TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky a mezioborových inženýrských studií

BAKALÁŘSKÁ PRÁCE

Liberec 2006 **Jiří Šťastný**

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky a mezioborových inženýrských studií

Studijní program: B2612 – Elektrotechnika a informatika
Studijní obor: 2612R011 – Elektronické informační a řídící systémy

Řídící a vizualizační aplikace elektrického servopohonu Siemens Masterdrives MotionControl

Application for control and visualization of electric servo-drive Siemens Masterdrives MotionControl

Bakalářská práce

Autor: **Jiří Šťastný**
Vedoucí práce: Ing. Martin Diblík
Konzultant: Doc. Ing. Pavel Rydlo, Ph.D.

Rozsah práce:
54 stran textu
32 obrázků
8 tabulek
3 přílohy
1 CD

V Liberci 15. 5. 2006

Prohlášení

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

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o užití mé bakalářské práce a prohlašuji, že **s o u h l a s í m** s případným užitím mé bakalářské práce (prodej, zapůjčení apod.).

Jsem si vědom toho, že užít své bakalářské práce či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum: 16.5.2006

Podpis: .....

Jiří Šťastný

Na tomto místě bych velmi rád poděkoval Ing. Martinu Diblíkovi, vedoucímu bakalářské práce, za poskytnutí mnoha cenných poznatků a zkušeností z oblasti teorie i praxe řízení servopohonů a Doc. Ing. Pavlu Rydlovi, Ph.D. za poskytnuté konzultace. Bez jejich rad a věcných připomínek by tato práce nemohla vzniknout v této kvalitě. V neposlední řadě chci poděkovat svým rodičům za materiální i duševní podporu během celého studia.

Liberec, květen 2006 Autor

Anotace

Cílem bakalářské práce je vytvořit vizualizační aplikaci v prostředí Control Web 5. Vizualizace běží na klientském počítači, který komunikuje se serverem pomocí OPC technologie. Tento server je připojen k řízenému pohonu pomocí průmyslové sběrnice Profibus. Pohon je tvořen frekvenčním měničem Siemens Masterdrives Motion Control, který ovládá synchronní motor. Vizualizace má sloužit k výukovým účelům a proto je nutná její přehlednost. Umožňuje vizualizovat a ovládat základní režimy servopohonu - momentový, rychlostní a polohový režim. Umožňuje nastavení žádaných hodnot řízených veličin a zobrazuje potřebné údaje o aktuálních hodnotách sledovaných veličin. Dále program dovoluje zobrazení diagnostických údajů (chyby a výstrahy) a průběhy sledovaných veličin zobrazuje v grafech.

Vizualizace, elektrický pohon, synchronní servopohon, OPC.

Annotation

The aim of this baccalaurete thesis is to create application for visualization of controlled drive in programming environment Control Web 5. Visualization runs on client computer, which is communicating with server by OPC technology. This server is connected to controlled drive by means of industrial bus Profibus. Drive consists of frequency converter Siemens Masterdrives Motion Control and synchronous servomotor. Visualization is intended for education and tutorial purposes, that is why its lucidity is necessary. Application allows visualizing and controlling basic modes of servo-drive – torque, speed and position mode. User can set desired values of controlled values and displays theirs actual values. Is is also possible to watch diagnostic data (faults and warnings) and display watched values in simple graphs.

Visualization, electric drive, synchronous servo-drive, OPC.

