyšuje pohodlí uživatele a usnadňuje práci při tvorbě aplikace pro sběr a uchování provozních dat.

5.1 Popis funkčních bloků

5.1.1 EventsManager

Tento funkční blok zaznamenává posledních dvacet detekovavných událostí. Parametry systému se ukládají do vstupně-výstupní proměnné arstrcEvents. Každé položce z pole arstrcEvents přísluší položka z pole ardtTimeStamp, která udává čas a datum vzniku události.

[1] V CoDeSys označované jako Referenced libraries.

Tabulka 5.1: Popis vstupů a výstupů - funkční blok EventsManager

	Typ	**Poč. hodn.**	**Popis**
VAR_IN			
i_xEventTrig	BOOL	FALSE	Při náběžné hraně načte parametry i_wParam1 až I_wParam1 a uloží do vstupně výstupní proměné.
i_xBlockEnable	BOOL	TRUE	Uvede funkční blok do aktivity
i_wParam1	WORD		Sledovaný parametr 1.
i_wParam2	WORD		Sledovaný parametr 2.
i_wParam3	WORD		Sledovaný parametr 3.
i_wParam4	WORD		Sledovaný parametr 4.
i_wParam5	WORD		Sledovaný parametr 5.
i_wParam6	WORD		Sledovaný parametr 6.
i_iParam7	INT		Sledovaný parametr 7.
i_iParam8	INT		Sledovaný parametr 8.
i_iParam9	INT		Sledovaný parametr 9.
i_xReset	BOOL	FALSE	Při nastavení TRUE vymaže uložené události.
i_xAlarmReset	BOOL	FALSE	Při nastavení TRUE vynuluje výstupní proměnnou q_xAlarm.
VAR_OUT			
q_lastEvent	strcRecData		Parametry načtené při poslední zachycené události.
q_timeStamp	DT		Datum a čas vzniku poslední zachycené události.
q_xAlarm	BOOL	FALSE	Proměnná se nastaví po detekci události vstupem i_xEventTrig.
VAR_IN_OUT			
ardtTimeStamp	ARRAY [1..20] of DT		Pole časových značek posledních dvaceti zachycených událostí.

Tabulka pokračuje na další straně

arstrcEvents	ARRAY [1..20] of strcRec-Data		Pole uložených parametrů posledních dvaceti událostí.

Poznámky:

- Silně se doporučuje deklarovat vstupně výstupní proměnnou s parametrem retain. Retain proměnná zachovává svou hodnotu i po výpadku napájení (viz 4.3). Ostatní funkční bloky nepoužívají vstupně-výstupní proměnné, ale jen výstupní, proto jsou již výstupy samotných bloků nastaveny jako retain.
- Časová značka přiřazovaná událostem je odečítána ze systémového času PLC, proto je pro správnou funkci nezbytné správné nastavení tohoto času. Lze využít funkční blok SySTimeSetting z knihovny Monitoring.

5.1.2 Statistics

Tento funkční blok slouží k počítání statistických údajů (maximum, minimum, průměrná hodnota) vstupního signálu. Výstupní proměnné jsou typu real z toho důvodu, že při výpočtu se využívá operace dělení. Jelikož proměnné určené pro uložení statistických údajů jsou celočíselného typu word (16 bitů), doporučuje se pro potřebu následného stahování vynásobit výstup příslušným násobkem deseti a přetypovat na word. Pokud nejsou hodnoty za desetinou čárkou podstatné, násobení není nutné provádět.

5.1.3 ImpulseCounter

Tento funkční blok slouží k čítání impulsů vstupního signálu. Lze určit aktivní hranu pro čítání vstupem i_xRiseOrFall.

Tabulka 5.2: Popis vstupů a výstupů - funkční blok Statistics

	Typ	**Poč. hodn.**	**Popis**
VAR_IN			
i_rInputSignal	REAL		Vstupní signal.
i_xBlockEnable	BOOL		Uvede funkční blok do aktivity.
i_xReset	BOOL		Nastaví výstupy na počáteční hodnoty.
VAR_OUT			
q_rMin	REAL	3E+38	Minimalní hodnota vstupního signálu.
q_rMax	REAL	1E-38	Maximální hodnota vstupního signálu.
q_rAverage	REAL	0	Průměrná hodnota vstupního signálu.

Tabulka 5.3: Popis vstupů a výstupů - funkční blok ImpulseCounter

	Typ	**Poč. hodn.**	**Popis**
VAR_IN			
i_xInputSignal	BOOL		Vstupní signál, jehož impulsy jsou čítány.
i_xReset	BOOL	FALSE	Vynuluje čítač.
i_xBlockEnable	BOOL		Aktivuje funkci čítače.
i_xRiseOrFall	BOOL		Určuje aktivní hranu vstupu i_xInputSignal. TRUE...naběžná hrana, FALSE...sestupná hrana
VAR_OUT			
q_wImpulseCount	WORD		Aktuální hodnota čítače.

5.1.4 EnergyMonitor1

Tento funkční blok slouží počítání spotřebované energie na základě aktuálního výkonu.

Tabulka 5.4: Popis vstupů a výstupů - funkční blok EnergyMonitor1

	Typ	**Poč. hodn.**	**Popis**
VAR_IN			
i_wActPower	WORD		Aktuální výkon v jednotkách určených vstupem i_iEnergyUnit
i_iEnergyUnit	INT		Určení jednotek výkonu. 0...W, 1...kW, 2...MW
i_xBlockEnable	BOOL		Uvedení funkčního bloku do aktivity.
i_xReset	BOOL		Vynulování načtených hodnot.
i_xAlarmReset	BOOL	false	Vynulování výstupního alarmu.
i_wMaxPower	WORD	0	Maximalní dovolená hodnota výkonu. Při překročení vznikne alarm.
i_wEnergyLimit	WORD	0	Hranice spotřeby, při které se vystaví alarm.
VAR_OUT			
q_wUsedEnergy	WORD		Spotřebovaná energie za dobu aktivity funkčního bloku. Hodnota je v jednotkách dle výstupu q_bEnergyUnit
q_wElapsTimeH	WORD		Uplynulý čas měření enregie v hodinách.
q_bEnergyUnit	BYTE		Jednotky naměřené enregie. 0...Wh, 1...kWh, 2...MWh
q_xAlarm	BOOL	FALSE	Výstupní alarm spuštěný překročením max. výkonu nebo dosažení hranice spotřeby.
q_bAlarmID	BYTE		0...dosaženo hranice spotřeby, 1...překonána maximální hodnota výkonu, 2...oba předchozí alarmy jsou aktivní.

5.1.5 EnergyMonitor2

Tento funkční blok slouží k počítání spotřebované energie na základě pulsů z pulsního elektroměru. Hrana na vstupu i_xInputSignal zvýší výstupní hodnotu výkonu o velikost převodního koeficientu i_rEnergyCoef.

