České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra řídicí techniky

DIPLOMOVÁ PRÁCE

Automatické bezobslužné řízení systémů záložních motorgenerátorových zdrojů

Praha, 2009 Jan Bednář

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra řídicí techniky

Automatické bezobslužné řízení systémů záložních motorgenerátorových zdrojů

Řešitel: Jan Bednář
Vedoucí práce: Ing. Pavel Burget
Oponent: Ing. Milan Egart

Diplomová práce v rámci prezenčního magisterského studia na katedře Řídicí techniky, fakulty elektrotechnické, Českého vysokého učení technického v Praze.

Studijni obor: MKM01 - Řídicí technika

Prohlášení

Prohlašuji, že jsem zadanou diplomovou práci zpracoval sám s přispěním vedoucího práce a konzultanta a používal jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

Mám důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb. , o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne _____ _____

podpis

Poděkování

Je mou milou povinností poděkovat lidem, bez nichž by tato diplomová práce nevznikla. Děkuji především vedoucímu práce Ing. Pavlu Burgetovi za konzultace k vývojovému prostředí CoDeSys a připomínky k navrhům strategie řízení. Dále konzultantu Ing. Milanu Egartovi za cenné rady při návrhu řídicího algoritmu. Díky patří též Ing. Martinu Kůrkovi za rady k vývojovému prostředí ControlWeb. V neposlední řadě mým rodičům a sestře za podporu během celého studia.

Abstrakt

Předkládaná diplomová práce pojednává o vývoji programového řešení pro automatické bezobslužné připojování záložního motorgenerátoru k zálohované soustavě. Fyzická část systému byla vyprojektována pro danou aplikaci - „řešení na míru“. Cílem vytvářeného programu pro PLC bylo řešení přepojování zálohované zátěže mezi primární sítí a záložním zdrojem. Údaje o technologii jsou vizualizovány v panelovém PC ve dveřích rozvaděče. Kvalita napájecí soustavy i zálohované soustavy je měřena speciálními přístroji. Součástí řešení kromě programu pro ovládání přepínání je i monitorování jednotlivých stavů obou motorgenerátorů, stavů výkonových přepínacích prvků v rozvaděčích, sledování a archivování dat z měřicích přístrojů.

Fyzické propojení PLC kontroléru s panelovým PC je provedeno dnes standardním Ethernetem 100 Mb/s a data se přenášejí pomocí otevřeného protokolu pro vzájemnou komunikaci různých typů zařízení MODBUS TCP/IP. Měřicí přístroje jsou ke kontroléru PLC fyzicky připojeny průmyslovou sériovou datovou linkou RS-485 a protokol je MODBUS RTU. Celý systém je vzdáleně monitorován z centrálního dohledu. K tomuto účelu jsou důležité stavy a hodnoty technologie sledovány a zasílány pomocí síťového protokolu pro správu síťových prvků SNMP na server. Zde jsou archivovány a je možné sestavovat různé statistiky.

Abstract

Presented diploma thesis discuss the development of software solution of operation free connection back-up motorgenerator with deposit system. Hardware of the solution was done and was designed directly for this aplication. The goal of program in PLC was solution of making reconnection of back-up part of technology between primary power line and secondary power source. Data of whole technology can bee seen in vizualization program in panel PC, which is situated in doors of control panel.Quality of power system line and back up system line is measured with special devices. Is included besides program to contorl switching and monitor individual states of both dieselagregates, states of power switch units in control panels, watch and archive of dates from measurement units.

Hardware connection between PLC and panel PC is realized by nowadays standard Ethernet 100 Mb/s. Data are transmited with open source protocol, which is designed to communication of different types of units, MODBUS TCP/IP. Measuring units are connected with PLC via industrial serial bus named RS-485 and protocol is MODBUS RTU. Whole system is remotely monitored by central supervisor. To this purpose are significant states and values of technology transmited via network protocol SNMP. In central supervisor station are data archived and is able to make various statistics.

České Vysoké Učení Technické v Praze
Fakulta elektrotechnická
Katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Jan Bednář**

Studijní program: Elektrotechnika a informatika (magisterský), strukturovaný
Obor: Kybernetika a měření, blok KM1 - Řídicí technika
Název tématu: Automatické bezobslužné řízení systémů záložních motorgenerátorových zdrojů
Pokyny pro vypracování

1. Teoretický rozbor funkce systému motor-generatorových zdrojů a výkonových přepínacích systémů ATS-Automatic Transfer Switch.
2. Teoretický rozbor časového harmonogramu a topologie řízení.
3. Návrh řešení ŘS - sestava pro modulární programovatelný automat (PLC).
4. Návrh řešení vzdáleného řízení, dohledu a vizualizace na PC-komunikace ModBus.
5. Komplexní realizace ŘS.
6. Implemetace registru událostí a diagnostiky systému.
7. Vizualizace operátorského grafického rozhraní (LCD-touchscreen) s využitím prostředí ControlWeb5.
8. Popis ostatních možností řídicího systému.

Seznam odborné literatury:
Dodá vedoucí práce

Vedoucí práce: Ing. Pavel Burget

Platnost zadání: do konce zimního semestru 2008/2009

L.S.

prof. Ing. Michael Šebek, DrSc. doc. Ing. Boris Šimák, CSc. — děkan, vedoucí katedry

V Praze dne 10.9.2007

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

Symbol	Význam
AES	Advanced Encryption Standard
ARP	Address Resolution Protocol
ATS	Automatic Transfer Switch
DES	Data Encryption Standard
IAB	Internet Acitivities Board
IANA	Internet Assigned Numbers Authority
IP	Internet Protocol
ISO	International Organization for Standardization
MTG, MG	Motor-generator
MIB	Management information base
OID	Object Identifier
SNMP	Simple Network Protocol
TCP	Transmission Control Protocol
UPS	Uninterruptible Power Supply (Source)
	Protokoly
ADU	Administration Data Unit
HDLC	High-Level Data Link Control
PDU	Protocol Data Unit
SDLC	Standard-Level Data Link Control
	Kódování dat
NRZ	Non Return to Zero
NRZI	Non Return to Zero Invert

Symbol	Význam
	PLC automat WAGO
CFC	Continuous Function Chart
FBD	Function Data Box
IL	Instruction List
LD	Ladder Diagram
PLC	Program Logic Controller
POU	Program Organisation Unit
SFC	Sequential Function Chart
ST	Statement List
	aplikace ControlWeb
CW	ControlWeb
	UPS
VD	Voltage Dependent
VFD	Voltage and Frequency Dependent
VFI	Voltage and Frequency Independent
VI	Voltage Independent

Kapitola 1

Úvod

V dnešní době, kdy se společnost stává stále více závislá na elektrické energii, může její nedostatek či dokonce výpadek znamenat velké problémy v mnoha případech doprovázené finančními ztrátami. Proto jsou důležité technologie vybavovány záložními zdroji energie. Ty lze jednoduše rozdělit na záložní zdroje pro krátké výpadky a záložní zdroje pro dlouhodobé dodávky výkonu.

Pokud je požadavek pouze na krátkodobé zálohování, postačí jako záložní zdroje UPS. Ty jsou v závislosti na konstrukci rychlé a některé typy dokonce překlenou výpadek bez odpojení zátěže. Takový záložní zdroj je možné použít např. pro málo významné datové centrum, kdy UPS dodávají energii po dobu potřebnou k bezpečnému vypnutí počítačů.

V případě, kdy je požadavek na dlouhodobé zálohování elektrickou energií, je nutné uvažovat např. o motorgenerátoru. Tato varianta umožňuje zálohovat v podstatě neomezenou dobu (pokud bude průběžně doplňováno palivo a splněny požadavky na start stroje). Proto se v praxi často využívá spojení UPS a motorgenerátoru, kdy UPS udržují napájení po dobu startování agregátu. Tato varianta se označuje jako okamžité, nepřerušitelné převzetí zátěže. Když je agregát nastartován a připojený synchronní stroj je schopen dodávat energii do zátěže, je nutné přepojit technologie z primární sítě na záložní motorgenerátor. Na trhu existují tzv. ATS (Automatic Transfer Switch)jednotky, které při napětí z motorgenerátoru požadovaných parametrů automaticky zajistí přepnutí.

Pojem ATS, nebo-li Automatic Transfer Switch je definován americkým patentem pod číslem US Patent 6876103[1] s názvem Automatic Transfer Switch system and controller. V abstraktu lze nalézt základní specifikace. Například ukazuje, jak zahrnout do jediného systému primární zdroj energie (jeho případnou regulaci, filtraci a přeměnu na užitečný výkon) a záložní zdroj energie. Řídicí systém také implementuje přepínací prvky (hovoří se o elektromagnetickém principu), vestavěný mikrokontrolér pro monitorování síťového a generovaného napětí a v neposlední řadě také uživatelské rozhraní pro ovládání celého systému a získávání potřebných informací o systému. Pro přehlednost se užívají LED kontrolky pro rychlé získání souhrnné informace o ATS systému.

Tato diplomová práce se zabývá řešením obdobného systému, monitorováním jeho stavu a také specifickým řešením připojování a odpojování zátěže v závislosti na hladině pohonných hmot. Celá technologie je řízena pomocí dvou PLC automatů WAGO. První

[1] Automatic transfer switch systems and controllers, Patent ID: US6876103, Issue Date: April 05, 2005, více na http://www.patents.com/Automatic-transfer-switch-systems-controllers/US6876103/en-US/

obsluhuje rozvaděče ATS a druhý řeší systém připínání a odpínání zátěže v závislosti na množství paliva v nádrži. Zároveň při prvním připnutí na motorgenerátor po výpadku zajišťuje postupné připnutí celé záteže, aby nedocházelo k odběrovým špičkám. Technologie je monitorována pomocí panelového PC vestaveného do dveří rozvaděče ATS. S oběma PLC je komunikace zajištěna pomocí ethernetu s užitím deterministického průmyslového protokolu MODBUS TCP/IP.

Časové odezvy v řádu mikrosekund jsou potřebné v aplikacích pro řízení různých zařízení a nejlépe jsou dosažitelné zařízeními se zabudovaným speciálním softwarem. Používat pro takovou aplikaci standardní protokol Ethernet není vhodné. Nutností je určitá modifikace pro zvládnutí řízení v reálném čase. Proto byl upraven standard MODBUS i pro rozhraní ethernet a označuje se jako MODBUS TCP/IP.

Pro sběr informací o celém systému a jejich archivaci na jednom místě je využit protokol SNMP. Při definovaných změnách na zařízení, v rozvaděči (např. přepnutí stykače) je odeslán SNMP Trap na definovanou adresu. Zde je zpracován pomocí filtru v programu SNMPc a uložena příslušná informace. V sestavené vizualizaci je také možné se podívat, jaké zařízení odeslalo zprávu a zprávu si přečíst, případně reagovat. Popsaným způsoben je řešeno informování o stavech technologie, tedy binární informace (zapnuto/vypnuto). SNMP Trap zprávy mají výhodu, že je příslušné zařízení odesílá asynchronně, vždy, když dojde k určité události. Zároveň SNMP protokol umožňuje použití MIB databáze. Takto jsou přenášeny informace o analogových hodnotách, jako jsou např. proudy a napětí v primární síti, kdy naposledy byl nastartován agregát apod. Na tyto informace je nutné se příslušného zařízení zeptat pomocí standardních příkazů, které jsou rozebrány v následujícíh kapitolách.

1.1 Záložní zdroje el. energie a jejich funkce

1.1.1 Motorgenerátor a jeho funkce

Motorgenerátor se spalovacím motorem je soustrojí složené ze spalovacího motoru a generátoru na společné hřídeli. Zařízení slouží k výrobě elektrické energie. Spalovací motor může být benzínový nebo naftový. Generátor může být buď dynamo, nebo alternátor.

Zařízení se používá jako zdroj elektrické energie v místech, kde není k dispozici rozvodná síť, jako špičkový zdroj, nebo jako záložní zdroj pro zajištění nepřetržité dodávky elektrické energie pro zpravidla dlouhodobější dodávky energie (hodiny až dny). Může být konstruován jako stacionární, nebo jako mobilní.

Označuje se též jako agregát (obecnější pojem), případně dle typu spalovacího motoru (nafta, benzin, zemní plyn, popř. bioplyn) jako naftový (častěji diesel), nebo jako benzínový agregát.

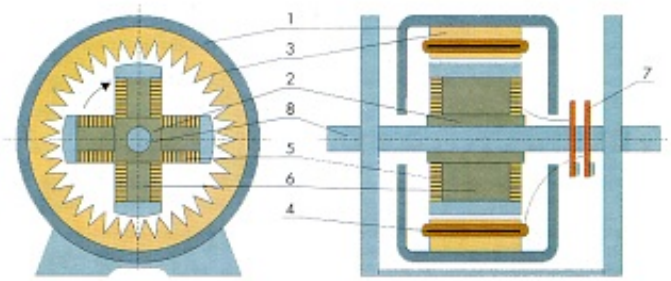
Rozsah využití motorgenerátorů je velice široký, např. jde o oblasti, kde výpadek energie může mít významně negativní dopad s možností ohrožení zdraví či života (zdravotnická zařízení, ozbrojené složky, letecká doprava, požární ventilace, globální záchranný systém při živelných pohromách). V těchto oblastech je často využití motorgenerátoru určeno normativně. Lze předpokládat dlouhodobější výpadky energie, např. v energeticky nadměrně zatížených provozech s možností negativního dopadu výpadku energie na materiální hodnoty a to jak výrobního zařízení, tak zpracovávané produkce při přerušení činnosti (výrobní či obchodní).

Výpadek může mít nepříznivý vliv na dobré obchodní jméno při poskytování služeb zákazníkovi (obchodní domy, banky, ...). Pokrytí odběrových špiček je další oblastí použití motorgenerátorů, protože při pravidelném překračování limitu energetického odběru (čtvrhodinové maximum určené smlouvou s rozvodnými závody) vznikají vysoké finanční náklady - penále. Motorgenerátor je zapojen tak, aby tento úsek nárazové potřeby e-nergie svým provozem překlenul.

Motorgenerátor může být vybaven externí nádrží s automatickým přečerpáváním paliva, čímž se prodlužuje doba chodu zařízení.

1.1.1.1 Princip činnosti

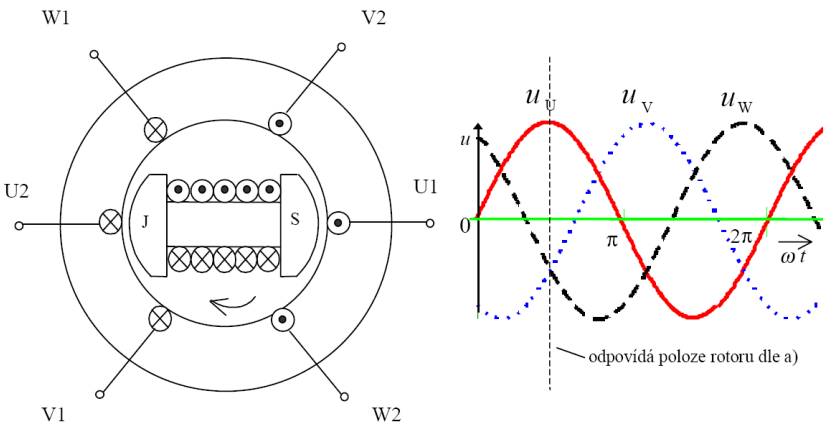
Spalovací motor vytvoří točivý moment. Ten je přenášen rotační spojkou na alternátor. Celé soustrojí motorgenerátoru převádí kinetickou energii na energii elektrickou. Budícím vinutím prochází stejnosměrný proud přiváděný přes sběrací kroužky, vzniká točivé magnetické pole, které ve statorovém vinutí indukuje třífázové střídavé napětí s kmitočtem úměrným otáčkám hřídele a počtu pólů.



Obrázek 1.1: Konstrukce synchronního stroje

Konstrukce dle obr. 1.1 obsahuje následující části: stator (1), rotor (2), magnetický obvod statoru (3), statorové vinutí (4), rotorové vinutí (5), póly (6), sběrací kroužky (7), hřídel (8).

Ve statoru alternátoru, který se podobá dutému válci, je magnetický obvod složený z plechů. Plechy se izolují lakem nebo zvláštním druhem papíru. Jsou v nich vytvořeny chladicí kanály, kudy vzduch nebo jiný plyn odvádí z magnetického obvodu teplo. Na vnitřním obvodu plechů jsou drážky s měděnými vodiči, které vytvářejí trojfázové vinutí. Začátky vinutí jsou připojeny na svorky alternátoru, odkud se střídavý elektrický proud odebírá a vede se do rozvodny a dále ke spotřebitelům. Konce vinutí jsou spojeny do uzlu. Jednotlivé vodiče jsou izolovány.



Obrázek 1.2: Princip synchronního stroje

Pro dvojpólový alternátor platí:

$$\omega \cdot \mathrm{T} = 2\pi \qquad \Rightarrow \qquad \omega = \frac{2\pi \cdot n}{60}\,, \tag{1.1}$$

kde $\mathrm{T} = \frac{60}{n} \quad \rightarrow \quad f_1 = \frac{n}{60}$. Otáčky rotoru synchronního stroje jsou označovány jako n.

Pro obecný p-pólový alternátor platí:

$$f_1 = p \cdot \frac{n}{60}\,, \tag{1.2}$$

kde f_1 je frekvence indukovaného napětí do statoru (kotvy[2]).

Připojí-li se ke svorkám vinutí statoru (kotvy) trojfázová zátěž (zátěž alternátoru), vinutím statoru bude procházet střídavý elektrický proud, jehož směr je dán Lenzovým zákonem. Tím, že vinutím statoru prochází střídavý trojfázový proud, vzniká podobně jako u asynchronního stroje točivé magnetické pole s otáčkami n_{s1}, které má stejnou rychlost otáčení jako rotor a jeho magnetické pole. Skluz stroje je tedy nulový a stroj se nazývá synchronní. Platí

$$n_{\mathrm{s1}} = n, \quad s = 0, \quad f_1 = p \cdot \frac{n_{\mathrm{s1}}}{60} \tag{1.3}$$

Synchronní alternátor má na statoru trojfázové vinutí a na rotoru tzv. budicí vinutí. Jestliže pohon otáčí rotorem a v jeho budicím vinutí prochází stejnosměrný proud, vzniká točivé magnetické pole, které v trojfázovém vinutí statoru vyvolá (indukuje) trojfázové střídavé napětí. Druhé točivé magnetické pole vyvolá střídavý proud, který začne procházet trojfázovým vinutím statoru při připojení alternátoru ke spotřebiči. Stroj se nazývá synchronní, protože se obě točivá magnetická pole otáčejí se stejnými otáčkami.

Alternátory se podle zařízení, které je pohání, dělí na turboalternátory, alternátory poháněné spalovacími motoryhydro a alternátory, které jsou pomaloběžné s velkým průměrem a malou délkou rotoru. Rotor pracuje většinou i jako setrvačník, aby vyrovnával nerovnoměrnosti způsobené chodem pístových motorů.

1.1.1.2 Provedení a regulace

Motorgenerátory lze rozdělit z pohledu dílenského provedení na:

- compact - provedení nekryté, neodhlučněné
- silent - provedení s kapotáží, odhlučněné
- super silent - provedení s kapotáží, speciálně odhlučněné
- provedení v kontejneru - kryté provedení, určené pro venkovní umístění
- mobilní, stacionární

[2]kotva - místo, kde se indukuje napětí

Obrázek 1.3: Ukázka motorgenerátorů

Silové výstupní obvody ve spojení s řídicí automatikou zajišťují především automatické přepínání mezi sítí dodavatele elektrické energie a motorgenerátorem a také elektrické a mechanické blokování za účelem ochrany sítě dodavatele proti dodávce elektrické energie při činnosti motorgenerátoru.

Návrat k napájení zátěže z rozvodné sítě po obnově dodávky napětí lze provést třemi základními způsoby:

1. bez zpětné synchronizace
2. se zpětnou synchronizací
3. s postupným předáváním zátěže

1.1.1.3 Návrat k síti

Při návratu sítě motorgenerátor **bez zpětné synchronizace** určitou dobu dále napájí zátěž a sleduje kvalitu sítě. Pokud jsou po definovanou dobu parametry sítě v definovaných tolerancích, řídicí automatika odpojí stykač napájení zátěže z motorgenerátoru a s definovaným zpožděním připojí zátěž na napájecí síť. Zátěž je v tomto typu řízení motorgenerátoru během výše uvedeného definovaného zpoždění bez napájení. Při tomto typu přepínání není provedena synchronizace chodu motorgenerátoru se sítí.

Při návratu sítě motorgenerátor **se zpětnou synchronizací** určitou dobu dále napájí zátěž a sleduje kvalitu sítě. Pokud jsou po definovanou dobu parametry sítě v definovaných tolerancích, řídicí jednotka provede synchronizaci motorgenerátoru se sítí a připojí síť na zátěž. Po definovaném čase (cca 0,5 s - nastavitelné) řídicí automatika odpojí motorgenerátor od zátěže. Po výše uvedený čas je motorgenerátor připojen paralelně k síti. Při návratu sítě je tedy zátěž napájena bez přerušení.

Při návratu sítě motorgenerátor **s postupným předáváním zátěže** určitou dobu dále napájí zátěž a sleduje kvalitu sítě. Pokud jsou po určenou dobu parametry sítě v definovaných tolerancích, řídicí jednotka provede synchronizaci motorgenerátoru se sítí, připojí síť na zátěž a postupně předá napájení zátěže na rozvodnou síť. Tento proces trvá cca 10 s a je při něm zajištěno, že nedojde ke zpětnému toku proudu do sítě (zpětnému

napájení). Přejímání zátěže probíhá bez proudových rázů. Parametry přejímání zátěže jsou volitelné. Pokud v době předávání zátěže dojde k výpadku sítě, motorgenerátor převezme napájení zátěže bez zpoždění. Při návratu sítě je tedy zátěž rovněž napájena bez přerušení.

Ve všech případech motorgenerátor po definovanou dobu (cca 5 min) dále běží v tzv. dochlazovacím režimu a následně je vypnut a je znovu připraven k provozu. Pokud v době běhu motoru v dochlazovacím režimu (po odpojení stykače motorgenerátoru) dojde k výpadku sítě, motor s malým zpožděním přebírá napájení zátěže. Celý tento proces probíhá automaticky a nevyžaduje žádný zásah obsluhy. Testování soustrojí je vždy prováděno bezvýpadkovým způsobem.

Motorgenerátory jsou většinou vybaveny elektrickým předehřevem chladicí kapaliny dieselmotoru. To umožňuje dosáhnout 100% výkonu stroje do 10 s. Chladič uzavřeného chladicího systému je umístěn na stroji, ale lze ho umístit i mimo. U chladiče umístěného na stroji je potřeba vyřešit vhodným způsobem přívod a odvod vzduchu.

1.1.1.4 Vybavení a technologie

Motorgenerátory lze obecně rozdělit do skupin dle dodávaných výkonů, místa použití, ovládání apod. Příkladem může být následující dělení. Do první skupiny mohou patřit přenosná soustrojí ve výkonech do cca 20 kVA, existují varianty s ručním startem, elektrickým startérem. S kapotáží i bez ní. Vyznačují se nenáročným provozem a tichým chodem. Druhá skupina zahrnuje motorgenerátory s výkony do cca 2000 kVA. Třetí skupina je specifická, zahrnuje motorgenerátory se středními výkony (cca do 200 kVA). Výhodou je robustní konstrukce, místa určení jsou především práce v náročném a pracném prostředí.

Vznětové motory soustrojí bývají dnes vybaveny moderními prostředky pro dosažení vysokých výkonů. Těmi jsou např. systém Common-rail přímého vysokotlakého vstřiku nafty. Palivo vstřikované do válců pod vysokým tlakem tvoří lépe hořlavou směs, čímž se dosahuje vyšší účinnosti motoru, vyššího výkonu a točivého momentu. Důležitým faktorem je nižší spotřeba, hlučnost apod. Oproti jiným systémům je tlak paliva vytvářen nezávisle na otáčkách motoru a vstřikovaném množství paliva - díky zásobníku tlaku.

Jako alternátory se dnes používají čtyřpólové bezkartáčové jednoložiskové samobudicí a samoregulační stroje vybavené elektronickou regulací. Alternativní mají vlastní chladicí systém, bezúdržbová ložiska a jsou vybaveny automatickým elektronickým napěťovým regulátorem pro zvládnutí skokové zátěže.

Obecně lze říci, že motorgenerátor v provedení Silent oproti provedení bez kapotáže dosahuje snížení úrovně hluku minimálně o 20 dB ve vzdálenosti jeden metr od stroje v každém směru.

Ve standardním vybavení je chladič s ventilátorem poháněným od motoru, vzduchový filtr, plnoprůtokový olejový a palivový filtr, uzavřený chladicí okruh, elektrický předehřev chladicího okruhu pro hladký start, elektrický startér, mechanický nebo elektronický regulátor otáček motoru, společný ocelový rám pro motor i generátor s antivibračními elementy a zvedací oka pro transport. Standardem je též vybavení kompletním řídicím systémem. Ten obsahuje voltmetr, ampérmetr, měřič frekvence, počitadlo motohodin, automatiku kontroly motoru, zásuvky a svorkovnice.

Správný výkon soustrojí je určován více faktory. Nejdůležitější informací je velikost, charakter a chování napájené zátěže. Smíšená zátěž se obvykle sestává ze spotřeby převážně činného charakteru (žárovkové osvětlení), induktivních spotřebičů (motory výtahů, čerpadel, ventilátorů) a v neposlední řadě bývají na straně MTG zátěží i UPS. Pro stanovení jmeno-vitého výkonu je nutné znát i rozběhové proudy a účiníky alespoň nejvýznamnějších spotřebičů. Podstatný je i činitel harmonického zkreslení vstupního proudu THDI a vlastnost soustrojí, která určuje toleranci náhradního zdroje vůči skokově připojené zátěži. Pamatovat je nutné také na symetrii zatížení.

Dalším parametrem ovlivňujícím volbu systému energocentra je vstupní Power Factor. Obvyklá hodnota pro velké systémy je cca 0,8 a lze dosáhnout až hodnotu 1. Pro nižší hodnoty Power Factoru je nezbytný zvýšený výkon generátoru.

Obrázek 1.4: příklad motorgenerátorového zdroje 2000 kVA

1.1.1.5 Provozní módy záložních motorgenerátorových zdrojů

Ostrovní provoz je provozem jednoho zdroje v místech mimo dosah veřejné distribuční sítě. Motorgenerátor slouží jako jediný zdroj elektrické energie. Řídicí systém zajišťuje především dohled nad všemi technologickými parametry soustrojí. Náhradní zdroj obvykle pracuje v manuálním provozu.

Záskok s dvojím přerušením se používá všude, kde technologická potřeba vyžaduje náhradu elektrické energie z veřejné sítě záskokovým zdrojem při jejím selhání. Vedle technologických parametrů vlastního motorgenerátoru monitoruje řídicí systém i parametry distribuční sítě. Náhradní zdroj obvykle pracuje v automatickém režimu. Po výpadku standardního zdroje, na základě nastavených parametrů automatika rozhodne o chování a startu soustrojí. Po nastartování přebírá motorgenerátor spotřebu chráněné technologie. Toto řešení je charakteristické dvěma přerušeními v dodávce elektrické energie. Poprvé v okamžiku kdy dojde k selhání hlavního zdroje a podruhé při návratu sítě. První výpadek je způsoben nenadálou poruchou a lze jej očekávat kdykoli. Doba přerušení je dána nas-

tavitelnou dobou rozhodování o startu soustrojí, startem a přípravou na převzetí zátěže. Druhý výpadek nastává po návratu sítě. Doba přerušení je pak dána nastavitelnou dobou rozhodování o odstavení soustrojí a časem nutným pro předání zátěže zpět síti. Přepnutí z motorgenerátoru na síť nelze provést bez jisté prodlevy, při níž jsou současně odpojeny stykač sítě i stykač motorgenerátoru. Toto řešení nepočítá se zpětnou synchronizací náhradního zdroje.

Krátkodobý chod se sítí odpovídá situaci výše popsanému stavu s tím rozdílem, že po návratu veřejné sítě dochází k synchronizaci generátoru. Oba zdroje jsou tak ve fázi a při zajištění limitace zpětné dodávky do veřejného zdroje jsou obě sítě krátkodobě propojeny. Výpadek tak nastává pouze v okamžiku náhlého síťového výpadku. Řídicí elektronika zajišťuje také synchronizaci se sítí a regulaci výkonu generátoru tak, aby byla zpětná dodávka limitována a převzetí zátěže z MTG na síť bylo měkké. V případě testovacího provozu za přítomnosti sítě generátor zátěž převezme a síti opět předá bez poruch nebo přerušení.

Paralelní chod se sítí je umožněn výše popsaným způsobem, řídicí systém pak zajišťuje řízený export/import energie do soustavy chráněné náhradním zdrojem. Výhoda tohoto řešení spočívá v možnosti připojit motorgenerátor paralelně k síti před očekávaným (plánovaným) výpadkem sítě a dosáhnout tak zcela nerušeného technologického procesu. Ekonomicky zvláště výhodnou vlastností tohoto řešení je využití soustrojí pro překlenování energetických špiček nad sjednaný energetický diagram.

Multi mode - provoz dvou a více paralelních zdrojů. Paralelní chod zdrojů je vhodný pro případy, kdy je vyžadována větší spolehlivost (redundance zdrojů), celkově větší pohotovostní výkon náhradního zdroje nebo proměnlivá, technologicky podmíněná výkonová potřeba (není nutno vždy provozovat jeden velmi výkonný ale nezatížený zdroj).

Paralelní systémy lze provozovat prakticky ve stejných funkčních modech jako samostatné motorgenerátory, od ostrovního provozu po paralelní chod soustavy se sítí. Řídicí systém plní ještě funkci výkonového managementu a sleduje stejnoměrný proběh jednotlivých strojů. Výkonový management zajistí, aby bylo vždy v chodu jen nezbytné množství soustrojí nutných pro bezproblémový chod tohoto energocentra. Soustrojí jsou tak připojována, nebo odpojována dle okamžitých a předpo-kládaných potřeb kritické zátěže, což přispívá ke stabilitě celého napájecího sytému a zároveň dokáže držet provozní náklady v předpokládaných ekonomických mezích. Paralelní řazení zdrojů lze provádět i dodatečně.

1.1.2 Zdroje UPS - Uninterruptible Power Supply

UPS jsou zařízení jejichž funkcí je zpravidla krátkodobá (minuty až hodiny) dodávka energie v případě nestability vstupního napětí či při úplném výpadku sítě. Jejich úlohou je chránit data a citlivá zařízení před poškozením vlivem nepředvídaných událostí na síti jako jsou šumy, rázy, napěťové špičky, poklesy napětí nebo úplné výpadky. Dojde-li k výpadku elektrické energie, záložní zdroj dodává spotřebiči energii ze svých akumulátorů. Vzhledem k ceně elektronických zařízení a přenášených dat jsou UPS nezbytným vybavením všech informačních systémů.

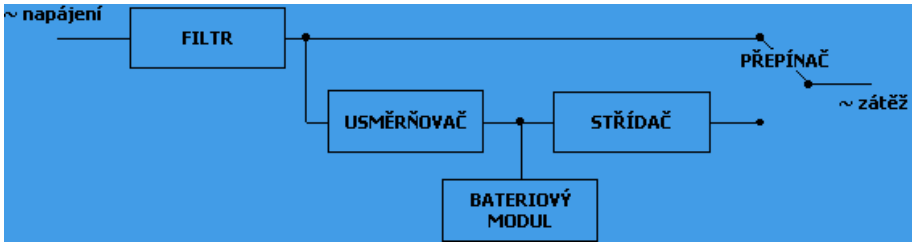
Obrázek 1.5: příklady UPS zdrojů

1.1.2.1 Princip činnosti

Existují tři základní skupiny UPS zdrojů lišících se dle metody přepnutí mezi sítí a bateriemi.

Voltage Dependent - napěťově závislý, též označovaný jako off-line. Přepíná chráněnou technologii pomocí relé na záložní měnič napájený z akumulátorů. Dochází zde ke krátkodobému (zhruba 5 ms) výpadku.

UPS v normálním (síťovém) režimu napájí zátěž přímo ze sítě, kdy parametry sítě mohou být upravovány pasivními filtry. Střídač nepracuje. Při výpadku vstupního síťového napětí se aktivuje střídač během několika ms (cca 5 ms) a zátěž je napájena z baterií. Modifikací je **Voltage** & **Frequency Dependent**. Tyto typy odstraňují problémy s výpadkem proudu a částečné rušení sítě.



Obrázek 1.6: UPS - Voltage Dependent

Voltage Independent - napěťově nezávislý, též line-interactive. Principem je přes regulační transformátor vyrovnávat mimořádné anomálie elektrické sítě.

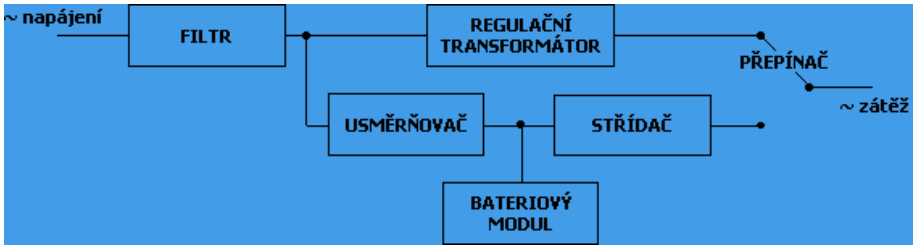
Základním rysem těchto UPS je napěťová a frekvenční nezávislost pouze v bateriovém režimu. UPS v normálním (síťovém) režimu napájí zátěž přes filtry přímo ze sítě. Parametry sítě jsou upravovány síťovým přizpůsobovacím členem tak, aby vstupní napětí bylo korigováno v závislosti na jeho hodnotě. Tato korekce může být realizována skokově (přepínání odboček na autotransformátoru) nebo lineárně (Delta konverze).

Střídač je synchronizován se sítí (line interactive), je však většinou ve stavu naprázdno. V režimu bez přítomnosti sítě je během několika ms energie dodávána střídačem z baterií.

Nejdokonalejší UPS tohoto typu mají plynulé řízení výstupního napětí a bezspínačový

přechod mezi síťovým a bateriovým režimem (Delta konverze).

Tento typ odstraňuje nedostatky jako jsou výpadek proudu, napěťové rázy, pokles napětí, podpětí a přepětí.

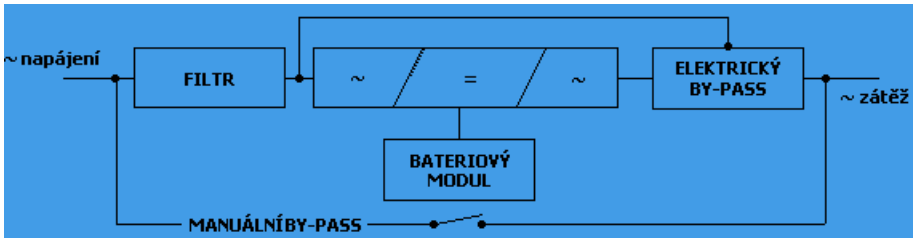


Obrázek 1.7: UPS - Voltage Independent

Voltage Frequency Independent - napěťově a frekvenčně nezávislý, též true on-line double conversion. Principem je napájení spotřebiče prostřednictvím měniče trvale z akumulátorů, které jsou současně dobíjeny ze sítě. UPS provádí stabilizaci a filtraci napětí. V případě výpadku či poklesu napětí dodávají akumulátory energii bez jakéhokoliv přerušení.

Zajišťuje napěťovou a frekvenční nezávislost ve všech režimech. V normálním (síťovém) režimu je vstupní napětí usměrněno a touto energií se napájí střídač. V případě potřeby jsou baterie dobíjeny.

V zálohovacím režimu je napájen střídač z akumulátorů. Všechny UPS jsou vybaveny automatickým by-passem. Základním znakem těchto UPS je trvalá napěťová a frekvenční nezávislost na vstupním napětí (mimo režim by-pass). Umožňují mimořádně přesnou stabilizaci výstupního napětí a kmitočtu. Střídač je trvale v provozu a energie přechází dvěma přeměnami AC/DC/AC - což je někdy uváděno jako dvojí konverze. Tento typ odstraňuje nedostatky typu výpadek proudu, napěťové rázy, pokles napětí, podpětí a přepětí, rušení, transientní a harmonické[3] zkreslení, frekvenční kolísání.