Obsah

Seznam obrázků

Seznam tabulek

Značky a symboly

α		Reálná osa statorového souřadného systému
β		Imaginární osa statorového souřadného systému
C	[F]	Kapacita
d		Reálná osa rotorového souřadného systému
f	[Hz]	Frekvence
i, I	[A]	Proud
L	[H]	Indukčnost
m	[N.m]	Elektromagnetický moment
M	[N.m]	Moment
p_p	[-]	Počet polpárů
q		Imaginární osa rotorového souřadného systému
R	[Ω]	Činný odpor
t	[s]	Čas
u, U	[V]	Elektrické napětí
ω	[rad.s^{-1}]	Úhlová rychlost
w_φ, φ_{set}		Žádaná hodnota polohy
w_ω		Žádaná hodnota rychlosti
ψ	[Wb]	Spřažený magnetický tok

Zkratky

AC	Střídavý proud (Alternating Current)
B	Binektor (konektor typu *boolean)*
BICO	Název technologie využívající binektory a konektory (BInector-COnector)
CUMC	Control Unit MotionControl
CW	Řídící slovo (Control Word)
DC	Stejnosměrný proud (Direct Current)
DI	Digitální vstup (Digital Input)
DO	Digitální výstup (Digital Output)
DW	Položka adresového prostoru typu *DoubleWord*
FM	Frekvenční Měnič
Free blocs	Volně programovatelné bloky
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HW	Fyzické vybavení (HardWare)
IGBT	Bipolární tranzistor s izolovanou řídicí elektrodou (Insulated-Gate Bipolar Transistor)
IN	Vstup (INput)
INT	Položka adresového prostoru typu *INTeger*
K	Konektor typu *word*
KK	Konektor typu *doubleword*
LU	Logická jednotka polohy (Logical Unit)
MC	Typ frekvenčního měniče (Motion Control)
OPC	Open Process Control
OUT	Výstup (OUTput)
PC	Počítač
PLC	Programovatelný řídící automat (Programable Logic Controler)
PM	Permanentní Magnet
Profibus	Typ průmyslové sběrnice
PWM	Pulsně-šířková modulace (Pulse-Width Modulation)
RAM	Typ paměti (Read Access Memory)

RFG	Rampový funkční generátor (Ramp Function Generator)
RS 232/485	Označení druhu sériové komunikační linky
SCADA/HMI	Označení systémů pro vizualizaci technologických procesů sloužící jako rozhraní mezi technologickým zařízením a jeho obsluhou (Supervisory Control And Date Acquisition/Human Machine Interface)
SM	Synchronní Motor
SP	Blok jednoduchého polohování (Simple Positioner)
SW	Stavové slovo (Status Word)
SW	Programové vybavení (SoftWare, firmware)
tag	Ukazatel na datový element do adresového prostoru OPC serveru
W	Položka adresového prostoru typu *Word*

1 Úvod

Automatizace průmyslu, dopravy a dalších technických odvětví dosáhla v poslední době výrazného vzestupu. Zvyšování kvality v technologii výroby a bezpečnosti je hlavním předpokladem k úspěchu v technické praxi. K tomu přispívají také programy pro řízení a vizualizaci, které jsou dnes moderním trendem.

V oblasti řízení a vizualizace technologických procesů vznikají stále nové zobrazovací možnosti. 3D objekty v programech pro vizualizaci mají stále věrnější vzhled, a stále více odpovídají skutečným zařízením. V současné době existuje mnoho programových produktů SCADA/HMI (Supervisory Control and Date Acquisition/Human Machine Interface) určených pro tvorbu rozsáhlých dispečerských a operátorských pracovišť. Jedním z nich je také systém *Control Web*, který vyvíjí a dodává firma Moravské přístroje a.s. Umožňuje dvoucestné programování (v textovém i grafickém módu), tvorbu spolehlivých vizualizačních aplikací pro řízení, sběr dat, jejich ukládání a vyhodnocování.

Základem správné vizualizace je její jednoduchost a přehlednost na jedné straně a její bezproblémová funkčnost i při nasazení do náročných aplikací na straně druhé. Control Web 5 má oba tyto předpoklady. Obsahuje všechny komponenty nutné pro tvorbu vizualizačních aplikací – zobrazovací a ovládací prvky, alarmy a archivy. Umožňuje programovat datově řízené aplikace, ale také náročné aplikace reálného času. Je možná i práce v síti s využitím modulů pro sekvenční řízení procesů a vizualizaci technologií pomocí internetových standardů HTTP a HTML libovolným www klientem. Aplikace tvořené v systému Control Web jsou nasazovány do řídicích technologicky vyspělých procesů, ať už se jedná o jaderné elektrárny nebo jen drobné aplikace jako přímé řízení jednotlivých strojů.