Tabulka 5.5: Popis vstupů a výstupů - funkční blok EnergyMonitor1

	Typ	**Poč. hodn.**	**Popis**
VAR_IN			
i_xInputSignal	BOOL		Vstupní signál z impulzního elektroměru. Reaguje na hranu určenou vstupem i_xRiseOrFall.
i_rEnergyCoef	REAL		Udává počet Wh na jeden impuls.
i_xReset	BOOL	FALSE	Vynuluje naměřené hodnoty.
i_xBlokEnable	BOOL		Aktivuje meření energie.
i_xRiseOrFall	BOOL		Určuje aktivní hranu vstupu i_xInputSignal. TRUE...naběžná hrana, FALSE...sestupná hrana
i_xAlarmReset	BOOL	FALSE	Vynuluje výstupní alarm.
i_wEnergyLimit	WORD	0	Hranice spotřeby pro vystavení alarmu.
VAR_OUT			
q_wUsedEnergy	WORD		Spotřebovaná energie za dobu aktivity funkčního bloku. Hodnota je v jednotkách dle výstupu q_bEnergyUnit
q_wElapsTimeH	WORD		Uplynulý čas měření enregie v hodinách.
q_bEnergyUnit	BYTE		Jednotky naměřené enregie. 0...Wh, 1...kWh, 2...MWh
q_xAlarm	BOOL	FALSE	Je nastaveno po překročení nastavené hranice spotřeby.

5.1.6 OperHoursCounter

Tento funkční blok slouží k čítání operačních hodin (motohodin). Po uplynutí definovaného počtu operačních hodin je aktivováno upozornění na potřebu pravidelného servisu.

Tabulka 5.6: Popis vstupů a výstupů - funkční blok OperHoursCounter

	Typ	**Poč. hodn.**	**Popis**
VAR_IN			
i_xBlokEnable	BOOL		Čítání pracovních hodin je aktivní pokud vstup roven TRUE.
i_xResetAll	BOOL	FALSE	Resetuje celý blok čítání oper. hodin.
i_xRestartSerCnt	BOOL	FALSE	Restartuje počítání hodin do příštího servisu.
i_wHoursToServis	WORD	4000	Počet hodin do příštího servisu.
VAR_OUT RETAIN			
q_dwOperHrsTotal	DWORD	0	Celkový počet operačních hodin.
q_wOperHrs	WORD	0	Počet operačních hodin od posledního servisu.
q_wHrsToServis	WORD	4000	Počet hodin zbývajících do příštího servisu.
q_xServisAlarm	BOOL	FALSE	TRUE pokud q_wHrsToServis dosáhne nulové hodnoty.

5.1.7 SysTimeSetting

Tento funkční blok složí k nastavení systémového času PLC. Pro nastavení je použita časová hodnota zadaná uživatelem přes uživatelské rozhraní v sešitu Microsoft Excel. Proto musí být na vstup i_dwNewPlcDT přivedena proměnná uložená na adrese %MD498 (viz tabulka 4.2).

Tabulka 5.7: Popis vstupů a výstupů - funkční blok SysTimeSetting

	Typ	**Poč. hodn.**	**Popis**
VAR_IN			
i_dwNewPlcDT	DWORD		Nový čas PLC v sekundách od 1.1.1970
i_xSet	BOOL		Nábežná hrana provede nastavení data a času.
VAR_OUT			
q_dtSysTime	DT		Poslední čas nastevený pomocí tohoto funkčního bloku.
q_bError	BOOL		TRUE pokud vznikla chyba pri změně systémového času.

5.2 Výpočty výstupních hodnot funkčních bloků

Výpočet průměrné hodnoty ve funkčním bloku Statistics je prováděn dle rovnice:

$$q\_rAverage = \frac{1}{n} \sum_{t=1}^{t=n} i\_rInputSignal(t), \tag{5.1}$$

kde n je počet cyklů PLC od spuštění počítání. Hodnota vstupu i_rInputSignal je v každém cyklu přičítána a výsledná suma je dělena počtem cyklů.

Výpočet průměrné hodnoty ve funkčním bloku EnergyMonitor1 je prováděn dle:

$$q\_wUsedEnergy = \frac{36 \cdot 10^5}{T} \int_t^{t+T} P(t)dt \ [Wh;\ ms,\ W], \tag{5.2}$$

kde P je aktuální výkon na vstupu i_wActPower a T je čas měření. Naměřená energie je násobena příslušnou mocninou deseti podle vstupu i_iEnergyUnit. Ve skutečnosti se nejdá o spojitou integraci, ale o nespojité přičítání elementární spotřeby za dobu jednoho cyklu PLC. V každém cyklu se čte aktuální hodnota systémového času, od které se odečte časová hodnota cyklu předchozího. Tímto způsobem se měří délka předchozího cyklu.

Funkční blok EnergyMonitor2 pracuje jako impulsní elektroměr. Při detekci aktivní hrany vstupu i_xInputSignal se aktuální hodnota naměřené spotřeby zvýší o hodnotu čtenou ze vstupui _rEnergyCoef.

Funkční blok ImpulseCounter pracuje jako čítač. Při detekci aktivní hrany na vstupu i_xInputSignal je inkrementována výstupní proměnná.

Měření počtu operačních hodin ve funkčním bloku OperHoursCounter se provádí pomocí časovače se spožděným sepnutím, který je nastaven na jednu minutu. Po uplynutí jedné minuty se časovač restartuje a čítač minut se inkrementuje. Když čítač minut nabývá hodnoty 60, inkrementuje se počet hodin a vynuluje se čítač minut.

Vývojové diagramy všech funkčních bloků jsou uvedeny v příloze B.

Kapitola 6

Uživatelské rozhraní

Uživatelské rozhraní je tvořeno sešitem aplikace MS Excel. Ten obsahuje jeden hlavní list pro ovládání a dalších sedm listů sloužících jako šablony, do kterých se následně stahují data. Uživatel má možnost šablony upravovat podle toho jaká data se budou do tabulky načítat. Tím je myšlena zejména modifikace hlaviček sloupců a názvů řádků tabulek nebo skrytí nepoužívaných sloupců. Většina funkčnosti uživatelského prostředí byla vytvořena pomocí výše zmíněného jazyka Visual Basic for Application.

V hlavním menu (viz obr. 6.1) jsou k dispozici čtyři tlačítka pro ovládání aplikace. Po stisknutí tlačítka Stáhnout data se přenesou data uložená v PLC do PC a zobrazí se do nového sešitu, který je naformátován podle uživatelem modifikovaných šablon. Po stisknutí tlačítka Uložit stažená data je uživateli nabídnuto uložení nově vzniklého sešitu, název nového souboru je implicitně nastaven podle data uložení. Po stisku tlačítka Nastavit čas PLC se provede nastavení systémového času PLC podle hodnoty vyplněné v sousedním poli. Poslední tlačítko Vynulovat v PLC slouží k nulování bloků v paměti PLC. Po stisku je vynulována ta část paměti, která je vybrána v sousedním rozvinovacím seznamu. Aby nedošlo k neúmyslné ztrátě dat, je uživatel nucen vymazání dat explicitně potvrdit.

Ostatní listy označené jako Events a Statistics, slouží jako šablony, do kterých se budou data zobrazovat. Příklad konkrétní modifikace listu Events ukazuje obr. 6.2, jedná se o první reálné využití tohoto uživatelského rozhraní coby OPC klienta v praxi. Aplikace slouží ke sledování a sběru provozních dat z mostových jeřábů.