Obrázek 1.8: UPS - Voltage and Frequency Independent

[3]způsobeno nelinearitou - vznikají vyšší harmonické složky, liché se projevují jako rušení

1.2 Přepínací, monitorovací prvky a jejich funkce

1.2.1 Motorový přepínač SOCOMEC ATyS

Motorové přepínače řady ATyS jsou 3- nebo 4-pólové přepínače dálkově ovladatelné pomocí beznapěťových kontaktů. Jsou kombinací dvou odpojovačů zátěže vzájemně mechanicky a elektricky blokovaných (montáž „zády k sobě“). Umožňují přepojení napájecí sítě a přepnutí pod zátěží dvou napájecích obvodů při dodržení bezpečné izolace. Při přepínání se může nacházet ve třech stabilních polohách označovaných I, 0, II.

Obrázek 1.9: Ukázka výkonového přepínče SOCOMEC ATyS

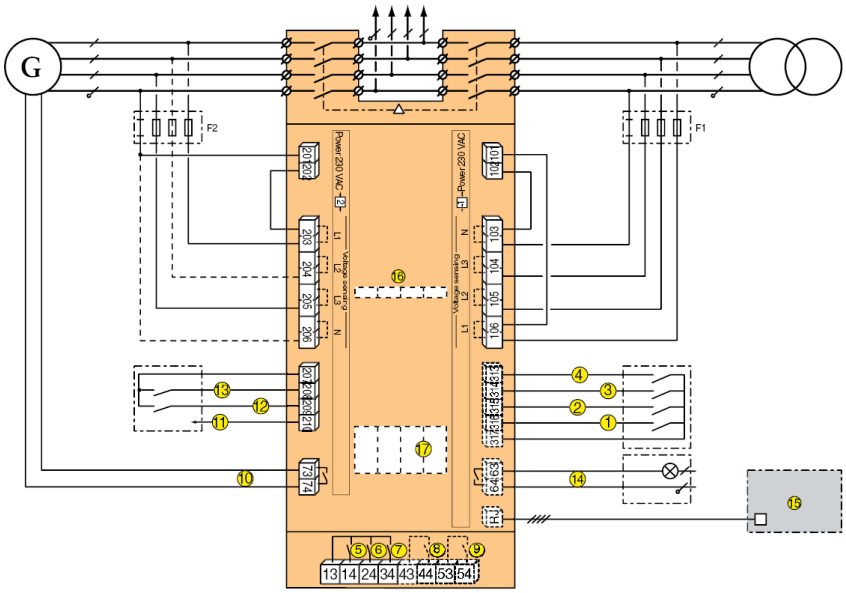
Přepínače jsou vybaveny motorovým natahováním pružiny, která poté přemístí kontakty. Tento způsob je zvolen s ohledem na co nejvyšší rychlost přepnutí. Tím se eliminuje vznik oblouku, který poškozuje kontakty opalováním. Všechny přístroje mají rovněž možnost ručního ovládání pomocí dodávané páky. Bezobslužný provoz zajišťuje elektronicky řízený modul motorizovaného ovládání:

- dálkově řízený ... ATyS 3 je řízen beznapěťovými kontakty, umožňujícími operaci přepnutí mezi polohou I, 0 nebo II, z externí řídicí logiky,

- automaticky řízený ... ATyS 6 zahrnuje kontrolní relé, časovače a testovací funkce k zajištění přepínání Normální / Záložní síť.

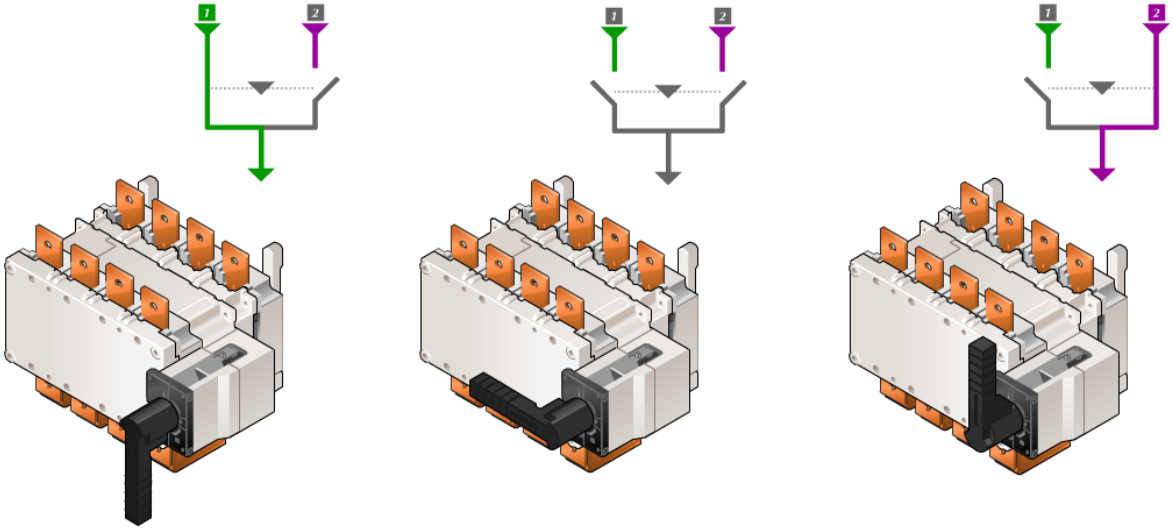
Informace o polohách, přítomnosti zdroje, napětí a kmitočtu jsou zobrazeny přímo na přístroji. Jako příslušenství slouží také kontrolní panel dálkového ovládání modul vestavitelný do dveří rozvaděče. K připojení se využívá konektor RJ45. Po jeho připojení se znepřístupní funkce čelního panelu na přepínači.

Jako další příslušenství je možné použít komunikační modul rozhraní RS-485 s protokolem JBUDS/MODBUS.



1: řízení položky 0
2: řízení položky I
3: řízení položky II
4: znepřístupnění řízení poloh (0-I-II)
5: pomocný kontakt, sepnutý v případě že přepínač je v poloze I
6: pomocný kontakt, sepnutý v případě že přepínač je v poloze II
7: pomocný kontakt, sepnutý v případě že přepínač je v poloze 0
8: pomocný kontakt, sepnutý v případě že přepínač je v režimu "AUT"
9: pomocný kontakt, sepnutý v případě že přepínač je uzamčený zámky
10: povel ke spuštění záložního zdroje
11: pomocné napájení (pro doplňkové moduly)
12: externí vstup "testování pod zátěží"
13: blokovací vstup DTT se aktivuje okamžotě po jeho sepnutí když DDT = maximální hodnota
14: chybový výstup
15: rozhraní pro externí ovládání
16: proudové transformátory
17: zásuvky pro doplňkové moduly ATyS 6e a ATyS 6m

Obrázek 1.10: Zjednodušené schéma připojení



Obrázek 1.11: Mechanismu přepínání u SOCOMEC ATYSu

1.2.2 Automatický přepínač sítí LOVATO ATL

Jedná se o řídicí jednotku pro ovládání výkonových přepínačů. Modul je určen pro automatické přepínání mezi hlavní sítí a záložním napájením. Přístroj je v panelovém provedení v krytu a má dva výstupy automatické a / nebo manuální ovládání stykačů nebo jističů s motorovým pohonem. Pro napájení kontroleru je možné zvolit variantu na 230 V, nebo 12 / 24 V. Pro sledování sítě jsou k dispozici vstupy pro tři fáze a nulový vodič. Na čelním panelu je možné sledovat fázové a sdružené napětí a stav stykačů. Modul disponuje čtyřmi pracovními režimy (OFF, MANUAL, AUTO, TEST). Konfigurace se provádí pomocí dodaného softwaru připojením na rozhraní RS-232. Varianta ATL-30 disponuje i rozhraním RS-485 s protokolem MODBUS.

Obrázek 1.12: Ukázka automatického přepínače sítí ATL

1.2.3 Multifunkční měřicí přístroj SOCOMEC DIRIS A4x

Přístroje DIRIS A40 a A41[4] jsou multifunkční měřicí přístroje určené pro měření v sítích nízkého a vysokého napětí. Veškeré parametry lze zobrazit na displeji umístěném na čelním panelu, některé parametry lze nastavit (např. převodní poměr proudového transformátoru). Na panelu se zobrazují rovněž měřené hodnoty a časové funkce. Přístroj lze snadno rozšířit pomocí dodatečných modulů s doplňkovými funkcemi pro

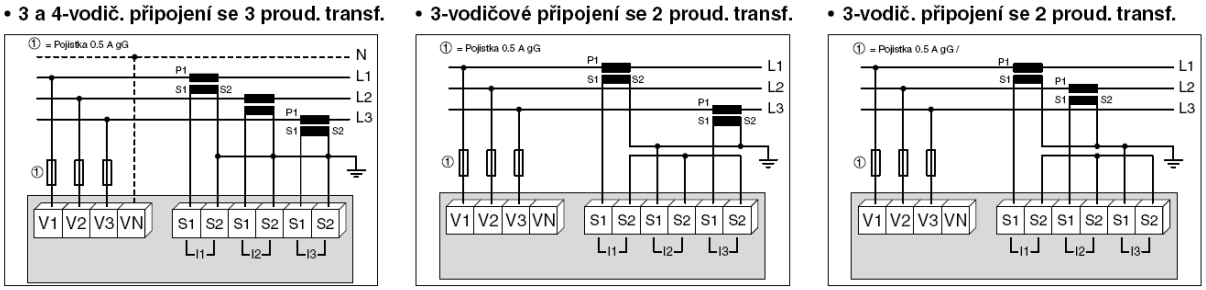
- měření spotřeby
- měření a zpracování informací o harmonických složkách
- signalizaci alarmů
- signalizaci alarmů přes RS-485 s protokolem JBUS/MODBUS

[4] oproti A40 umožňuje měřit proud i nulovým vodičem třífázové soustavy v zapojení do hvězdy

Obrázek 1.13: Ukázka multifunkčního měřicího přístroje DIRIS

Přístroj DIRIS pomocí 6 tlačítek na předním panelu nahradí několik jednofunkčních analogových nebo digitálních měřících přístrojů jako jsou panelové voltmetry, ampermetry nebo měřidla výkonu. Navíc umožňuje centralizovat tyto parametry na PC nebo PLC pomocí komunikace RS-485 a použití protokolu JBUS/MODBUS nebo PROFIBUS-DP. Rozměry a design přístrojů jsou přizpůsobeny pro jejich snadnou montáž do dveří rozvaděčů. Funkci přístrojů lze snadno rozšířit pomocí zásuvných modulů, tyto moduly lze doplňovat postupně, dle potřeb uživatelů.

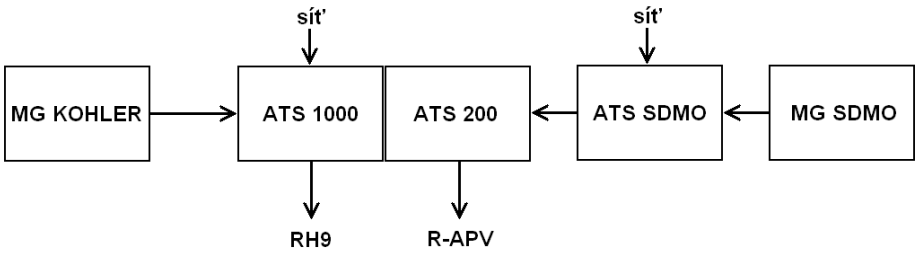
Měření okamžitých efektivních hodnot(TRMS) zahrnuje běžný proud fází, průměrné a maximální hodnoty ve volitelném časovém rozmezí, fázová a sdružená napětí, kmitočet, činný fázový výkon ve 4 kvadrantech, jalový fázový výkon ve 4 kvadrantech, zdánlivý fázový výkon, účiník (PF) jednotlivých fází s indikací induktivní nebo kapacitní, míra harmonických složek (THD) až do 49. harmonické. Měření celkové spotřeby, činné spotřeby ve 4 kvadrantech, jalové spotřeby ve 4 kvadrantech, zdánlivé spotřeby, provozní hodiny s rozlišením 1/100 hod. pro měření provozní doby.



Obrázek 1.14: Schémata zapojení

1.3 Obecné požadavky na řídicí systém

Celá řízená část se skládá ze dvou závislých celků, jak ukazuje následující obrázek.



Obrázek 1.15: Blokové schéma celého systému

Základním prvkem celého systému zálohování je motorgenerátor MG KOHLER, který je primárním záložním zdrojem a jeho výkon má při dnešní spotřebě rezervu. Standardní cesta napájení je z primární sítě přes rozvaděč ATS 1000 do pole nové rozvodny RH 9, odkud je dále rozváděno. Samostatnou větev napájení tvoří R-APV. Jedná se o nejdůležitější část technologie a záloha je zde dvojnásobná. Standardně je tato část napájena z primární sítě. Pokud síť vypadne, bude napájena z MG KOHLER přes ATS 1000 a ATS 200. V případě poruchy budou v ATS 200 a ATS SDMO přepnuty stykače do takové polohy, aby napájení R-APV šlo z MG SDMO.

1.3.1 Řízení rozvaděčů ATS 1000 a ATS 200

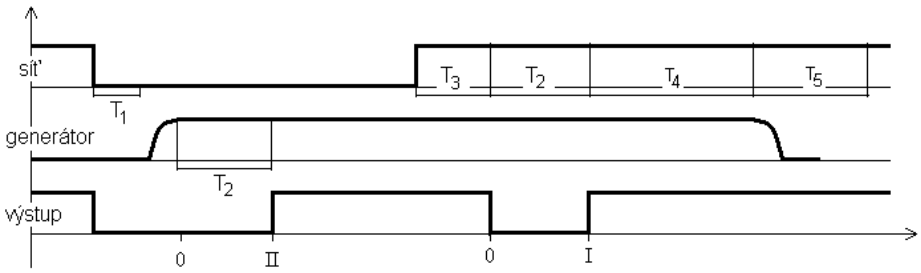
Základním prvkem řídicího algoritmu je tzv. přepínací schéma. V projektu není řešeno přifázování motorů obsažených v technologii (čerpadla, ...). Oproti tomu je použit jednodušší princip, kdy při výpadku napájení je přepnut výkonový přepínač SOCOMEC ATYS do tzv. nulové polohy (0) a zde setrvává určitý čas. V tomto čase by mělo dojít k zastavení točivých strojů a vybití indukčností. Tato doba je závislá na prvcích použitých v technologii a je nutné ji nastavit individuálně v každé aplikaci. Až po uplynutí této doby je možné provést druhou fázi přepnutí.

Jedná se zde o přepínání proudů v řádu 10^2 A, proto je samotný přepínač vybaven motorickým natahováním pružiny, která přesune v co nejkratší době (s potlačním jiskření) kontakty do nové polohy. Tento pohon je ovládán logikou. V případě popisovaného řešení se jedná o pulsní logiku. Délka, tvar a amplituda pulsu jsou definovány - pravoúhlý puls amplitudy do 24V a délky minimálně 1s. Tyto údaje bylo nutné vzít v úvahu při návrhu algoritmu.

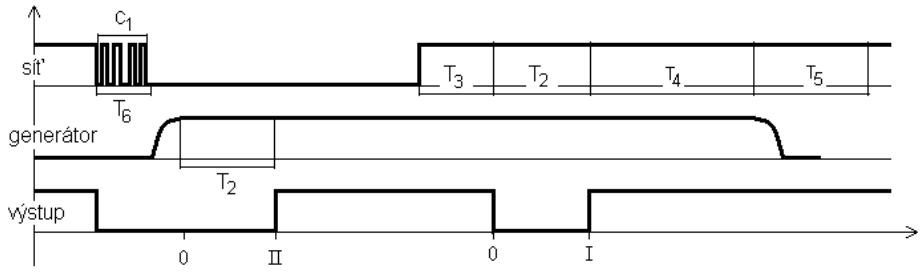
Dalším krokem při návrhu je první spuštění programu (např. po resetu PLC). Musí být zajištěny následující požadavky. Je-li přítomna primární síť, nastavit výkonové přepínače SOCOMEC ATYS do polohy I. Při náběhu programu a současném napájení z primární sítě nesmí dojít k manipulaci s kontakty. Pokud by byl realizován algoritmus, který by zajistil přepnutí do nulové polohy a opětovné přepnutí na síť, došlo by k výpadku

napájení technologie a zároveň ke zbytečnému přepínaní mezi kontakty a tím i k jejich opotřebování. Samozřejmostí je dodržení přepínacího schématu. Pokud nebude přítomna primární síť, bude se čekat nastavený čas (může být shodný s časem T_1) a pokud v této době nebude přítomna síť provede se cyklus ošetření výpadku.

Pokud primární síť na začátku bude přítomna, program bude testovat síť na výpadek. Je požadována reakce na dva typy výpadků. Prvním (viz obr. 1.16) je **dlouhodobý výpadek**. Trvá delší čas než je doba T_1. Ve většině případů se nastavuje v závislosti na připojené technologii v rozmezí 10 ÷ 30s. Druhým typem výpadku (viz obr. 1.17) je určitý počet C_1 **opakovaných krátkodobých výpadků** za nastavenou dobu T_6. Počet výpadků i doba T_6 se liší v závislosti na charakteru a kvalitě primární sítě a připojené technologii. Platí ovšem, že $T_6 \gg T_1$.



Obrázek 1.16: Dlouhodobý výpadek



Obrázek 1.17: Krátkodobý opakovaný výpadek

Po zjištění výpadku je startován motorgenerátor. Určitou dobu trvá, než na jeho výstupu je přítomno napětí požadované kvality. Je vhodné poznamenat, že v tomto momentě není k dispozici napájecí napětí, proto nelze přepnout výkonové přepínače. Na obr. 1.16 a obr. 1.17 je tato doba naznačena logaritmickou křivkou (generátor) před dobou T_2. V momentě, kdy je napětí požadované kvality, lze začít s přepnutím přepínače SOCOMEC ATYS. Nejprve do nulové polohy. Zde se setrvává dobu T_2. Ta se volí dle připojené technologie, je nutné vzít v úvahu, jak rychle doběhne nejpomalejší elektrický točivý stroj a dle toho volit celkovou dobu T_2. Význam této doby je popsán výše. Po jejím uplynutí je možné v přepnutí pokračovat a nastavit přepínač do polohy II.

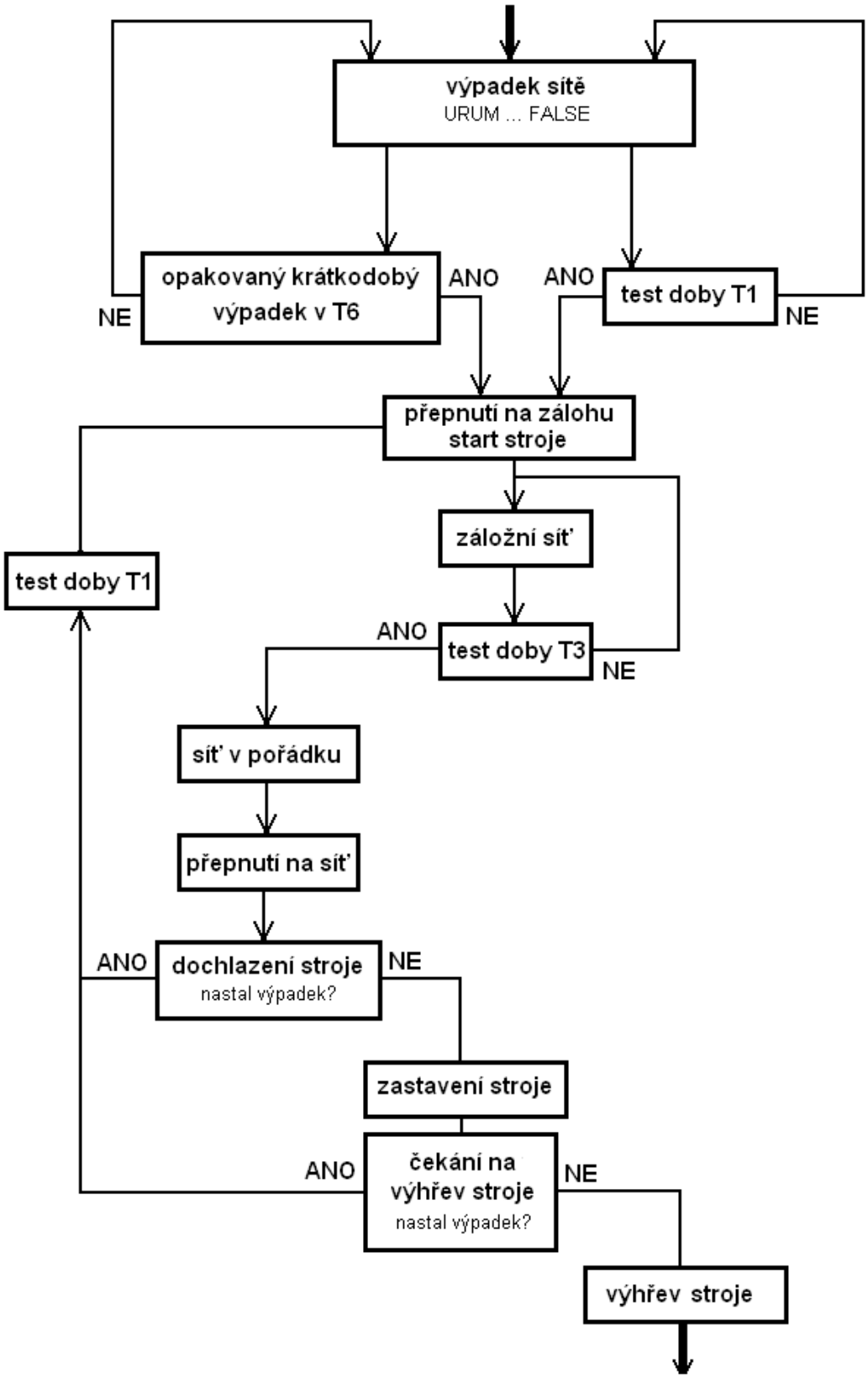
V této poloze je dodávána elektrická energie z motorgenerátoru. Řídící program testuje primární síť. V případě, že je vyhodnocena přítomnost primární sítě, je spuštěno testování kvality sítě po dobu T_3. Zde musí být uplatněna následující podmínka. Pokud z jakéhokoliv důvodu bude primární síť ztracena, při jejím znovuobjevení se musí čas T_3 začít počítat znovu. Tato podmínka vychází ze skutečnosti, kdy primární síť může po náběhu ještě kolísat. S opětovným přepojením technologie je nutné vyčkat až do doby, kdy odezní počáteční problémy s „nárazy“ způsobené náběhem sítě a okamžitým připojením mnoha zařízení.

Dále se na obr. 1.16 a obr. 1.17 vyskytuje časový interval T_4. Ten je možné nazvat **časem pro dochlazení stroje**. Vznětový motor při práci do zátěže vyvíjí kromě mechanické práce také teplo. To je odváděno chladičem. Startovací teplota, nastavená na termostatu vyhřívače kolem 80 °C, je menší než je teplota vznětového motoru při chodu. Po skončení dodávky elektrické energie se nechá bežet motor bez zátěže a tím dojde k poměrně účinnému ochlazení na teplotu v blízkosti teploty vyhřívací. Při implementaci řídicího algoritmu je nutné brát v úvahu následující. Pro „velké“ agregáty může doba T_4 dosáhnout řádu minut (např. pro motor 600 kVA se obecně doporučuje $T_4 = 300$s). Pokud v této době dojde k výpadku, je nutné na něj reagovat. Mnoho jednoduchých ATS kontrolerů od renomovaných firem s touto variantou nepočítá. Jeden z hlavních požadavků na tento vyvíjený systém byl, aby v případě výpadku primární sítě v době dochlazení T_4 bylo reagováno a síť testována na výpadek. V případě výpadku poté jeho ošetření, aniž by došlo k zastavení motorgenerátoru. Pokud proběhne dochlazení v pořádku (nedojde k výpadku a je dopočítán čas T_4) lze zastavit motorgenerátor.

Posledním časovým intervalem na obr. 1.16 a obr. 1.17 je doba vyčkávání na vyhřívání stroje T_5. Tato doba následuje ihned po skončení doby dochlazení T_4. Má velký význam, vznětový motor nemusíme dochlazovat na přesnou teplotu nastavenou termostatem vyhřívače, stačí pouze začít dochlazovat a vlivem nelinearity procesu (po uvolnění zátěžě motorgenerátoru - odpojí se alternátor - bude pracovat bez zátěže a nebude vyvíjet nadměrné teplo) se motor zpočátku bude ochlazovat pomalu, ale postupně se teplota bude snižovat rychleji. Proto lze motor zastavit dříve a určitou dobu vyčkat do vyhřívání. Efekt dochlazení a vyhřívání bude velmi podobný, ale ušetří se pohonné hmoty a elektrická energie na vyhřívání. Tato doba se volí v rozmezí 60 ÷ 120s.

Tabulka 1.1: Vysvětlení časových intervalů

T_1	doba testování dlouhodobého výpadku
T_2	čekání na zastavení točivých strojů
T_3	doba testování kvality primání sítě
T_4	doba dochlazení motorgenerátoru
T_5	čekání do začátku vyhřívání
C_1	počet krátkých opakovaných výpadků
T_6	doba testování opakovaných výpadků



Obrázek 1.18: Princip ošetření výpadku

1.3.2 Řízení rozvodny

V tomto projektu je připojená technologie složena z různorodých zařízení. Jsou jimi hlavně UPS, dále obecná zátěž (charakteru kapacitní-induktivní) a další. Záložní zdroj je koncipován především pro napájení výpočetního centra.

Z několika poměrně závažných důvodů, mezi něž patří postupné proudové zatěžování jak motorgenerátoru, tak i primární sítě a také potlačení nadměrného opotřebení kontaktů vlivem jiskření z důvodů spínání velkých proudů, byl realizován tzv. prioritní systém připojování. Ten zajišťuje, že technologie je rozdělena na části s ohledem na požadavky jejich chodu v případě výpadku[5]. Zároveň s tím je rozloženo připnutí outletů na určitý, předem definovaný, časový horizont. Nedojde tedy k připnutí všech outletů najednou. To má kladný vliv nejen na opotřebení kontaktů, ale i na proudové odběrové špičky jednak ze sítě, ale také z motorgenerátoru při připínání technologie. Kdyby došlo k připnutí najednou, musí motorgenerátor dodat a pokrýt velký proudový náraz, to ovšem nepřispěje k jeho bezporuchovému chodu (musí se zvýšit otáčky, aby mohl pracovat v oblasti vyššího výkonu a zároveň je mechanicky zatěžován na hřídeli alternátorem).

Při výpadku elektrické energie dojde k automatickému odepnutí (odpadnutí) všech vývodů v rozvodně. Jednotlivým outletům byla přiřazena priorita dle jejich důležitosti, přičemž vývod s nejvyšší prioritou je označen jako 1 a vzrůstající číslo označuje nižší prioritu. Ta představuje pořadí v jakém se mají outlety připojovat. Mezi jednotlivými připojeními je ponechán časový rozestup. Vše je přehledně zpracováno v následující tabulce.

Tabulka 1.2: Vysvětlení priorit připojování outletů

vývod	priorita	čas připnutí	celkový čas
`RH_1_1`	p1	30s	30s
`RH_2_1`	p2	p1 + 20s	50s
`RH_2_2`	p3	p2 + 30s	80s

Při chodu záložního zdroje v dlouhodobém režimu (dlouhodobý výpadek el. energie) je nutné zachovat nepřetržité napájení vybraných důležitých technologií a stanovit priority vývodů, které budou při určité hladině pohonných hmot a v definovaný čas odepnuty. Systém řízeného odpínání je navržen na nepřetržitý provoz po dobu 24 hodin při dodávce el. energie do technologie s prioritou 1 (není zde uvažována varianta doplnění paliva do nádrže po dobu ošetřování výpadku - nejhorší varianta). Priorita zde opět představuje pořadí v jakém se budou outlety připojovat. Odpojování tedy bude v opačném pořadí.

[5]Dále bude pojem *část technologie* označován slovem *outlet*

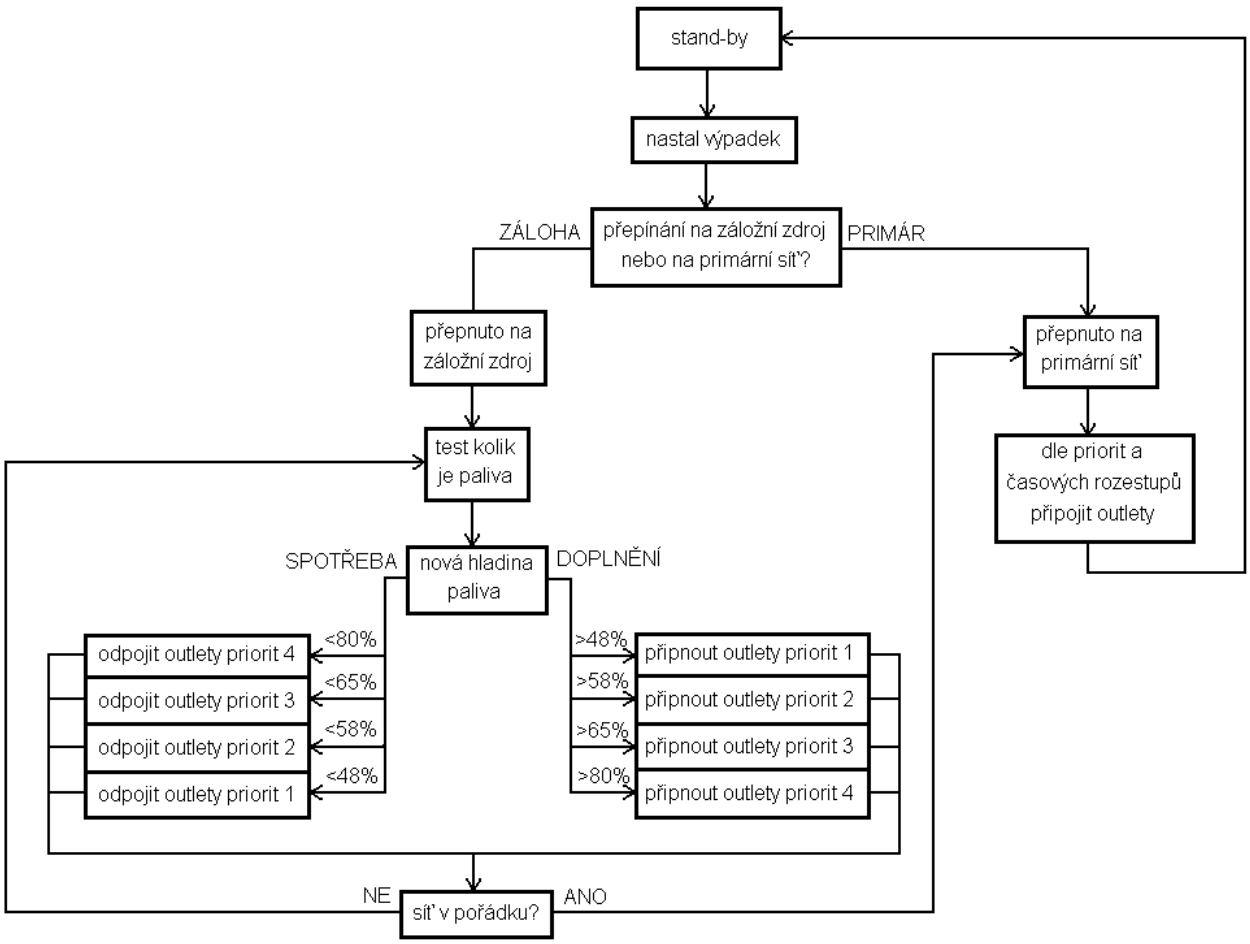
Tabulka 1.3: Odpojování outletů v závislosti na množství paliva

vývod	priorita	≥80%	≥50%	≤30%
RH_1_1	p1	I	I	I
RH_2_1	p2	I	I	0
RH_2_2	p3	I	0	0

Poslední variantou, která může nastat je, že při ošetření výpadku dojde k doplnění paliva. Tím může dojít ke změně hladiny o jeden i více stavů (vždy je však změna natolik pomalá, aby byly připnuty příslušné outlety včas). Je proto nutné uzpůsobit řídicí program, aby kromě odpojování technologií dokázal i technologie dle jejich priorit připojovat. Zde již není striktní požadavek na časové odstupňování, protože ke změně hladiny nebude docházet často[6].

Empiricky byla zjištěna četnost a délka výpadků. Zpracováním poskytnutých dat byly získány přibližné úrovně hladin a příslušné požadavky na outlety. Z toho vychází údaje v tabulce 1.3. V případě neočekávané události (např. porucha hlavního motorgenerátoru KOHLER) je záložní SDMO.

[6]Nádrž pro výkon 600kVA může mít objem 1m^3, spotřeba motoru přibližně 80l/h. Cyklus zjištění nové hladiny je v řádu 10^{1}s. I s připojením stykače příslušných outletů je zajištěna rychlost dostatečně velká oproti čerpadlu na naftu.



Obrázek 1.19: Princip připínání a odepínání outletů

Kapitola 2

Datová komunikace a použité protokoly

2.1 Ethernet

Ethernet je jeden z typů lokálních sítí, který realizuje vrstvu síťového rozhraní. Popularita spočívá v jednoduchosti protokolu a tím i snadné implementaci i instalaci.

2.1.1 Princip a formát rámce

Klasický Ethernet používal sběrnicovou topologii se sdíleným médiem. Jednotlivé stanice jsou na něm identifikovány svými hardwarovými adresami (MAC adresa). Když stanice obdrží paket s jinou než vlastní adresou, zahodí jej.

Pro přístup ke sdílenému přenosovému médiu se používá metoda CSMA/CD (Carrier Sense with Multiple Access and Collision Detection).

Stanice, která potřebuje vysílat, naslouchá, co se děje na přenosovém médiu. Pokud je v klidu, začne stanice vysílat. Může se stát (v důsledku zpoždění signálu), že dvě stanice začnou vysílat přibližně ve stejný okamžik. Jejich signály se pochopitelně navzájem sečtou. Tato situace se nazývá kolize a vysílající stanice ji poznají podle toho, že na příposlechu nosné nemají shodný signál, který vysílají. Stanice, která detekuje kolizi, vyšle krátký signál (nazvaný „jam“ o 32 bitech). Poté se všechny vysílající stanice odmlčí a později se pokusí o nové vysílání.

Původní protokol s přenosovou rychlostí 10 Mbit/s byl vyvinut firmami DEC, Intel a Xerox pro potřeby kancelářských aplikací. Později byl v poněkud pozměněné podobě normalizován institutem IEEE jako norma IEEE 802.3. Tato norma byla převzata ISO jako ISO 8802-3.

Formát rámců lokální sítě Ethernet II a IEEE 802.3 se skládá z polí

- preambule - 8 byte, střídavě binární 0 a 1, poslední byte má tvar 10101011 a označuje začátek vlastního rámce, slouží k synchronizaci, poslední byte se někdy nazývá omezovač počátku rámce (Starting Frame Delimiter, SFD),

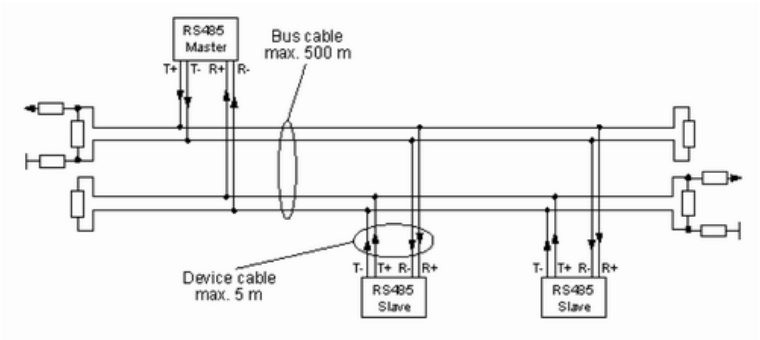
- příznak začátku rámce - 1B,
- cílová adresa - fyzická MAC adresa o délce 48 bitů, může být individuální (unicast), skupinová (multicast) a všeobecná (broadcast),
- zdrojová adresa - fyzická adresa stejného typu jako cílová, vždy individuální adresa konkrétní stanice,
- typ protokolu nebo délka
  - pro Ethernet II je to pole určující typ vyššího protokolu,
  - pro IEEE 802.3 udává toto pole délku pole dat,
- data - pole dlouhé minimálně 46 oktetů a maximálně 1500 oktetů, minimální délka je nutná pro správnou detekci kolizí,
- datová výplň - vyplní rámec, pokud je přepravovaných dat méně než 64B,
- kontrolní součet - (Frame Check Sequence, FCS) 32-bitový CRC, počítá se ze všech polí s výjimkou preambule a FCS.