Cílem této bakalářské práce je vytvořit program pro řízení a vizualizaci servopohonu Siemens Masterdrives Motion Control. Doprovodný text provede jeho uživatele teoretickým základem aplikace a následně ho seznámí s jejím praktickým využitím. Tato aplikace bude sloužit k výukovým účelům, a proto je zde kladen důraz na názornost a snadné ovládání. Je rozdělena do jednotlivých panelů, které přehledně zobrazují vnitřní datové a regulační struktury řídící jednotky a naznačují jejich vzájemné propojení a použití.

Stálé zdokonalování v automatickém řízení technologických procesů a s tím související rozmach prostředků pro vizualizaci bylo důvodem výběru tohoto tématu pro mou práci. Automatické řízení mě velmi zajímá a rád bych se naučil pracovat v některém z jeho vizualizačních prostředků. Tato práce je pro mé vzdělání velkým přínosem a doufám že Vás také zaujme.

2 Elektrické pohony

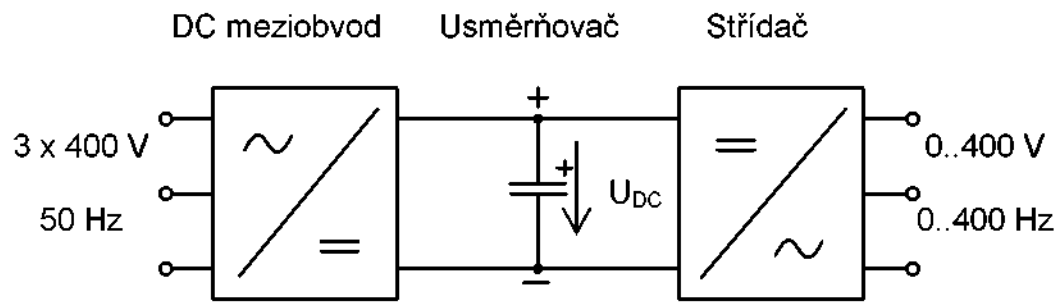
2.1 Druhy pohonů využívaných v průmyslu a automatizaci

Dominantní postavení v oblasti elektrických pohonů v průmyslových aplikacích mají frekvenčně řízené pohony s asynchronními motory. V automatizaci a především v robotice se však používají i mnohá speciální řešení, kde kritériem pro konstrukční návrh není jen účinnost pohonu, ale také jeho velká přesnost, dobrá zatížitelnost a malé rozměry. Co se týká servopohonů, měnič frekvence nejčastěji napájí synchronní motory s permanentními magnety (PM). Tyto motory v mnoha aplikacích obsadily místa, kde byly dříve nasazovány stejnosměrné (DC) motory. Pohony se synchronními motory s PM mají oproti DC motorům menší rozměry, nižší hmotnost a jsou kompaktnější. Používají se v široké škále výkonů od 10^0 kW do 10^2 kW. Jsou dražší než klasické servopohony s asynchronními motory, ale v automatizačních aplikacích uživatelé ocení především jejich malé rozměry a malou hmotnost. Dobré dynamické vlastnosti jako malý moment setrvačnosti a zároveň vyšší momentová přetížitelnost je předurčují pro dynamicky náročné aplikace.

Elektrické střídavé pohony využívají frekvenční řízení, které zajišťuje řídící jednotka pohonu (často nazývána zkráceně frekvenční měnič). Zastává hlavní potřeby regulace jako je ovládání rychlosti a směru otáčení, točivého momentu či úhlu natočení hřídele.