Obrázek 6.1: Hlavní menu uživatelského rozhraní

Schneider Electric

Zdvih - registr chyb

Datum	Kód chyby	Napětí meziobvodu [0.1V]	Proud motorem [0.1A]	Výstupní frekvence [0.1Hz]	Oteplení motoru [%]	ETA	ETI	CMD	Doba provozu [h]
12.03.2010 00:00:00	[Incorrect config.] (CFF)	123	23	3	2	213	12	32	12
12.03.2010 12:35:06	[Invalid config.] (CFI)	138	26	3	2	239	13	36	13
13.03.2010 01:10:12	[Modbus com.] (SLF1)	154	29	4	3	267	15	40	15
13.03.2010 13:45:18	[internal com. link] (ILF)	173	32	4	3	299	17	45	17
14.03.2010 02:20:24	[Com. network] (CnF)	194	36	5	3	335	19	50	19
14.03.2010 14:55:30	[Com. network] (CnF)	217	41	5	4	375	21	56	21
15.03.2010 03:30:36	[Overcurrent] (OCF)	243	45	6	4	420	24	63	24
15.03.2010 16:05:42	[Precharge] (CrF1)	272	51	7	4	471	27	71	27
16.03.2010 04:40:48	[Overcurrent] (OCF)	305	57	7	5	527	30	79	30
16.03.2010 17:15:54	[Load slipping] (AnF)	341	64	8	6	591	33	89	33
17.03.2010 05:51:00	[PTC1 probe] (PtF1)	382	71	9	6	662	37	99	37
17.03.2010 18:26:06	[PTC1 probe] (PtF1)	428	80	10	7	741	42	111	42
18.03.2010 07:01:12	[PTC1 overheat] (OtF1)	479	90	12	8	830	47	125	47
18.03.2010 19:36:18	[Drive overheat] (OHF)	537	100	13	9	929	52	140	53
19.03.2010 08:11:24	[Motor overload] (OLF)	601	112	15	10	1041	59	156	59
19.03.2010 20:46:30	[Overbraking] (ObF)	673	126	16	11	1166	66	175	66
20.03.2010 09:21:36	[Drive overheat] (OHF)	754	141	18	12	1306	74	196	74
20.03.2010 21:56:42	[Motor overload] (OLF)	845	158	21	14	1462	82	220	83
21.03.2010 10:31:48	[Overbraking] (ObF)	946	177	23	15	1638	92	246	93
21.03.2010 23:06:54	[Overbraking] (ObF)	1059	198	26	17	1835	103	276	104

Obrázek 6.2: Ukázka stažených událostí - aplikace pro sledování mostových jeřábů.

Kapitola 7

Komunikace

7.1 OPC

Jak již bylo zmíněno v předchozích kapitolách, komunikace mezi klientskou aplikací na PC a řídicím systémem probíhá přes OPC server. Schématické zapojení komunikace je uvedeno na obr. 3.1 příp. obr. 8.1. K plnění standardu OPC je použit software OFS od Schneider Elektric. Komunikace mezi serverem a cílovým zařízením (PLC) skýtá více variant připojení, které se odvíjí zejména od komunikačních možností daného PLC. Je důležité se zmínit o potřebě dynamických knihoven SAOFSGrpX.dll a SAOPCGrpX.dll, které jsou nezbytné pro vytváření proměnných na straně klientské aplikace v MS Excel. Tyto knihovny jsou obsaženy na přiloženém CD.

7.2 Možnosti připojení

Byly otestovány dvě možnosti připojení mezi PLC a OPC serverem a to po protokolu Modbus[1] RTU a protokolu Modbus TCP.

7.2.1 Modbus RTU

Modbus je přenosový protokol vyvinutý firmou Gould Modicon (nyní Schneider Electric) pro komunikaci mezi řídicími systémy. Konstruktéři využili již zavedené standardy komunikací, a hlavně „nesvázali“ Modbus s žádnou placenou licencí. To bylo začátkem

[1] Tzv. Modicon - bus, komunikační protokol, více informací na http://www.modbus.org

globálního rozšíření Modbusu [14]. Nyní je považován za veřejný protokol a stal se de facto standardem v komunikaci mezi zařízeními různých výrobců. Na rozdíl od jiných protokolů nedefinuje žádnou fyzickou vrstvu (OSI vsrtva 1) [3].

V této variantě připojení probíhá komunikace po sériové lince RS485 (EIA - 485). Více informací o standardu RS485 lze najít v literatuře [4]. Do PC je linka přivedena pomocí převodníku RS485-USB do USB portu. Je nezbytné provedení správného nastavení Modbus adresy PLC a parametrů přenosu (rychlost, parita, stop bity). Pro nastavení konfigurace OFS je na přiloženém CD obsažen konfigurační soubor OFS_Config_modbus.xml.

7.2.2 Modbus TCP

Druhá testovaná možnost připojení je přes ethernetovou síť pomocí protokolu Modbus TCP. Pro více informací o protokolu Modbus TCP viz [13]. I pro tento případ je na CD přiložen konfigurační soubor pro nastavení OPC Factory serveru OFS_Config_ethernet.xml.

Kapitola 8

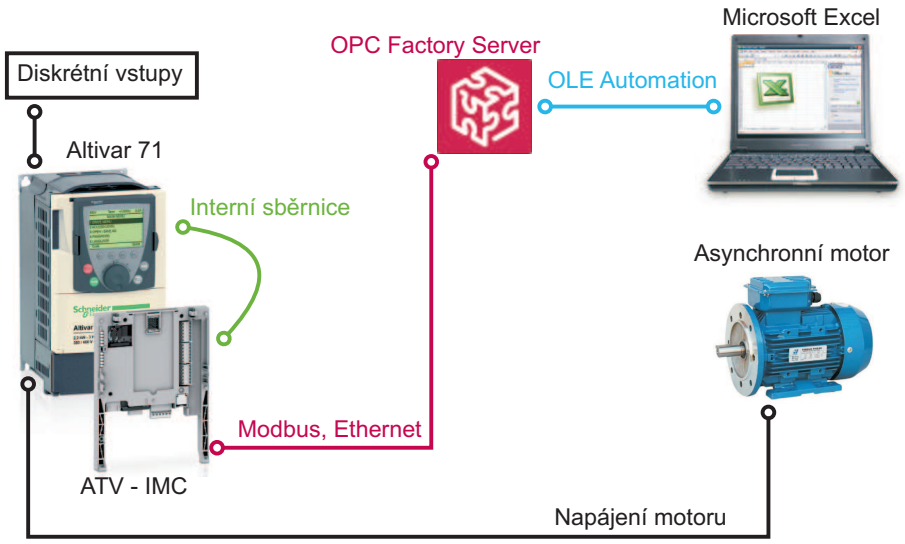
Testování

Pro otestování funkce knihovny Monitoring a OPC klienta byl vytvořen testovací projekt (ATV_IMC_monitoring_template.projectarchive). Tento projekt slouží zároveň jako šablona pro využití knihovny. Definuje globální proměnné určené k stahování a obsahuje instance všech funkčních bloků knihovny Monitoring. Ukázka tohoto projektu v jazyce CFC je zobrazena v příloze C.

8.1 Testovací zapojení

K testování bylo využito zapojení s frekvenčním měničem Altivar 71, programovatelnou řídicí kartou Altivar - IMC a trojfázovým asynchronním motorem. Z registrů frekvenčního měniče lze číst aktuální hodnoty proudu, napětí, otáček, výkonu nebo stavová slova. Tato možnost byla využita k testování funkčních bloků pomocí reálných vstupních hodnot.