úvod	SFD	cílová MAC adresa	zdrojová MAC adresa	typ / délka	data (payload)	CRC
7B	1B	6B	6B	2B	46-1500B	4B

Obrázek 2.1: Ethernetový rámec

2.2 RS-485

Jedná se o průmyslovou sériovou datovou komunikaci. Jejích služeb využívá mnoho různých průmyslových sběrnic (např. PROFIBUS-DP, MODBUS atd.).



Obrázek 2.2: Princip zapojení sběrnice RS-485

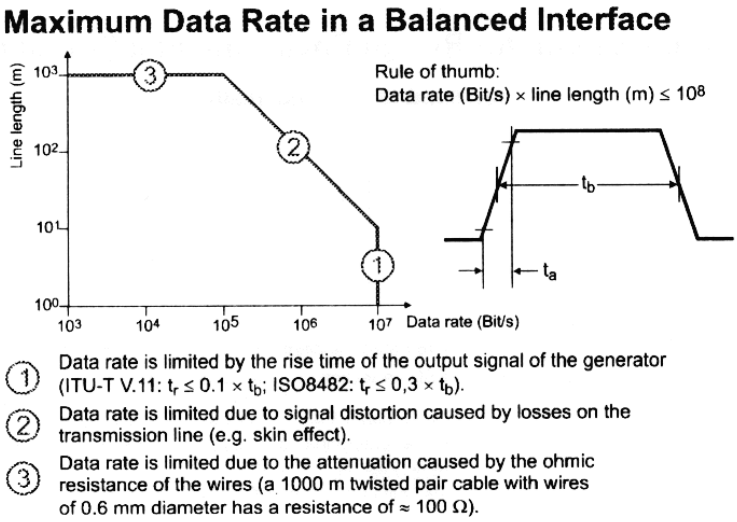
Linka RS-485 představuje klasickou sběrnici, kde lze jedním párem vodičů propojit mezi sebou až 32 zařízení, která si mají mezi sebou přenášet data. Jde o obousměrný přenos dat, ale protože nelze po jednom páru najednou přenášet data oběma směry, je zde vyžadováno řízení směru přenosu.

Běžnější je použití dvouvodičové verze sběrnice RS-485, existuje i méně známá čtyřvodičová verze, využívaná například sběrnicí Measurement Bus (DIN 66 348). Ta narozdíl od dvouvodičové nabízí obousměrnou komunikaci, tzn. není nutné se zabývat řízením směru komunikace, ale zase se u ní vyskytuje úskalí v podobě omezené vzájemné komunikace mezi zařízeními. Je totiž nutné použít komunikační model Master/Slave, kde výstup nadřazené Master jednotky je připojen na vstup všech Slave jednotek a výstupy všech Slave jednotek na vstup Master jednotky.

Terminace vedení je velice podstatná hlavně při vyšších rychlostech a delších kabelech. Hlavními důvody jsou především odrazy na konci vedení, definování úrovně na vedení a podmínka minimálního zatížení vysílače.

Pro RS-485 není terminace zcela jednoduchá. Jelikož každé zařízení komunikuje obousměrně, nejsme schopni určit, kde je vysílač a kde přijímač. To se mění podle toho, které zařízení právě vysílá. Proto je nutné na oba konce RS-485 vedení připojit 100 Ω terminátor. Protože všechny zařízení mají třístavové výstupy, nastávají situace, kdy jsou všechny vysílače ve stavu vysoké impedance a na vedení, díky terminačním odporům, není definovaný žádný stav (<200 mV). V tomto stavu je ale žádoucí aby napětí definovalo klidový stav, tedy <-200 mV. Hodnoty terminačních odporů jsou spočítány tak, aby zapojení vyhovělo maximálnímu počtu zařízení o vstupní impedanci 12 kΩ, tj. 32, na jednom vedení. Důležité je aby terminační odpory Rt byly pouze dva a to na konci kabelu (mohou být součástí posledního zařízení na vedení).

Jednotlivé vodiče jsou označeny buď jako A(-), B(+), kde A(-) označuje tzv. invertovaný vodič a B(+) neinvertovaný vodič. Logický stav 1 (někdy označený jako OFF), reprezentuje napěťový rozdíl $(A - B) < -0.3\,V$, zatímco logický stav 0 (ON) reprezentuje rozdíl $(A - B) > +0.3\,V$. Přenos pomocí rozdílového napětí eliminuje vliv naindukovaného rušivého napětí vztaženého k nulovému potenciálu, protože se na obou vodičích naindukuje stejná velikost napětí. Správný vysílač by měl generovat na výstupu úrovně +2 V a -2 V a přijímač by měl být ještě schopen rozlišit úroveň +200 mV a -200 mV jako platný signál.



Obrázek 2.3: Omezení na maximální přenosovou rychlost

Graf na obr. 2.3 ukazuje závislost přenosové rychlosti na několika základních podmínkách.

1. Maximální rychlost přenosu na krátké vzdálenosti, kde lze vliv vedení zanedbat, je dána výstupními parametry vysílače. Jedná se o dobu trvání náběžných a sestupných hran. Standard předpokládá rychlost 10Mbit/s.
2. Vedení delší než 10m. Musíme brát v úvahu ztráty způsobené kapacitami a tzv. skin efektem, při kterém se proud začíná šířit pouze po obvodu vodičů. Pravidlem pro standardní TP kabely je, že rychlost přenosu (Mbit/s) násobená délkou vedení (m) je menší než 10^8 . Například pro 100m dlouhý kabel dostáváme maximální rychlost 1Mbit/s.
3. Poslední omezení se týká velmi dlouhých vedení. Rychlost je limitovaná ohmickým odporem vedení a následným útlumem signálu. Maximální délka kabelu vyplývá z jeho odporu, který by neměl být vetší než impedance vedení, tedy 1000 Ω. Nezanedbatelná je také kapacita vedení.

Z podstaty RS-485 vyplývají jistá specifika, která musí software zajišťovat. Především se jedná o programové přidělování komunikačního média jednotlivým stanicím tak, aby nedocházelo ke vzájemným kolizím a zároveň aby byla odezva stanic dostatečně rychlá.

Je nutné přenášet data, i bitovou a bytovou synchronizaci. Pro přenos jednotlivých bitů je tedy vhodná jedna z modulací používaných v datových sítích (např. kódování NRZ[1], NRZI[2], fázová modulace NRZ, nebo diferenciální fázová modulace).

Pro zajištění bytové synchronizace máme opět několik možností. Asi nejjednodušší je definování jednoho specifického bytu jako synchronizační znaku. Lze také využít přenosových protokolů SDLC/HDLC[3], vhodných pro vysokorychlostní přenosy dat.

[1]Non Return to Zero
[2]Non Return to Zero Invert
[3]High-Level Data Link Control - detekuje chyby a řídí tok dat, synchronní, asynchronní přenosy

Ze síťového hlediska se u RS-485 jedná o sběrnicovou topologii. Jelikož Slave stanice nemají žádnou možnost, jak začít bez možné kolize vysílat, musí jim být přiděleno právo vysílat Master stanicí. Mluvíme tedy o centrálním přidělování, konkrétně přidělování na výzvu (pooling), kde se centrální stanice (Master) periodicky dotazuje všech Slave stanic, zda mají data k odeslání. Pakliže ano, dotazovaná stanice je ihned odešle. Nemá-li data k odeslání, odpoví pouze potvrzovacím paketem, nebo neodpoví vůbec. Tato metoda se jeví jako nejvýhodnější pro Multipoint zapojení.

V systémech, kde Master stanice nemá prioritní funkci, je možné použít i jiné přístupové metody. Například metoda náhodného přístupu ALOHA. Jakákoliv stanice vyšle svoje data bez ohledu na stav přenosového kanálu. Dojde-li ke kolizi, pak tato stanice nedostane potvrzení o doručení a opakuje vysílání. U RS-485 je implementována metoda ALOHA vylepšená o příposlech „nosné“. Stanice v tomto případě zahájí vysílání pouze v případě, je-li kanál v klidovém stavu. U obou těchto metod je nezbytně nutný přenosový protokol s detekcí chyb.

2.3 Protokol TCP/IP

Vzhledem ke složitosti problémů je síťová komunikace rozdělena do tzv. vrstev, které znázorňují hierarchii činností. Výměna informací mezi vrstvami je přesně definována. Každá vrstva využívá služeb vrstvy nižší a poskytuje své služby vrstvě vyšší.

Komunikace mezi stejnými vrstvami dvou různých systémů je řízena komunikačním protokolem za použití spojení vytvořeného sousední nižší vrstvou. Architektura umožňuje výměnu protokolů jedné vrstvy bez dopadu na ostatní. Příkladem může být možnost komunikace po různých fyzických médiích - ethernet, token ring, sériová linka.

Architektura TCP/IP využívá čtyři vrstevy z referenčního modelu OSI:

- aplikační vrstva (application layer)
- transportní vrstva (transport layer)
- síťová vrstva (network layer)
- vrstva síťového rozhraní LLC (logical link control)

Vrstva síťového rozhraní LLC je část spojové (linkové) vrstvy a umožňuje řídit přístup k fyzickému přenosovému médiu. Je specifická pro každou síť v závislosti na její implementaci.

Síťová vrstva MAC zajišťuje především síťovou adresaci, směrování a předávání datagramů. Obsahuje protokoly IP, ARP, RARP, ICMP, IGMP, IGRP, IPSEC. Je implementována ve všech prvcích sítě - směrovačích i koncových zařízeních.

Transportní vrstva je implementována až v koncových zařízeních a umožňuje přizpůsobit chování sítě potřebám aplikace. Poskytuje spojované (protokol TCP, spolehlivý) či nespojované (UDP, nespolehlivý) transportní služby.

Aplikační vrstva je nejvyšší vrstvou standardního modelu ISO. To jsou programy (procesy), které využívají přenosu dat po síti ke konkrétním službám pro uživatele. Příkladem

mohou být Telnet, FTP, HTTP, DHCP, DNS apod.

Aplikační protokoly používají vždy jednu ze dvou základních služeb transportní vrstvy: TCP nebo UDP, případně obě dvě (např. DNS). Pro rozlišení aplikačních protokolů se používají tzv. porty, což jsou domluvená číselná označení aplikací. Každé síťové spojení aplikace je jednoznačně určeno číslem portu, transportním protokolem a adresou počítače.

2.3.1 TCP - Transmission Control Protocol

Protokol TCP je jedním ze základních protokolů Internetu, konkrétně implementuje transportní vrstvu. Jedná se o spojově orientovaný protokol pro přenos toku bajtů. Použitím TCP mohou aplikace na různých počítačích propojených do sítě vytvořit spojení, přes které se budou přenášet data. Vytvořené spojení garantuje spolehlivé doručování dat ve správném pořadí. Protokol TCP podporuje aplikační protokoly jako např. WWW, emailu nebo SSH. V současnosti je zdokumentován v IETF RFC 793.

V sadě protokolů Internetu je TCP prostřední vrstvou mezi IP protokolem pod ním a aplikací nad ním. Aplikace ke vzájemné komunikaci využívají spolehlivé spojení na způsob roury, zatímco IP protokol poskytuje jen nespolehlivé pakety. TCP používá služby IP protokolu opakovaným odesíláním nespolehlivých paketů. Při ztrátě paketu zajišťuje spolehlivost a přeuspořádáváním přijatých paketů zajišťuje správné pořadí. Tím TCP plní úlohu transportní vrstvy ve zjednodušeném modelu ISO/OSI počítačové sítě.

Aplikace posílá proud bajtů TCP protokolu k doručení. Protokol TCP rozděluje tento proud bajtů do segmentů[4]. TCP následně předá takto vzniklé pakety IP protokolu k přepravě sítí do TCP modulu na druhé straně spojení. Dále TCP ověří, zda-li se některé pakety neztratily (každému paketu je přiděleno číslo sekvence, které se také použije k ověření správného pořadí dat).

Modul TCP na straně příjemce posílá zpět potvrzení pro pakety které byly úspěšně přijaty. Pokud by se odesilateli potvrzení nevrátilo do definované doby (RTT - Round-Trip Time), data by byla považována za ztracená a bylo by nutné je vyslat znovu. Testování na poškození dat např. šumem kanál se provádí kontrolním součtem.

K rozlišení komunikujících aplikací používá TCP protokol tzv. čísla portů. Každá strana TCP spojení má přidruženo 16 b[5] bezznaménkové číslo portu přidělované aplikacemi. Porty jsou rozčleněny do třech skupin

- dobře známé porty - přiřazovány organizací IANA (Internet Assigned Numbers Authority) a jsou vesměs používané systémovými procesy, Tyto porty užívají aplikace běžící jako servery a pasivně přijímající spojení, příkladem jsou FTP (porty 21, 20), SMTP (port 25), DNS (port 53) a HTTP (port 80),
- registrované porty - jsou typicky používané aplikacemi koncových uživatelů při otevírání spojení k serverům, také mohou identifikovat služby,
- dynamické/privátní porty - mohou být také používány koncovými aplikacemi, ale není to obvyklé.

[4]Velikost segmentů je určena parametrem MTU (Maximum Transmission Unit) linkové vrstvy sítě, ke které je počítač připojen.

[5]existuje max. 65 535 portů

2.3.2 IP - Internet Protocol

Protokol IP je datový protokol používaný pro přenos dat přes paketové sítě. Tvoří základní protokol dnešního Internetu.

Data se v IP síti posílají po blocích nazývaných datagramy. Jednotlivé datagramy putují sítí zcela nezávisle, na začátku komunikace není potřeba navazovat spojení či jinak vytvářet datovou cestu, přestože spolu třeba příslušné stroje nikdy předtím nekomunikovaly.

Protokol IP v doručování datagramů poskytuje nespolehlivou službu, označuje se také jako best effort, v překladu „nejlepší úsilí“. Tedy, všechny stroje na trase se datagram snaží podle svých možností poslat blíže k cíli, ale nezaručují prakticky nic. Datagram vůbec nemusí dorazit, nebo naopak může být doručen několikrát a není ani zaručeno pořadí doručených paketů. Pokud aplikace potřebuje spolehlivost, je potřeba ji implementovat v jiné vrstvě síťové architektury, typicky TCP protokol, který je bezprostředně nad IP. Pokud by síť často ztrácela pakety, měnila jejich pořadí nebo je poškozovala, výkon sítě pozorovaný uživatelem by byl malý. Na druhou stranu příležitostná chyba nemívá pozorovatelný efekt. Navíc se obvykle používá vyšší vrstva, která ji automaticky opraví.

2.3.2.1 Směrování, verze IP, IP adresy a jejich struktura

Každé síťové rozhraní komunikující prostřednictvím IP má přiřazeno jednoznačný identifikátor, tzv. IP adresu. V každém datagramu je pak uvedena IP adresa odesílatele i příjemce. Na základě IP adresy příjemce pak každý počítač na trase provádí rozhodnutí, jakým směrem paket odeslat - tzv. směrování. Na starosti to mají specializované stroje označované jako směrovače (routery).

Dnes se nejčastěji používá verze označovaná číslem 4 (nazývaná IPv4), obsahující 32 b adresy. Navrhovaným nástupcem je IPv6. V internetu pozvolna docházejí adresy a IPv6 má krom jiného adresy 128 b, které poskytují větší adresní prostor (IPv4 má miliardy adres, IPv6 stovky sextiliónů).

Adresa IP je jednoznačná identifikace konkrétního zařízení v prostředí Internetu. Veškerá data posílaná sítí obsahují IP adresu odesilatele i příjemce. Protože by pro běžné uživatele počítačových sítí bylo velice obtížné pamatovat si číselné adresy, existuje systém specializovaných počítačů DNS (Domain Name System), které převádějí doménová jména na IP adresy a opačně.

Adresa IPv4 je 32 b číslo, zapisované po jednotlivých bajtech desítkově, oddělených tečkami, např. 192.168.1.10. Takových čísel existuje celkem $2^{32} \approx 4 \cdot 10^9$. Určitá část adres je ovšem rezervována pro vnitřní potřeby protokolu a nemohou být přiděleny. Dále pak praktické důvody vedou k tomu, že adresy je nutno přidělovat hierarchicky, takže celý adresní prostor není možné využít beze zbytku.Nedostatek adres se řeší různými způsoby:

- dynamickým přidělováním - každý uživatel dostane dočasnou IP adresu, při příštím připojení může tentýž uživatel dostat úplně jinou adresu

- překladem adres - (NAT - Network Address Translation), ke správě tohoto přidělování slouží specializované síťové protokoly, jako např. DHCP[6].

Adresa IPv4 se dělí na tři základní části - adresa sítě, adresa podsítě a adresa počítače. Koncepce adresy má velký význam pro směrování – mimo cílovou síť se směruje podle adresy sítě, a až když je IP datagram doručen do ní, začíná se brát ohled i na detailnější části adresy.

Adresu sítě přiděluje poskytovatel připojení - lokální registrátor. Strukturu lokální části adresy (zda bude rozdělena na podsítě, jaká část bude věnována adrese podsítě a jaká adrese počítače) určuje správce sítě. Ten také přiděluje adresy.

Hranici mezi adresou podsítě a počítače určuje maska podsítě. Jedná se o 32 b hodnotu zapisovanou podobně jako IP adresa. V binárním tvaru obsahuje jedničky tam, kde se v adrese nachází síť a podsíť, a nuly tam, kde je počítač. Maska podsítě je společně s IP adresou součástí základní konfigurace síťového rozhraní, obvykle se předává protokolem DHCP.

V začátcích Internetu bylo rozdělení adresy na síť a lokální část fixní. Prvních 8 b adresy určovalo síť, zbytek pak stroj v síti. To však umožňovalo pouze 256 sítí a v každé mohlo být přes 16 milionů stanic. Nyní je možno předěl mezi adresou sítě a lokální částí adresy umisťovat libovolně. Daná adresa se pak značí kombinací prefixu a délky ve formě 192.168.24.0/21 což znamená, že tato síť je určena prvními 21 b adresy, zbytek je adresa stanice. Tato síť používá rozsah adres 192.168.24.0 – 192.168.31.255.

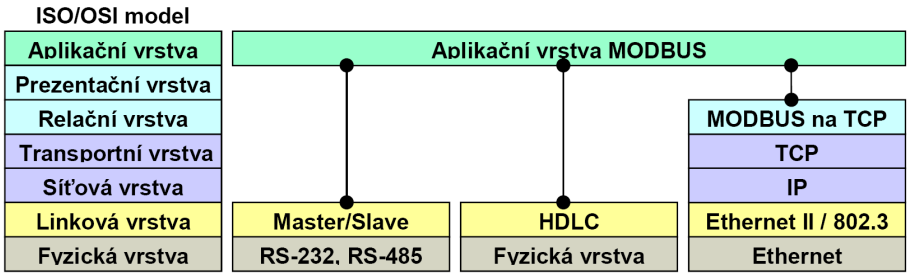
Nejnižší adresa v síti (s nulovou adresou stanice) slouží jako označení celé sítě, nejvyšší adresa v síti slouží jako adresa pro všesměrové vysílání (broadcast), takové adresy tedy nelze použít pro normální účely. Adresy 127.x.x.x (tzv. localhost, nejčastěji se používá adresa 127.0.0.1) jsou rezervovány pro tzv. loopback, logickou smyčku umožňující posílat pakety sám sobě. Dále jsou vyčleněny rozsahy tzv. interních (neveřejných) IP adres, které se používají pouze pro adresování vnitřních sítí (např. lokálních)

- ve třídě A: 10.0.0.0 až 10.255.255.255 (celkem 16 777 214 adres)
- ve třídě B: 172.16.0.0 až 172.31.255.255 (celkem 16krát 65 534 adres (tj. celkem 1 048 544))
- ve třídě C: 192.168.x.0 až 192.168.x.255 (celkem 256krát 254 adres)

[6]Dynamic Host Configuration Protocol umožňuje nastavit všem stanicím sadu parametrů nutných pro komunikaci včetně parametrů doplňujících a uživatelsky definovaných, významným způsobem tak zjednodušuje a centralizuje správu počítačové sítě, parametry nastavitelné pomocí DHCP (IP adresa, maska sítě, brána, DNS servery)

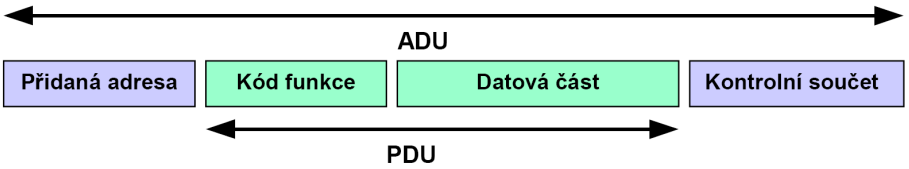
2.4 Protokol MODBUS

Jedná se o otevřený komunikační protokol na úrovni aplikační vrstvy ISO/OSI modelu, umožňující komunikaci typu klient-server mezi zařízeními na různých typech sítí a sběrnic. Vytvořen byl v roce 1979 francouzskou firmou MODICON, dnes součástí SCHNEIDER ELECTRIC. V současné době je podporována celá řada komunikačních médií např. sériové linky typu RS-232, RS-422 a RS-485, optické a rádiové sítě nebo síť Ethernet s využitím upraveného protokolu MODBUS TCP/IP. Komunikace probíhá metodou požadavek-odpověď a požadovaná funkce je specifikována pomocí kódu funkce, jež je součástí požadavku.



Obrázek 2.4: Příklad implementace protokolu MODBUS

Protokol MODBUS definuje strukturu zprávy na úrovni protokolu (PDU – Protocol Data Unit) nezávisle na typu komunikační vrstvy. V závislosti na typu sítě, na které je protokol použit, je PDU rozšířen o další části a tvoří tak zprávu na aplikační úrovni (ADU – Application Data Unit).

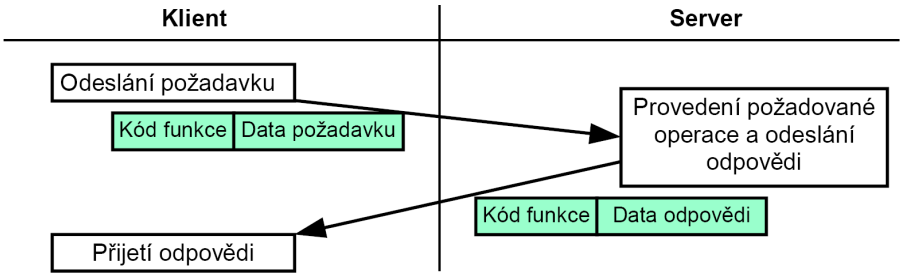


Obrázek 2.5: Základní tvar MODBUS zprávy

Kód funkce udává serveru jaký druh operace má provést. Rozsah kódů je 1 až 255, přičemž kódy 128 až 255 jsou vyhrazeny pro oznámení záporné odpovědi (chyby). Některé kódy funkcí obsahují i kód podfunkce upřesňující blíže požadovanou operaci. Obsah datové části zprávy poslané klientem slouží serveru k uskutečnění operace určené kódem funkce. Obsahem může být například adresa a počet vstupů, které má server přečíst nebo hodnota registrů, které má server zapsat. U některých funkcí nejsou pro provedení operace zapotřebí další data a v tom případě může datová část ve zprávě úplně chybět.

Pokud při provádění požadované operace nedojde k chybě (obr. 2.6), odpoví server zprávou, která v poli „Kód funkce“ obsahuje kód požadované funkce jako indikaci úspěš-

ného vykonání požadavku. V datové části odpovědi předá server klientovi požadovaná data (pokud jsou nějaká).



Obrázek 2.6: MODBUS transakce s bezchybným provedením požadavku

Pokud při vykonávání požadované operace dojde k chybě, je v poli „Kód funkce“ vrácen kód požadované funkce s nastaveným nejvyšším bitem indikujícím neúspěch (exception response). V datové části je vrácen chybový kód (exception code) upřesňující důvod neúspěchu.

Maximální velikost PDU je zděděna z první implementace MODBUSu na RS-485, kde byla maximální velikost ADU 256 bytů. Tomu odpovídá maximální velikost PDU 253 bytů.

velikost ADU na RS-485	253 B PDU + adresa (1 B) + CRC (2)	$\approx$ 256 B
velikost ADU na TCP/IP	253 B PDU + MBAP	$\approx$ 260 B

Protokol MODBUS definuje 3 základní typy zpráv (PDU):

- Požadavek (Request PDU)
  - 1 byte ... Kód funkce
  - n bytů ... Datová část požadavku – adresa, proměnné, počet proměnných
- Odpověď (Response PDU)
  - 1 byte ... Kód funkce (kopie z požadavku)
  - m bytů ... Datová část odpovědi – přečtené vstupy, stav zařízení
- Záporná odpověď (Exception Response PDU)
  - 1 byte ... Kód funkce + 80h (indikace neúspěchu)
  - 1 byte ... Chybový kód (identifikace chyby)

Kódování dat

Protokol MODBUS používá tzv. „big-endian“ reprezentaci dat. To znamená, že při posílání datových položek delších než 1 byte je jako první posílán nejvyšší byte a jako poslední nejnižší byte.

2.4.1 Datový model

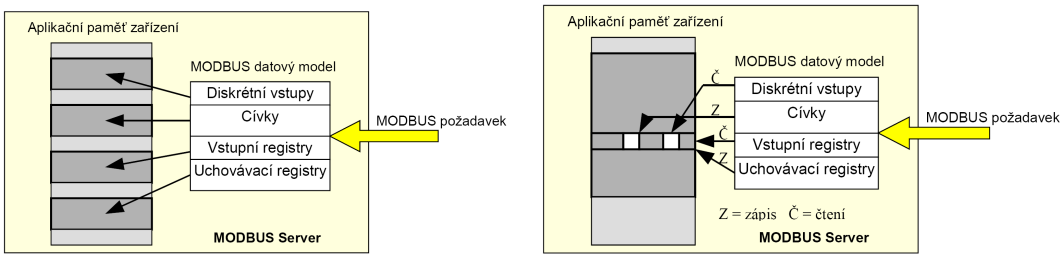
Datový model MODBUSu je založen na sadě tabulek, s charakteristickým významem. Definovány jsou čtyři základní tabulky:

Tabulka 2.1: Datový model MODBUS

tabulka	položka	přístup	popis	adresa
diskrétní vstupy	1 b	čtení	data poskytována I/O systémem	10000÷19999
cívky	1 b	čtení/zápis	data modifikovatelná aplikačním programem	0÷9999
vstupní registry	16 b	čtení	data poskytovaná I/O systémem	30000÷39999
uchovávací registry	16 b	čtení/zápis	data modifikovatelná aplikačním programem	40000÷49999

Mapování tabulek do adresního prostoru je závislé na konkrétním zařízení. Každá z tabulek může mít vlastní adresní prostor nebo se mohou částečně či úplně překrývat. Každá z tabulek může mít dle protokolu až 65 536 položek. Z důvodu zpětné kompatibility bývá adresní prostor rozdělen na bloky o velikosti 10 000 položek tak, jak je uvedeno ve sloupci „adresa“ (tabulka 2.1). Přístupná je každá položka jednotlivě nebo lze přistupovat ke skupině položek najednou. Velikost skupiny položek je omezena maximální velikostí datové části zprávy.

Na obr. 2.7 jsou znázorněny dva možné způsoby organizace dat v zařízení, obr. 2.7(a) znázorňuje zařízení, u kterého není žádný vztah mezi položkami jednotlivých tabulek a každá tabulka má tedy svůj oddělený prostor v aplikační paměti zařízení. Do jednotlivých tabulek lze přistupovat prostřednictvím příslušné funkce MODBUSu. Oproti tomu obr. 2.7(b) znázorňuje zařízení, které má pouze jeden datový blok. K položkám lze přistupovat prostřednictvím různých funkcí MODBUSu v závislosti na tom, co je pro aplikaci v daném okamžiku výhodné.



Obrázek 2.7: Datový model (a) s oddělenými bloky, (b) s jediným blokem

Adresovací model

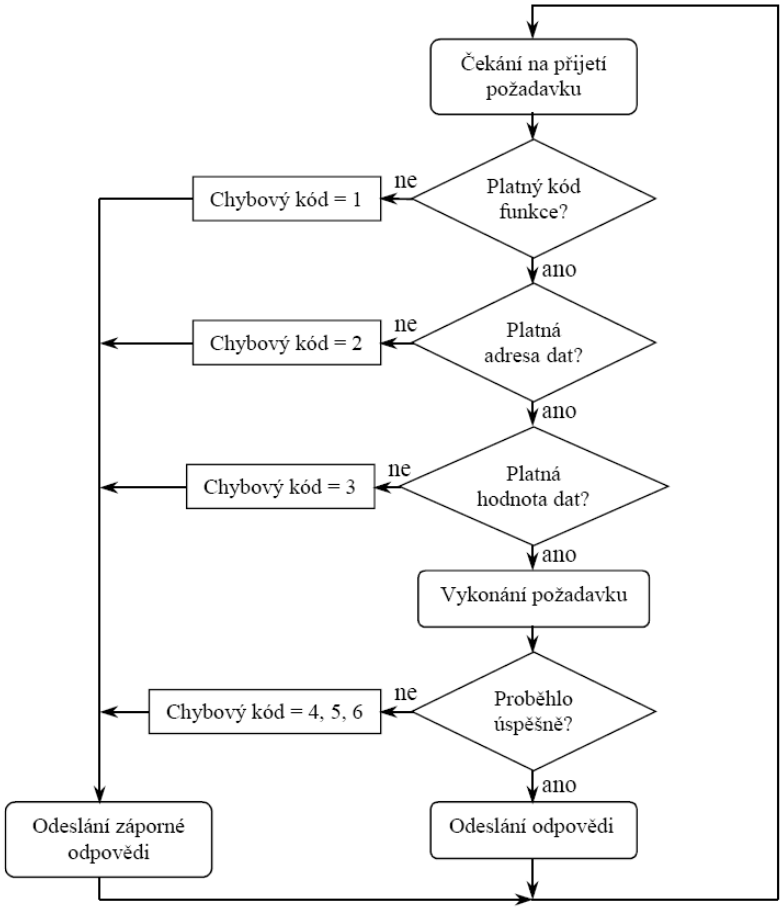
Protokol MODBUS přesně definuje adresovací pravidla ve zprávách (PDU). V MODBUS

zprávách jsou datové položky adresovány od 0 do 65 535. Dále je definováno adresování v rámci datového modelu složeného ze 4 datových bloků (tabulek), datové bloky jsou číslovány od 1 do n. Mapování položek datového modelu do aplikace v serveru je závislé na výrobci.

Definice MODBUS transakcí

Stavový diagram na obr. 2.8 ukazuje obecný postup zpracování MODBUS požadavku na straně serveru. Jakmile server zpracuje požadavek, sestaví odpověď a odešle ji klientovi. V závislosti na výsledku zpracování požadavku je vytvořena jedna ze dvou možných odpovědí:

- Pozitivní odpověď (Response)
  - kód funkce v odpovědi = kód funkce v požadavku
- Negativní odpověď (Exception Response)
  - kód funkce v odpovědi = kód funkce v požadavku + 80h
  - je vrácen kód chyby udávající důvod neúspěchu



Obrázek 2.8: Obecný postup zpracování MODBUS požadavku na straně serveru

Kategorie kódů funkcí

MODBUS protokol definuje tři skupiny kódů funkcí

- veřejné kódy funkcí - jasně definované, garantována unikátnost, schvalovány společností MODBUS-IDA.org, veřejně zdokumentované, dostupný test shody, zahrnují veřejné přiřazené kódy funkcí i nepřiřazené kódy rezervované pro budoucí použití,
- uživatelsky definované kódy funkcí - dva rozsahy uživatelsky definovaných kódů funkcí: 65 ÷ 72 a 100 ÷ 110, umožňují uživateli implementovat funkci, která není definována touto specifikací, není garantována unikátnost kódů, lze je po projednání přesunout do veřejných kódů,
- rezervované kódy funkcí - kódy funkcí, které jsou v současnosti používány některými firmami a které nejsou dostupné pro veřejné použití

				Kódy funkcí			
				Kód	Podfunkce	hex	
Přístup k datům	**Bitový přístup**	Fyzické diskrétní vstupy	Čti diskrétní vstupy	**02**		**02**	
		Interní bity nebo fyzické cívky	Čti cívky	**01**		**01**	
			Zapiš jednu cívku	**05**		**05**	
			Zapiš více cívek	**15**		**0F**	
	16-bitový přístup	Fyzické vstupní registry	Čti vstupní registr	**04**		**04**	
		Interní registry nebo fyzické výstupní registry	Čti uchovávací registry	**03**		**03**	
			Zapiš jeden registr	**06**		**06**	
			Zapiš více registrů	**16**		**10**	
			Čti/zapiš více registrů	**23**		**17**	
			Zapiš registr s maskováním	**22**		**16**	
			Čti FIFO frontu	**24**		**18**	
	Přístup k záznamům v souborech		Čti záznam ze souboru	**20**	**6**	**14**	
			Zapiš záznam do souboru	**21**	**6**	**15**	
Diagnostika			Čti stav	**07**		**07**	
			Diagnostika	**08**	**00-18, 20**	**08**	
			Čti čítač kom. událostí	**11**		**0B**	
			Čti záznam kom. událostí	**12**		**0C**	
			Sděl identifikaci	**17**		**11**	
			Čti identifikaci zařízení	**43**	**14**	**2B**	
Ostatní			Zapouzdřený přenos	**43**	**13, 14**	**2B**	
			CANOpen základní odkaz	**43**	**13**	**2B**	

Obrázek 2.9: Přehled funkčních kódů

Záporné odpovědi

Když klient posílá serveru požadavek, očekává na něj odpověď. Mohou nastat čtyři situace.

- Server přijme bezchybně požadavek a je schopen jej normálně zpracovat, vrátí klientovi normální odpověď.
- Server požadavek nepřijme z důvodu komunikační chyby, není vrácena žádná odpověď. Na straně klienta dojde k vypršení časového limitu pro příjem odpovědi.
- Server přijme požadavek, ale detekuje komunikační chybu (parita, CRC apod.), nevrací žádnou odpověď. Na straně klienta dojde k vypršení časového limitu pro příjem odpovědi.
- Server přijme bezchybně požadavek, ale není schopen jej normálně zpracovat, vrátí klientovi zápornou odpověď s udáním důvodu neúspěchu.

Normální a záporná odpověď se liší nejvyšším bitem kódu funkce. Je-li bit nulový, jedná se o normální odpověď, je-li bit nastavený, jedná se o zápornou odpověď. V případě záporné odpovědi je v datové části předán kód chyby. V následující tabulce je seznam možných chybových kódů.

Tabulka 2.2: MODBUS chybové kódy

kód	jméno	význam
01	ilegální funkce	požadovaná funkce není serverem podporována
02	ilegální adresa dat	zadaná adresa je mimo serverem podporovaný rozsah
03	ilegální hodnota dat	předávaná data jsou neplatná
04	selhání zařízení	při provádění požadavku došlo k neodstranitelné chybě
05	potvrzení	kód určený k použití při programování. Server hlásí přijetí platného požadavku, ale jeho vykonání bude trvat delší dobu
06	zařízení je zaneprázdněné	kód určený k použití při programování. Server je zaneprázdněn vykonáváním dlouho trvajícího příkazu.
07	chyba parity paměti	kód určený k použití při práci se soubory. Server při pokusu přečíst soubor zjistil chybu parity
0A	brána – přenosová cesta nedostupná	kód určený k práci s bránou (gateway). Brána není schopná vyhradit interní přenosovou cestu od vstupního portu k výstupnímu. Pravděpodobně je přetížená nebo nesprávně nastavená.
0B	brána – cílové zařízení neodpovídá	kód určený k práci s bránou (gateway). Cílové zařízení neodpovídá, pravděpodobně není přítomno.

2.4.2 Implementace MODBUSu

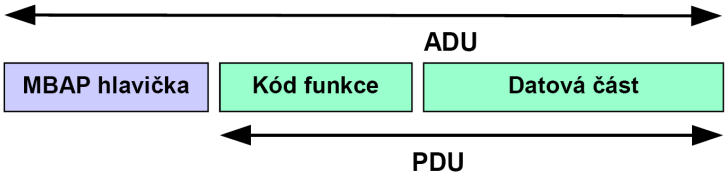
MODBUS standard definuje kromě aplikační vrstvy ISO/OSI modelu i některé implementace protokolu na konkrétní typ sítě nebo sběrnice. Příkladem je MODBUS TCP/IP a MODBUS na sériové lince pro fyzickou vrstvu RS-232 a RS-485.