2.2 Frekvenční měniče

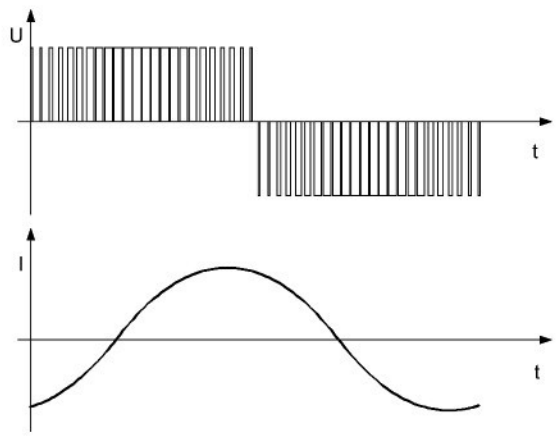
Frekvenční měniče (či měniče kmitočtu) jsou polovodičová zařízení, skládající se z napájecího řízeného usměrňovače, stejnosměrného meziobvodu s filtračním kondenzátorem a výstupního střídače, který pracuje s pulsně šířkovou modulací (PWM). Blokové schéma FM je uvedeno na obrázku (Obr. 2.1).



Obr. 2.1: Blokové schéma FM

Vstupní usměrňovač je napájen z jednofázové nebo třífázové rozvodné sítě, jednofázové napájení se používá u měničů nejmenších výkonů. Vstupní usměrňovač může být diodový (neřízený) nebo tyristorový (řízený) uspořádaný do můstku. Při použití neřízeného diodového vstupního usměrňovače není možné do sítě vracet energii, nachází-li se pohon v oblasti generátorového brzdění. FM téměř vždy umožňují nastavit režim brzdění motoru stejnosměrným proudem. Kromě toho, je-li vyžadováno brzdění s velkými výkony po delší dobu, nabízí výrobce velmi často možnost doplnit brzdné jednotky tzv. brzdným rezistorem. Toto je řešení, kdy je paralelně k filtračnímu kondenzátoru ve stejnosměrném meziobvodu pulsním měničem připínán brzdný rezistor tak, aby napětí v meziobvodu nepřekročilo nastavenou mezní hodnotu. Při generátorovém brzdění se činná energie motoru mění v tepelnou energii v brzdném rezistoru.

Výstupní střídač ze stejnosměrného napětí vytvoří harmonickou trojfázovou soustavu, jejíž napětí a frekvence je dána řídícím algoritmem měniče. Výstupní trojfázové napětí střídače je formováno pomocí pulsně šířkové modulace PWM (viz. Obr. 2.2). Ta umožňuje realizovat téměř spojitou změnu frekvence a efektivní hodnoty první harmonické výstupního napětí. Modulační frekvence je použitím IGBT dosti vysoká a u měničů bývá nastavitelná plynule nebo v několika stupních zpravidla v rozmezí 2 až 20 kHz. Díky poměrně vysoké modulační frekvenci je průběh proudu, který je filtrován indukčnostmi motoru, přibližně sinusový.



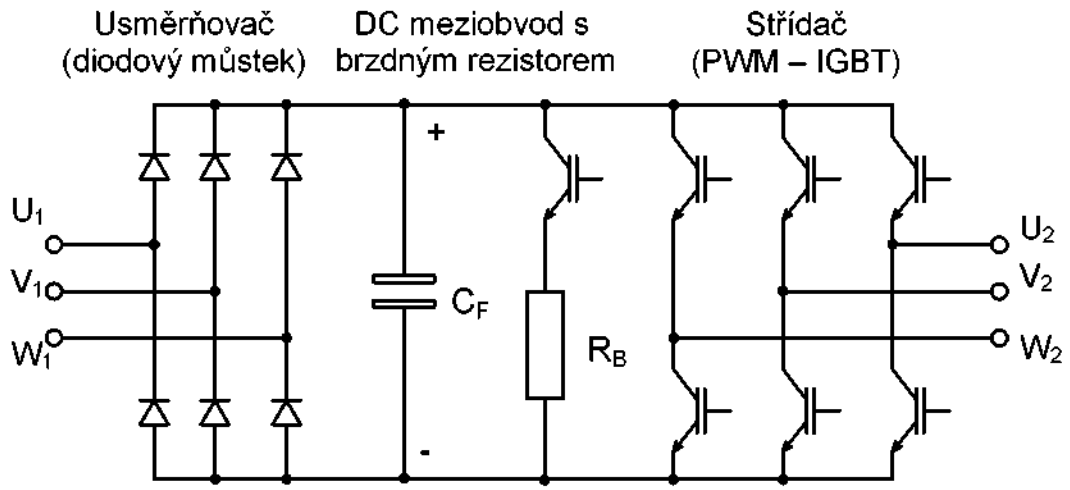
Obr. 2.2: Ukázka tvorby sinusového napětí pomocí PWM a průběh proudu