Komunikace mezi frekvenčním měničem a kontrolérem probíhá po interní sběrnici. Čtení proměnných z frekvenčního měniče probíhá na začátku každého cyklu volně běžící úlohy, která je v testovacím projektu nazvána freewheel_Task. Pro detekci vzniku je vhodné použít rošířené stavové slovo LR1, jehož druhý bit se nastaví pokud měnič přejde do stavu chyby. Paměť frekvenčního měniče poskytuje desítky parametrů, které lze zaznamenávat při vzniku údálostí, jejich úplný seznam lze najít v [6]. Schéma testovacího zapojení zobrazuje obr. 8.1



Obrázek 8.1: Schéma testovacího zapojení

8.2 Provádění testů

Jeden z diskrétních vstupů měniče byl nastaven jako signál externí chyby, který byl využit pro snadné přivedení měniče do chybového stavu. Výstupní parametry měniče, načítané po interní sběrnici se zaznamenají do pole arstrcEvents, které je vstupně-výstupní proměnnou bloku EventsManager. Po vzniku poruchy se rovněž nastaví výstupní alarm a nastaví ostatní výstupní proměnné.

Pro testování funkce bloku EnergyMonitor1 byla na vstup pro čtení aktuálního výkonu přivedena proměnná, která je naplněna hodnotou výkonu, kterou poskytuje přímo frekvenční měnič. Funkce spouštění a resetování alarmu byla testována změnou vstupních proměnných, které udávají hranice výkonu a energie pro spuštění alarmu.

Blok EnergyMonitor2, který zastává funkci impulsního elektroměru, byl testován pouze změnou vstupních proměnných, které nejsou nikterak čteny z měniče. Několikeré překlopení vstupu i_xInputSignal je postačující pro kontrolu správné funkčnosti bloku. Stejným způsobem byl otestován i blok ImpulseCounter.

Vstup i_rInputStatistics čte aktuální hodnotu rychlosti, kterou poskytuje frekvenční měnič. Jelikož jsou všechny vstupní i výstupní proměnné typu real, je zapotřebí nejprve přetypovat hodnotu rychlosti získanou z frekvenčního měniče z typu int na real.

Proměnné připojené na vstupy bloku SysTimeSetting jsou přepisovány uživatelským

přístupem ze sešitu aplikace Microsoft Excel. Po stisku tlačítka Nastavit čas PLC vznikne impuls na vstupu i_xSet, který provede nastavení času. Po správném nastavení času se musí výstup q_dtSysTime shodovat s požadovaným časem a výstup q_bError musí nabývat hodnoty false.

Hodnoty vstupních proměnných bloku OperHoursCounter byly při testování měněny pouze pomocí programovacího prostředí.

8.3 Výsledky testů

Blok EventsManager správně reaguje na vznik chyby díky čtení rozšířeného stavového slova LR1. Všechny přivedené proměnné typu parametr události se zaznamenávají do vstupně-výstupní proměnné. Čas události uložený ve vstupně-výstupní proměnné odpovídá okamžiku jejího vzniku. Po vzniku události se nastaví výstupní alarm, který lze bez potíží resetovat vstupem i_xReset.

Blok EnergyMonitor1 správně integruje energii jako součin aktuálního výkonu a aktuální délky cyklu PLC. Nastavování a reset alarmu funguje bez problémů.

Výstupní hodnota energie bloku EnergyMonitor2 odpovídá celočíselným násobkům vstupní proměnné i_rEnergyCoef. Spouštění a resetování alarmu funguje správně.

Funkční blok ImpulseCounter správně čítá impulsy vstupní proměnné i_xInputSignal. Po načítání definovaného počtu impulsů se nastaví alarm, který lze bez problémů resetovat. Nastavení vstupu i_xRiseOrFall změní spouštěcí hranu z vzestupné na sestupnou.

Počet motohodin načítaný za dobu testování odpovídal realnému času měřeného pomocí stopek. Alarm ohlašující potřebu pravidelného servisu se spustí po uplynutí definovaného počtu hodin.

Po stisku tlačítka Stáhnout data v uživatelském rozhraní aplikace Excel se všechny proměnné určené ke stažení zobrazí správně. Nulování registrů v PLC z uživatelského rozhraní pracuje bez problémů. Funkční blok SysTimeSetting není kompatibilní s PLC Altivar - IMC. Na ostatních PLC rodiny SoMachine funguje bez problémů.

Všechna data typu událostí i statistik jsou v PLC uchovány po resetu PLC díky použití proměnných s parametrem retain. Vstupně výstupní proměnné bloku EventsManager je definována uživatelem, pro správnou funkci je potřeba, aby uživatel definoval tuto proměnnou způsobem uvedeným v testovacím projektu.

Hodnoty naměřené při provádění testů jsou uvedeny v příloze A.

Kapitola 9

Závěr

Cílem práce bylo vytvoření knihovny pro prostředí CoDeSys V3, která bude obsahovat funkční bloky pro uchování událostí a souhrnných hodnot provozních veličin. Dále bylo zapotřebí vytvořit aplikaci pro uchování dat na PC, k tomu bylo využito aplikace MS Excel. Tato práce obsahuje důslednou dokumentaci funkčních bloků včetně popisu vstupů a výstupů a vývojových diagramů.

Požadavky na řešení uvedené v 2.2 byly splněny. Byla nalezena a definována vhodná struktura pro ukládání dat ve vnitřní paměti PLC. Při vypracování byl brán v úvahu požadavek na snadnou modifikovatelnost. Hlavní test byl proveden na řídicím systému Altivar - IMC, platformě, u které se očekává nejčastější využití - jeřábové aplikace, vleky a další. Byla však ověřena funkčnost i na ostatních PLC rodiny SoMachine - Modicon M238, Modicon M258 a Modicon LMC058. Uložená data jsou uchována v PLC i po výpadku napájení. Uživatel má možnost zasahovat do dat uložených v PLC z uživatelského prostředí na PC. Požadavek pro sledování přiblížení procesních údajů ke kritickým hodnotám byl naplněn. Funkční bloky, u kterých to bylo žádoucí, obsahují výstupní proměnnou, která oznamuje dosažení kritické hodnoty.

Značná část času byla věnována tvorbě aplikace k načtení a uchování dat na PC. V této fázi bylo s výhodou použito knihy [1], která poměrně do hloubky probírá základy programování ve VBA. Při samotném programování knihovny mi byla velice nápomocná propracováná nápověda v prostředí SoMachine. Využití dokumentace a manuálů k použitým produktům Schneider Electric ([6], [7], [8], [9], [10]) bylo také užitečné.

Modifikované řešení uživatelské aplikace pro MS Excel již bylo reálně využito pro monitoring jeřábových aplikací, což svědčí o možnosti reálného využití této práce. Na obr. 9.1 je zobrazen mostový jeřáb, na kterém byly využity výsledky této bakalářské práce. Aplikace pro stahování dat do PC značně zkrátila fázi uvedení stroje do provozu díky snadnému

Obrázek 9.1: Mostový jeřáb - ukázka z konkrétní aplikace této bak. práce

přístupu k provozním datům jeřábového systému. Technik se již nemusel připojovat do PLC přes programovací prostředí, aby mohl kontrolovat stav systému. Všechny jím zvolené parametry mohl pohodlně číst z přehledných tabulek v sešitě Microsoft Excel. V dohledné době bude tato práce využita na další jeřábové aplikaci. Na obr. 6.2 je ukázána možná modifikace grafického prostředí pro konkrétní aplikaci.

Jako možnost pro budoucí rozšíření této práce se nabízí použití OPC serveru, který je implementován přímo v prostředí CoDeSys. Pro platformu SoMachine se tato možnost objevila teprve s aktuální verzí SoMachine V3 (květen 2011). Toto rozšíření by mohlo pomoci ke zkrácení doby potřebné ke stažení dat z PLC do PC, která je nyní řádově v jednotkách vteřin. Dále by přibyla možnost komunikace přes programovací rozhraní Mini - USB.