2.4.2.1 MODBUS TCP/IP

Protokol Modbus TCP, vyvinutý v roce 1979 společností Modicon, je dnes levným a jednoduše použitelným protokolem, který je považován za standard. Byl začleněn do protokolu IP jako TCP 502 a v současné době se jeho vývoj opírá o organizaci Modbus.org, jejímž cílem je zdokonalovat a propagovat Modbus jako první průmyslový internetový protokol. Ke sběrnicím s protokolem Modbus je připojeno přes milion sériových zařízení, zatímco s protokolem Modbus TCP/IP jich pracuje několik set tisíc. Velkou předností protokolu je jeho stabilita. Protokol je sám o sobě otevřený a informace o něm jsou volně dostupné. V aplikacích klient/server umožňuje komunikaci v reálném čase mezi zařízeními přes Ethernet TCP/IP s dobou odezvy od 0,24 s až 1 s.

Implementace služby I/O scanning zvyšuje výkonnost protokolu, používá se pro rychlou výměnu dat se vzdálenými vstupně/výstupními jednotkami nebo frekvenčními měniči. Sledování je prováděno z řídicího systému pomocí dotazů Modbus TCP Read/Write. Pokud je třeba zajistit synchronizaci v reálném čase v rozsáhlejších distribuovaných aplikacích, přichází v úvahu služba Global Data. Umožňuje zařízením na síti zapisovat (publish), nebo získávat (subscribe) data. Doba výměny dat se tak redukuje na polovinu, není třeba se na data dotazovat.

Na obr. 2.10 je znázorněn formát zprávy na TCP/IP. Pro identifikaci MODBUS ADU je použita MBAP hlavička (MODBUS Application Protocol Header). Pro posílání MODBUS/TCP ADU je na TCP vyhrazen registrovaný port 502.



Obrázek 2.10: MODBUS TCP/IP zpráva

2.4.2.2 MODBUS na sériové lince

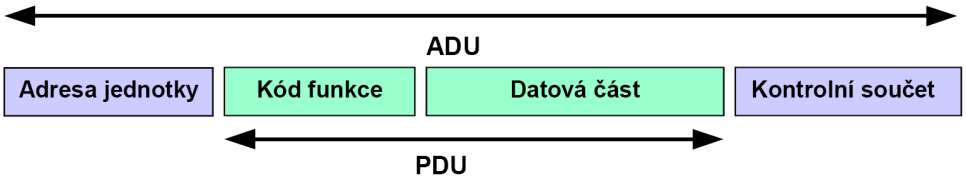
MODBUS Serial Line protokol je typu Master-Slave a je definován na síťové vrstvě ISO/OSI modelu. Na fyzické úrovni mohou být použita různá sériová rozhraní, například RS-232 nebo RS-485 a jejich varianty.

Princip komunikace je Master/Slave, tedy v jeden okamžik může být na sběrnici pouze jeden master a 1 až 247 slave jednotek. Komunikaci vždy zahajuje master, slave nesmí

nikdy vysílat data bez pověření mastera.
Master posílá požadavky slave jednotkám ve dvou režimech:

- unicast režim ... master adresuje požadavek jedné konkrétní slave jednotce a ta pošle odpověď,
- broadcast režim ... master posílá požadavek všem jednotkám, žádná jednotka neodpoví.

Adresní prostor zahrnuje 256 různých adres. Master nemá žádnou specifickou adresu, pouze slave jednotky musejí mít adresu a ta musí být v celé MODBUS síti jedinečná. Pravidlem je, že adresa 0 je vyhrazena pro broadcast a adresy $248 \div 255$ jsou rezervovány.



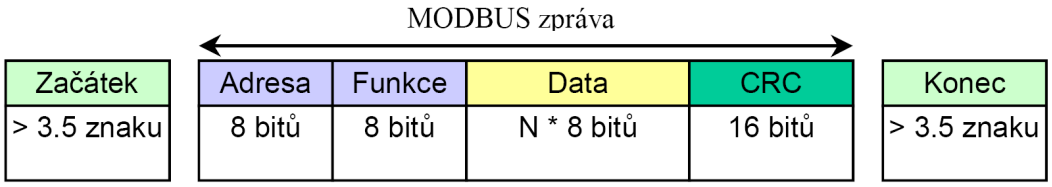
Obrázek 2.11: MODBUS zpráva na sériové lince

Na obr. 2.11 je znázorněn základní formát MODBUS aplikační zprávy na sériové lince. Zpráva kromě standardní MODBUS PDU obsahuje pole „Adresa jednotky“. Toto pole obsahuje adresu slave jednotky. Pole „Kontrolní součet“ slouží k detekci chyb a obsahuje CRC nebo LRC kód v závislosti na vysílacím režimu.

Protokol MODBUS definuje dva sériové vysílací režimy, MODBUS RTU a MODBUS ASCII. Režim určuje v jakém formátu jsou data vysílána a následně dekódována. Každá jednotka musí podporovat režim RTU, režim ASCII je nepovinný. Všechny jednotky na jedné sběrnici musejí pracovat ve stejném vysílacím režimu.

MODBUS RTU

V režimu RTU obsahuje každý 8 b byte zprávy dva 4 b hexadecimální znaky. Vysílání zprávy musí být souvislé, mezery mezi znaky nesmějí být delší než 1.5 znaku. Začátek a konec zprávy je identifikován podle pomlky na sběrnici delší než 3.5 znaku. Formát RTU rámce je znázorněn na obr. 2.12. K detekci chyb slouží 16-bitové CRC pole s generujícím polynomem $x^{16} + x^{15} + x^2 + 1$. Každá jednotka musí podporovat sudou paritu. Pokud není použita parita, je nahrazena druhým stop bitem.



Obrázek 2.12: RTU rámec zprávy

Formát bytu (11 bitů)

1	start bit
8	datových bitů
1	bit parita
1	stop bit

MODBUS ASCII

V režimu ASCII je každý 8 b byte posílán jako dvojice ASCII znaků. Oproti režimu RTU je tedy pomalejší, ale umožňuje vysílat znaky s mezerami až 1 s. Začátek a konec zprávy je totiž určen odlišně od RTU módu. Začátek zprávy je indikován znakem „:“ a konec zprávy dvojicí řídicích znaků CR, LF. Formát ASCII rámce je na obr. 2.13. K detekci chyb slouží 8 b LRC pole. Každá jednotka musí podporovat sudou paritu. Pokud není použita parita, je nahrazena druhým stop bitem.

Začátek	Adresa	Funkce	Data	LRC	Konec
znak „:“	2 znaky	2 znaky	0 až 2*252 znaků	2 znaky	2 znaky CR, LF

Obrázek 2.13: ASCII rámec zprávy

Formát bytu (10 bitů)

1	start bit
7	datových bitů
1	bit parita
1	stop bit

2.5 Protokol SNMP - Simple Network Management Protocol

Protokol SNMP byl původně určen pro usnadnění správy sítě, možnosti jeho využití jsou podstatně větší a stále častěji proniká i do průmyslové automatizace a měřicí techniky. Protokol je asynchronní, transakčně orientovaný, založený na modelu klient/server. Strana, která posílá požadavky (SNMP klient), může být např. jednoduchý SNMP browser či složitý NMS (Network Management System). Na straně zařízení je SNMP agent (SNMP server), který na požadavky odpovídá. Výjimku tvoří tzv. trapy, které agenti vysílají asynchronně při výskytu jednotlivých událostí (výpadek proudu, překročení mezních údajů, připojení nového zařízení). Je nutné předem definovat adresu, kam se informace posílá. Pro přenos dat se používá protokol UDP, přičemž je definováno přesně místo, kam se mohou připojovat uživatelské aplikace jednotlivých firem, které spravuje organizace IANA (Internet Assigned Numbers Authority - doslovně: Internetová autorita pro přidělování čísel).

Typickými úlohami pro SNMP protokol jsou

- zjišťování stavových informací o zařízeních (množství volné paměti, počet spojení, počet přihlášených uživatelů, ...)
- nastavování parametrů (atributů) na síťových prvcích
- monitorování uptimu
- monitorování verzí běžících systémů
- sběr dat o existujících síťových rozhraních (`ifName`, `ifDescr`, `ifSpeed`, `ifType`, `ifPhysAddr`)
- measuring network interface throughput (`ifInOctets`, `ifOutOctets`)
- dotazování ARP[7] (`ipNetToMedia`)

SNMP je uzpůsobeno k přenosu nad nespolehlivým kanálem, využívá protokol UDP. Běží na nejrůznějších síťových protokolech. Od verze SNMP v2 pracuje jako samostatný aplikační protokol. Typicky běží na portu 161 pro agenty a 162 pro management. Agent odpovídá na portu, na kterém přišel dotaz, TRAP zasílá na port 162.

SNMP v1 neposkytoval zabezpečení, autentizace se posílala po síti v plain textu. SNMP v1 a SNMP v2 (RFC 1441, RFC 1452) nejsou kompatibilní, dva způsoby spolupráce řeší RFC 1908 a to pomocí proxy agentu a dvouprotokolové sítě. SNMP v3 (RFC 3411, RFC 3418, 2004) je nejnovější standard. Současní klienti obvykle podporují všechny tři módy protokolu (viz RFC 3584). SNMP v3 řeší autentizaci a zabezpečení pomocí šifrování (AES - Advanced Encryption Standard).

2.5.1 Historie

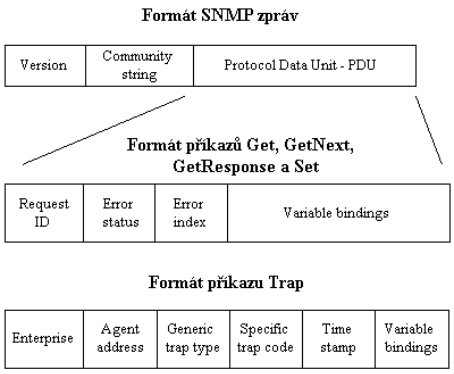
Protokol SNMP začal vznikat v roce 1988 jako reakce na potřebu efektivní platformy pro správy počítačových sítí. V roce 1990 byl institucí IAB (Internet Acitivities Board) potvrzen jako standard sítě internet. První specifikace protokolu, RFC 1157, vznikla v

roce 1989 a stanovovala vlastnosti SNMP v1. Specifikovala jen příkazy `get`, `get next`, `set`, `trap` a využívala pouze ochranu heslem - community string. Toto byl pravděpodobně hlavní nedostatek, který přetrval ještě ve v2, která dále definovala příkaz `get bulk` pro snazší získávání informací z tabulek (např. routing table atp.). Ochrana heslem je nedostatečná, protože heslo lze poměrně jednoduše zjistit analyzátorem paketů. Současná specifikace protokolu SNMP v3 pochází z roku 1998 (přijata 2002) a umožňuje ochranu dat i pomocí DES[8] šifrovacího algoritmu.

- `get-request` získání informace z MIB
- `get-next-request` umožňuje managerovi získat informace o objektech v MIB bez znalosti jejich přesných jmen, umožňuje postupné procházení celým hierarchickým stromem
- `set-request` změna hodnoty proměnné agenta
- `trap` jediný typ příkazu vysílaný bez předchozího vyžádání, agent jej zasílá managerovi jako reakci na specifikovanou událost, zpráva zůstává nepotvrzená, proto agent nemá jistotu, zda byla doručena
- `get-response` agent vykoná tuto operaci jako reakci na předchozí příkazy - je to vlastně odpověď agenta managerovi. Odpověď obsahuje i dotaz, protože protokol nezajišťuje souvislost mezi dotazem a odpovědí
- `get-bulk` operace, která je součástí SNMP v2, umožňuje vyžádat si k přečtení celou skupinu informací z MIB, čímž se mnohdy urychluje komunikace
- `inform` umožňuje komunikaci dvou managerů mezi sebou

Protokol SNMP v1 je základní, SNMP v2 obsahuje navíc autentizaci a SNMP v3 navíc šifrování. Nejvíce zařízení podporuje SNMP v2. Na monitorované straně jsou shromažďovány informace o stavu zařízení. Získaný obsah zpráv se na monitorovací straně může dále zpracovávat (tabulky, grafy). Na monitorované straně může existovat možnost konfigurace, kdy agent zašle managerovi informace automaticky bez jeho požadavku. K tomu dojde zpravidla potom, kdy byla splněna předem definovaná podmínka (výpadek, kolize, dosažení hraniční hodnoty), agent nečeká na odpověď. Takové konfiguraci agenta se říká SNMP TRAP.

[8] DES je symetrická šifra vyvinutá v 70. letech. V současnosti je tato šifra považována za nespolehlivou, protože používá klíč pouze o délce 56 b. Navíc algoritmus obsahuje slabiny, které dále snižují bezpečnost šifry. Díky tomu je možné šifru prolomit útokem hrubou silou za méně než 24 hodin.



Obrázek 2.14: Formát SNMP zprávy)

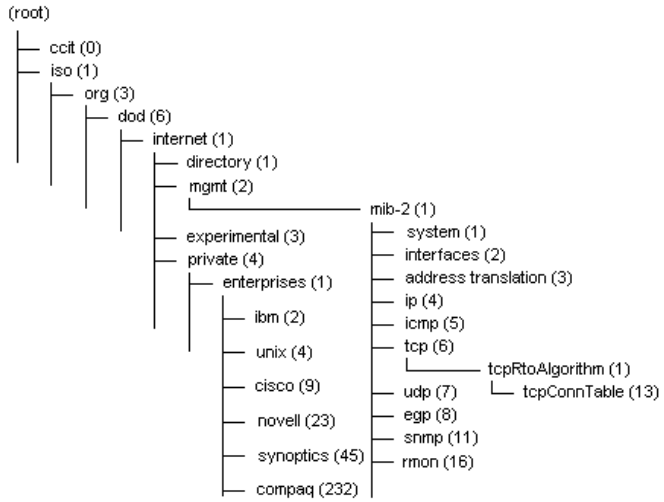
Každá hodnota v SNMP je jednoznačně identifikována pomocí číselného identifikátoru OID - Object Identifier. Tento identifikátor je tvořen posloupností čísel oddělených tečkou, hodnota vznikne doplněním nadřazeného OID o aktuální číslo. Celá tato stromová struktura je uložena v MIB databázi. Navíc databáze obsahuje jména a popisy jednotlivých hodnot (OID). MIB může být doplněna o další hodnoty pomocí části struktury uložené v MIB souboru.

Příkladem OID může být hodnota 1.3.6.1.2.1.2.2.1.6.1, které odpovídá textová verze z MIB databáze
`iso.org.dod.internet.mgmt.mib-2.interfaces.ifTable.ifEntry.ifPhysAddress`.

Textová reprezentace MIB je pojmenovaný modul, který musí vyhovovat syntaktickým pravidlům podmnožiny jazyka ASN.1 (Abstract Syntax Notation One) a seskupuje dohromady odpovídající definice. Jeden nebo více těchto modulů je uchováno jako standardní ASCII soubor.

2.5.2 SNMP Global Naming Tree

Každý objekt zařízení musí mít jedinečné jméno, aby se na něj dalo odkazovat. Protože jedno zařízení může obsahovat objekty definované nezávisle několika různými výrobci, schéma pro pojmenování těchto objektů muselo být navrženo tak, aby nemohlo dojít k záměně. Centrální registr všech možných objektů by byl nekonečně veliký, byla proto zvolena koncepce hierarchického stromu SNMP Global Naming Tree, vyvinutého organizací ISO (www.iso.org).



Obrázek 2.15: SNMP Global Naming Tree

Standardní MIB struktura tedy odpovídá tomuto SNMP Global Naming Tree, který se skládá z objektů root, subtree a leaf. Každá část tohoto stromu má označení, které se skládá ze dvou částí - stručného textového popisu a číselného kódu. Kořenový uzel (root) je sám bez popisu, ale pod ním jsou vždy minimálně tři uzly

- iso(1) spravován organizací ISO,
- ccitt(0) spravován organizací ITU-T (bývalé CCITT),
- set-request změna hodnoty proměnné agenta,
- joint-iso-ccitt(2) společně spravováno ISO a ITU-T.

Jednotlivým výrobcům zařízení jsou přidělovány subtree (jsou jmenováni výkonnými autoritami) a mohou si tak vytvářet vlastní neomezenou strukturu. Takto vzniklé privátní MIB popisují vlastnosti konkrétního zařízení. Většinou jsou ale výrobci zveřejňovány, právě z důvodu umožnění správy těchto prvků i aplikacemi jiných výrobců.

Jméno uzlu, tzv. `object identifier`, je tak tvořeno sekvencí číselných kódů na cestě z „root“ přes „subtree“ až k danému objektu typu „leaf“. Tato decimální notace reprezentuje tedy cestu ke každé z funkcí nebo schopností daného zařízení. Jde o podobný systém jako při specifikacích plných cest k souborům v systémech UNIX a DOS, přičemž nejvyšší úroveň začíná v objektu (root). Textový popis slouží ke snazší orientaci v této struktuře.

Na obr. 2.15 je schematicky znázorněna část této struktury, kde je vidět postavení jednotlivých objektů, funkcí a cestu k jejich dosažení.

Ve větvi internet(1) jsou vytvořeny tři logické skupiny

- Management Branch — standardní MIB, které byly vytvořeny orgánem IETF (www.ietf.org), jsou umístěny v části (... internet(1)mgmt(2)) hierarchické struktury MIB a obsahují definované objekty pro některé běžné síťové zařízení a protokoly. Tuto skupinu podporují zařízení většiny výrobců a tak umožňují jejich nezávislou správu
- Experimental Branch — tato větev zahrnuje MIB, které jsou zatím ve vývoji
- Private Branch — změna hodnoty proměnné agenta
- joint-iso-ccitt(2) — privátní MIB jednotlivých výrobců jsou lokalizovány v části (iso(1)org(3)dod(6)internet(1)private(4)enterprises(1)). Tato větev tak umožňuje jednotlivým výrobcům vytvářet MIB pro svá vlastní zařízení, jimž nedostačují standardní MIB, např. object identifier 1.3.6.1.4.1.45 reprezentuje cestu k objektům firmy SynOptics, 1.3.6.1.4.1.23 cestu k objektům Novell, 1.3.6.1.4.1.9 cestu k objektům Cisco, atd.

2.5.3 Typy SNMP objektů

SNMP objekty mohou být v zásadě dvou typů - skalární hodnoty a tabulky. Objekty typu skalár mohou nabývat pouze jednoduché nestrukturované hodnoty. Jedná se o standardní typy

- Integer — celé číslo, většina implementací omezuje tento typ velikostí 32 b,
- Counter — nezáporný integer, plynule se zvětšuje až do hodnoty (232 - 1), poté znovu od nuly, absolutní hodnota je méně důležitá než rozdíl od posledního vzorku, ze kterého lze usuzovat na rychlost změn,
- Gauge — nezáporný integer, hodnota může vzrůstat i klesat, ale nemůže překročit max. hodnotu (231),
- TimeTicks — nezáporný integer, čas v setinách sekundy od jisté doby (modulo (232 - 1)), např. doba chodu zařízení
- IpAddress — 32 b IP adresa
- OCTET STRING — sekvence bytů, užívá se k vyjádření řetězce znaků (jméno systému), nebo libovolných binárních dat (MAC adresy zařízení)
- OBJECT IDENTIFIER — reprezentuje jméno uzlu, SNMP dovoluje další typy skalárních hodnot (NULL, Opaque a Network Address), které se ale nepoužívají

Jako rozšíření těchto nestrukturovaných jednoduchých objektů dovoluje standard SNMP strukturovat data do tabulek. Tyto tabulky jsou pak uspořádány do řádků a sloupců (ob-

doba databázových záznamů). Jednotlivé položky takovéto tabulky mohou být libovolné skalární hodnoty. Tabulky nelze do sebe vnořovat.

Nelze provádět SNMP operace nad tabulkami jako celkem, ale jen nad jednotlivými skalárními objekty - hodnotami v tabulce. Příkladem tabulky mohou být směrovací tabulky routerů `tcpConnTable` (object identifier 1.3.6.1.2.1.6.13) a jednotlivé sloupce reprezentují hodnoty `tcpConnState`, `tcpConnLocalAddress`, `tcpConnLocalPort`, `tcpConnRemAddress` a `tcpConnRemPort`.

Identifikace jednotlivých polí v SNMP tabulkách se provádí pomocí indexů. Těmi jsou buď jednoduché hodnoty nebo sady hodnot. K jednotlivým polím pak můžeme přistupovat náhodně, tedy příkazem `get` se specifikací pozice nebo sekvenčně příkazem `get-next` procházet celou tabulku.

2.5.4 MIB databáze - Management Information Base

MIB je databáze, která dovoluje jednoznačně identifikovat informace využívané systémem správy. Aby mohl SNMP manager i agent tyto informace získat a předávat, tak je nutná znalost struktury MIB.

Báze dat je objektově orientovaná. Data jsou uložena jako objekty a sdružují se do tříd. Jednotlivé objekty mají hodnoty. Každý řízený objekt v MIB obsahuje veškeré informace potřebné pro popis. Způsob pojmenování objektů je založen na jejich vztahu. Jeden objekt může obsahovat jiné objekty nebo jiné třídy. MIB je tedy tvořena jedním stromem.

Každý agent by měl udržovat objekty standardní MIB (např. síťové adresy, typy rozhraní, čítače). Jsou definovány tři mechanismy pro přidání:

- přidání nových objektů prostřednictvím definice nové verze MIB-II
- přidání nestandardních objektů přidáním experimentální větve
- přidání vlastních objektů v rámci podstromu soukromé větve

Do MIB byly zařazeny jen nejnutnější objekty. Předem byly vyloučeny objekty svým způsobem nadbytečné, které mají konkrétní vztahy s jinými objekty v MIB. Jednoduchost definice a omezená velikost báze umožňuje zaručit minimální dopad na činnost a složitost agentů. To se projeví v nárocích na zpracovatelský systém.

Dnes lze nalézt spoustu existujících a v praxi používaných systémů, které 24 hodin denně, 7 dní v týdnu, 365 dní v roce monitorují stav systémů a služeb v počítačové síti. V případě výpadku informují definovanými kanály (nejčastěji email, SMS, vizuální výstraha) příslušnou obsluhu, popř. spustí definovanou akci.

Díky podpoře SNMP není problém realizovat systém pro sběr informací z agentů, kteří SNMP podporují, a sestavit tak konkrétní aplikaci, která se bude chovat podle našich požadavků. Následně pak lze realizovat skutečně velmi jednoduše systém, který bude příslušná data zpřístupňovat na webových stránkách. Navíc získáme možnost testování a monitorování řízeného systému třetí stranou.

Kapitola 3

Vývojová prostředí

3.1 CoDeSys - Controlled Development System

Software CoDeSys je univerzální vývojové prostředí pro aplikační programy řídicích systémů PLC. Bylo vytvořeno firmou 3S (Smart Software Solution) dle standardu IEC 61131-3[1], který popisuje jednotnost a částečnou zaměnitelnost díky tomu, že definuje povinné jazyky a jejich syntaxi tak, aby programátor nemusel znát množství navzájem různých jazyků. Norma však nic neříká o způsobu ovládání programu a způsobu konfigurace hardwarových periférií a komunikací, takže každý výrobce volí vlastní způsob řešení.

CoDeSys obsahuje celkem šest různých programovacích jazyků (přičemž norma IEC 61131-3 definuje pouze pět programovacích jazyků). Mezi jazyky je možné volně přepínat a na každou danou část programu použít nejvhodnější programovací jazyk. Jedná se o následující jazyky:

- IL ... instrukční list (PLC assembler),
- LD ... žebříčkový diagram (Ladder Diagram) – je v současné době nejoblíbenější jazyk užívaný programátory,
- FBD ... funkční diagram (Function Block Diagram) je LD rozšířený o volání funkcí,
- SFC ... sekvenční funkční diagram (Sequential Function Chart) je jazyk vhodný pro sekvenční aplikace, kdy vstup do další části programu je podmíněn vykonáním části předchozí a splněním zadaných podmínek, určitá varianta obecných Petriho sítí,
- ST ... strukturovaný text (Structured Text) – je vyšší programovací jazyk vhodný pro pokročilé matematické aplikace a složité algoritmy, jazyk je velmi podobný např. prostředí Pascal,

[1] Mezinárodní norma IEC 61131 (zpracovaná organizací PLCopen a oficiálně vydaná Mezinárodní elektrotechnickou komisí (International Electrotechnical Commission – IEC)) se věnuje programovatelným automatům – jejich funkci, provedení hardwaru a komunikací i způsobům jejich programování. Odborné veřejnosti je nejznámější její třetí část, tj. IEC 61131-3, týkající se programování automatů (forma uživatelských programů, deklarace proměnných a typů dat, způsoby aktivace programů a popis programovacích jazyků).

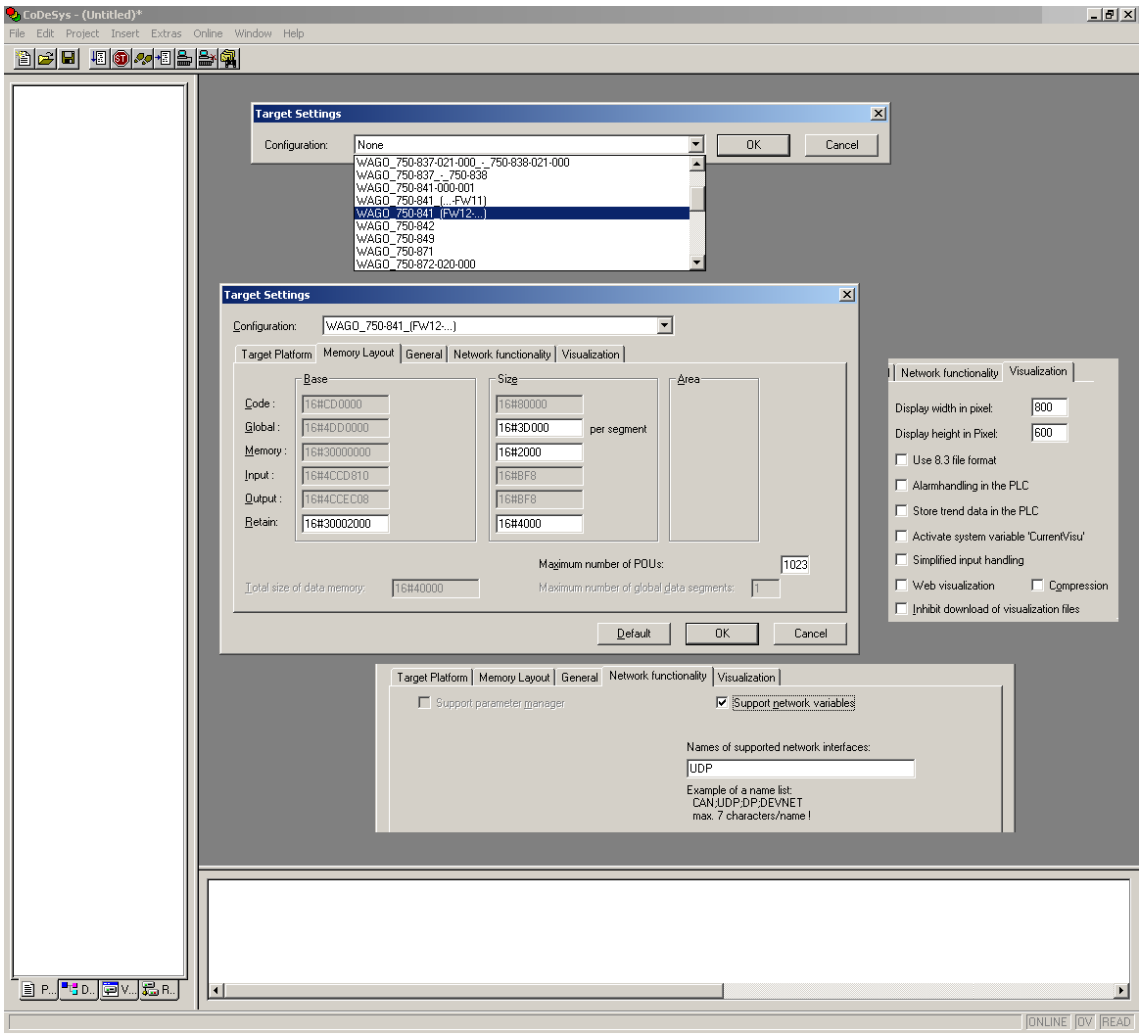
- CFC ... spojitý funkční diagram (Continuous Function Chart).

CoDeSys umožňuje při práci na projektu jeho naprogramování (textové a grafické editory), odladění a testování, vizualizaci procesu, uvedení do provozu, vytvoření dokumentace.

V popisovaném řešení je využíváno PLC od společnosti WAGO. Výrobce musí dodat knihovny a ovladače, aby bylo možné se k PLC připojit a vytvářet programy. V následující kapitole bude popsáno založení, sestavení a nakonec i vložení kompilované aplikace do PLC kontroléru. Tato verze prostředí je 2.3.8.5 (vydána 5.10.2007) a podporuje PLC s firmwarem FW14 a nižší. Jeho výhodou je plná podpora MIB tabulek pro SNMP protokol.

Založení nového projektu se skládá z výběru správného výrobce a kontroléru (v případě, že je nainstalována podpora více výrobců a jejich PLC kontrolerů). V tomto případě se jedná o model s ethernetovým rozhraním s označením WAGO_750-841. Zároveň je nutné vybrat verzi firmwaru. Je nabídnuta možnost firmware do verze 11 (`...-FW11`) a od verze 12 (`FW12-...`). Další dialog s názvem `Terget Settings` (viz obr. 3.1) obsahuje pět karet. Záložka `Memory Layout` ukazuje základní adresy pro kód, paměťový prostor, vstupy, výstupy apod. a jejich rozsah. Zároveň je zde zobrazena informace o maximálním počtu POUs (viz dále). Karta `General` obsahuje check box `Online Change`. Zaškrtnutím se paměť rozdělí na polovinu a je možné provádět při připojení k PLC jednoduché změny programu (není nutné pro změnu programu se odhlašovat od PLC). Karta `Network functionality` obsahuje check box `Support network variables`. Pokud jsou propojeny alespoň dvě PLC, je možné pomocí těchto síťových proměnných provádět výměnu informací. V této práci byly síťové proměnné použity a konfigurace bude popsána v další kapitole. Poslední kartou je `Visualization`. Zde je možné nastavit velikost obrazovky na kterou se tvoří vizualizace (optimalizace velikosti). Pokud je požadavek na zobrazení vizualizace na web-serveru v PLC, je nutné zaškrtnou check box `Web visualization`, dále pokud je obsaženo vykreslování trendů a ukládání hodnot, je možné je archivovat přímo v PLC v paměti - `Store trend data in the PLC`. To jsou podstatné konfigurační kroky při tvorbě programu do PLC.

Dalším krokem je založení programu. Ten se skládá z tzv. POU (Program Organization Unit). Podmínkou je alespoň jeden blok POU pojmenovaný `PLC_PRG(PRG)`.



Obrázek 3.1: Vývojové prostředí CoDeSys 2.3.8.5

Výsledný program se může skládat z **bloků programu (PRG)**. Program je POU, které vrací několik hodnot v průběhu operace. Jsou globálně řazené v projektu a všechny získané hodnoty jsou z předešlého běhu. Lze je volat (ne však z funkcí). Pokud je volán program z jiného POU a jeho hodnoty se změnily, změny se projeví až při dalším volání programu, dokonce i když je volán z jiného POU. Tyto změny proto hrají roli pouze, když probíhá volání stejné instance. Deklarace se provede klíčovým slovem `PROGRAM`.

Dalším POU je **funční blok (FB)**. Poskytuje jednu či více hodnot v průběhu operace. Oproti funkci nemá návratovou hodnotu. Deklarace se provede klíčovými slovy `FUNCTION_BLOCK`. Volání se provede pomocí instancí.

Posledním POU jsou **funkce (FUN)**, které jediné předávají návratový element. Ten se může skládat z jednotlivých proměnných nebo i celých struktur. Při deklaraci je nutné uvést návratový typ, dle `FUNCTION 'nazev': INT`.

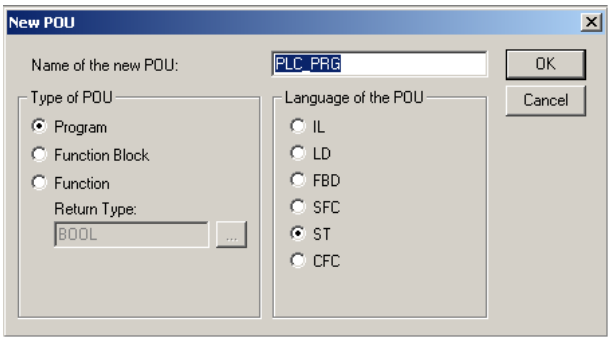
Dialogové okno pro založení nového bloku POU je na obr. 3.1. Je možné vybrat z následujících programovacích jazyků.

Tabulka 3.1: Seznam programovacích jazyků

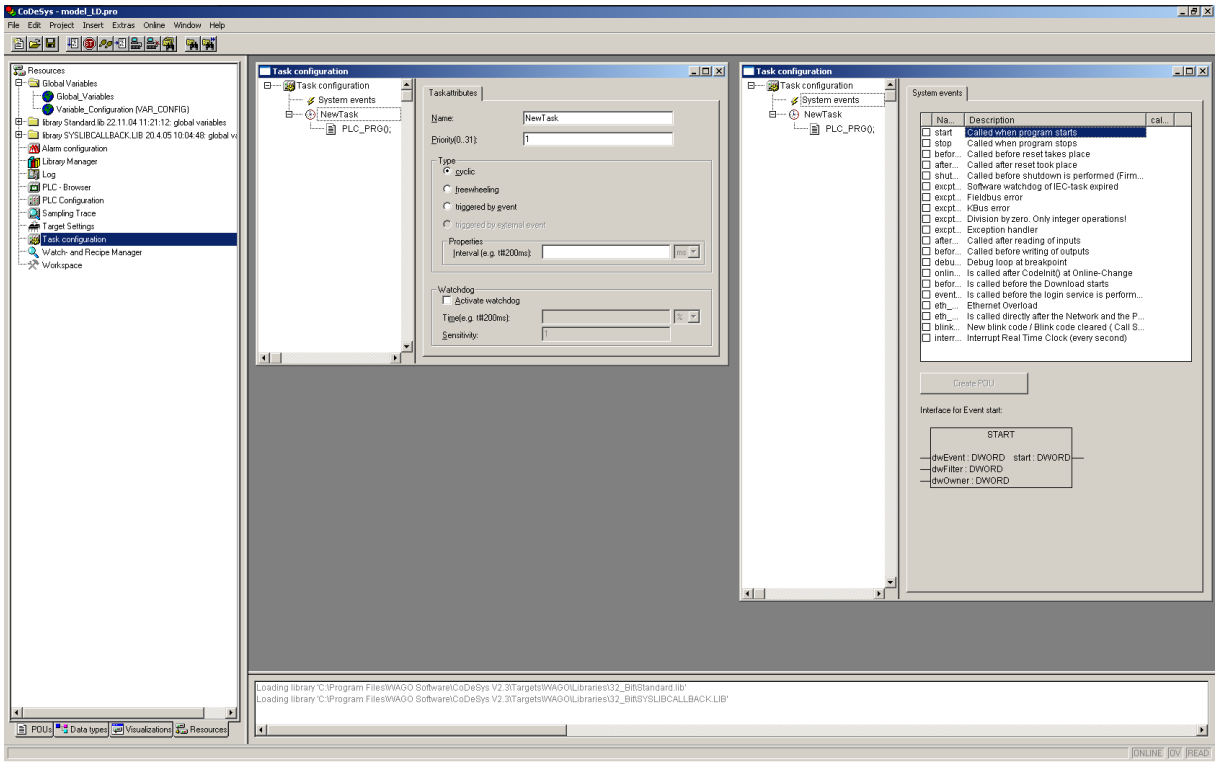
IL	Instruction List
LD	Ladder Diagram - žebříčkový diagram
FBC	Function Block Diagram
SFC	Sequential Function Chart
ST	Structured Text
CFC	Continuous Function Chart

Programovací jazyk SFC vychází z Grafcetu a je vhodný pro sekvenční program, jak bude ukázáno dále. Jednotlivé kroky SFC lze s výhodou programovat pomocí žebříčkových diagramů, kde se užívá tzv. kontaktů a cívek. Tím je sestaven požadovaný logický výraz. Další možností, která bude ukázána je jazyk ST. Jedná se o strukturovaný text a proto může vzdáleně připomínat jazyk Pascal.

Automat PLC pracuje v tzv. scan cyklu, kdy nejprve je zjištěn stav vstupů, následně je vyhodnocen program a zapsány hodnoty na výstupy. Dnešní PLC automaty toto schéma dodržují, ale zároveň umožňují definovat spuštění programů-tasků v několika režimech. Z tohoto důvodu je nutné vytvořený program přidat do seznamu spouštěných. K tomu slouží dialog Task configuration. Ten je možné najít na záložce Resources. Je možné vybrat vykonávání cyklické s definovanou periodou, volný běh (freewheeling) programu a spouštění (externí) událostí (triggered by event).