Měniče kmitočtu se používají v případech, kde je z technologického důvodu zapotřebí plynulá regulace otáček rotoru stroje a kde se požaduje plynulý rozběh a doběh otáček stroje. Při rozběhu pohonu lze s FM dosáhnout velkého záběrového

momentu. Typická místa nasazení měničů jsou tedy čerpadla, ventilátory, všechny druhy dopravníků, papírenské stroje atd. Dnešní průmyslová automatizace by se zkrátka bez FM jen těžko obešla. Nasazením měničů u pohonů ventilátorů a čerpadel se dosáhne kromě plynulé regulace množství média, také značné úspory elektrické energie, oproti dříve používanému ztrátovému způsobu regulace škrcením.

2.2.1 Konstrukce

Frekvenční měnič se skládá ze silové části podrobněji popsané v kapitole 2.2 a z řídící elektroniky. Silovou část tvoří vstupní usměrňovač (Invertor), DC meziobvod a výstupní střídač (Convertor), který provádí PWM. Blokové schéma silové části je na obrázku (Obr. 2.3). Pokud je správně dimenzován vstupní usměrňovač, je možné pomocí jednoho FM řídit větší množství Convertorů, to je cenově méně náročné řešení, něž pořizovat větší množství plně vybavených měničů.



Obr. 2.3: Blokové schéma provedení silové části FM

Velkým zvratem v konstrukci FM se stalo zavedení výkonových polovodičových součástek, tranzistorů IGBT (*Insulated-Gate Bipolar Transistor)* do jejich výkonové části. Jejich výhodou je velký vstupní a malý výstupní odpor v sepnutém stavu, který zaručuje nízký budící příkon a malé spínací ztráty. Dále mají vysokou spínací frekvenci a velký rozsah pracovních napětí (až do 1700 V) a proudů (do 500 A). Významné jsou také jejich krátké spínací časy okolo 1 μs, vypínací čas asi 2 μs. IGBT tranzistory proto mohou pracovat se spínací frekvencí až 20 kHz, s jejím nárůstem však rostou i spínací ztráty a tedy dochází ke snížení výstupního výkonu celého pohonu.

Řídící elektroniku měniče tvoří regulační struktura s řídicím mikroprocesorem, který zajišťuje vnitřní řízení měniče digitálně implementovanými algoritmy. Tento mikroprocesor zajišťuje také vzájemnou komunikaci s vnějšími zařízeními. Standardně FM obsahují PID regulátory umožňující velmi přesnou regulaci. Měniče umožňují připojení k nadřazenému programovatelnému řídicímu automatu (PLC) nebo PC po průmyslové sběrnici. Dále mají vstupní a výstupní svorky pro připojení analogových a digitálních signálů, díky kterým lze funkce měniče rozšířit o řadu doplňujících zařízení, volených podle konkrétních požadavků.