Literatura

[1] Král, M.: *Excel VBA Výukový kurz*. Computer Press, Brno 2010

[2] Šmejkal, L. Martinásková, M.: *PLC a automatizace, 1. díl*. BEN - technická literatura, Praha 1999

[3] Mackay, S. Wright, E. Reynders, D. Park, J.: *Practical Industrial Data Networks: Design, Installation and Troubleshooting*. Elsevier, Oxford 2004

[4] Black, U.: *Physical Layer Interfaces and Protocols*. IEEE Computer Society Press, Los Alamitos 1996

[5] *OPC Factory Server V3.33 User Manual*. Schneider Electric,3/2009

[6] *Altivar71 Communication parameters, User manual*. Schneider Electric,11/2009

[7] *Modicon M238 Logic Controller, Programming Guide*. Schneider Electric,1/2011

[8] *Modicon LMC058 Motion Contoller, Sytem Functions and Variables LMC058 PLCSystem Library Guide*. Schneider Electric,10/2010

[9] *Modicon M258 Logic Controller, Programming Guide*. Schneider Electric,10/2010

[10] *Altivar ATV - IMC Drive Controller, Hardware Guide*. Schneider Electric,01/2010

[11] *OPC Foundation*[online]. 2011 [cit. 2011-05-08]. Dostupné z WWW: http://www.opcfoundation.org/

[12] *3S - Smart Software Solutions*[online]. 2011 [cit. 2011-05-08]. Dostupné z WWW: http://www.3s-software.com

[13] *Modbus Organization*[online]. 2011 [cit. 2011-05-08]. Dostupné z WWW: http://modbus.org

[14] Fischmann, P.: *Modbus slaví 30 let: průmyslový protokol dobyl svět.* Schneider Magazín, Jaro 2010, s. 12

Příloha A

Výsledky testů - naměřené hodnoty

EnergyMonitor1

Na vstupu i_wActualPower byla hodnota 7000 W po dobu 5 minut.

Teoretická hodnota naměřené energie: 0,583 kWh,

naměřená hodnota pomocí funkčního bloku: 0,583 kWh.

OperHoursCounter

Ve stejnou dobu bylo spuštěno měření funkčního bloku a čítání stopek. Po 40 minutách nebyl zaznamenán časový rozdíl v řádu vteřin.

Statistics

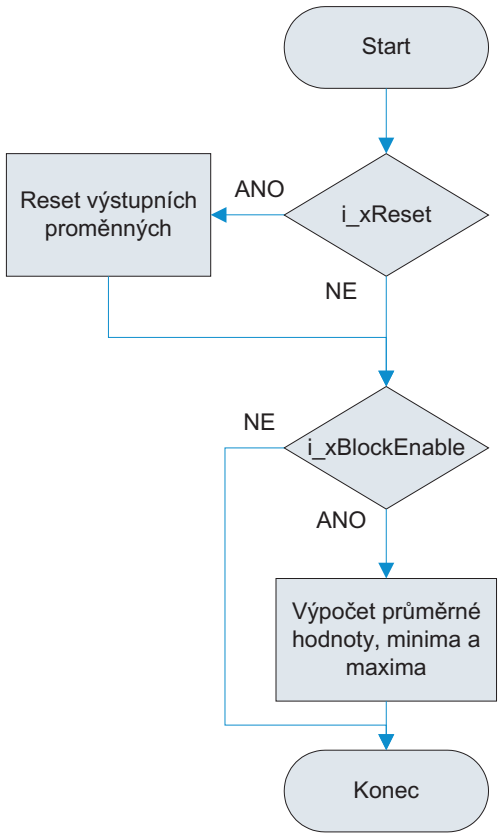
Pro ověření správného výpočtu průměrné hodnoty byla na vstup i_rInputSignal přivedena na 3 minuty hodnota -100 a na 3 minuty hodnota +100. Po těchto 6 minutách byla hodnota výstupu q_rAverage rovna 0.

EnergyMonitor2

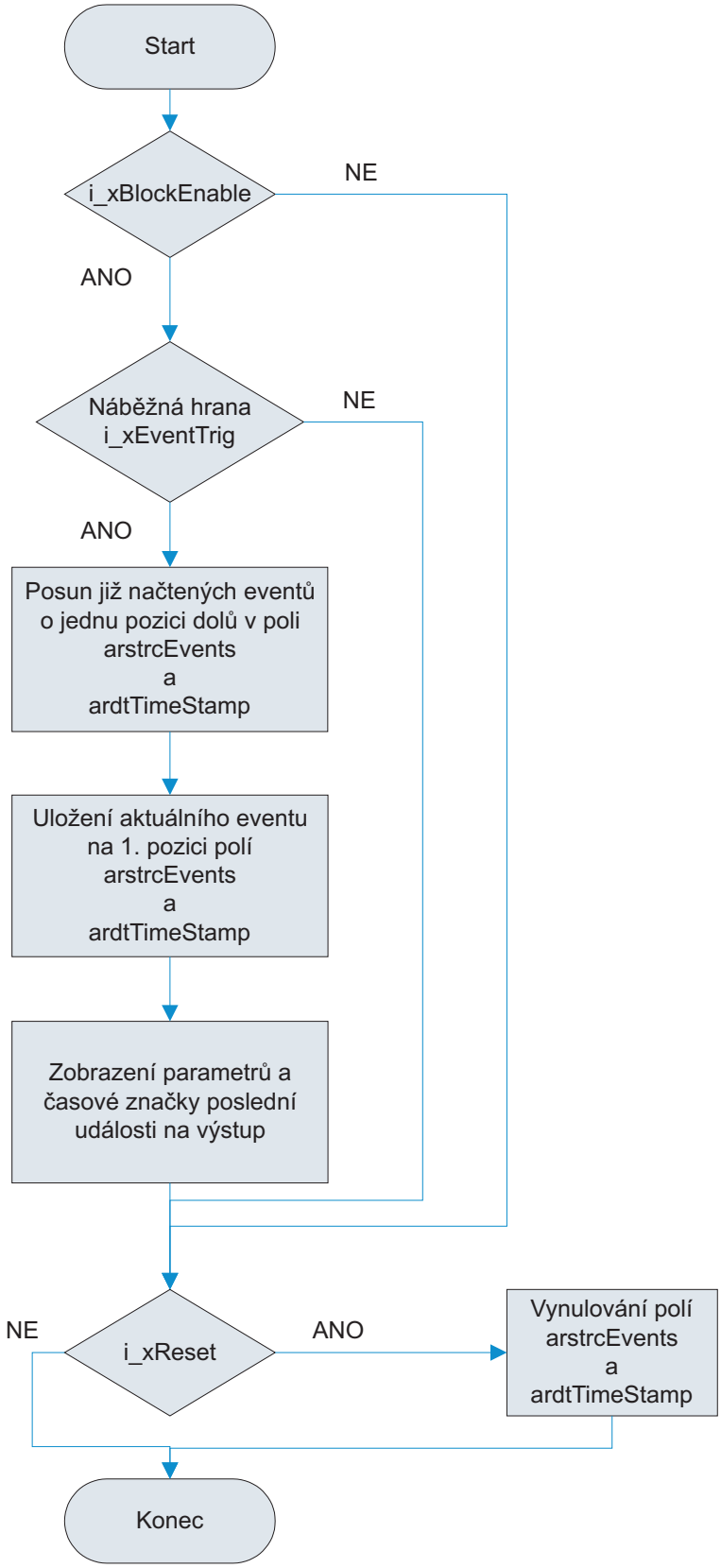
Na vstup i_rEnergyCoef byla přivedena hodnota 1000. Po detekci 10 sestupných hran na vstupu i_xInputSignal byla hodnota výstupu rovna 10 000.