Zároveň je možné přidat vykonání programu při určité události. Tou může být např. start nebo zastavení hlavního programu, reset kontroleru apod. Uvedené možnosti jsou zobrazeny na obr. 3.2



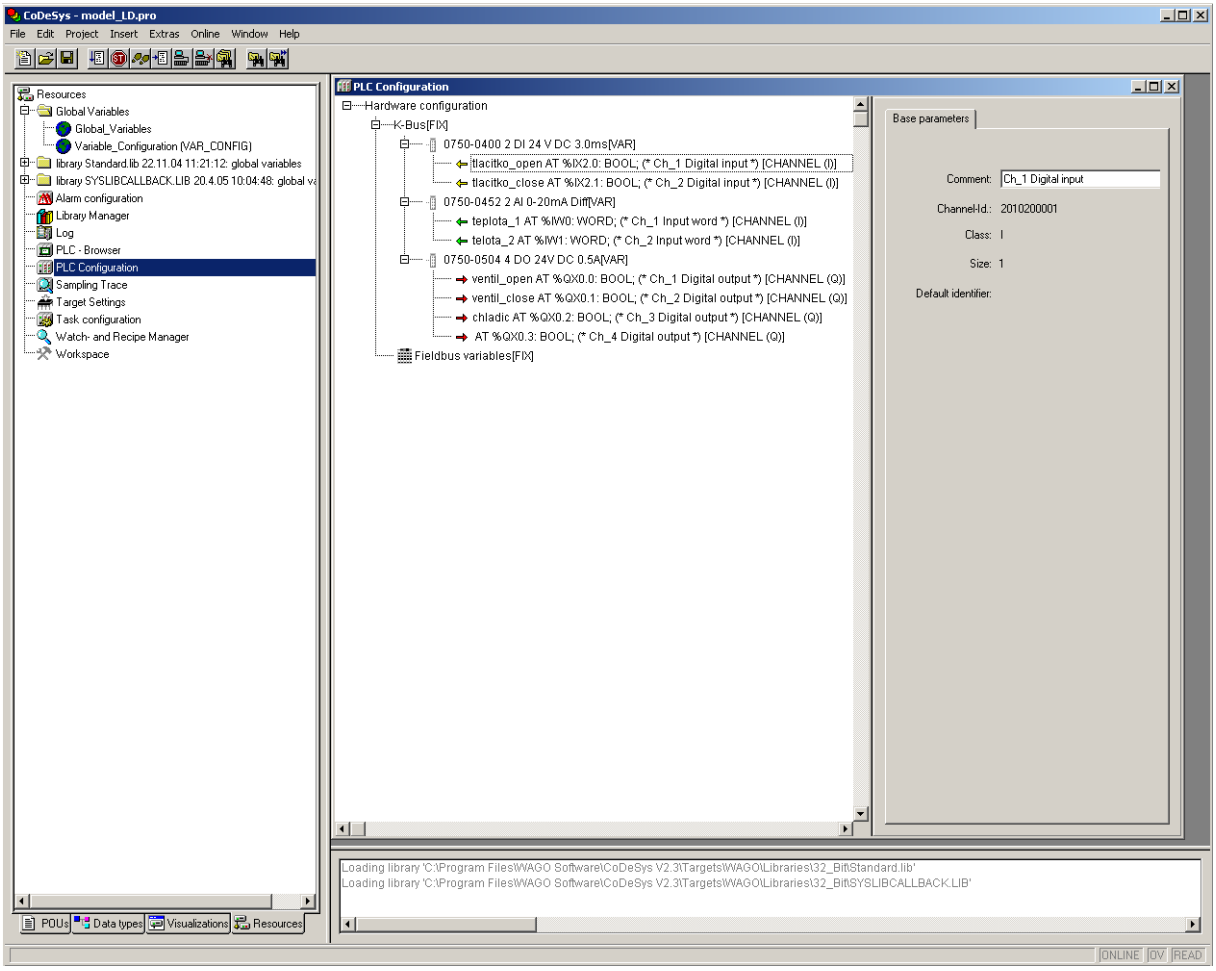
Obrázek 3.2: Task manager

3.1.1 Ukázkový program

V této kapitole budou ukázány možnosti programování PLC s užitím jazyka SFC, LD a ST. Ukázkový program bude kombinovat řízení a monitoring technologie pomocí vizualizace přímo na webserveru PLC.

Operátor má dvě tlačítka k ovládání ventilu, pro otevření a zavření. Logická hodnota log. 1 představuje akci. Ventil se otevírá 5s. Dále jsou v technologii umístěny dva teploměry (TTY 4-20mA) ke sledování teploty tekutiny. Při překročení nastavené teploty na prvním teploměru je automaticky spuštěn ventilátor chladiče. K monitorování technologie bude sestavena vizualizace.

Nejprve je nutné dle zadání definovat vstupy a výstupy pro PLC. Zde se jedná o jeden logický (tlačítko) a dva analogové (teploměry) vstupy. Dále tři logické výstupy, první pro ovládání ventilu a dva pro ovládání ventilátoru chladiče.



Obrázek 3.3: HW konfigurace PLC

Nyní jsou definované globální proměnné. Jedná se o vstupy (uvozené %I) tlacitko_open na adrese %IX2.0, tlacitko_close na adrese %IX2.1, teplomer_1 na adrese %IW0 a teplomer_2 na adrese %IW1. Výstupy jsou ventil_open na adrese %QX0.0, ventil_close na adrese %QX0.1, chladic na adrese %QX0.2. Adresové prostory pro vstupy a výstupy jsou oddělené. Zkratka %IW0 znamená vstupní slovo (1WORD ≈ 2BYTE ≈ 16bit).

Tabulka 3.2: Ukázka adresace

WORD	BYTE	bits
%IW0	%IB0, %IB1	%IX0.0 .. %IX0.15
%QW0	%QB0, %QB1	%QX0.0 .. %QX0.15

Je mnoho variant řešení. V takto jednoduchém případě je vcelku zbytečné používat SFC, ale pro přehlednost a vysvětlení pojmů bude použito. Základní kostra programu je v

jazyce ST. Program PLC_PRG plní funkci „task manageru“. Cyklicky se spouští uvedené dva programy. První `chlazení` je pro řízení ventilátoru a druhý `ventily` pro ovládání ventilu.

Na obr. 3.5 je v okně chlazeni(PRG) základní kostra v jazyce SFC.

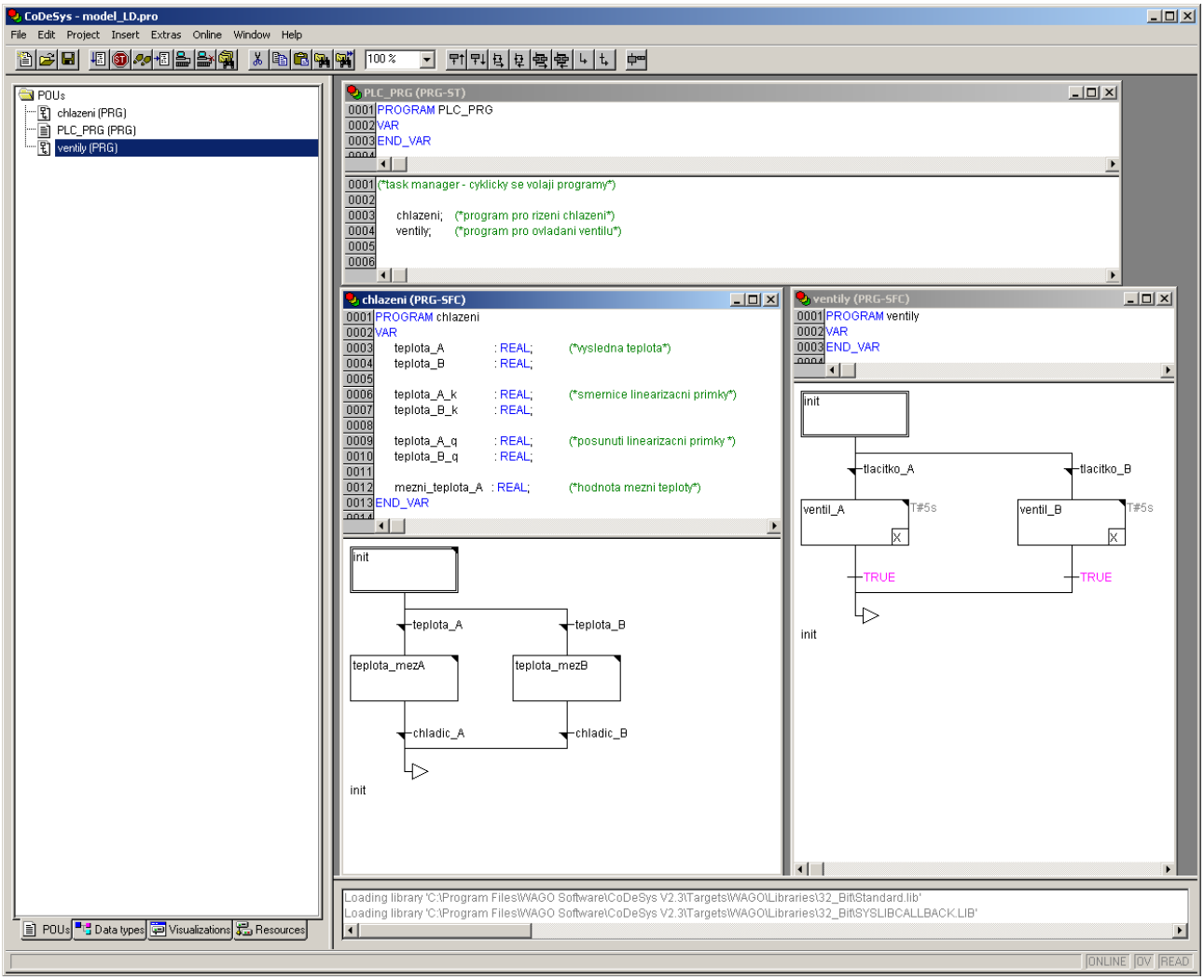
- init ... inicializace programu, přepočet teploty,
- teplota_mezA ... při překročení mezní hodnoty teploty dojde k zapnutí ventilátoru,
- teplota_mezB ... při poklesu teploty pod mezní teplotu dojde k vypnutí ventilátoru,

Na obr. 3.6 je v okně `ventily (PRG)` základní kostra v jazyce SFC.

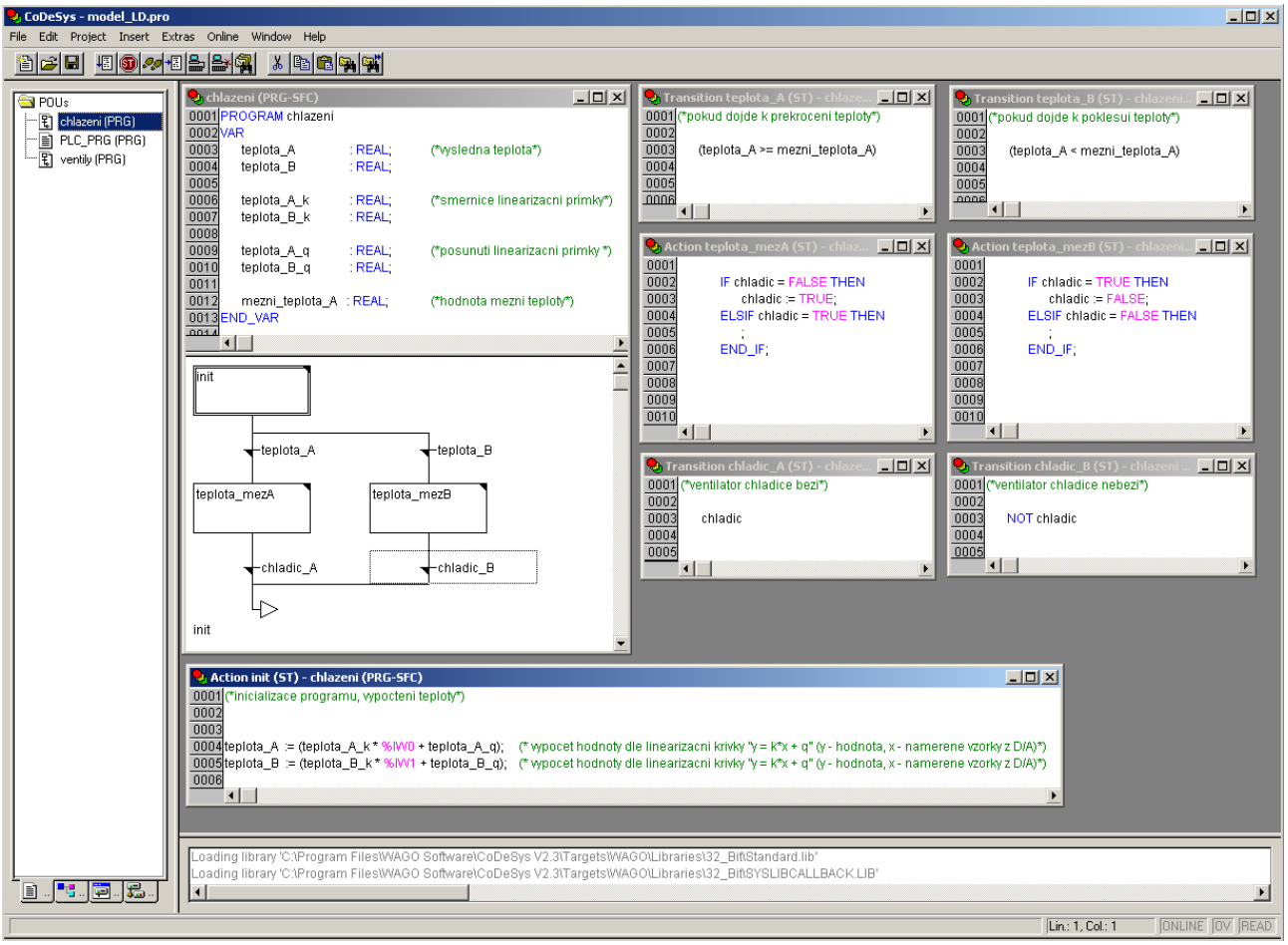
- init ... inicializace programu, přepočet teploty,
- ventil_A ... otevření ventilu, ventil se otevírá 5 s (akce délky 5 s). Křížek v pravém dolním rohu bloku představuje akci při opuštění bloku. Toho lze užít pro vypnutí servopohonu ventilu,
- ventil_B ... uzavření ventilu, ventil se uzavírá 5 s (akce délky 5 s). Křížek v pravém dolním rohu bloku představuje akci při opuštění bloku. Toho lze užít pro zavření servopohonu ventilu,

Na obr. 3.6 je pro otevření ventilu použit jazyk ST a pro zavření ventilu jazyk LD. Je zde vidět jeho hlavní charakteristika, kdy jako podmínka je kontakt[2] tlacitko_close a jako cívky[3] jsou ventil_close.

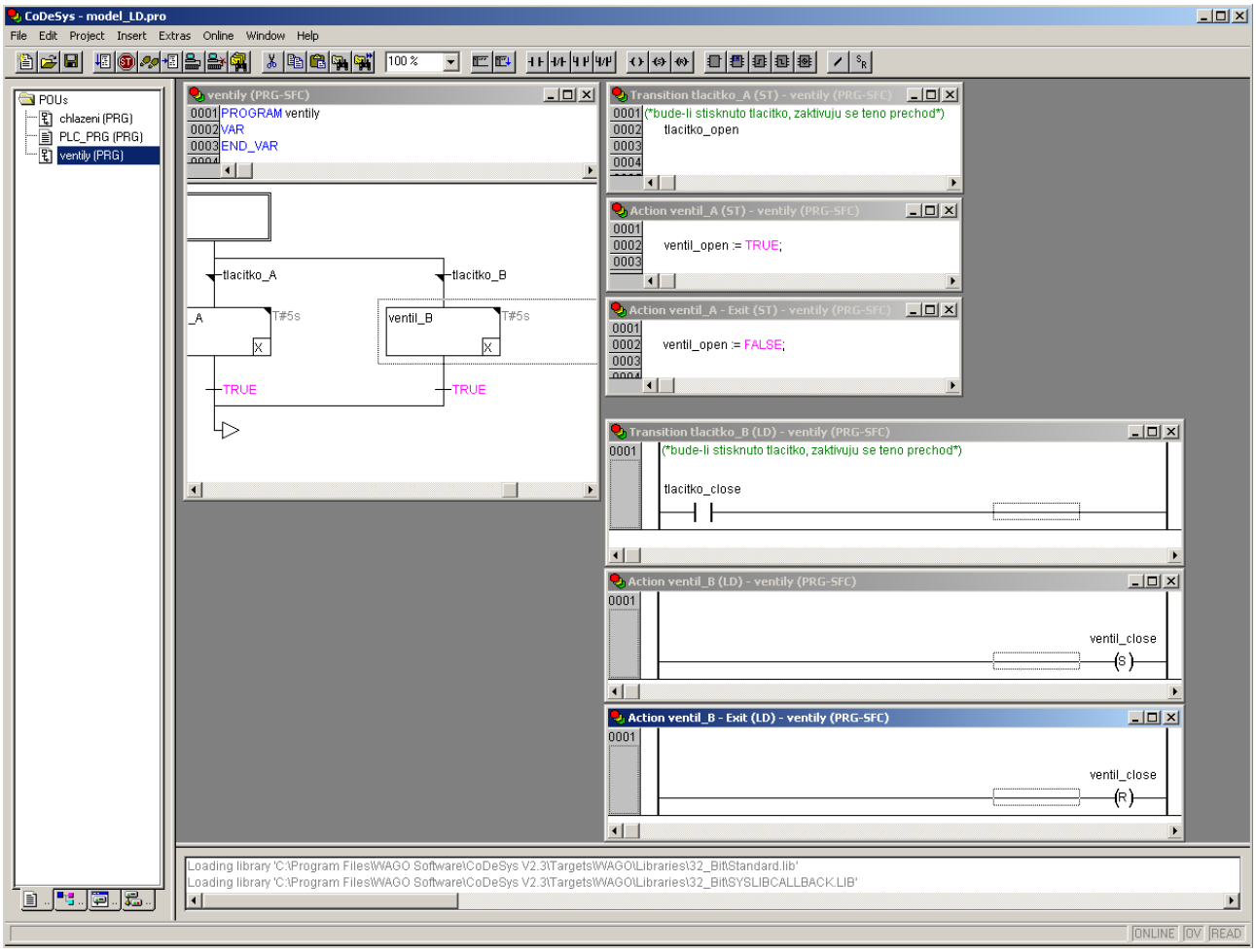
[2] v levé častí žebříkového diagramu, vlastnost logické podmínky - např. log. vstup
[3] v pravé části žebříkového diagramu, vlastnost logické akce - např. log. výstup



Obrázek 3.4: Základní struktura programu - softwarový task manager



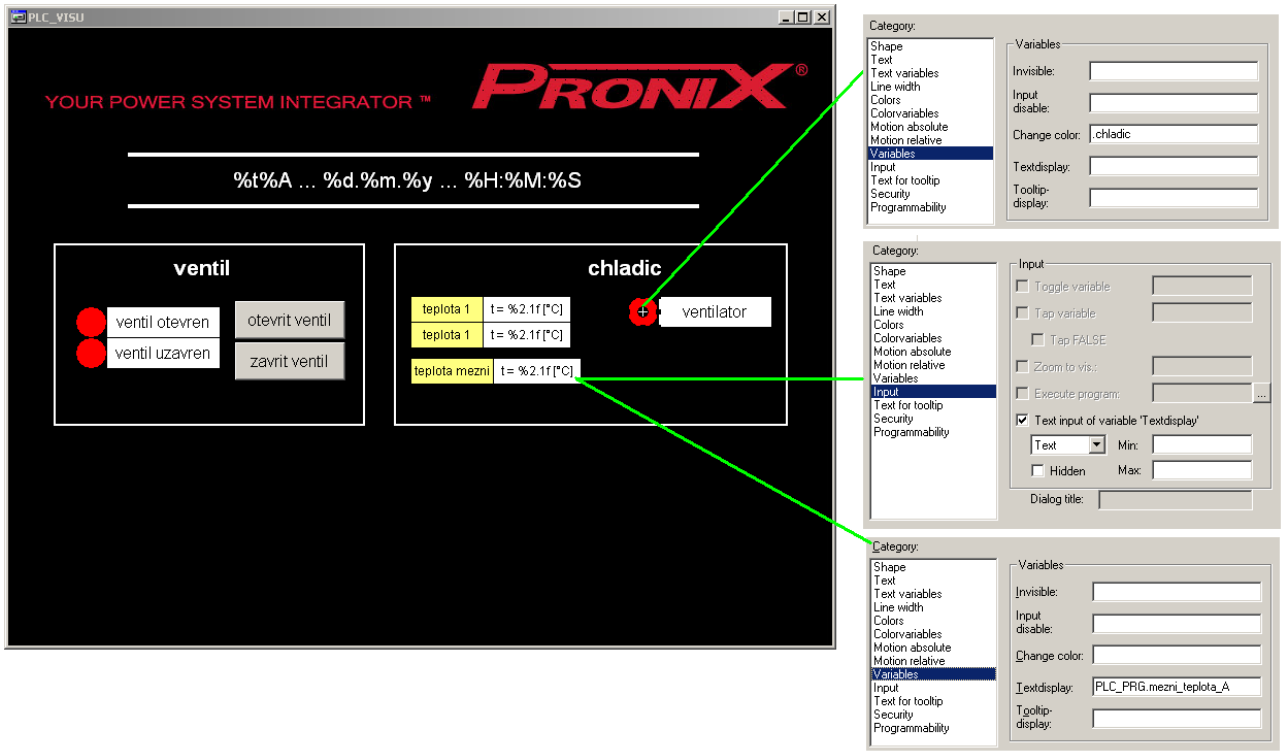
Obrázek 3.5: Program pro řízení ventilátoru chladiče



Obrázek 3.6: Program pro ovládání ventilu v kombinaci jazyků LD a ST

Ke sledování stavu technologie může sloužit jednoduchá vizualizace. Tu je možné sestavit (v základním okně CoDeSysu karta Visualization) pomocí základních nabízených objektů. Stav binárních proměnných je možné zobrazit kontrolkami. Proměnná, dle jejíhož stavu se bude měnit barva, se uvede do políčka „Change color“. Je nutné ještě doupravit barvy.

V horní části obrazovky je zobrazován den v týdnu, datum a aktuální čas v PLC. Kód `%t%A` představuje označení pro den v týdnu, `%d.%m.%y` je označení pro datum a `%H:%M:%S` pro aktuální čas.



Obrázek 3.7: Vizualizace

Pro zobrazení teploty je použit kód `t = %2.1f [°C]`. Zadá se jako text a do políčka textdisplay se uvede jméno proměnné, která se bude zobrazovat. Uvedený formát je možné použít jen pro proměnné typu `REAL` a znamená zobrazení jako float typ, v řádu desítek s jedním desetinným místem.

V tuto chvíli je již program ve fázi, kdy je možné jej nahrát do PLC. Předtím je ještě vhodné zkompilovat celý projekt klávesou F11. Zároveň CoDeSys upozorní, zda-li jsou nějaké chyby při překladu a napoví, jak mohly vzniknout.

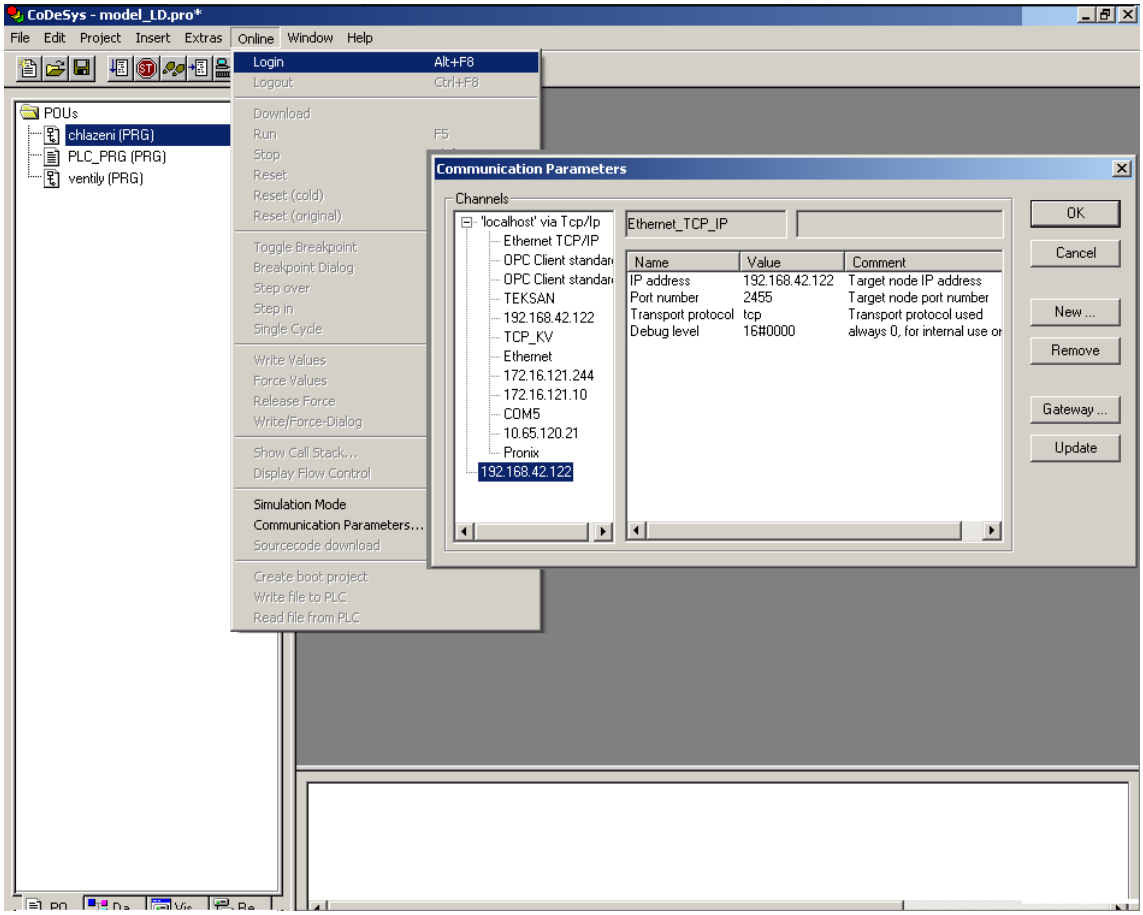
Pokud je projekt bez chyb, je možné se připojit k PLC a nahrát program. K tomu je nutné vybrat nebo zadat jednak metodu přístupu (pro WAGO 750-841 přichází v úvahu RS-232 a Ethernet) a port (COMx) nebo IP adresu[4].

Připojení k PLC se provede položkou v menu Online-Login. Poté bude program nahráván do PLC. Lze použít rozhraní RS-232 nebo Ethernet. Ten je řešen jako emulace sériové linky na rychlosti 119kBd/s. Když dojde k nahrání projektu, je nutné ještě provést spuštění programu v PLC. Bez toho nebude PLC plnit požadovanou funkci.

V případě, kdy dojde k výpadku napájení PLC, jeho resetování apod., dojde ke ztrátě programu a je nutné PLC znovu nahrát. Existuje varianta, kdy se program zavede do bootROM paměti a při restartu PLC nabootuje z tohoto programu. To se provede v menu Online-Create boot project. Zároveň je nutné přímo na PLC dát DIP přepínač do

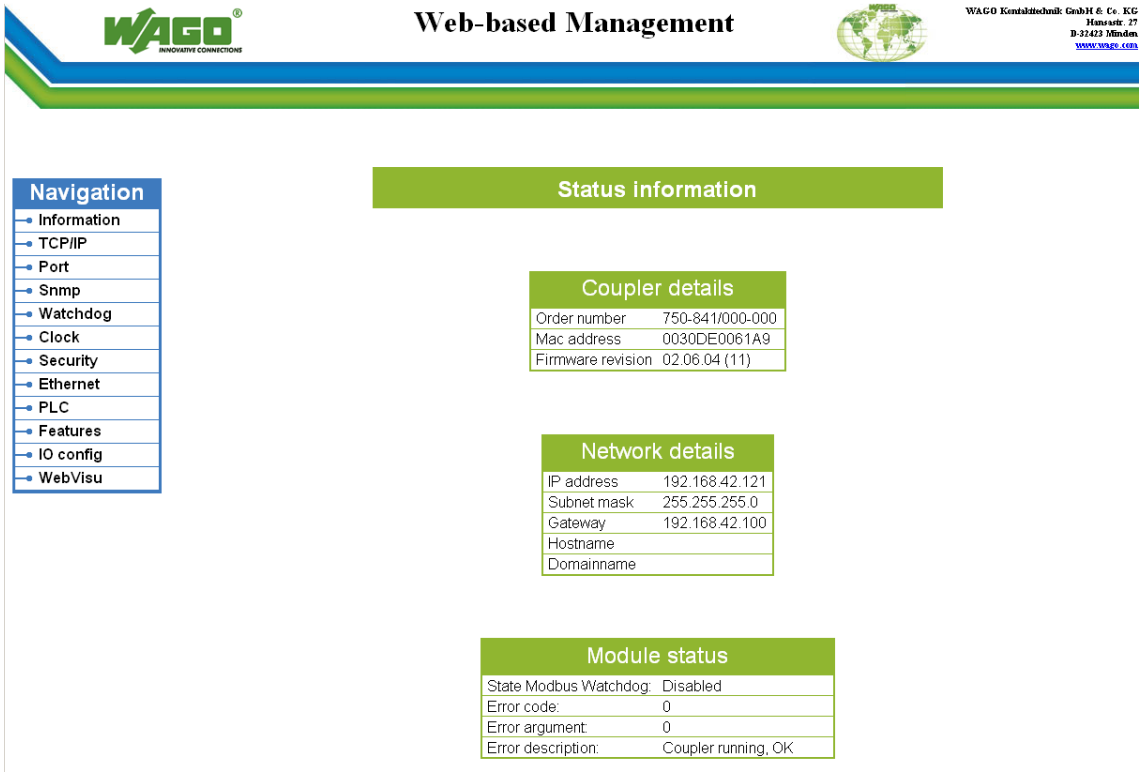
[4]pozn. pokud je CoDeSys instalován při zapnutém firewallu na lokálním PC, nebude správně instalován blok komunikačních paramatrů

„bootovací polohy“.



Obrázek 3.8: Komunikační parametry a nahrávání projektu do PLC

K PLC je možné se připojit vzdáleně přes internet, adresa je `http://'ip_adresa'/webserv/index.ssi`, většinou stačí zadat pouze IP adresu a automaticky se požadavek přesměruje na tuto.
Ale k ochraně nastavení PLC je možné nastavit přesměrování přímo na vizualizaci `http://'ip_adresa'/plc/webvisu.htm`.



Obrázek 3.9: webserver PLC

3.2 ControlWeb 5 SP11

ControlWeb je vývojové prostředí od společnosti Moravské přístroje. Jedná se o otevřený komponentový průmyslový řídicí a informační systém reálného času pro operační systémy Windows. Vytvořené aplikace je možné nasadit i na embedded systémech na bázi Windows CE. Control Web je na našem trhu špičkou v oboru.

Obrázek 3.10: Logo ControlWebu

V současnoti je k dispozici verze ControlWeb 6. Sestavená aplikace byla vyvinuta v ControlWebu 5 SP11. Koncepčně vychází z osvědčené architektury předchůdců ControlPanel a ControlWeb2000. Nasazení těchto systémů je možné od jaderných elektráren a celopodnikových informačních systémů přes tramvaje až po přímé řízení jednotlivých strojů dokazuje velmi široké možnosti těchto produktů. Distribuovanost, propojitelnost v počítačových sítích a vestavěný HTTP server je u tohoto systému i nadále naprostou samozřejmostí.

ControlWeb může pracovat stejně jako jiné SCADA/HMI systémy používaných v průmyslu. K dispozici jsou historické trendy apod. Navíc ale dodává skutečnou programovatelnost a otevřenou, komponentovou architekturu. Množina virtuálních přístrojů není pevně dána a zabudována v systému. Každý přístroj je dynamicky linkovaná knihovna detekovaná při startu systému. Není proto problém množinu virtuálních přístrojů libovolně upravovat.

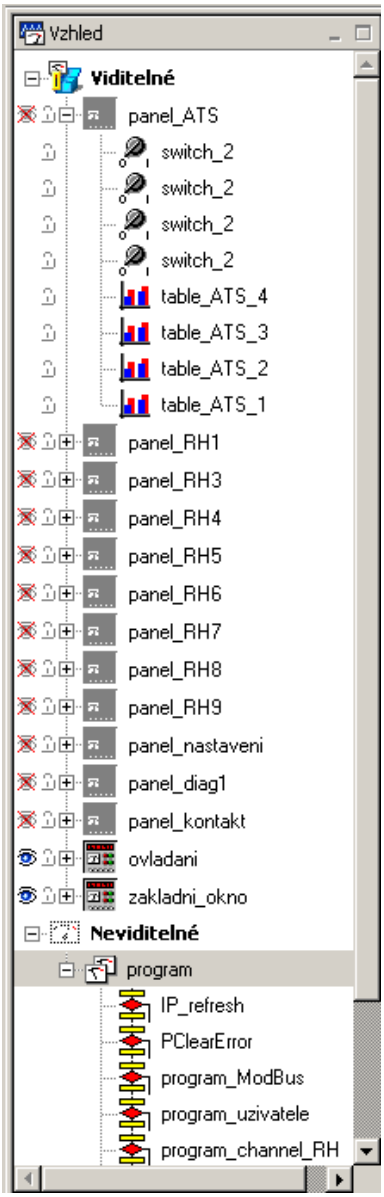
Prostředí umožňuje práci v reálném čase. Nespoléhá se na tzv. databázi tagů, která je aktualizována „maximální možnou rychlostí“(což v praxi může znamenat i intervaly několika desítek sekund mezi komunikacemi s automaty připojenými přes DDE). Každý vstupně/výstupní kanál je čten přesně v době, kdy jej nějaký virtuální přístroj (nebo skupina virtuálních přístrojů) požaduje. Real-time časování je přesně monitorováno a řízeno.

Vizualizovat technologie je možné i prostřednictvím standardů HTTP a HTML. ControlWeb obsahuje plnohodnotný HTTP server dynamicky tvořící stránky podle stavu technologie pracující i na starších operačních systémech (např. Windows 95). Navíc dokáže prostřednictvím HTTP a HTML technologii i řídit.

Výsledná aplikace nezávisí na použitém hardware. Native ovladače dokáží pracovat efektivněji než např. DDE ovladače. DDE je samozřejmě plně podporováno spolu s OPC (OLE for Process Control) a řadou dalších standardů pro průmyslové automaty, samostatné moduly a měřicí karty. Rozhraní ovladačů je plně dokumentováno a otevřeno, takže každý si může doplnit ovladač podle svých potřeb. Je zajištěna podpora nejrozšířenějších průmyslových standardů pro výměnu dat a spolupráci mezi aplikacemi - COM/OLE, ActiveX, ODBC, SQL.

3.2.1 Založení nové aplikace

Nejprve je nutné poznamenat, že lze aplikaci vytvářet v grafickém prostředí, nebo pro zkušené programátory existuje i textová verze.



Při prvním vytváření aplikace v prostředí ControlWeb je možné využít služby průvodce novou aplikací. První důležitý krok je volba mezi módem reálného času a aplikací řízenou daty.

V módu reálného času má autor aplikace zcela pod kontrolou veškeré časování systému. Základním požadavkem na takovéto aplikace je vhodný návrh struktury, aby se na daném počítači stihlo vše vykonat v předepsaném čase. Tento mód je vhodný, pokud se budou počítačem přímo regulovat spojité soustavy, řídit stroje a výrobní linky v reálném čase, realizovat složitější řídicí algoritmy, simulovat, modelovat a vizualizovat reálný systém ve vazbě na reálný čas. S výhodou je možné realizovat soft-PLC, problémem je ovšem malá odolnost vůči nevhodně zvolené konstrukci programu.

Oproti tomu v aplikaci řízené daty je činnost jednotlivých komponent odvozena od změn datových elementů, kterými jsou proměnné a kanály. Tedy pokud některý virtuální přístroj změní svou výstupní hodnotu, aktivují se veškeré přístroje, které tuto proměnnou čtou. Systém tuto činnost provádí nepřetržitě s maximální rychlostí, je ovšem nemožné tuto rychlost předem stanovit.