2.2.2 Způsoby řízení střídavých elektromotorů

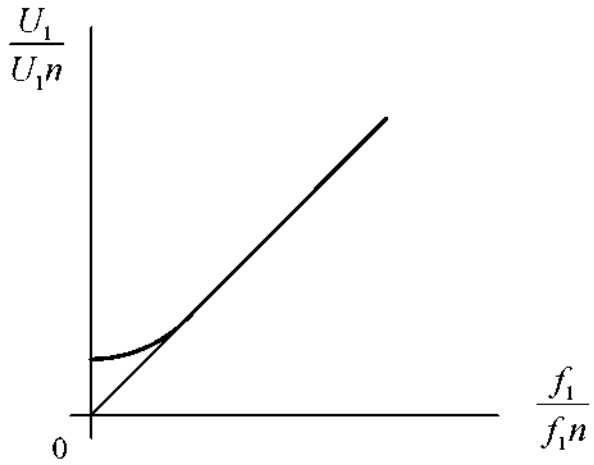
Měniče se vyrábějí v provedení se skalárním nebo vektorovým řízením. Pro pohony s velkými nároky na přesnost regulace rychlosti se používá vektorové řízení a pro méně přesnou regulaci skalární řízení. V obou se využívá mnoha poznatků z teorie automatizace, potřebné jsou velmi rychlé výpočty matematického modelu pohonu v podobě diferenciálních rovnic. V případě skalárního řízení dochází pouze ke změně amplitudy proudu. Vektorové řízení umožňuje měnit velikost momentotvorné i tokotvorné složky proudu a tak poskytuje mnohem kvalitnější řízení pohonu.

Ve skalárním řízení je kmitočet a další parametry považován za skalární veličinu. Zmíníme se o nejpoužívanějším typu skalárního řízení, a to kmitočtově-napěťovém. Při tomto řízení je pomocí změny kmitočtu regulováno napětí na základě vztahu mezi ω_1 a U_1. Moment je totiž přímo úměrný poměru $\left(\frac{U_1}{\omega_1}\right)^2$, což plyne z rovnice pro maximální moment asynchronního motoru (viz. [1]):

$$M_m = \frac{3}{2} \cdot \frac{U_1^2}{\omega_1^2} \cdot \frac{p_p}{L_k}.$$

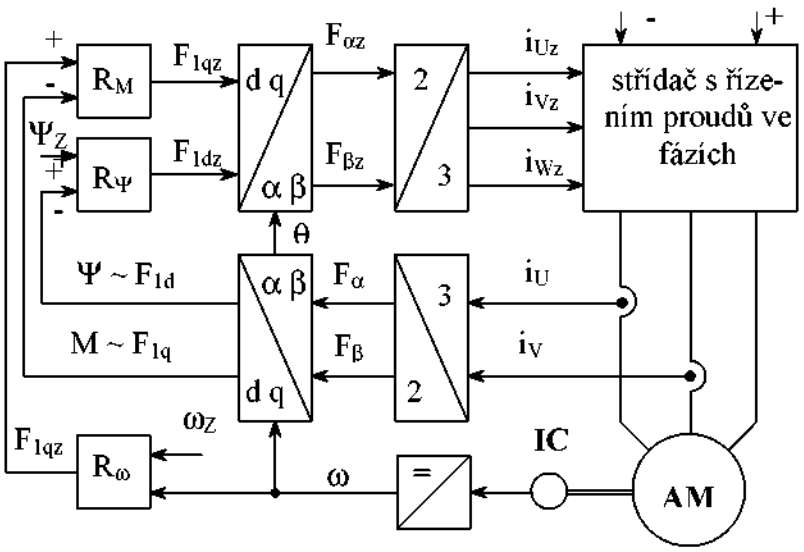
Pro nízké kmitočty, kdy dochází ke zvyšování úbytku napětí na statorovém odporu vzhledem k úbytku napětí na reaktanci vinutí, musíme na vstup motoru dodat vyšší napětí než odpovídá lineární závislosti $\frac{U_1}{U_1 n}$ na $\frac{f_1}{f_1 n}$ (viz. Obr. 2.4). Skalární řízení je tedy vhodné pro regulaci při ustálených nebo pomalu se měnících zatíženích nebo otáčivých rychlostech. U tohoto řízení není třeba měřit aktuální otáčivou rychlost. Existuje celá řada variant skalárního řízení, které zkvalitňují vlastnosti

regulace, princip ovšem zůstává stále stejný. Z těchto variant bych jmenoval například použití tzv. estimátoru skluzu.