Příloha B

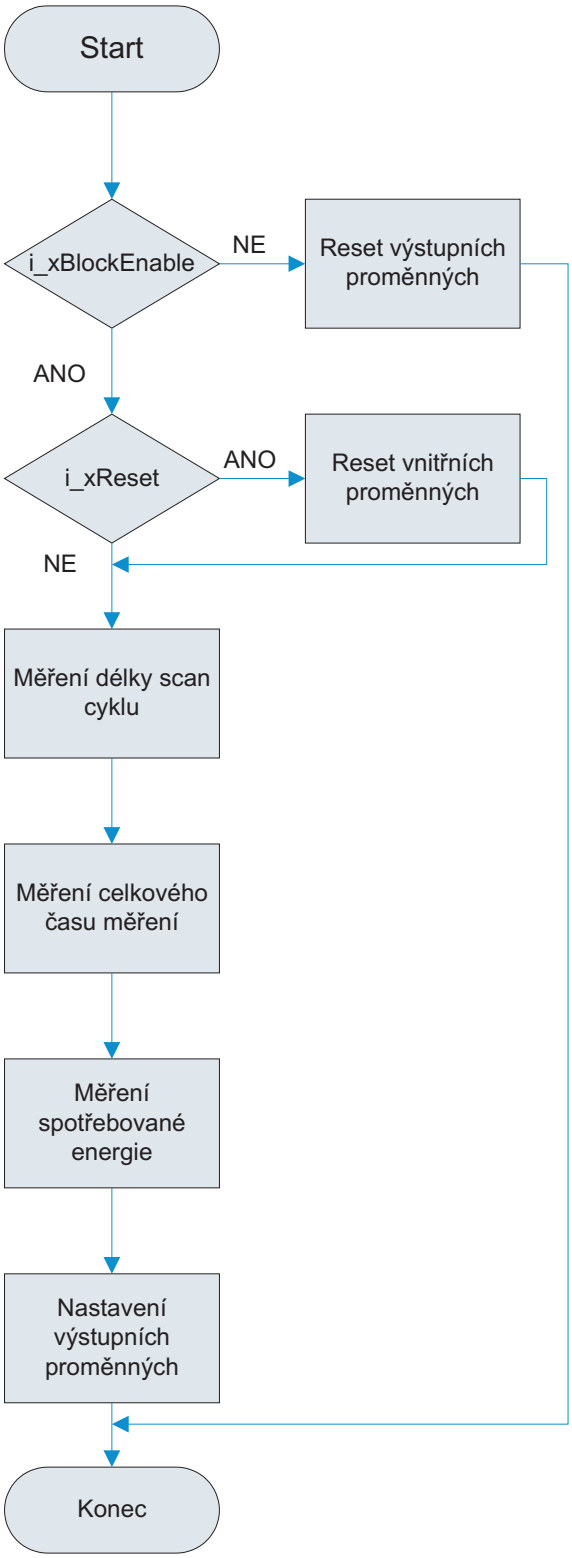
Vývojové diagramy funkčních bloků knihovny Monitoring



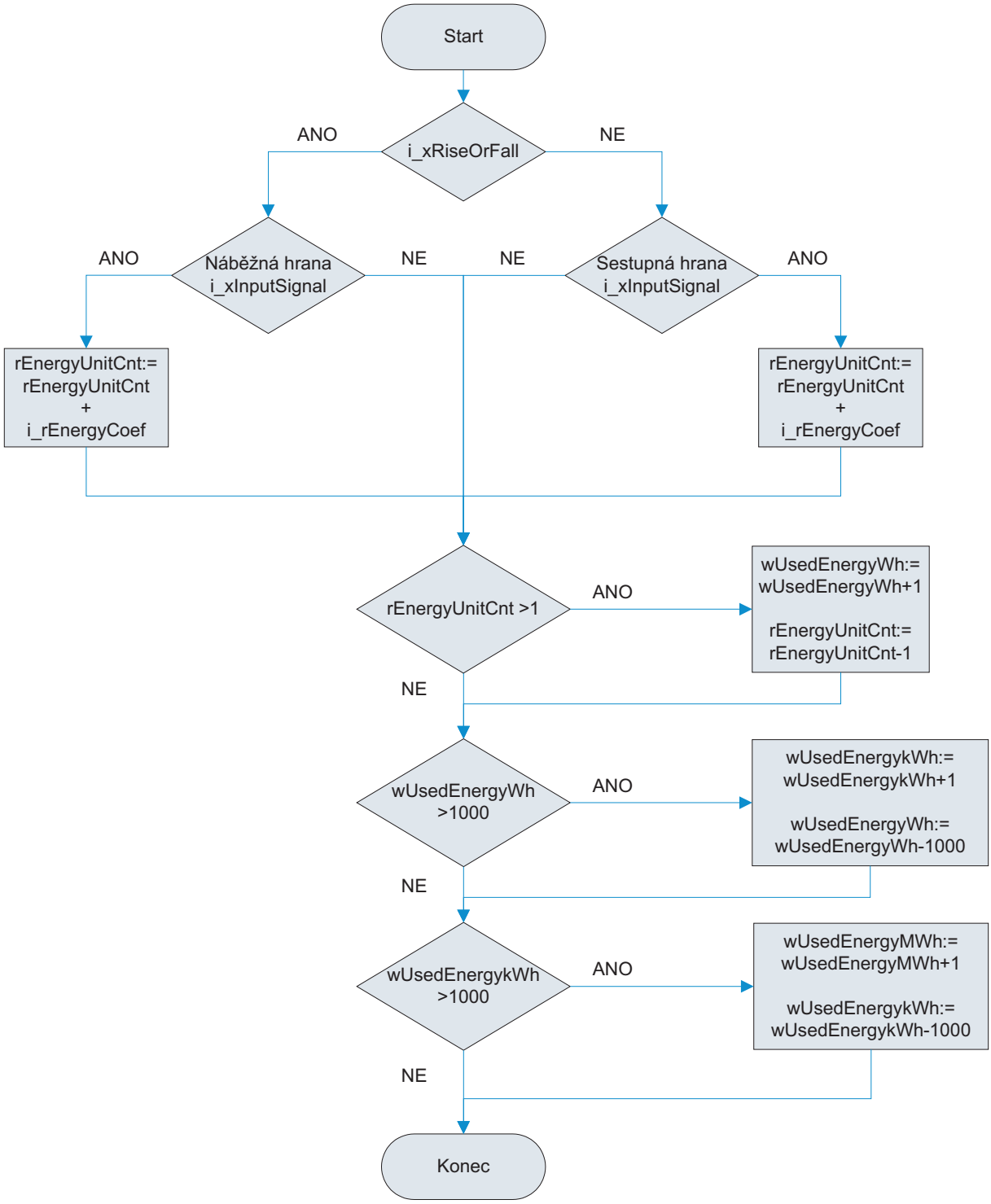
Obrázek B.1: Vývojový diagram funkčního bloku Statistics