Aplikační program se skládá z jednotlivých virtuálních přístrojů, jedná se o samostatné programové komponenty. Tato koncepce je výhodná především pro otevřenost. Není omezen počet a typ používaných přístrojů. Aplikace je zpracovávána plným nativním výkonem počítače, kompilací dojde k překladu na ekvivalent kódu v jazyce C. Aplikace jsou plně přenositelné mezi platformami i verzemi systému.

Jednotlivé virtuální přístroje jsou propojeny hierarchickou strukturou časování a viditelnosti (levý sloupec v grafickém vývojovém prostředí). Viditelnost určuje, kde na obrazovce se bude příslušná komponenta zobrazovat a časování určuje, kdy a za jakých podmínek bude přístroj aktivován. Je vhodné jednotlivé přístroje pojmenovat unikátními jmény, aby je bylo možné volat a předávat jim hodnoty.

Jako základ aplikace slouží tzv. `panel`. Označovaný též jako kontejner. Jedná se o standardní okno systému MS Windows, lze navíc definovat kdo a jak může zavírat okna, či měnit jejich velikost. Do těchto panelů se vkládají instance komponent.

Pro vytvoření základního okna slouží následující kód. Pokud by byla aplikace reálného času, je zároveň nutné uvést položku `timer`. První parametr je periodicita spouštění přístroje a druhý představuje posunutí prvního spuštění od půlnoci. Přednastavené parametry se do kódu nezapisují.

```
rem = 'hlavni okno aplilace'; (*poznamka*)
(* timer = krok, pocatek; *)
owner = background; (*kdo je vlastnikem pristroje*)
position = 0, 0, 672, 570; (*x,y poloha leveho dolniho rohu, sirka, vyska panelu*)
win_title = 'Popisek';
win_disable = move, zoom, minimize, maximize; (*zakazane operace s panelem*)

procedure OnStartup(); (*procedura pri startu aplikace*)
begin
  Show(); (*zobrazeni panelu*)
  Select(); (*vytazeni panelu na povrch*)
end_procedure;
```

Do základního panelu bude umístěn další panel. Při poklepání myší bude otevřen panel nazvaný `panel_RH1`.

```
owner = zakladni_okno;
position = 6, 9, 54, 206;
mode = border_only;

procedure OnMouseDown( MouseX, MouseY : longint; LeftButton, MiddleButton, RightButton : boolean );
begin
     panel_RH1.Show();
     panel_RH1.Select();
end_procedure;
```

3.2.2 Datové elementy, ovladače a parametrické soubory

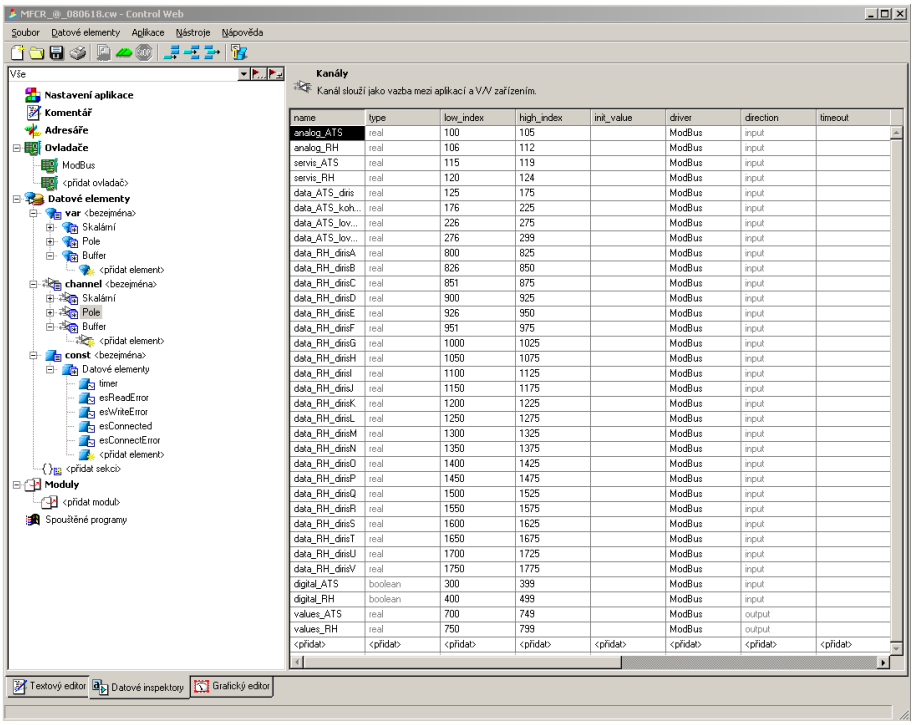
Přístroje si musí vyměňovat data. K tomu slouží globální a lokální datové elementy,

- globální ... použitelné v celém aplikačním programu, pro přenos a sdílení dat,
- lokální ... přísluší jednomu virtuálnímu přístroji, existují lokální konstanty (neměnné), proměnné (nastavovány na inicializační hodnotu) a statické proměnné (uchovávají aktuální hodnotu).

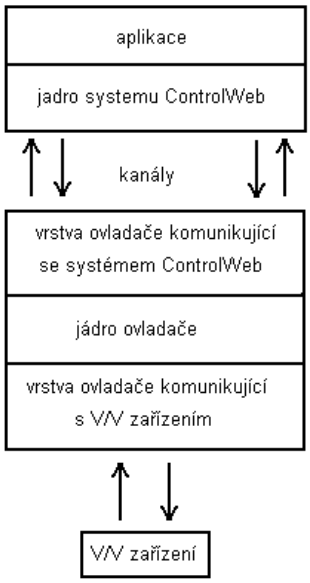
Z hlediska způsobu používání jsou rozlišeny tři základní druhy datových elementů,

- konstanty ... obsahují inicializační hodnotu,
- proměnné ... uchování dat, které se mění v průběhu běhu aplikace, globální/lokální,
- kanály ... pro přenos dat mezi aplikačním programem a vstupně/výstupními zařízeními, nutné ovladače kanálů, které zajišťují automatický přenos
  - vstupní ... z pohledu CW je možné pouze číst,
  - výstupní ... lze číst i zapisovat, čtená hodnota je však hodnotou naposledy zapsanou,
  - obousměrné ... čteny/zapisovány, čtení je provedeno přímo ze zařízení,

Kanály vzhledem k vazbě na ovladače mohou být pouze globální. Ovladače jsou totiž také globální.



Obrázek 3.11: Správa datových elementů a kanálů



Ovladač je programová komponenta, která spojuje aplikaci[5] sestavenou v prostředí ControlWeb s konkrétním zařízením. Ovladač dostává požadavky na čtení nebo zápis dat prostřednictvím kanálů.

Celý mechanismus výměny dat probíhá dle obrázku vlevo. Jádro systému informuje ovladač, že dle programu je nutné přečíst určitý údaj z kanálu. Ovladač sestaví požadavek odpovídající komunikačnímu protokolu pro příslušné zařízení a odešle požadavek do zařízení. To odpoví na požadavek a pošle ovladači odpověď v definovaném formátu. Ten dekóduje zprávu a získá příslušný dotazovaný údaj. Jádro je informováno, že je k dispozici požadovaný údaj. Tato hodnota je přiřazena do kanálu a dle kódu je hodnota dopočítána.

[5]může využívat služeb neomezeného počtu ovladačů současně

Parametrické soubory `*.par` slouží k definici činnosti ovladačů. Jejich struktura není předepsána a záleží jen na typu ovladače. Zpravidla bývají textové a lze je libovolně editovat. Důležitá poznámka se týká používání tabulátoru. Při jeho použití dojde k chybě překladu aplikace. Parametrické soubory by měly obsahovat

- nastavení komunikace s vstupně/výstupním zařízením,
- mapování kanálů do paměti vstupně/výstupního zařízení,
- další informace ovlivňující funkčnost ovladače.

Mapovací soubory `*.dmf` ukládají informace o typech a směrech všech kanálů, ketré jsou spojeny s daným ovladačem. Jsou vždy textové.

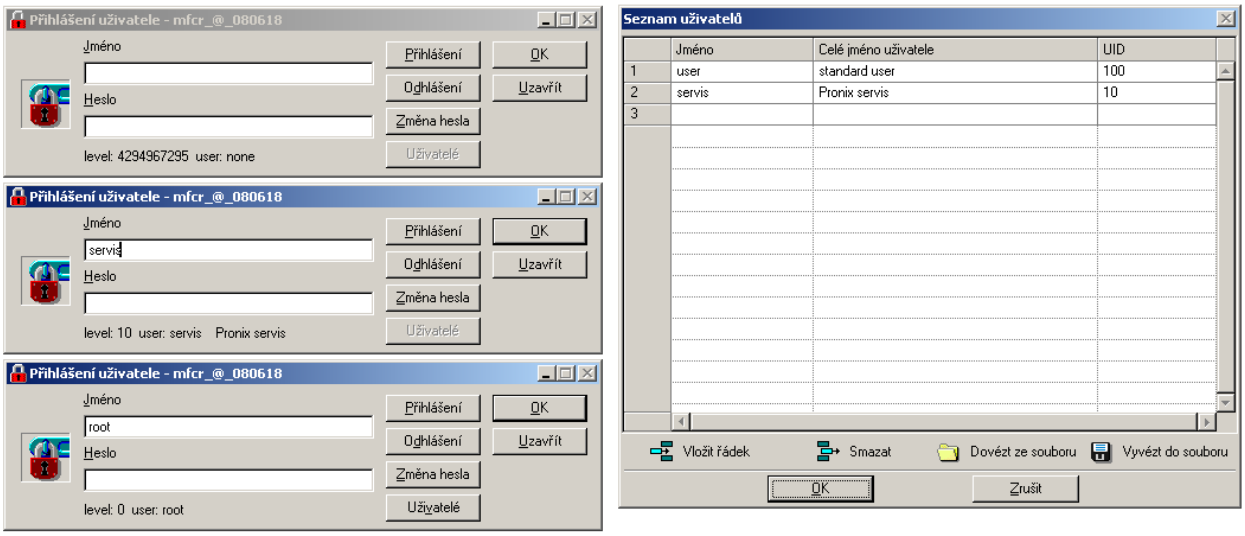
Pro správnou funkci a používání kanálů jsou potřeba ovladač `'nazev'.dll`, mapovací soubor `'nazev'.dmf` a parametrický soubor `'nazev'.par`.

3.2.3 Uživatelská práva

Vytvořené aplikace často slouží pro řízení složitých technologií. Proto je vhodné vytvořit autorizovaný přístup k technologii a vést evidenci osob pracujících se zařízením. K dispozici je nápověda, jak vytvořit uživatelské skupiny a v těchto skupinách potom vytvářet přímo jednotlivé uživatele. Hlavní správce systému označovaný jako `root` má ID `0` a nepřihlášený uživatel `none` má ID `2^32 ~ 4294967296`. To je možné zjistit z obr. 3.12.

Povolení přístupových práv je možné několika způsoby. V datových inspektorech, nebo jednodušší způsob je využít průvodce. Není-li dosud stanoveno heslo správce, průvodce upozorní na tuto skutečnost a umožní jeho zadání. Poprvé toto heslo může zadat kdokoliv.

Přidávat uživatele může jen správce. Uživatel je definován přístupovým jménem a heslem, dále lze uživatele seskupovat do skupin. To je výhodné, protože se může u stroje střídat při směnném provozu více lidí se stejným oprávněním a je vhodné rozlišit, kdo v danou chvíli na stroji pracoval. Pro lepší orientaci je možné zadávat i celé jméno a další vlastnosti.

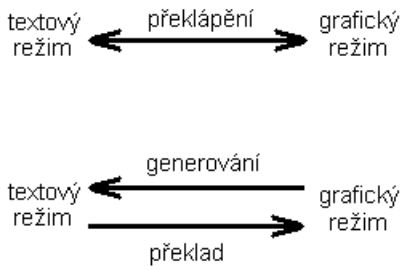


Obrázek 3.12: Uživatelské účty

Aby bylo možné pracovat s uživatelskými jmény přímo v aplikaci, je sestavený program `program_uzivatele`, který přímo čte systémovou proměnnou a ukládá do globální proměnné `uzivatel` přihlašovací jméno právě přihlášeného uživatele. Tuto proměnnou lze zobrazovat přímo v komponentě `string_display`.

```
owner = ovladani;
position = 66, 88, 57, 20;

procedure OnActivate();
var
  UserLevel : longcard;
  UserLogin : string;
  UserFull  : string;
begin
  system.GetActualUser( UserLevel, UserLogin, UserFull );
 if UserLogin = 'root' then
  uzivatel = 'root';
 elsif UserLogin = 'technolog' then
  uzivatel = 'technolog';
 elsif UserLogin = 'servis' then
  uzivatel = 'servis';
 elsif UserLogin = 'user' then
  uzivatel = 'user';
 elsif UserLogin = 'none' then
  uzivatel = 'none';
end; (*hhh*)
end_procedure;
```

3.2.4 Překlad a generování aplikace



V prostředí ControlWeb lze využít k tvorbě aplikace grafické nástroje, nebo textové programování. Oba způsoby jsou označovány jako dvojcestné programování. Každou aplikaci lze vytvářet dle aktuálního požadavku buď graficky nebo textově. Přechod mezi režimy se nazývá překlápění.

Textová i grafická podoba aplikace je navzájem zástupná. Bylo nutné zvolit jednu pro ukládání a zálohování. Vybrána byla textová podoba, protože je možné ji otevřít v textovém editoru a představuje snazší manipulaci. Dalším neméně důležitým hlediskem je v případě poruchy archivního média možnost obnovit alespoň část kódu.

Přechody mezi režimy se označují překlad (z textové do grafické) a generování (z grafické do textové).

Překlad kontroluje formální správnost textu a podle něj vytváří binární podobu aplikace. Existují tři druhy překladů. **Překlad po částech**, též označovaný jako inkrementální překlad. ControlWeb překládá aplikaci po jednotlivých částech - komponentách. Pokud se vyskytne chyba, po jejím opravení se bude pokračovat v překladu od místa, které skončilo překladem řádně bez chyby. **Jednoduchý překlad** je nejrychlejší, kontroluje se pouze syntaktická správnost aplikace. Tedy všechny třídy přístrojů exitují a systém zná knihovny, výrazy obsahují známé funkce a datové elementy mají korektní inicializační hodnoty. Ale nezavádí ovladače a nepřekládá vzdálené moduly. **Překlad pro překlopení** musí zajistit kontrolu datových souborů (zda se dají načíst) a zavést do paměti. Tento překlad se užívá při přechodu z textového do grafického režimu. **Překlad pro spuštění** kontroluje kompletní aplikaci, syntaktickou i sémantickou stránku. Provádí se vždy, při požadavku na spuštění aplikace.

Generování provádí převod z grafické formy na textovou. Výsledný generovaný text nebývá identický s tím před překlopením. Je to dáno optimalizací při generování. Především se ponechává v textu jen kód, který se liší od přednastaveného. Probíhá standardní formátování. Zároveň je zajištěna zpětná kompatibilita s předchozími generacemi systému (ControlPanel). Komentáře je nutné uvodit klíčovým slovem `rem = 'text'`. Jinak bude při překlopení ztracen.

3.2.5 Datově řízené aplikace

Takto vytvořené aplikace vyžadují pouze minimální znalost vizualizované technologie. Aplikace se o výměnu dat mezi počítačem a technologií stará sama a optimalizuje tak datový tok a rychlost odezvy vizualizace. Programátor tak získává maximální zjednodušení. To je ovšem vykoupeno nemožností časování kanálů a přístrojů, není vazba na reálný čas a tím nemožnost mít plnou kontrolu nad náročnou technologii. Doporučuje se využít tento typ aplikace v případech, kdy je požadavek archivace a vizualizace bez potřeby řídit časování dějů. Dále např. vizualizace „pomalých dějů“.

Aplikace pracuje v závislosti na množství komunikovaných dat, celý běh je řízen

jádrem. To optimalizuje rychlost komunikace dle jejich množství a zároveň spouští závislé přístroje, jejichž obsah je dán daty získanými z kanálu. Jádro pracuje cyklicky, v každém kroku změří potřebná data, vyhodnotí výsledky, aktivuje přístroje a zapíše patřičné výsledky zpět do technologie.

Aplikace vnitřně definuje umělé zdržení mezi jednotlivými kroky jádra, aby byl zajištěn chod aplikace a zbytku operačního systému. Pro verzi ControlWeb 2000 byla s ohledem na OS Windows 2000 a dostupný hardware doba zdržení 5 ms. Tato doba odpovídá dvěma stům operací za sekundu. Délka kroku jádra bývá různá, závisí na délce komunikací. Každý krok dokončí své komunikace a teprve poté se spouští přístroje. Jádro neprovádí kroky, pokud to není vyžadováno.

Komunikace kanálů probíhá automaticky. Systém zajistí vhodnou periodicitu obnovování dat, jejich bezpečné odměření a podle změny hodnot i aktivaci přístrojů, které kanály používají. Z principu se jádro snaží získávat data z kanálů co nejčastěji. To znamená, maximální rychlostí kterou dovolí komunikace a běh přístrojů. Některé aplikace nepotřebují měřit rychlostmi v řádech milisekund. Proto kanály obsahují parametr s doporučenou periodou měření kanálu. Analýzou těchto dob může dojít k značnému zvýšení zefektivnění běhu výsledné aplikace.

Výstupní kanály není třeba specialně ošetřovat. Stačí zapsat jejich novou hodnotu a o vše ostatní se starají ovladače příslušných kanálů.

3.2.5.1 Rozběh a zastavení aplikace

Je nutné zajistit, aby při rozběhu byly vždy stejné podmínky. Mechanismus rozběhu zahrnuje inicializaci výstupních kanálů. Tím se zajistí shodnost dat mezi aplikací a technologií. Dále inicializace přístrojů, nastavení počátečních hodnot přístrojům, obnovení dat ze záloh přístrojů backup a provedení první komunikace, která přenese připravená data do technologie. Následně jsou spuštěny přístroje.

Je možné také použít přístroj `startup`. Ten zajistí spuštění jádra jako první před ostatními přístroji.

Zastavení aplikace je jednodušší, lze povolit použití standardního zavíracího křížku v prostřdí MS Windows, nebo doplnit do aplikace tlačítko s následujícím kódem.

```
owner = ovladani;        (*vzdy musi byt vyplnen vlastnik pristroje*)
position = 5, 532, 118, 26;
win_disable = zoom, maximize; (*urcuje jake operace budou zakazany s pristrojem*)
mode = text_button; (*jak bude pristroj vypadat*)
font = 'Arial (Central European)', 10, normal;
true_text = 'Konec';
false_text = 'Konec';
logic = set_true_on_press; (*jakou bude pristoj vykonavat logickou funkci*)

procedure OnMouseDown( MouseX, MouseY : longint; LeftButton, MiddleButton, RightButton : boolean );
begin
        core.StopApplication();        (*samotny prikaz pro zastaveni chodu aplikace*)
end_procedure;
```

Aplikace může obsahovat přístroj `terminate`. Je nepovinný, aktivován při zastavení aplikace. Mohou v něm být obsaženy akce pro bezpečné zastavení, nebo nový start aplikace v budoucnu. Příkladem je rušení dočasných souborů či tvorba záloh. Přístroj může běžet

libovolnou dobu, zároveň probíhá komunikace i aktivování ostatních přístrojů. Při doběhu přístroje `terminate` se aplikace ukončí.

3.2.5.2 Periodické časování

Při vizualizaci v aplikaci řízené daty je také uvažovat periodické časování. Vyžaduje to tvorba grafů z naměřených dat či archivace dat. Existují tři základní způsoby.

- **Relativní (jednoduchá)perioda** - jádro aktivuje přístroj nejdříve po uplynutí definované periody, ta je vázána ke startu aplikace.
- **Absolutní perioda (s posunutím)** - perioda je určena časovým posunem vůči reálnému času. Aktivace proběhne v momentě součtu posunutí a celistvého násobku periody počítané od půlnoci
- **Časovač** - CW obsahuje přistroje, které vlastní aktivaci šíří na ostatní a tím je časují. **Sequencer** zajišťuje aktivaci dle určité posloupnosti, **selector** aktivuje na základě vyhodnocení určité podmínky, **iterator** provádí aktivaci cyklicky.

Kapitola 4

Program pro PLC WAGO

4.1 Řídicí systém rozvaděčů ATS1000 a ATS200

K sestavení konečné verze programového vybavení pro PLC automat WAGO byl použit produkt společnosti 3S-software GmbH (Smart Software Solution) nazvaný CoDeSys ve verzi 2.3.8.5 (build Oct 5 2007).

Program je sestaven jako multitaskový. CoDeSys umožňuje nastavit spouštění podprogramů pomocí menu - karta Resources-Task configuration. Lze volit mezi:

- cyclic — cyklické vykonávání tasku s definovanou periodou
- freewheeling — volně běžící task, pro technologie bez nutnosti reálného času
- triggered by event — spouštění vnější událostí

Uvedenou možnost spouštění tasků nebylo možné použít. Při přepínání mezi jednotlivými úlohami docházelo k chybám. Proto byla úspěšně vyzkoušena varianta, kdy v hlavním programu PLC_PRG(PRG) jsou volány jednotlivé programy. Je tak zajištěno cyklické vykonávání programů. Uvedené podprogramy řeší jednotlivé operace mezi nimiž je přepočet počtu vzorků z analogových měřičů proudu na výslednou hodnotu, inicializace SNMP TRAP zpráv pro změny stavu binárních vstupů, inicializace zpráv z databáze MIB, komunikace po datové lince RS-485 s užitím protokolu MODBUS RTU, řízení výkonového přepínače SOCOMEC 1250A v rozvaděči ATS1000, ovládání motorgenerátorů (start), řízení výkonového přepínače SOCOMEC 250A v rozvaděči ATS200, ovládání a řízení testu bez zátěže a testu se zátěží.

Dále jsou sestaveny funkční bloky pro přepnutí kontaktů u výkonových přepínačů. Jedná se o rutiny, které jsou vždy stejné a jejich struktura jako funkční blok je výhodná pro použití v ostatních programech.

Pro datovou komunikaci po RS-485 jsou sestaveny strukturované datové prvky pro fázová a sdružená napětí, proudy, výkony (činný, jalový, zdánlivý) a účiník. Měření probíhá na třífázové síti, proto objekty musí být strukturované a jejich položky jsou právě jednotlivé vlastnosti sítě ve fázích.

```
TYPE 'datovy typ':
        STRUCT
                f1 : [WORD, INT];
                f2 : [WORD, INT];
                f3 : [WORD, INT];
        END_STRUCT
END_TYPE
```

Objekt pro uchování informací o motorgenerátoru vypadá následovně.

```
TYPE zarizeni :
        STRUCT
                proudy : proudy_fazove;
                sdruzena_napeti : napeti_sdruzena;
                fazova_napeti : napeti_fazova;
                frekvence : WORD;
                cinny_vykon : vykon_cinny;
                jalovy_vykon : vykon_jalovy;
                zdanlivy_vykon : vykon_zdanlivy;
                ucinik : power_factor;
                total_kW : WORD;
                total_kVAr : WORD;
                total_kVA : WORD;
                total_PF : WORD;
                tlak_oleje : WORD;
                teplota_oleje : WORD;
                teplota_chlazeni : WORD;
                otacky_stroje : WORD;
                napeti_baterky : WORD;
                spotreba : WORD;
                teplota_okoli : WORD;
                ECM_baterka : WORD;
                pocet_startu : WORD;
                pocet_hodin : WORD;
                aktualni_den : WORD;
                aktualni_den_2 : STRING;
                aktualni_mesic : WORD;
                aktualni_rok : WORD;
                aktualni_hodina : WORD;
                aktualni_minuta : WORD;
                palivo : WORD;
        END_STRUCT
END_TYPE
```

Aby bylo možné k těmto údajům přistupovat i z panelového PC s prostředím ControlWeb, je každému zařízení přiřazena počáteční adresa v paměti, od které se začínají ukládat načtené hodnoty. Objekt `zarizeni` zabírá vždy 48 WORDů, ale využívají se pouze některé.

```
diris AT %MW200 : zarizeni; (*pouzito 23xWORD*)
kohler AT %MW250 : zarizeni; (*pouzito 38xWORD*)
lovato_DMK AT %MW300 : zarizeni; (*pouzito 23xWORD*)
lovato_RGK AT %MW350 : zarizeni; (*pouzito 30xWORD*)
```

4.1.1 Program `analogy`

Hodnota proudů ve fázích primární sítě je snímána proudovými měřiči, kvantována a vzorkována. Výslednou číselnou reprezentaci je nutné přepočítat pomocí linearizační křivky na skutečnou hodnotu proudu. K tomu slouží uvedený program.

```
PROGRAM analogy
VAR
        chyba_PSU : BOOL;
        proud_L1 : REAL;
        proud_L1_k : REAL := 1.0E-4;
        proud_L1_q : REAL := -3.1402E-16;
-----------------------------------------
(*====== vypocet hodnoty dle linearizacni krivky 'y = k*x + q' (y - hodnota, x - namerene vzorky z D/A) ==== *)
        proud_L1 := (proud_L1_k*%IW0 + proud_L1_q); (*_proud_L1 = %IW0*)
        IF analogy.psu < 20 THEN                (*hlidani poruchy nabijecky*)
                chyba_PSU := TRUE;
        ELSE
                chyba_PSU := FALSE;
        END_IF;
END_VAR
```

```
%REGRESE...funkce pro MATLAB%
%    function [k, q] = regrese (x,y)
%    funkce vypocita linearni regresi ze zadanych vektoru X a Y dle y = kx + q
%
%    INPUT:
%       x ... vektor hodnot osy X
%       y ... vektor funkcnich hodnot (osa y)
%    OUTPUT:
%       k ... smernice primky
%       q ... posunuti primky
%    TESTOVACI DATA:
%        x = [1 2 4 5];
%        y = [5.5 4 2 1.5];

function [k,q] = regrese(x,y)
x = x;
y = y;
delkaX = length(x);
delkaY = length(y);
if delkaX ~= delkaY
    disp(sprintf('velikost vektoru X je %d a vektoru Y je %d',delkaX, delkaY));
end
soustava(1) = delkaX;
soustava(2) = sum(x);
soustava(3) = sum(x);
soustava(4) = sum(x(:).^2);
soustava(5) = sum(y);
soustava(6) = sum(x(:).*y(:));
soustavaA = [soustava(1), soustava(2);...
             soustava(3), soustava(4)];
soustavaB = [soustava(5);...
             soustava(6)];
koeficienty = soustavaA \ soustavaB;
k = koeficienty(2);
q = koeficienty(1);
end
```

Konstanty linearizační funkce jsou vypočtené pro stávající analogovou kartu v PLC, v případě výměny senzoru nebo karty je možné konstanty změnit. Jejich výpočet lze provést pomocí lineární regresní funkce. Výše je uvedena funkce v MATLABu pro výpočet konstant lineární regresní funkce.

4.1.2 Program `init_SNMP`

Uvedený program slouží pro definici SNMP TRAP zpráv. Ty se nejprve sestaví z parametru a textu, poté probíhá inicializace fronty. SNMP TRAPy jsou posílány zároveň s jejich hod-

notou v případě, že dojde ke změně logického stavu dané logické proměnné odpovídající vstupu PLC[1].

```
PROGRAM init_SNMP
VAR
        k        : INT;                 (*index cyklu FOR (vynulovani FRONTY pri startu aplikace)*)
END_VAR
-----------------------------------------------------------------------
(*definice parametru a textu*)
(*                aaa_bbb_ccc_ddd
                        aaa ... zakaznik
                        bbb ... lokalita
                        ccc ... rozvadec, nebo urcita skupina celku
                        ddd ... logicky signal*)

Init_Variables[1].parametr:=1;           Init_Variables[1].text:= 'aaa_bbb_ccc_ddd';
...
(*inicializace fronty*)
FOR k:= 0 TO 500 DO
        traps_to_send[k].VALUE := 0;
END_FOR;
```

4.1.3 Program komunikace_MIB

Program zajišťuje poskytování analogových informací o technologii. Je vytvořena množina analogových hodnot, které je možné pomocí protokolu SNMP a tabulky MIB vzdáleně vyčítat.

Tuto funkci zpřístupňuje až poslední firmware PLC verze 14, která podporuje tabulky MIB. Zároveň je k dispozici knihovna do prostředí CoDeSys, která zajišťuje rozhraní pro vzdálený přístup. Jedná se o funkci

```
SNMP_SET_PLCDATA_WRITEAREA(writeArea_index, writeArea_Value);
```

```
PROGRAM komunikace_MIB
VAR
(********************SNMP MIB - write area*************************)
        writeArea_Value                                 : DWORD;                (*posilana hodnota*)
        writeArea_index                                 : BYTE:= 1;                (*index zpravy*)
        writeArea_wasSend                       : BOOL;                (*posilani se zdarilo*)
END_VAR
-----------------------------------------------------------------------
FOR writeArea_index :=1 TO 255 BY 1 DO
        IF                (writeArea_index = 1) THEN writeArea_Value:= REAL_TO_DWORD(analogy.proud_L1);
        ...;
        ELSIF         (writeArea_index = 7) THEN writeArea_Value:= diris_A.fazova_napeti.f1;
        ...;
        ELSIF         (writeArea_index = 27) THEN writeArea_Value:= kohler.fazova_napeti.f1;
        ...;
        ELSIF         (writeArea_index = 64) THEN writeArea_Value:= lovato_RGK.fazova_napeti.f1;
        ...;
        ELSIF         (writeArea_index = 92) THEN writeArea_Value:= lovato_DMK.fazova_napeti.f1;
        END_IF;

        SNMP_SET_PLCDATA_WRITEAREA(writeArea_index, writeArea_Value);
END_FOR;
```

[1]SNMP TRAP zprávy posílány asynchronně, viz...kapitola[2.1.1]

4.1.4 Program komunikace_ModBus

Uvedený program slouží pro přepočet hodnot z ControlWebu na použitelný formát pro PLC. Jedná se především o zadávání hodnot časů z ControlWebu, kdy je potřeba kvůli časovačům v PLC převést údaj z [s] na [ms]. Dále je zde zpracována adresa SNMP serveru. Z CW se přenáší jako čtyři čísla v rozsahu 0-255 a zde jsou čísla převedena na formát STRING a operací skládání STRING řetězců CONCAT složena celá IP adresa SNMP serveru.

```
PROGRAM komunikace_ModBus
VAR
END_VAR
-----------------------------------------------------
(*prevedeni konstant zadanych v ControlWebu na hodnoty casu*)
        IF CW_T1 = 0 THEN        (*pripad, kdy WAGO startuje a CW nedava cyklicky hodnoty na vystup*)
                CW_T1 := 10;
                ...;
        END_IF;
        IF CW_SNMP_1a = 0 THEN
                CW_SNMP_1a := 123;
                CW_SNMP_1b := 456;
                CW_SNMP_1c := 789;
                CW_SNMP_1d := 123;
        END_IF;
        T1 := DWORD_TO_TIME(CW_T1 * 1000);
        ...;
        ipconfig := CONCAT(CONCAT(CONCAT(CONCAT(CONCAT(CONCAT( WORD_TO_STRING(CW_SNMP_1a) , '.' ),
        WORD_TO_STRING(CW_SNMP_1b)) , '.'),  WORD_TO_STRING(CW_SNMP_1c)) , '.') ,  WORD_TO_STRING(CW_SNMP_1d)) ;
```

4.1.5 Program komunikace_RS485

Program zajišťuje cyklické vyčítání informací ze zařízení připojených na lince RS-485. Jako protokol je použit MODBUS RTU, kdy jednotka master je PLC a připojená zařízení jsou jednotkami slave. Pro správnou funkci je nutné přidat knihovny `Ethernet.lib`, `mod_com.lib`, `WagoLibEthernet_01.lib`, `Modb_l05.lib`, `SerComm.lib` a `serial_interface_01.lib`. Pro zajištění cyklického vyčítání zařízení je sestavena kombinace generátor-pulsů (užití komponenty BLINK) a řadiče moduloX (komponenta COUNTER UP).

```
(*Komponenta BLINK - generator pulsu*)
casovani_RS485(ENABLE:= TRUE, TIMELOW:= T#0.5s, TIMEHIGH:=T#0.5s);
```

Generátor pulsů je použit pro časování řadiče, který inkrementuje svou hodnotu. Podle počtu vyčítaných zařízení je určen parametr X řadiče.

```
(*Komponenta CTU - citac modulo*)
radic_RS485(CU:=casovani_RS485.OUT, RESET:=radic_RESET, PV:=radic_PV);
radic_RESET := radic_RS485.Q;
radic_CV := radic_RS485.CV;
```

Samotné vyčítání a zpracování načtených údajů probíhá následovně. V momentě, že je v řadiči požadovaná hodnota a generátor má aktivní puls, předá se příslušná adresa a

délka vyčítaných dat a provede se vyčtení, zpracování údajů. V neaktivní půlperiodě je vyčkávání a ukládání.

```
(*Zarizeni na RS485 na ModBus adrese 5 - DIRIS *)
IF radic_CV = 0         THEN
        zarizeni_A :='DIRIS_1@1';
        adresa_Slave := 5;
        adrese_v_zarizeni_slave := 768;
        pocet_prenasenych_dat := 22;
        start_RS485        := TRUE;

                diris_A.proudy.f1        := ((prijata_data.Data[5] * 256 + prijata_data.Data[6])/100);
ELSIF casovani_RS485.OUT = FALSE THEN
        start_RS485 := FALSE;
END_IF;
```

Nastavení parametrů se provede následovně.

```
PROGRAM komunikace_RS485
VAR
        casovani_RS485 : BLINK; (*definice generatoru pulsu BLINK*)
        radic_RS485 : CTU; (*definice citace CTU*)
        ModBus_Master : MODBUSMASTER_RTU; (*komunikace po kanale MODBUS*)

(*kruhovy radic RS485 realizovany jako citac modulo 8*)
        radic_RESET        : BOOL;
        radic_PV : WORD := 9;        (*prednastavena hodnota - pocet zarizeni na sbernici*)
        radic_CV : WORD; (*aktualni hodnota*)

(*RS485 - komunikace po kanale MODBUS*)
        adresa_Slave : BYTE; (*adresa Slave zarizeni, typ BYTE, standardne 5, jinak nastavit*)
        kod_funkce : BYTE; (*kod pozadovane funkce - 3~ReadHoldingRegister, 5~ForceSingleCoil*)
        pocatecni_adresa_pameti : WORD;        (*pocatecni adresa v zarizeni Slave od niz se cte/zepisuje*)
        pocet_prenasenych_dat : WORD;        (*pocet dat prenasenych v jednom cteni/zapisu*)
        zarizeni_na_sbernici : BYTE := 2; (*poradove cislo zarizeni na sbernici RS485 - Master~16#02*)
        rychlost_prenosu : COM_BAUDRATE := 960; (*rychlost prenosu po sbernici - rychlost delena 10*)
        typ_parity : COM_PARITY := 2;        (*typ parity 0~NO, 1~ODD-licha, 2~EVEN-suda *)
        pocet_stopbitu : COM_STOPBITS := 1; (*pocet StopBitu 1~jeden, 2~dva*)
        pocet_bitu : COM_BYTESIZE := 8; (*pocet datovyxh bitu 7~sedm, 8~osm*)
        rizeni_prenosu : COM_FLOW_CONTROL := 4;        (*0~NO, 1~Xon/Xoff, 2~RTS/CTS, 3~FullDuplex, 4~HalfDuplex*)
        start_RS485 : BOOL; (*zahajeni komunikace po RS485*)

        prijata_data : typRING_BUFFER;(*buffer, kam se ukladaji prijata data, nutno RING_BUFFER*)
        odesilana_data : typRING_BUFFER; (*buffer, odkad se ctou data, nutno RING_BUFFER*)

        ErrorCode        : BYTE;        (*Kod Chyby - viz.navod ke zarizeni Slave*)
END_VAR
```