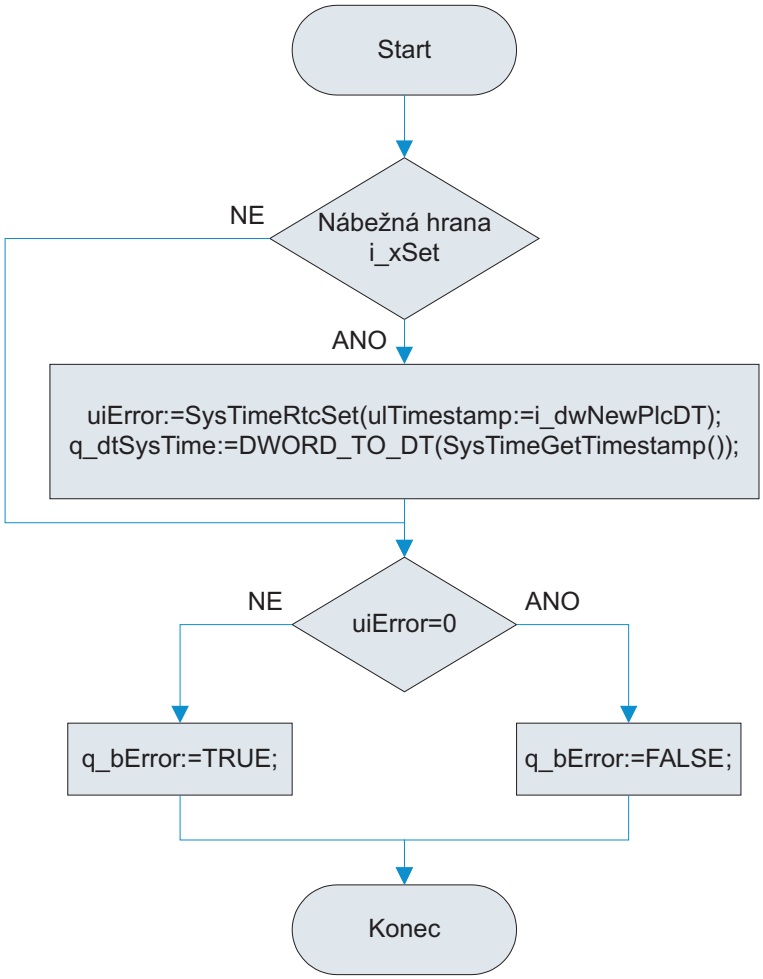
Obrázek B.2: Vývojový diagram funkčního bloku EventsManager



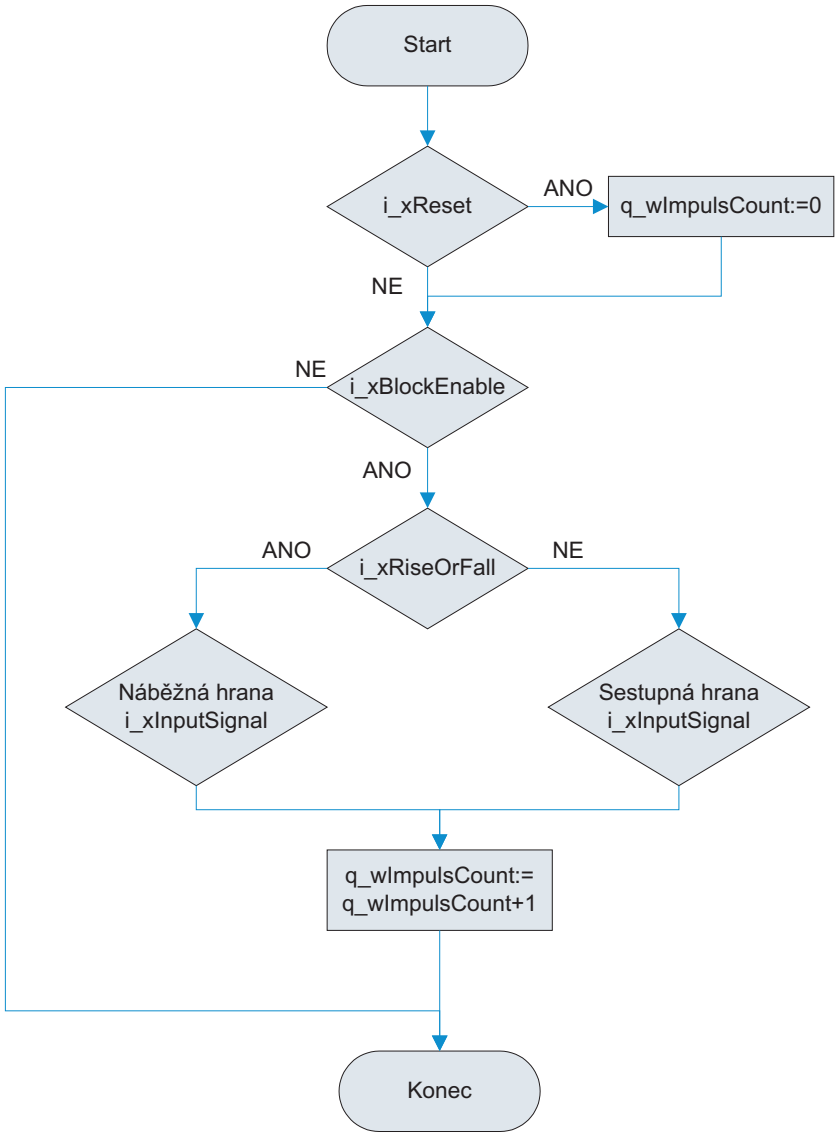
Obrázek B.3: Vývojový diagram funkčních bloků EnergyMonitor1 a EnergyMonitor2



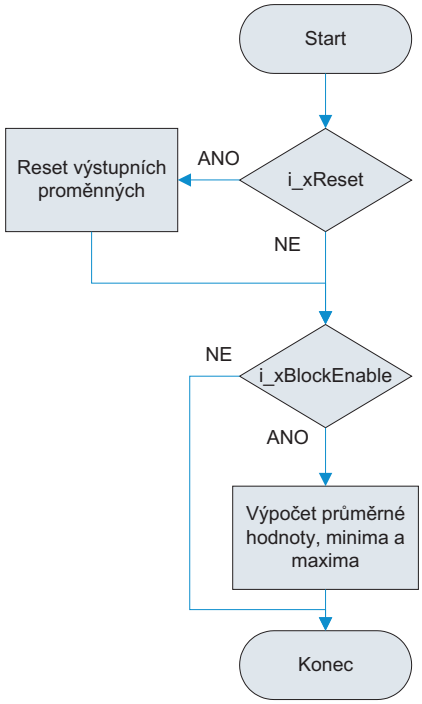
Obrázek B.4: Vývojový diagram měření energie ve funkčním bloku EnergyMonitor2



Obrázek B.5: Vývojový diagram funkčního bloku SysTimeSetting



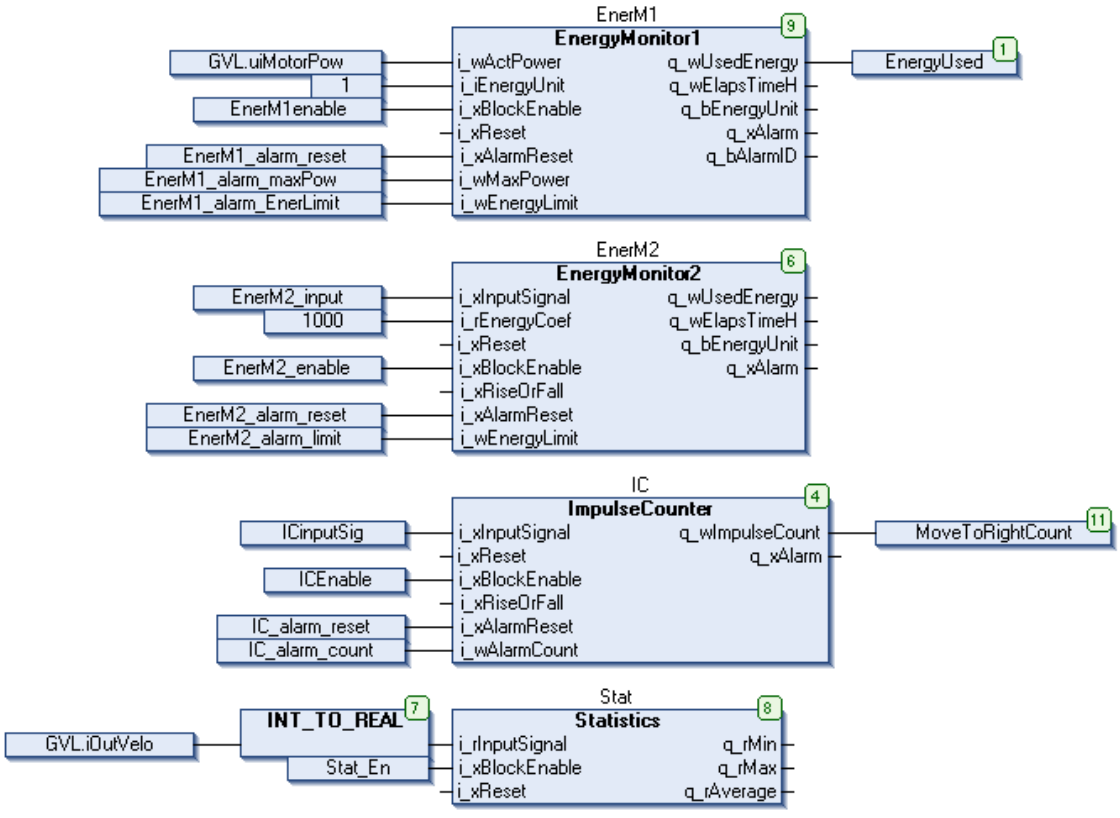
Obrázek B.6: Vývojový diagram funkčního bloku ImpulsCounter

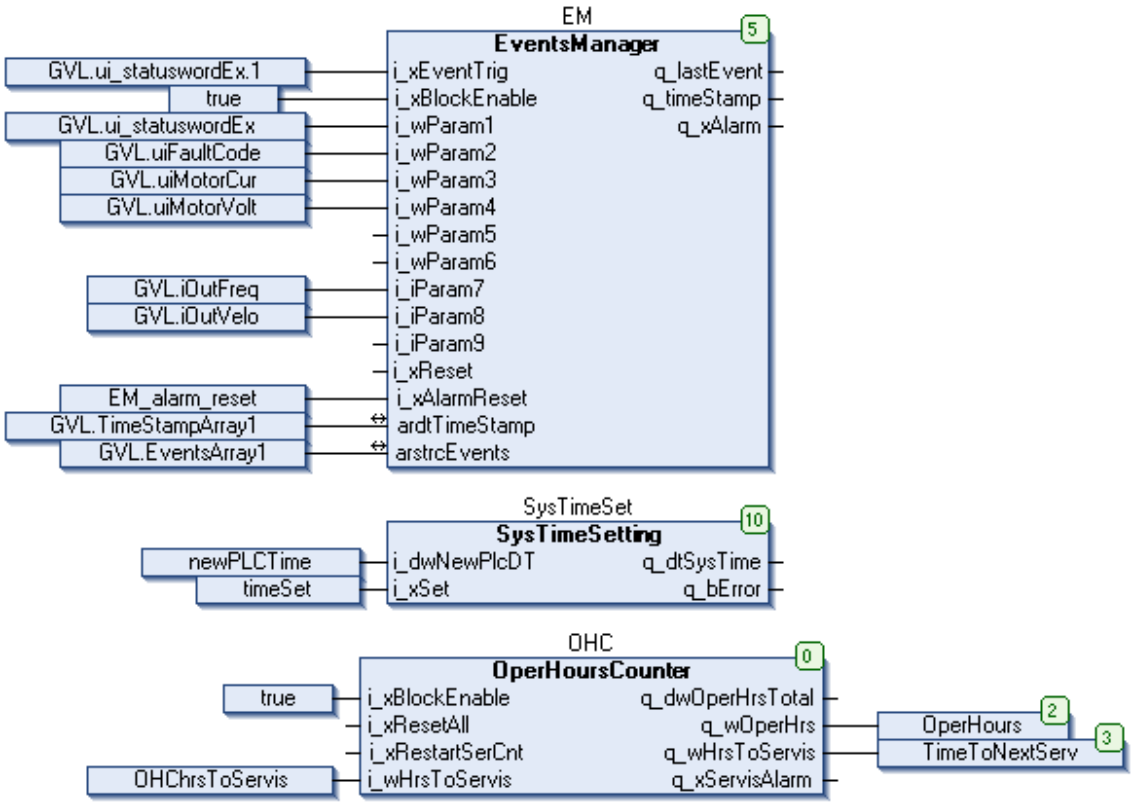


Obrázek B.7: Vývojový diagram funkčního bloku OperHoursCounter

Příloha C

Ukázka testovacího programu




EM
EventsManager
5
GVL.ui_statuswordEx.1
true
GVL.ui_statuswordEx
GVL.uiFaultCode
GVL.uiMotorCur
GVL.uiMotorVolt
GVL.iOutFreq
GVL.iOutVelo
EM_alarm_reset
GVL.TimeStampArray1
GVL.EventsArray1
i_xEventTrig
i_xBlockEnable
i_wParam1
i_wParam2
i_wParam3
i_wParam4
i_wParam5
i_wParam6
i_iParam7
i_iParam8
i_iParam9
i_xReset
i_xAlarmReset
ardtTimeStamp
arstrcEvents
q_lastEvent
q_timeStamp
q_xAlarm
SysTimeSet
SysTimeSetting
10
newPLCTime
timeSet
i_dwNewPlcDT
i_xSet
q_dtSysTime
q_bError
OHC
OperHoursCounter
0
true
OHChrsToServis
i_xBlockEnable
i_xResetAll
i_xRestartSerCnt
i_wHrsToServis
q_dwOperHrsTotal
q_wOperHrs
q_wHrsToServis
q_xServisAlarm
OperHours
2
TimeToNextServ
3

Příloha D

Seznam použitých zkratek

CFC - Continuous Function Chart
FBD - Function Block Diagram
HMI - Human Machine Interface
IEC - International Electrotechnical Commision
IMC - Integrated Motion Controller
LMC - Lexium Motion Controller
MS - Microsoft
OEM - Original Equipment Manufacturer
OFS - OPC Factory Server
OLE - Object Linking and Embedding
OPC - OLE for Process Control
OPC DA - OPC Data Access
PLC - Programmable Logic Controller
RTU - Remote Terminal Unit
USB - Universal Serial Bus
VBA - Visual Basic for Excel

Příloha E

Obsah přiloženého CD

K této práci je přiloženo CD s tímto obsahem:

- root
  - Monitoring.library
  - VycteniGeneral.xls
  - ATV_IMC_monitoring_template.projectarchive
  - SAOFSGrpX.dll
  - SAOPCGrpX.dll
  - OFS_config_modbus.xml
  - OFS_config_ethernet.xml