Následuje samotný program pro vyčítání dat po lince RS-485. Může se stát, že je potřeba vyčítat větší množství dat, ale na jedno vyčtení je omezený rozsah. Proto je nutné provést více přístupů od různých adres, jak je naznačeno.

```
(*Komponenta BLINK - generator pulsu*)
casovani_RS485(        ENABLE:= TRUE,
                                        TIMELOW:= T#0.5s,
                                        TIMEHIGH:=T#0.5s);

(*Komponenta CTU - citac modulo*)
radic_RS485(        CU:=casovani_RS485.OUT,
                              RESET:=radic_RESET,
                              PV:=radic_PV);

radic_RESET := radic_RS485.Q;
radic_CV := radic_RS485.CV;

        ErrorCode := ModBus_Master.Error;
                 IF ErrorCode = 0        THEN
                        error := 'OK';
                ELSIF ErrorCode = 1        THEN
                        error := 'Illegal Function or Address';
                ELSIF ErrorCode = 2        THEN
                        error := 'Illegal DataValue';
                ELSIF ErrorCode = 3        THEN
                        error := 'Parameter OutOfRange';
                ELSIF ErrorCode = 4        THEN
                        error := 'Illegal Variable Format';
                ELSIF ErrorCode = 6        THEN
                        error := 'Slave Busy'; (*...*)
                ELSIF ErrorCode = 152        THEN
                        error := 'Not Connected Er152'; (*...*)
                ELSIF ErrorCode = 153        THEN
                        error := 'Not Connected Er153'; (*...*)
                END_IF;

(*Zarizeni na RS485 na adrese 5 - DIRIS *)
IF radic_CV = 0 THEN
        zarizeni :='DIRIS_1@1';
        adresa_Slave := 5;
        adrese_v_zarizeni_slave := 768;
        pocet_prenasenych_dat := 22;
        start_RS485        := TRUE;

                diris_A.proudy.f1 := ((prijata_data.Data[5] * 256 + prijata_data.Data[6])/100); (*~0.1 A*)
                diris_A.proudy.f2 := ((prijata_data.Data[9] * 256 + prijata_data.Data[10])/100);
                diris_A.proudy.f3 := ((prijata_data.Data[13] * 256 + prijata_data.Data[14])/100);
                diris_A.proudy.f0 := ((prijata_data.Data[17] * 256 + prijata_data.Data[18])/100);

ELSIF casovani_RS485.OUT = FALSE THEN
        start_RS485 := FALSE;
END_IF;
(*-------------------------------------------------------*)

IF radic_CV = 1        THEN
        zarizeni :='DIRIS_1@2';
        adresa_Slave := 6;
        adrese_v_zarizeni_slave := 798;
        pocet_prenasenych_dat := 30;
        start_RS485 := TRUE;

                diris_A.cinny_vykon.f1 := prijata_data.Data[5] * 256 + prijata_data.Data[6];
                diris_A.cinny_vykon.f2 := prijata_data.Data[9] * 256 + prijata_data.Data[10];
                diris_A.cinny_vykon.f3 := prijata_data.Data[13] * 256 + prijata_data.Data[14];

ELSIF casovani_RS485.OUT = FALSE THEN
        start_RS485 := FALSE;
END_IF;
```

4.1.6 Program PLC_PRG

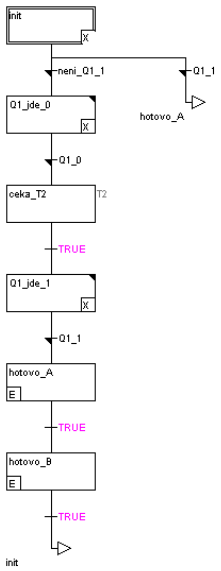
Hlavní program, vždy musí být obsažen jeden s tímto jménem. V tomto systému je použit pouze jako taskmanager.

```
PROGRAM PLC_PRG
VAR
                monitor : monitor_SFC;
                memory : memory_SFC;
END_VAR
----------------------------------------------------------------
(**AUTOMATICKY REZIM -- osetreni MG1 - ATS1000 -- SOCOMEC 1250A*)
        IF (System_On AND Auto AND Q1_On_Automat_Mode AND NOT Q1_On_Blokade_Mode) THEN
                rizeni_gen1;(*spusteni programu-tasku pro rizeni rozvadece ATS1000*)
        END_IF;

(**AUTOMATICKY REZIM -- osetreni MG2 - ATS200 -- SOCOMEC 250A*)
        IF (System_On AND Auto AND Q2_On_Automat_Mode AND NOT Q2_On_Blokade_Mode) THEN
                rizeni_gen2;(*spusteni programu-tasku pro rizeni rozvadece ATS200*)
        END_IF;

(**TEST BEZ ZATEZE**)
        IF (System_On AND  Test_Without_Load  AND NOT Q1_On_Blokade_Mode AND NOT Q2_On_Blokade_Mode) THEN
                rizeni_test_A;(*spusteni programu-tasku pro rizeni testu bez zateze*)
        END_IF;

(**TEST SE ZATEZI**)
        IF (System_On AND Test_With_Load AND Q1_On_Automat_Mode AND NOT Q1_On_Blokade_Mode AND NOT Q2_On_Blokade_Mode) THEN
                rizeni_test_B;(*spusteni programu-tasku pro rizeni  testu se zateze*)
        END_IF;


        analogy; (*spusteni programu-tasku pro prepocet namerenych hodnot z A/D prevodniku*)
        komunikace_RS485; (*spusteni programu-tasku pro komunikaci po RS-485*)
        komunikace_ModBus; (*spusteni programu-tasku pro komunikaci s panelem CW*)
        monitor; (*spusteni instance FB pro sledovani novych bool udalosti*)
        memory; (*spusteni instance FB pro ukladani bool udalosti do pole udalosti*)
        komunikace_MIB; (*spusteni programu-tasku pro definici SNMP analogovych hodnot*)

(**nastaveni promennych        pro sdilene promenne pres UDP-ethernet**)
        GEN_Volt_OK                := GEN1_Voltage_Is_OK;
```

4.1.7 Program rizeni_gen1

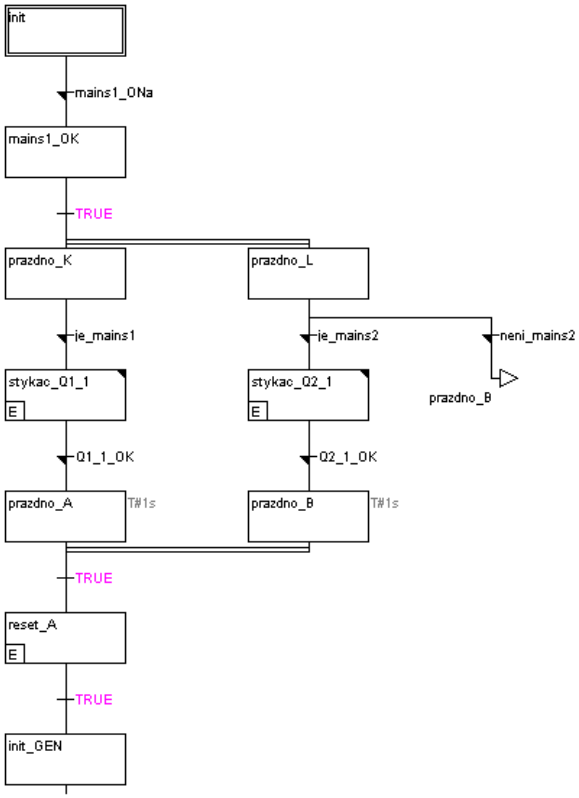
Tento program je pro přehlednost a především pro svůj sekvenční charakter sestaven v SFC (Sequential Function Chart).

Pokud dojde k zapnutí programu, musí být zajištěno definované chování a uvedení celého systému do určitého stavu, který je možné označit jako výchozí. Jedná se především o nastavení výkonových přepínačů do polohy na primární síť v případě, že je síť přítomna a přepínač je v jiné poloze. Naopak, pokud již na síť přepnuto je, musí tak zůstat. Proto je v mechanismu přepínání (viz obr. 4.1) nejprve testována podmínka v jaké je stykač poloze. Následně je přepnuto do nulové pozice a čekáno dobu T_2.



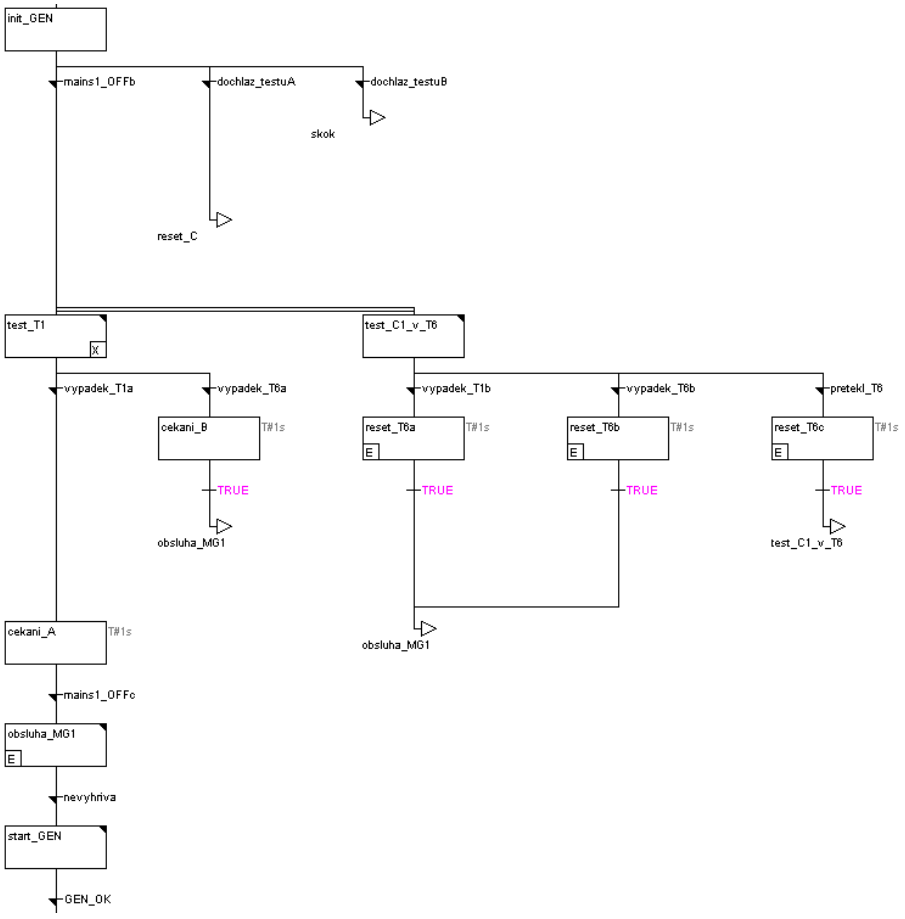
Obrázek 4.1: Blok programu přepnutí stykače do polohy 1

Po přepnutí bude program čekat v `init_GEN`, dokud nenastane výpadek na primární síti.

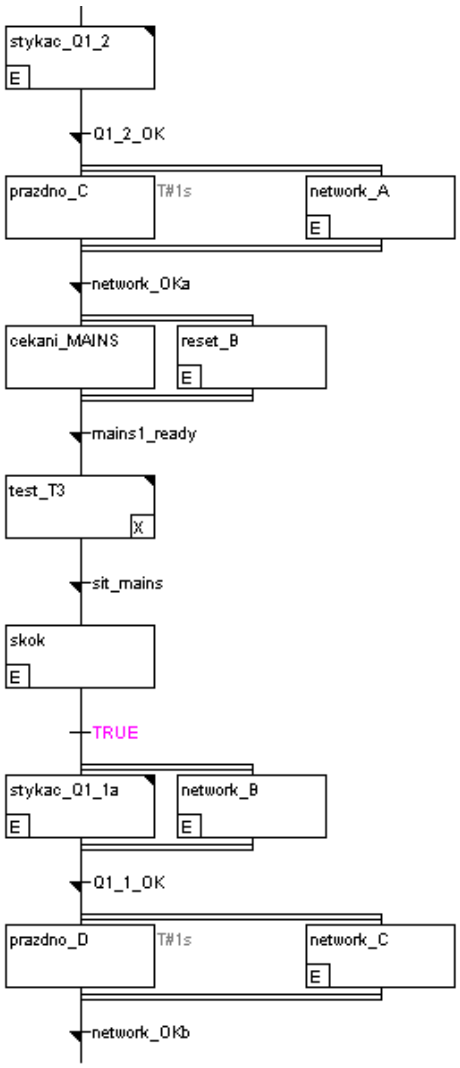


Obrázek 4.2: První část programu - nastavení do výchozího stavu

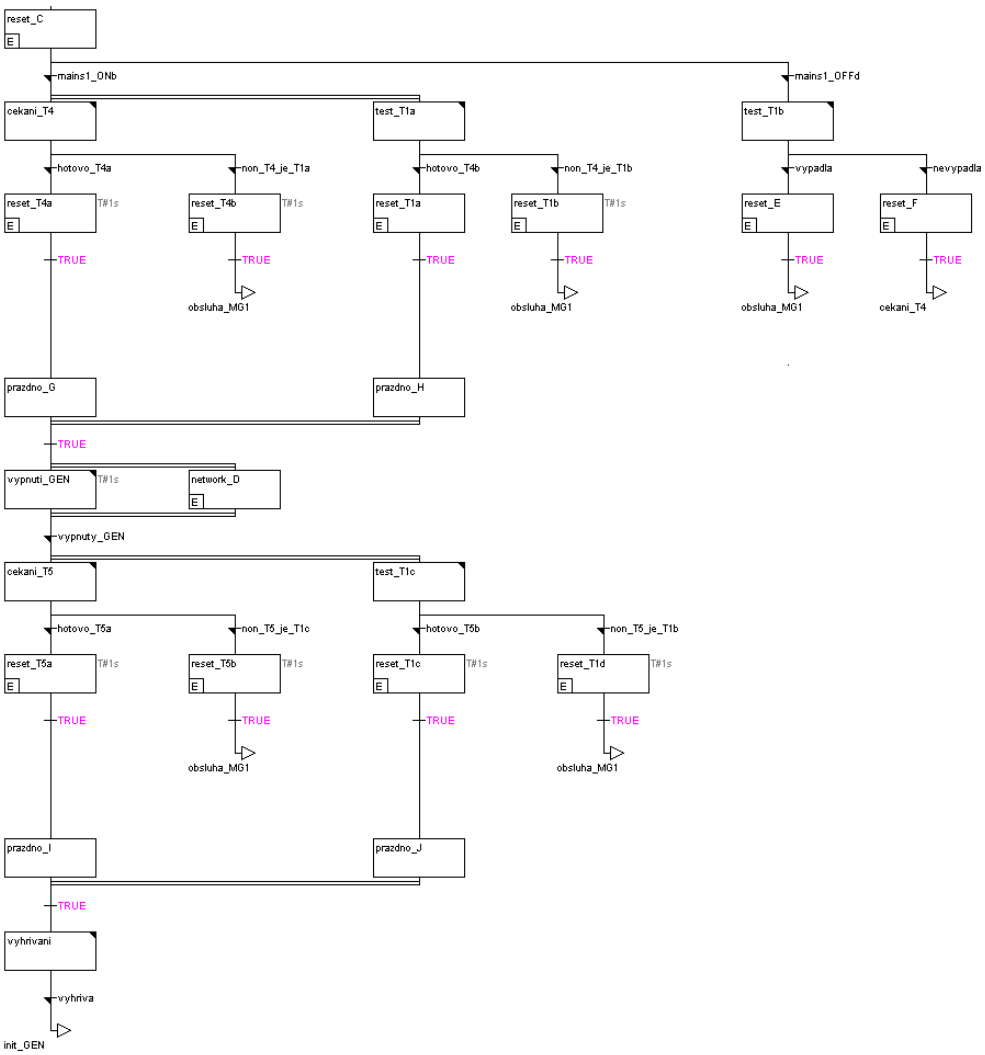
Pokud nastane výpadek, je paralelně testována délka výpadku T_1 a zároveň počítáno, kolikátý je to výpadek v době T_6. Dále je startován agregát. Na přechodu `GEN_OK` se čeká, až bude generátor schopen dodávat energii v požadované kvalitě.



Obrázek 4.3: Druhá část programu - testování výpadku, start agregátu



Obrázek 4.4: Třetí část programu - přepnutí na generátor, testování sítě po dobu T_3, přepnutí zpět na síť



Obrázek 4.5: Čtvrtá část programu - dochlazení a vyhřívání agregátu

4.2 Řídící systém nové rozvodny

Jak již bylo poznamenáno dříve, pro zajištění záložního napájení nejdůležitější části technologie byl realizován systém odpínání a připínaní outletů v závislosti na množství pohonných hmot v nádrži motorgenerátoru. Instalovaný motorgenerátor KOHLER má spotřebu asi 80 l nafty na hodinu provozu při plném zatížení. Nádrž je dimenzovaná pro bezproblémový chod na plný výkon po dobu 24 hodin bez nutnosti doplnění paliva. Kromě toho je k dispozici ještě záložní motorgenerátor SDMO o přibližně desetinovém výkonu. Ten slouží pouze v kritických situacích, kdy dojde k poruše primárního zálohování.

Program je opět řešen jako multitaskový, kdy jsou jednotlivé podprogramy cyklicky spouštěny. K řízení jsou použity i vybrané signály z rozvaděče ATS 1000. Jedná se o signály informující o přítomnosti primárních sítí, napětí dodávaném motorgenerátory apod.

Hlavní program pro řízení je sestaven v SFC diagramu. Obsahuje dvě základní větve. Druhá, jednodušší na popis, řeší připínání outletů po znovupřipojení technologie k primární síti. Zde je nutné připnout outlety v určitých časových intervalech. Logická podmínka v přechodu označeném `MAINS` na obr. 4.6 je přepnutí stykače SOCOMEC ATyS do polohy na síť, signál z rozvaděče ATS 1000 (proběhlo v pořádku testování kvality sítě po dobu T_3), je přítomna primární síť a zatím po objevení sítě připnutí neproběhlo (tzn. outlety budou přinuty poprvé, aby nedošlo k zacyklení programu).

Všechny akce mají stejnou funkci, proto bude popsána jen jedna. Akce označená `priorita_2_B` připojuje outlety z druhé priority. Nejprve je v náběžné akci (označena v levém dolním rohu písmenem `E`) provedeno spuštění časovače pojmenovaného `outlet_T1` příkazem `start_priorita2:=TRUE;`, který zajistí časové rozestupy mezi připnutím outletů. Následující část kódu ukazuje počítání časového rozestupu časovačem `outlet_T1`, po dopočítání je podmínka pro start časovače zakázána a je spuštěn druhý časovač `prepnutí_stykace`. Ten zajistí požadovaný časový interval, dostatečný k přepnutí stykače a připnutí outletů.

Dále je nutné určit, které outlety v dané prioritě budou připnuty a které nikoliv. Uživatel může ve vizualizaci určit seznam připínaných outletů. Stykače zároveň plní nadproudovou ochranu. Tedy v případě, kdy je nadproudová ochrana vypadlá, není potřeba posílat požadavek na připnutí stykače, protože je nutné, aby přišla obsluha osobně a nejprve odstranila chybu a následně ručně provedla natáhnutí. V momentě, kdy určitý outlet je již připnut nebo vypadla nadproudová ochrana (signál `_7_FA7_E`), nebude se posílat požadavek na připnutí (signál `_7_FFA7_ON`).

Po dopočítání času je nastavena proměnná `priorita_2_OK`, která informuje, že je možné pokračovat v připínaní další priority. Nejprve je zkontrolována podmínka `(NOT _7_FA7_E AND _7_FA7_ON) OR (_7_FA7_E AND NOT _7_FA7_ON)`. Tedy je připnuto (v tom případě není vypadlá nadproudová ochrana), nebo je vypadlá nadproudová ochrana a nemůže být připnuto.

```
outlet_T1(IN := start_priorita2, (*pocitani casu k pripojeni technologie priority 2*)
                        PT:= T1);

start_priorita2 := FALSE;

prepnuti_stykace(        IN := NOT outlet_T1.Q, (*puls delky T_stykac k prepnuti DEONu*)
                                        PT:= T_stykac);

IF prepnuti_stykace.Q THEN
        IF  NOT _7_FA7_E AND NOT _7_FA7_ON THEN        (*V11_ucebna*)
                _7_FFA7_ON := TRUE;
        ELSIF _7_FA7_E THEN
                _7_FFA7_ON := FALSE;
        END_IF;
ELSE
        _7_FFA7_ON := FALSE; (*V11_ucebna*)
END_IF;

(*reseni podminky osetreni dalsi priority*)
IF NOT prepnuti_stykace.Q AND        (((NOT _7_FA7_E AND  _7_FA7_ON) OR (_7_FA7_E AND NOT _7_FA7_ON)) THEN
        priorita_2_OK := TRUE; (*pokud je vypadla nadproud. ochr. a neni ZV od pripojeneho DEONu*)
END_IF;
```

Před opuštěním akce se ještě resetují oba časovače. Je to z praktických důvodů, aby byl jasně definován jejich stav při dalším cyklu. Stávalo se, že časovače zůstaly dopočítané, nebo se přednastavily nesmyslné hodnoty.

```
prepnuti_stykace(IN := FALSE, (*puls delky T_stykac k prepnuti DEONu*)
                                        PT:= T_stykac);
outlet_T1(IN  := FALSE,
                        PT := T1);
```

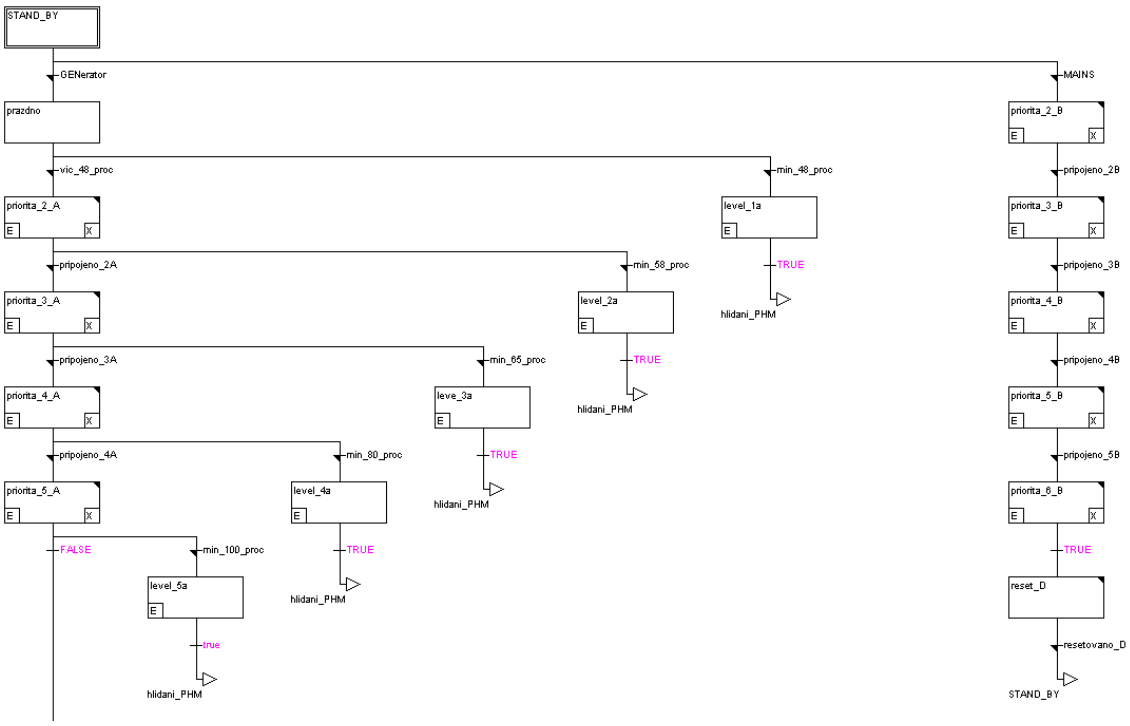
Následující přechod `pripojeno_2B`) testuje podmínky:

```
                priorita_2_OK
AND        NOT prepnuti_stykace.Q
```

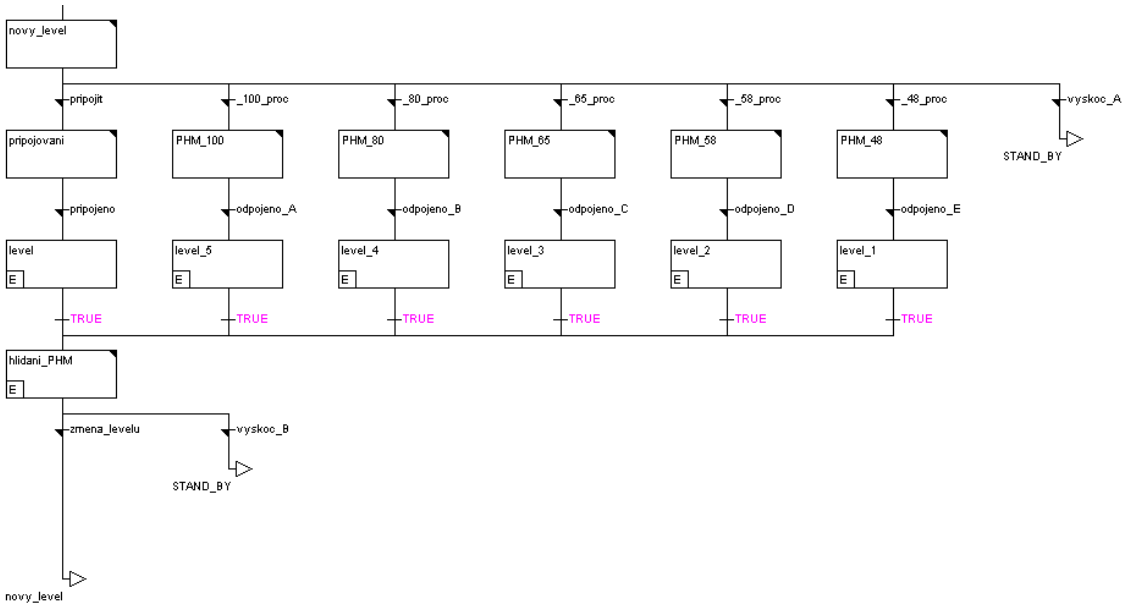
První větví programu (viz obr. 4.6) je řešení připnutí outletů v případě zálohování z motorgenerátoru. Zde je nutno řešit připínání s ohledem na množství paliva a zároveň respektovat časové rozestupy.

Logická podmínka v přechodu označeném `GENerator` je podobná jako v předchozím. Musí být přepnut stykač SOCOMEC ATyS do polohy na generátor, signál z rozvaděče ATS 1000 (byl vyhodnocen výpadek) a je přítomno napětí z generátoru. Dále se větev dělí podle množství paliva. Přechod `vic_48_proc` testuje podminku `PHM_ted >= 48`. Pokud podmínka není splněna (paliva je méně), pokračuje se druhou větví, nastaví se proměnná `level_minule:=1` a skáče se na akci `hlidani_PHM`. Zde se čeká až do doby, než dojde k doplnění paliva.

V akci nazvané `hlidani_PHM` se provádí přiřazení aktuální hodnoty do proměnné `level_minule` v závislosti na hladině.



Obrázek 4.6: První část hlavního programu - ošetření připínání outletů



Obrázek 4.7: Druhá část hlavního programu - ošetření připínání/odpínání outletů dle množství paliva

```
(*---------------------rozmezi 0 - 48%-----------------------------------*)
        IF PHM_ted < 48 THEN
                level_ted := 1;
(*---------------------rozmezi 48 - 58%----------------------------------*)
        ELSIF (PHM_ted >= 48 AND PHM_ted <= 58) THEN
                level_ted := 2;
(*---------------------rozmezi 58 - 65%----------------------------------*)
        ELSIF (PHM_ted > 58 AND PHM_ted <= 65) THEN
                level_ted := 3;
(*---------------------rozmezi 65 - 80%----------------------------------*)
        ELSIF (PHM_ted > 65 AND PHM_ted <= 80) THEN
                level_ted := 4;
(*---------------------rozmezi 80 - 100%---------------------------------*)
        ELSIF (PHM_ted > 80 AND PHM_ted <= 100) THEN
                level_ted := 5;
        END_IF;
```

Pokud dojde ke změně hladiny, vyhodnotí se podmínka v přechodu `zmena_levelu` - (`level_minule <> level_ted`) a program skočí na akci `novy_level`. Zde se bude rozhodovat. Pokud došlo k doplnění paliva, bude program pokračovat hned první větví a provede připnutí outletu z nového levelu. Program poté bude opět čekat na novou změnu hladiny.

```
(*---------------------rozmezi 0 - 48%-----------------------------------*)
        IF PHM_ted < 48 THEN
                pripnuto_OK := TRUE;
(*---------------------rozmezi 48 - 58%----------------------------------*)
        ELSIF (PHM_ted >= 48 AND PHM_ted <= 58) THEN
                IF prepnuti_stykace.Q THEN
                        IF  NOT _7_FA6_E AND NOT _7_FA6_ON THEN (*V11_test prac./ucebna*)
                                _7_FFA6_ON := TRUE;
                        ELSIF _7_FA6_E THEN
                                _7_FFA6_ON := FALSE;
                        END_IF;
                ELSE
                        _7_FFA6_ON := FALSE; (*V11_test prac./ucebna*)
                END_IF;
                IF NOT prepnuti_stykace.Q AND (((NOT _7_FA6_E AND  _7_FA6_ON) OR (_7_FA6_E AND NOT _7_FA6_ON)) THEN
                        pripnuto_OK := TRUE; (*pokud je vypadla nadproud. ochr. a neni ZV od pripojeneho DEONu*)
                END_IF;
(*---------------------rozmezi 58 - 65%----------------------------------*)
        ELSIF (PHM_ted > 58 AND PHM_ted <= 65) THEN
                IF prepnuti_stykace.Q THEN
     ...
                END_IF;
(*---------------------rozmezi 65 - 80%----------------------------------*)
        ELSIF (PHM_ted > 65 AND PHM_ted <= 80) THEN
                IF prepnuti_stykace.Q THEN
     ...
                END_IF;
(*---------------------rozmezi 80 - 100%---------------------------------*)
        ELSIF (PHM_ted > 80 AND PHM_ted <= 100) THEN
                IF prepnuti_stykace.Q THEN
     ...
                END_IF;
END_IF;
```

V případě poklesu hladiny musí dojít k odpojení outletů z nového levelu. To se děje opačným způsobem než jaký byl naznačen pro připínání. Zde již není potřeba kontrolovat signál nadproudové ochrany, ale jen natažení jističe.

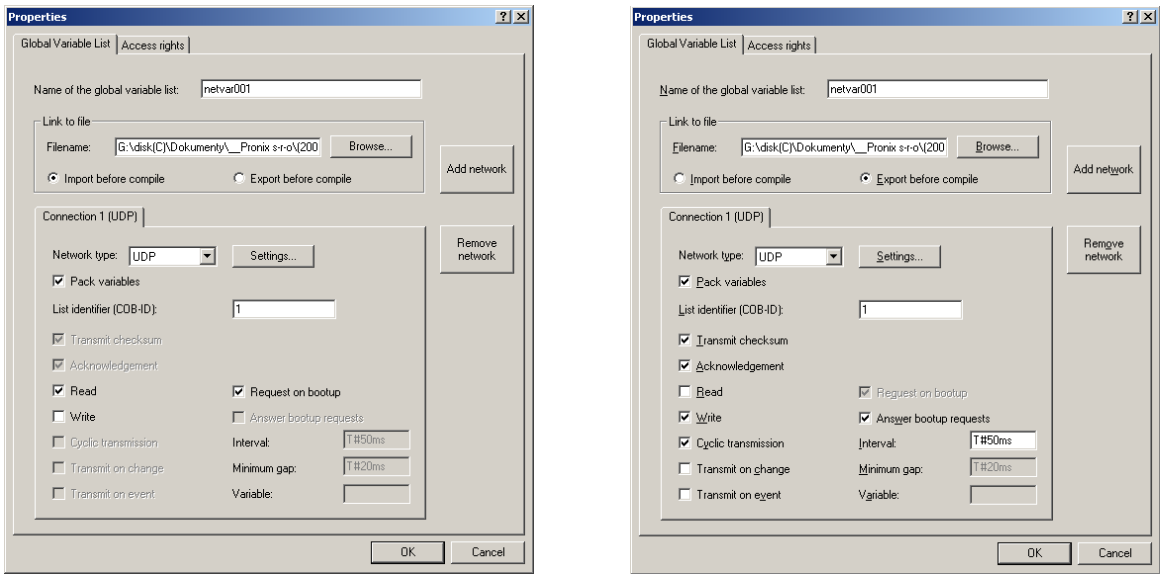
```
IF prepnuti_stykace.Q THEN
        IF  _7_FA4_ON THEN (*V11_kuchyne*)
                _7_FFA4_OFF := TRUE;
        ELSIF NOT _7_FA4_E THEN
                _7_FFA4_OFF := FALSE;
        END_IF;
ELSE
        _7_FFA4_OFF := FALSE;        (*V11_kuchyne*)
END_IF;

IF NOT prepnuti_stykace.Q AND  NOT _7_FA4_ON         THEN
        priorita_2_OK := TRUE; (*pokud je vypadla nadproud. ochr. a neni ZV od pripojeneho DEONu*)
END_IF;
```

Tímto je v podstatě princip programu pro řízení rozvodny popsán. Dále se zde vyskytují podprogramy pro řešení stavových informací pomocí SNMP trap zpráv, zápisu analogových hodnot do MIB databáze, vyčítání dat z měřících přístrojů DIRIS po lince RS-485 apod. Princip těchto podprogramů je však shodný jako v rozvaděči ATS 1000.

4.3 Síťové proměnné

Celá aplikace je rozložena na dva autonomní PLC automaty. Aby bylo možné provádět spolehlivé řízení v rozvodně, je nutné znát stav - polohu výkonových přepínačů. V úvahu přicházely dvě základní varianty. První byla natažení zpětné vazby od pomocných kontaktů obou přepínačů do obou PLC jako binární vstupy. Druhou možností bylo použití síťových proměnných, tato varianta se nakonec realizovala. Níže uvedený text pojednává o přidání a definici síťových proměnných do vytvořeného projektu.

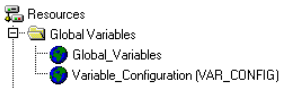
Přidání a konfigurace síťových proměnných v prostředí CoDeSys vypadá následovně. Na kartě Resources se přidá nový objekt, zadá se jeho jméno a cesta, kde se bude nacházet soubor (ve formatu `'nazev'.exp`) uchovávající potřebné informace o proměnných.

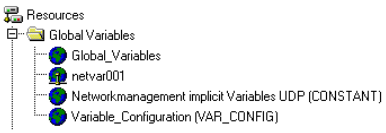


Obrázek 4.8: Vlasnosti sdílených proměnných

Dále je nutné uvést, zda se bude tento soubor v daném programu generovat (export before compile) nebo se bude načítat (import before compile). Takto je možné přidat další soubor proměnných. Pokud je požadavek na síťové proměnné, musí se použít tlačítko „Add network“. Tím se zobrazí dialog, kde je možné uvést, zda-li se v daném programu bude používat pouze čtení síťových proměnných, nebo pouze zápis, nebo jejich kombinace. Dále je možné určit jak bude probíhat přenos aktuálního stavu proměnných. Jsou tři možnosti:

- cyklický přenos ... lze nastavit časový interval ve formátu `T#_ms`,
- přenos na změnu ... k přenosu dojde při změně stavu některé z proměnných, nebo po uplynutí nastaveného minimálního času („Minimum gap“),
- přenos na změnu určité proměnné ... uvede se proměnná při jejíž změně dojde k přenosu.





Obrázek 4.9: (a) Bez sdílených proměnných (b) Sdílené proměnné „netvar001“

Předchozí obrázek zachycuje situaci před přidáním a po přidání síťových proměnných. Jejich definování se provede standardním způsobem a je naznačeno v příkladu, kde je definována jedna síťová proměnná nazvaná `GEN_network`, jejíž typ je `BOOL` (binární proměnná).

```
VAR_GLOBAL
        GEN_Volt_OK :BOOL;
END_VAR
```

Na závěr k síťovým proměnným ještě jedna praktická zkušenost. Při použití síťových proměnných je nutné uvést a použít v programu alespoň jednu, jinak při kompilaci programu dojde k ohlášení chyby Error4061. Pro odstranění chyby postačí pouze uvedení v libovolném spouštěném POU.

Kapitola 5

Program pro panelové PC v ControlWebu

ControlWeb je k dispozici ve dvou základních verzích. První je vývojová verze nutná pro vytvoření aplikace. Výsledkem je soubor ve formátu `*.cw`. Toto samotné prostředí umožňuje vytvořit plnohodnotnou aplikaci. Pokud je ale nutné používat komunikační kanály či jiné další doplňující funkce, je nutné dokoupit a doinstalovat přídavné moduly. Jejich seznam je možné najít na webu společnosti Moravské přístroje v sekci Ceník.

Druhá verze je označována jako Runtime. Vytvořená aplikace se nejprve přeloží na formát `*.cwx`. Takto získanou aplikaci je možné spustit přímo v místě určení. Výhodou Runtime verze je mnohem menší cena[1], nevýhodou naopak nemožnost úpravy sestavené aplikace.

5.1 Propojení ControlWebu a PLC WAGO

Fyzické propojení je provedeno dnes již pro průmysl standardním ethernetem 100Mbit/s. Jako protokol je použit již zmíněný ModBus TCP/IP.

Přenos dat mezi prostředím ControlWeb a PLC WAGO probíhá čtením a zápisem do paměťových buněk v PLC. Pro nastavení požadovaného přenosu jsou nutné dva soubory. Prvním je `'nazev'.dmf`. Zde se definují tzv. kanály. V následující části je ukázka takového souboru. Je nutné uvést o jaký typ kanálu půjde (`real / boolean`) a směr přenosu z pohledu ControlWebu. Tento soubor s definicí kanálu je shodný pro všechny typy ovladačů kanálů. Pro uživatelské užití jsou vyhrazeny kanály od čísla 100. Nižší mají funkci konfigurace a diagnostiky samotných PLC kontrolerů. Tedy soubor `'*'.dmf` může vypadat následovně.

```
begin
   100 - 199 real input (*analogové hodnoty...vstupy*)
   300 - 399 boolean input (*digitální údaje...vstupy*)
   500 - 599 boolean output (*digitální údaje...výstupy*)
   700 - 799 real output (*analogové hodnoty...výstupy*)
end.
```

[1] ControlWeb 6 Vývojová verze - 21 700 Kč, ControlWeb 6 Runtime - 6 500 Kč, ovladač Modicon MODBUS TCP/IP - 10 700 Kč, citováno 21.10.2008

Rozsah kanálů může být libovolný. Pro ovladač kanálu postaveného nad RS-232 se nedoporučuje měnit a editovat výše popsaný soubor, jinak hrozí nefunkčnost systému a z toho plynoucí škody. V této diplomové práci byl použit kanál MODBUS TCP/IP nad ethernetem a samozřejmostí byla zkouška změny typu kanálů. Vše fungovalo bez problémů. Jediný problém, který se vyskytl, byl s prostředím ControlWeb. Po editaci souboru `'*'.dmf` bylo nutné restartovat prostředí, aby došlo k požadovaným změnám.

Druhým důležitým souborem pro definici komunikačních kanálů je soubor `'nazev'.par`, označovaný jako parametrický. Zde probíhá definice cesty k zařízení. Jedná se o zadání adresy zařízení (`Address`). Tou může být IP adresy v případě MODBUS kanálu nad ethernetem, nebo například portu v případě kanálu nad RS-232. Výhodou takovéto definice je, že aplikace může využívat více fyzických zařízení (např. PLC kontrolerů), se kterými probíhá komunikace. Dále to je přesný rozsah kanálů (`ChFrom` a `ChTo`). Lze definovat samotné kanály, nebo jejich rozsah. Definice rozsahem je výhodná, pokud je požadavek přiřadit určité vlastnosti většímu počtu kanálů. Následně se definuje tzv. oblast (`Area`). Výstižnější pojmenování je modifikátor přenosu. Existuje čtveřice modifikátorů, které byly využity při tvorbě aplikace.

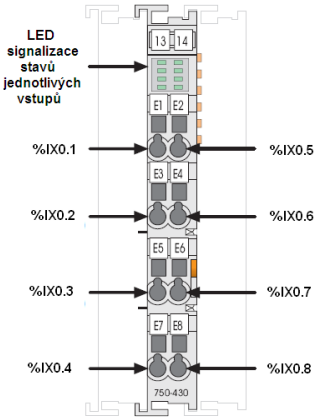
Tabulka 5.1: Modifikátory přenosu z CW do PLC

0x	OUTPUT - nastavení binárních výstupů v PLC
1x	INPUT - čtení binárních vstupů z PLC
3x	REGISTER - čtení/zápis analogových karet PLC (WORD)
4x	HOLD - čtení/zápis do paměti PLC (WORD)

Posledním povinným údajem je offset (`Ofs`). Ten určuje počátení adresu v zařízení, od které se bude číst nebo zapisovat. S určením hodnoty offsetu bývá problém. Při tvorbě této práce bylo zjištěno, že funguje následující mechanismus.

Pokud bude požadavek na čtení binárních vstupů PLC, bude použit modifikátor přenosu `1x` a offset začíná od `1`, protože bude čten binární vstup na první adrese. Na obr. 5.1 je naznačeno 8DI[2]. Protože adresa prvního vstupu je `%IX0.1`. Tedy offset bude `1`. V tabulka 5.2 je naznačen princip přiřazení offsetu.

[2]Digital Inputs



Obrázek 5.1: Binární vstupy PLC WAGO 750-430

Tabulka 5.2: Modifikátory přenosu z CW do PLC

WAGO	Area	Ofs
`%IX0.1`	1x	1
`%IX0.2`	1x	2
`%IX0.8`	1x	8
`%MW0`	3x	1
`%MW1`	3x	2

Podobně funguje i přístup k analogovým kartám. Tam se používá modifikátor `3x` a jako offset je opět `1`, pokud se bude číst od prvního kanálu první karty. V obr. 5.1 je opět naznačen přístup k jednotlivým WORDům analogových karet. Pozor na správně uvedenou oblast `Area`.

Problém ovšem nastává v případě, kdy by se mělo číst/zapisovat do paměti PLC. V tomto případě je nutné zjistit rozsah adres, které jsou rezervované pro práci PLC a počáteční adresu pro uživatelská data. Pro tento případ byla použita počáteční adresa v PLC 3064_{hex}, tedy adresa 12388_{dec}. Tato adresa byla zvolena, protože na adrese 3000_{hex} začínají paměťové registry a následuje oblast pro diagnostiky modulů. Ty zabírají 100xWORD oblast.

Naznačený problém se vyskytuje až v momentě, kdy uživatel-programátor chce číst nebo zapisovat. Z předchozího odstavce plyne mechanismus výpočtu adresy. Takto získaná adresa 12388_{dec}[3] však není správná. Bylo vyzkoušeno, že je nutné použít až následující adresu, tedy 12389_{dec}.

Dalšími nepovinnými údaji jsou přesnější typ kanálu `[Subtype]`. Může se jednat o `int`, `int16`, jejich neznaménková verze, binární kanál atd. K typu kanálu se váže i směr přenosu `[Bidirect]`. Jsou dva typy, implicitně je nastaven jednosměrný z pohledu ControlWebu a

[3] $3000_{\text{hex}} + 100_{\text{dec}} \rightarrow 12388_{\text{dec}} \approx 12388_{\text{dec}}$

nezapisuje se. Pokud požadujeme obousměrný, pak se zapisuje klíčovým slovem `bidirect`. Předposlední nepovinný parametr je označení kanálů `[ID:x]`. Číslování je plně v rukou programátora a nemá vliv na funkčnost systému. Posledním údajem je komentář kanálů `[;comment]`. Opět dle uvážení programátora a slouží především k přehlednosti.

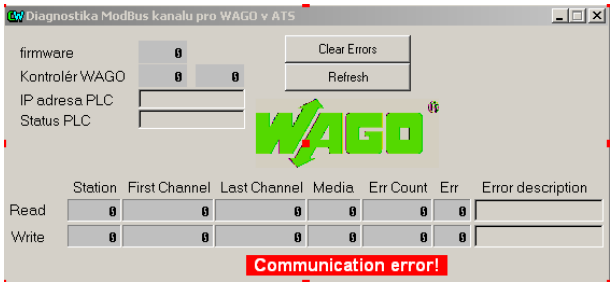
Na tomto místě je vhodné poznamenat subjektivní dojem z úpravy parametrického souboru. Je nevhodné používat tabulátor při psaní. V případě jeho použití se objeví při kompilaci aplikace v ControlWebu chyba v parametrickém souboru a je nutné jej celý napsat znovu. Obecně může vypadat parametrický soubor následovně. Hlavičku je nutné uvádět vždy. V závislosti na požadavcích řízení nebo monitorování se volí parametry přenosu, jako je `CheckTime` nebo `Timeout`. Příliš nízká hodnota může vést k nestabilitě komunikace mezi PLC a aplikací ControlWeb, nebo dokonce k pádu aplikace. Parametr `TraceOutput` slouží k lazení komunikace. Do vytvořeného souboru se zapisuje veškerá proběhlá komunikace po kanálu. Jedná se o výpis, který je k dispozici při tvorbě aplikace v prostředí ControlWeb.

```
[Modbus] (*typ protokolu kanálu*)
Mode = RTU (*definice kanálu*)
CheckTime = 50000
ConnectOnStartup = false
Timeout = 1000
EnableMonitor = true
DisablePresetSingleRegister = false
MaxRegistersInBlock = 100
TraceOutput = None (*trasovaci soubor, C:\XTrace.log*)

[Channels]
;-----------------------------------------------------------------------------------------------
;Block = Address,    ChFrom, ChTo, Area, Ofs [,Subtype] [,Bidirect] [,ID:x] [;comment]
;-----------------------------------------------------------------------------------------------
 Block = 00@192.168.121.201,   100,  105,   3X,   1,        uint16,        id:0101 ;AI modules

[Settings]
  Timeout = 500
  NumRepeat = 1
```

Pokud by byl použit kanál nad RS-232, lze užít protokol SAIA S-BUS.

```
[ComPort]
 Com = com1
 Protocol = sbus
 SBusMode = Parity
 BaudRate = 9600
 Timeout = 200
 RTS = enable

[Channels]
;-----------------------------------------------------------------------------
;Block = Station, ChannelFrom, ChannelTo, Media, Offset, [Subtype], [Bidirect]
;-----------------------------------------------------------------------------
 Block = 01,  500,  515, F,  0  ;flagy (bitove promenne)
 Block = 01,  600,  607, I,  0         ;bitove vstupy
 Block = 01,  700,  707, O,  16 , bidirect        ;bitove vystupy
 Block = 01,  900,  915, R, 100, uint8, bidirect ;analogove vystupy

[Settings]
 ComDriver = CWCOMM.DLL COM1
 Timeout = 500
 NumRepeat = 1
```

Jak bylo poznamenáno, v souboru `*.dmf` lze kromě uživatelských kanálů definovat i kanály pro diagnostiku PLC. Pro přehlednost a možnost zjištění chyb lze v sestavené aplikaci nahlédnout, v jakém stavu se právě PLC nachází, jeho výrobce, označení a typ, verze firmware apod. Také jsou zde zobrazeny vlastnosti aktuálně zpracovávaných kanálů.



Obrázek 5.2: Diagnostický nástroj

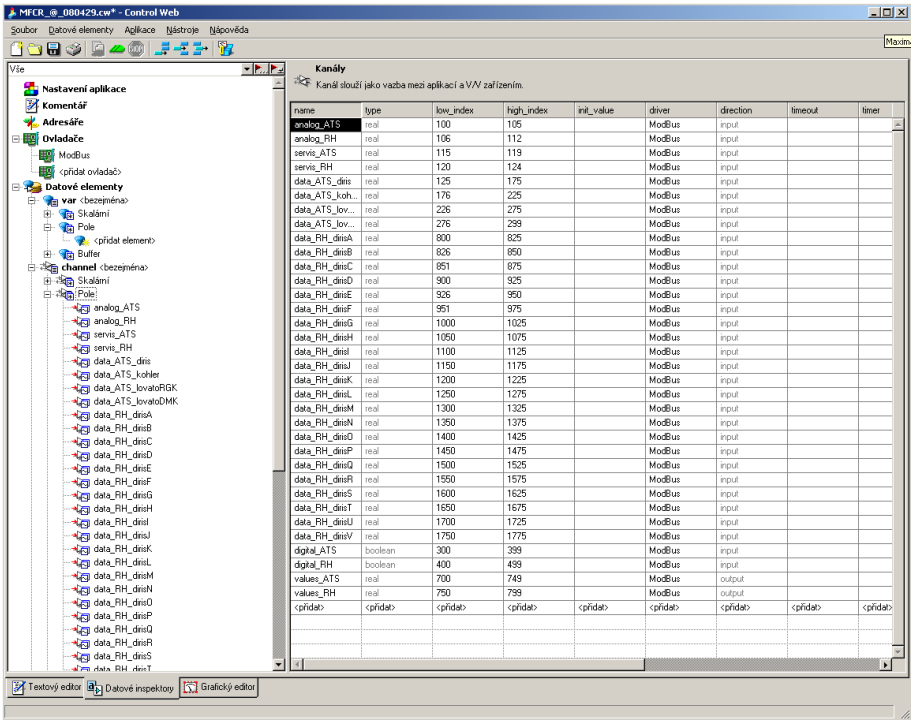
Aby bylo možné tyto diagnostické kanály používat, je nutné je uvést i v parametrickém souboru. Oba použité soubory jsou zobrazeny následně, zároveň jsou uvedeny i jednoduché vysvětlivky.

```
begin
     1 real input (*ExceptionStatus*)
     2 string input (*ConnectedStation*)
    10 real input (*StationR*)
    11 real input (*ErrCodeR*)
    12 string input (*ErrStringR*)
    13 real input (*ErrCountR*)
    14 real input (*FirstChannelR*)
    15 real input (*LastChannelR*)
    16 real input (*MediaR*)
    17 boolean output (*ResetR*)
    18 boolean output (*ClearErrorCounterR*)
    20 real input (*StationW*)
    21 real input (*ErrCodeW*)
    22 string input (*ErrStringW*)
    23 real input (*ErrCountW*)
    24 real input (*FirstChannelW*)
    25 real input (*LastChannelW*)
    26 real input (*MediaW*)
    27 boolean output (*ResetW*)
    28 boolean output (*ClearErrorCounterW*)
   100 - 199 real input (*AnalogInputs & DiagWAGO*)
   200 - 299 real input
   300 - 399 boolean input (*DigitalInputs WAGO_ATS*)
   400 - 499 boolean input (*DigitalInputs WAGO_RH*)
   500 - 599 boolean output
   600 - 699 boolean output
   700 - 799 real output
   800 - 2000 real input (*DataRS485 WAGO_RH*)
end.
```

```
[Modbus] (*typ protokolu kanálu*)
Mode = RTU (*definice kanálu*)
CheckTime = 50000
ConnectOnStartup = false
Timeout = 1000
EnableMonitor = true
DisablePresetSingleRegister = false
MaxRegistersInBlock = 100
TraceOutput = None (*trasovaci soubor, C:\XTrace.log*)

[Channels]
;-----------------------------------------------------------------------------------------------
;Block = Address,    ChFrom, ChTo, Area, Ofs [,Subtype] [,Bidirect] [,ID:x] [;comment]
;-----------------------------------------------------------------------------------------------
 Block = 00@172.16.121.201,   100,  105,   3X,   1,        uint16,                id:0101
       ;AI modules WAGO_1 ... ATS 1000 (napeti UPS a proudy L1,L2,L3)
;-------------------------
 Block = 00@172.16.121.202,   106,  112,   3X,   1,        uint16,                id:0102
       ;AI modules WAGO_2 ... RH_x (pt100 a napeti UPS)
;-------------------------
 Block = 00@172.16.121.201,   115,  115,   4X,   8209,     uint16,   bidirect     id:0103
       ;WAGO_ATS-firmware WAGA
 Block = 00@172.16.121.201,   116,  116,   4X,   8210,     uint16,   bidirect     id:0104
       ;WAGO_ATS-serie WAGA
 Block = 00@172.16.121.201,   117,  117,   4X,   8211,     uint16,   bidirect     id:0105
       ;WAGO_ATS-vezre kontroleru WAGA
 Block = 00@172.16.121.201,   118,  118,   4X,   8212,     uint16,   bidirect     id:0106
       ;WAGO_ATS-serie WAGA
 Block = 00@172.16.121.201,   119,  119,   4X,   8213,     uint16,   bidirect     id:0106
       ;WAGO_ATS-vezre kontroleru WAGA
;-------------------------
 Block = 00@172.16.121.202,   120,  120,   4X,   8209,     uint16,   bidirect     id:0107
       ;WAGO_ATS-firmware WAGA
 Block = 00@172.16.121.202,   121,  121,   4X,   8210,     uint16,   bidirect     id:0108
       ;WAGO_ATS-serie WAGA
 Block = 00@172.16.121.202,   122,  122,   4X,   8211,     uint16,   bidirect     id:0109
       ;WAGO_ATS-vezre kontroleru WAGA
 Block = 00@172.16.121.202,   123,  123,   4X,   8212,     uint16,   bidirect     id:0110
       ;WAGO_ATS-serie WAGA
 Block = 00@172.16.121.202,   124,  124,   4X,   8213,     uint16,   bidirect     id:0111
       ;WAGO_ATS-vezre kontroleru WAGA
;-------------------------
 Block = 00@172.16.121.201,   125,  299,   4X,   12489,    uint16,                id:0112
       ;WAGO_ATS-data nactena po RS485(Diris, Kohler, Lovato RGK, DMK)
;-------------------------
 Block = 00@172.16.121.201,   300,  399,   1X,   1,                               id:0113
       ;DI modules WAGO_ATS ... ATS 1000 (40 DI)
;-------------------------
 Block = 00@172.16.121.202,   400,  499,   1X,   1,                               id:0114
       ;DI modules WAGO_RH ... RH_x        (56 DI)
;-------------------------
 Block = 00@172.16.121.201,   700,  749,   4X,   12389,    uint16,                id:0115
       ;WAGO_ATS ... zapis hodnot z CW do WAGA (casove T1...T6)
 Block = 00@172.16.121.202,   750,  799,   4X,   12389,    uint16,                id:0116
       ;WAGO_RH ... zapis hodnot z CW do WAGA (casy odpinani)
;-------------------------
 Block = 00@172.16.121.202,   800,  1800,  4X,   12489,    uint16,               id:0117
       ;WAGO_RH - data nactena po RS485(25xDiris - 25xword)
```

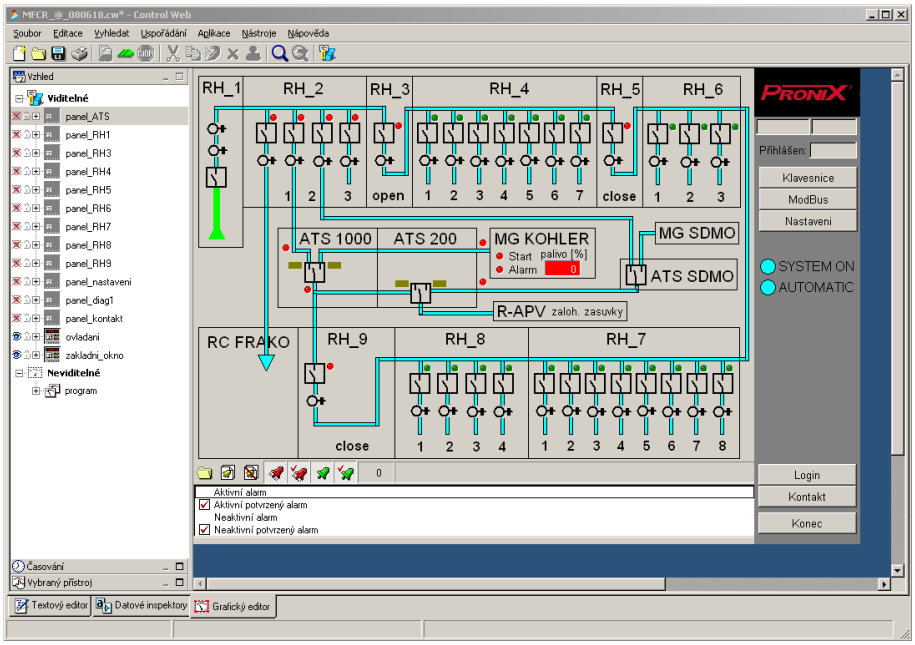
5.1.1 Tvorba aplikace

V momentě, kdy jsou definovány kanály pomocí parametrického souboru, je možné přímo v prostředí CW udělat jejich deklaraci a tím je začít používat. Jak vypadá záložka Datové inspektory je na následujícím obrázku.



Obrázek 5.3: Deklarace kanálů v prostředí CW

Celkový pohled na grafickou stránku výsledné aplikace přináší následující obrázek.



Obrázek 5.4: Vývoj grafické podoby aplikace

Protože vizualizace je navržena na panelové PC s rozlišením obrazovky 800x600 bodů,

bylo nutné dostat maximum informace o technologii na malý prostor. Proto byla oříznuta standardní Windows lišta. K zastavení aplikace slouží příslušné tlačítko, jehož kód i s popisky je uveden níže.

```
owner = ovladani (*panel ktery je vlastnikem tlacitka-kazdy pristroj musi mit vlastnika*);
position = 5, 532, 118, 26;
win_disable = zoom, maximize; (*zakazane operace s pristrojem*)
access = [ 0, 10, 100, 1000];        (*uzivatelske skupiny ktere mohou pristroj spustit*)
mode = text_button; (*jak bude pristroj vypadat*)
font = 'Arial (Central European)', 10, normal;
true_text = 'Konec';
false_text = 'Konec';
logic = set_true_on_press; (*logicka funkce pristroje*)

procedure OnMouseDown( MouseX, MouseY : longint; LeftButton, MiddleButton, RightButton : boolean );
begin
  core.StopApplication(); (*funkce jadra-zastaveni aplikace*)
end_procedure;
```

Dalším tlačítkem je možné zobrazit klavesnici na obrazovce. Panelové PC má dotykový displej a ovladaní se provádí přes obrazovku. Příkaz `osk.exe`[4] je standardní funkce operačního systému Windows. Jak je patrné, chybí zde řádek definující přístupová práve (`access=[...]`). Pokud mají k přístroji přístup všichni uživatelé (`access=[0, 4294967296]`), CW automaticky řádek smaže.

```
owner = ovladani;
position = 6, 118, 118, 26;
win_disable = zoom, maximize;
mode = text_button;
font = 'Arial (Central European)', 10, normal;
true_text = 'Klavesnice';
false_text = 'Klavesnice';
logic = set_true_on_press;

procedure OnMouseDown( MouseX, MouseY : longint; LeftButton, MiddleButton, RightButton : boolean );
var
    ErrorCodeKeyboard : longcard;
    ParametersKeyboard : string;
    x : longint;
    y : longint;
    w : longint;
    d : longint;
    Minimized : boolean;
    ThreadIdentifier : longcard;
begin
    system.RunProgram(ThreadIdentifier, 'C:\WINDOWS\system32\osk.exe', ParametersKeyboard, x, y, w, d, Minimized);
end_procedure;
```

Lze nastavit, že po spuštění se automaticky nebude zobrazovat dialogové okno pro přihlášení uživatele. K jeho zobrazení slouží následující kód.

[4] On Screen Keyboard

```
owner = ovladani;
position = 5, 474, 118, 26;
win_disable = zoom, maximize;
mode = text_button;
font = 'Arial (Central European)', 10, normal;
true_text = 'Login';
false_text = 'Login';
logic = set_true_on_press;

procedure OnMouseDown( MouseX, MouseY : longint; LeftButton, MiddleButton, RightButton : boolean );
begin
  system.ShowLoginWindow();
end_procedure;
```

V pravé horní části obrazovky, pod logem výrobce, je zobrazení právě přihlášeného uživatele. K tomu slouží následující procedura.

```
owner = ovladani;
position = 66, 88, 57, 20;

procedure OnActivate();
var
  UserLevel : longcard;
  UserLogin : string;
  UserFull  : string;
begin
  system.GetActualUser( UserLevel, UserLogin, UserFull );

  if UserLogin = 'root' then
  uzivatel = 'root';
 elsif UserLogin = 'technolog' then
  uzivatel = 'technolog';
 elsif UserLogin = 'servis' then
  uzivatel = 'servis';
 elsif UserLogin = 'obsluha' then
  uzivatel = 'obsluha';
 elsif UserLogin = 'none' then
  uzivatel = 'none';
end; (*hhh*)

SetValue(uzivatel);
end_procedure;
```

Následující dvě procedury využívají funkci `date` a její parametry pro zobrazení aktuálního datumu a času. Alarmy jsou zapisovány do logovacího souboru a je vhodné mít přímo na obrazovce přehled o aktuálním čase (lišta je automaticky schovávána pro větší prostor na obrazovce).

```
timer = 100, 0.1;
owner = ovladani;
position = 2, 60, 65, 20;

procedure OnActivate();
var
  datum : string;
begin
  datum = date.TodayToString();
  SetValue(datum);
end_procedure;
```

```
timer = 5, 0.1;
owner = ovladani;
position = 68, 60, 55, 20;
justify = left;

procedure OnActivate();
var
  cas   : string;
begin
  date.GetTimeString(cas);
  SetValue(cas);
end_procedure;
```

Kontrolky nadproudových ochran jsou řešeny jako soubory typu `*.ico` a zobrazovány jako červená v případě vypadnutí ochrany vlivem nadproudu nebo zelená pokud je vše v pořádku.

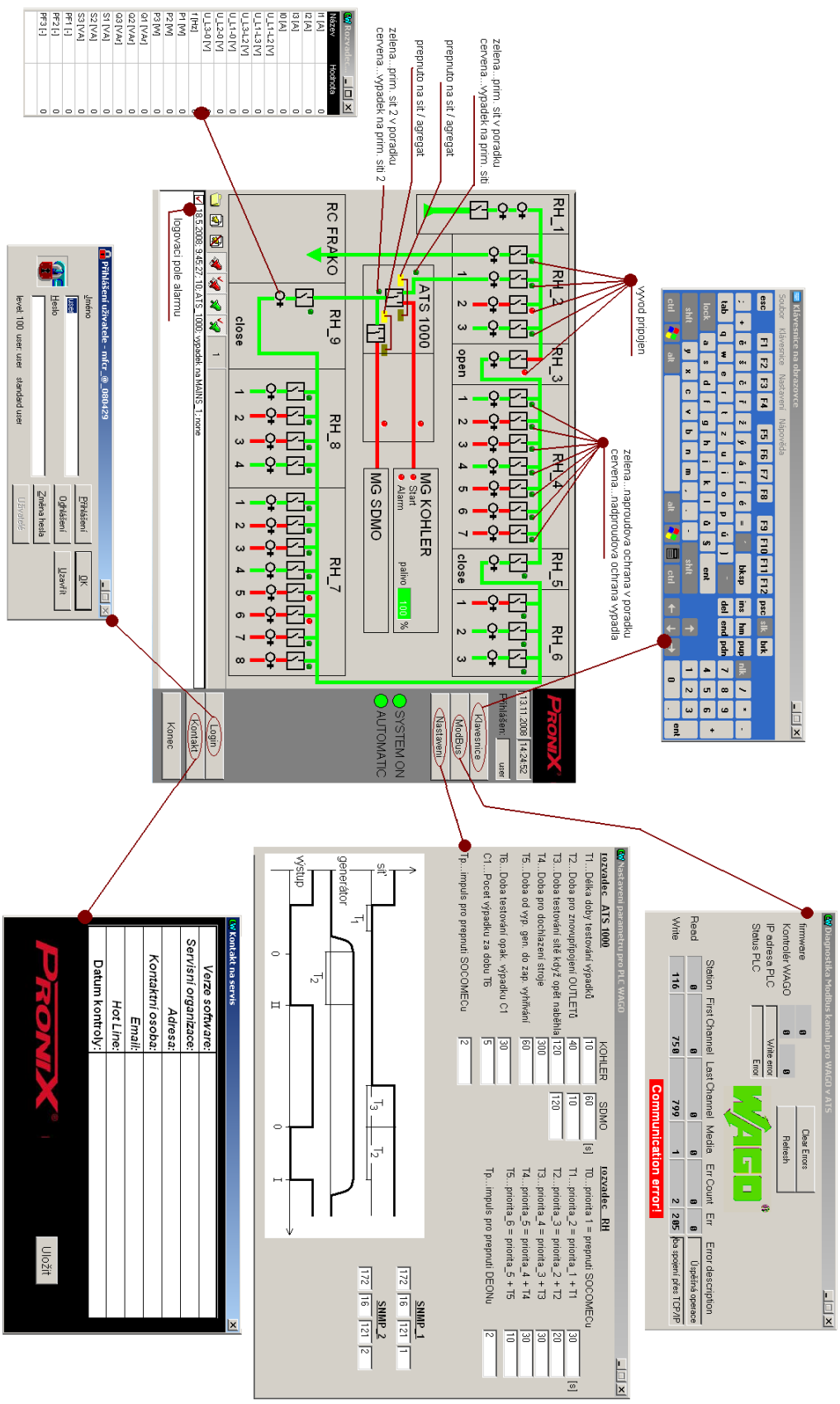
```
owner = zakladni_okno;
position = 276, 355;
win_disable = zoom, maximize;
expression = _8_FA1_E;        (*logicky signal na ktery se reaguje*)
true_icon = 'gledon2.ico';
false_icon = 'gledoff2.ico';
blink_colors
  true_paper = white;
  false_paper = white;
end_blink_colors;
```

Podobně je řešeno i zobrazení, zda-li větví prouchází proud.

```
owner = zakladni_okno;
position = 269, 389, 5, 50;
procedure OnActivate();
begin
 if _8_FA1_ON = true then
   box_8_FA1.SetInteriorColor(0, 255, 0);        (*prochazi proud-cara je zelena (Red,Green,Blue)*)
   box_8_FA1.SetBorderColor(0, 255, 0);
 else
   box_8_FA1.SetInteriorColor(255, 0, 0); (*neprochazi proud-cara je cervena*)
   box_8_FA1.SetBorderColor(255, 0, 0);
 end;
end_procedure;
```

Pro archivování stavů systému je použita komponenta `alarm`. Je možné určit strukturu, jak budou záznamy vapadat a také, které údaje o události budou obsaženy. Záznamy mohou obsahovat datum a čas vzniku, dále je lze třídit do skupin dle priorit nebo podle částí celku (ATS, RH, MTG). Pro kritické události (např. dochází palivo) lze nastavit požadavek na potvrzení nastalé události operátorem.

```
owner = zakladni_okno;
position = 0, 469, 674, 139;
record_structure = classify, group, text, value, operator, priority, none;
message_content = rise_date, rise_time, priority, group, text, operator, none, none, none, none, none;
confirm_access = [ 0..100 ];
file
  name = 'Alarmy';
  type = absolute;
  length = day;
  history = 14;
  access = [ 10..100, 100..1000 ];
  name_type = long_name;
end_file;
item
  text = 'palivo - mene nez 50% nadrze';
  group = 'MTG';
  priority = 100;
  condition = palivo < 50;
  finish_condition = palivo < 50;
  expression = palivo;
  rise_action = write;
  finish_action = write;
  confirm_action = write;
end_item;
item
  text = 'palivo - mene nez 10% nadrze ';
  group = 'MTG';
  priority = 1;
  condition = palivo < 10;
  finish_condition = palivo < 10;
  expression = palivo;
  rise_action = write;
  finish_action = write;
  confirm_action = write;
end_item;
```



Obrázek 5.5: Celkový pohled na vizualizaci

Kapitola 6

Závěr

Předládaná práce zpracovává téma řízení rozvaděče pro automatické bezobslužné přepínání záložních zdrojů. Cílem bylo vytvořit řídicí program přímo „na míru“ již vyprojektovanému fyzickému řešení rozvaděčů s přepínacími prvky ATS.

Na záčátku práce proběhlo seznámení jednak se samotnou technologií, ale predevším s principem a činností ATS rozvaděčů. Součástí celého systému je i řízení připínaní a odpínání outletů rozvodny. V zadání jsou přesně specifikovány požadavky pro odpínání a připínání jednotlivých outletů.

V přípravné fázi bylo rozhodnuto, že řízená technologie bude rozdělena na část rozvadečů ATS a rozvodny RH, každá s vlastní jednotkou PLC. Rozdělením došlo k potřebám komunikace mezi oběma jedotkami PLC. Z několika možností byla vybrána metoda sdílených proměnných.

Po přípravné fázi následoval principiální návrh strategie řízení. Zohledněnými parametry návrhu byly kromě základních požadavků písemného zadání také požadavek na modularitu jednotlivých programů a hlavně přehlednost celého kódu. S přihlédnutím k požadavkům byly koncepce obou hlavních programů sestaveny v jazyce SFC (strukturní funkční schéma).

Program pro řízení ATS kopíruje sekvenční strukturu cyklu přepínání, čímž je splňen požadavek na přehlednost. Modularita je zajištěna pomocí tzv. task-manageru, kdy s ohledem na funkční cyklus PLC jsou postupně volány jednotlivém programy. Využití jazyka SFC je výhodné i z pohledu testování finálního programu přímo na technologii. Je na první pohled vidět, v jakém stavu se právě technologie nachází a lze jednoduše kontrolovat podmínky dalšího postupu.

Vizualizace technologického celku se realizována s využitím programovacího prostředí ControlWeb. Vzhledem k rozsáhlosti grafického prostředí vizualizace a použitému panelovému PC může na první pohled působit těžkopádně a přeplněně. Jsou však zobrazeny veškeré důležité informace na jediné obrazovce, což přispívá přehlednosti. Vzhledem k faktu, že je jedná o řídicí systém to přináší i další výhodu, že je možné z jediného okna sledovat celou technologii. K ovládání slouží dialogové okno, kde je možné nastavovat časové parametry přepínacího cyklu a časová zpoždění při připínaní outletů.

Možnosti testování systému řizení byly omezené a mohly probíhat pouze mimo pracovní dobu. Při testování byly upraveny základní časové parametry, které jsou uloženy jako přednastavené hodnoty. Pro zajištění přehlednosti byla modífikována i grafická část

vizualizace. Jednalo se o celkové zobrazení rozvodny - pohled shora a umístění obou agregátů doprostřed.

K uvedenému řešení „na míru“ exitují také ekvivalentní standardizované. Jedním z nich může být použití přepínačů využívající technologii označovanou jako STS (Static Transfer Switch). Velkou výhodou tohoto řešení je téměř nulová doba odpojení záteže. Toto řešení je však nevhodné v popisované aplikaci. Důvodem je, že jsou obsaženy točivé stroje a je nutné vyčkat na jejich zastavení (řešení bez přifázovávání záteže).

Na trhu existuje několik modulů řídicích kontrolerů konstruovaných přímo pro řízení přepínačů. Příkladem může být model Lovato RGK popsaný na začátku práce. Zmíněné jednotky mají ovšem pouze omezené možnosti vizualizoce řízené technologie a jejich ovládaní je též omezené.

Literatura

[1] Bílý, R., Cagaš, P., Cagaš, R., Hladůvka, D., Kolařík, M., Sobotík, J., Zálešák, M., Zgarba, Z.: *ControlWeb 2000*, Computer Press, Praha 1999,

[2] Zezulka, F., Bradáč, Z., Fiedler, P., Kučera, P., Štohl, R.: *Programovatelné automaty*, skripta FEKT VUT, Brno 1999,

[3] Martinásková, M., Šmejkal, L.: *PLC a automatizace 1*, BEN, Praha 1999, *Programovatelné automaty*, skripta FEKT VUT, Brno 1999,

[4] Šmejkal, L.: *PLC a automatizace 2*, BEN, Praha 2005,

[5] Ronešová, A.: *Přehled protokolu MODBUS*, Plzeň 2005, [on-line], ⟨http://home.zcu.cz/ ronesova/bastl/files/modbus.pdf⟩, [cit. 2008-10-30]

[6] WAGO Elektro s.r.o: web firmy a manuály k modulům [on-line], ⟨http://www.wago.com⟩, [cit. 2008-10-30]

[7] Moravské přístroje a.s.: web firmy [on-line], ⟨http://www.mii.cz/⟩, [cit. 2008-10-30]

[8] KOHLER Power: web firmy [on-line], ⟨http://www.kohlerpower.com⟩, [cit. 2008-10-30]

[9] WIKIPEDIA: internetová otevřená encyklopedie [on-line], ⟨http://www.wikipedia.org⟩, [cit. 2008-10-30]

[10] SOCOMEC Group: web firmy [on-line], ⟨http://www.socomec.com⟩, [cit. 2008-10-30]

[11] LOVATO Electric: web firmy [on-line], ⟨http://www.lovatoelectric.com⟩, [cit. 2008-10-30]

[12] Regulacni pohony: Programy pro řízení - CoDeSys [on-line], ⟨http://www.regulacni-pohony.cz⟩, [cit. 2008-10-30]

[13] SNMP [on-line], ⟨http://www.fi.muni.cz/ kas/p090/referaty/2007-podzim/ct/snmp.html⟩, [cit. 2008-10-30]

Příloha A

Obsah přiloženého CD

K této práci je přiloženo CD, na kterém jsou uloženy zdrojové kódy pro

- řízení rozvaděčů ATS 1000 a ATS 200 pro PLC ... `rizeni_ATS@081007`
- žízení nové rozvodny pro PLC ... `rizeni_RH@081007`
- vizualizace v CW pro panelové PC ... `MFCR_@_080618`
- zdrojové kódy pro LaTeX 2_ε a obrázky tohoto dokumentu