Obr. 2.4: Kompenzace vlivu odporu statoru při nízkých otáčkách

Při realizaci vektorového řízení, jehož struktura je na obrázku (Obr. 2.5) je třeba mít informaci o poloze os souřadného systému *d*, *q* vůči statoru. Fázové složky napájení statoru se převedou do souřadných os α a β. U vektorového řízení se využívá transformace ze souřadnic α,β svázaných se statorem motoru do souřadnic d,q svázaných s točivým magnetickým polem stroje. Ve výsledku původně rotující vektor statorového proudu „stojí" a mění pouze svou amplitudu. Na hřídeli motoru je umístěno čidlo rychlosti, které tvoří rychlostní zpětnou vazbu. Ve dvou fázích statorového vinutí se měří proudy i_U a i_W, které se transformují ze systému U, V, W do souřadnic α, β, výstupem jsou pak signály F_α a F_β. V návaznosti je pak provedena



Obr. 2.5: Strukturní schéma vektorového řízení s čidlem rychlosti

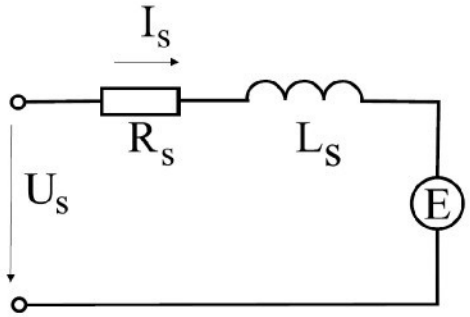
transformace do souřadnic *d*, *q*, které se otáčí synchronně s magnetickým polem stroje. Výstupem jsou hodnoty F_{1d} a F_{1q}, které již regulátor převede na momentotvornou *(Isq)* a tokotvornou *(Isd)* složku proudu. Tyto hodnoty jsou následně převedeny do soustavy souřadnic α, β a poté na tři složky fázových proudů i_{Uz}, i_{Vz}, i_{Wz}. Mimo zde uvedeného řešení existují další modifikace vektorového řízení i bez otáčkové zpětné vazby (viz. [1]).

2.3 Synchronní motor

Základní vlastností synchronního motoru je shoda otáček rotoru s otáčkami točivého magnetického pole statoru. Třífázový střídavý proud, jehož složky jsou vzájemně fázově posunuty, teče do vinutí statoru vzájemně natočených pod úhlem odpovídajícím počtu pólů motoru. To vyvolá točivé magnetické pole ve statoru. Rotor může být tvořen vinutím napájeným přes kroužky tvořící elektromagnet nebo permanentními magnety se střídavě uspořádanými póly po obvodu.

Nabuzený synchronní motor se po přímém připojení na střídavou síť sám neroztočí. Točivé magnetické pole statoru se otáčí rychlostí danou frekvencí napájecí sítě a počtem pólů motoru. Stojícím rotorem, prochází stejnosměrný proud, který budí stacionární magnetické pole. Vzájemným působením těchto dvou polí dochází k silovému působení pole statoru na pole rotoru. Orientace této síly se však mění s rychlostí otáčení točivého pole statoru. Z toho plyne, že rotor synchronního motoru je vystaven střídavému působení tažné síly v opačných směrech. Rotor se tedy vzhledem ke své hmotě nemůže sám rozběhnout. Roztočí-li se však rotor na synchronní rychlost nebo alespoň na rychlost blízkou synchronní a připojí se následně k síti, bude se motor otáčet synchronní rychlostí i po odpojení rozběhového zařízení. Rozběh je převážně realizován právě FM, který frekvenční rampou zajistí měkký rozběh motoru a roztočí rotor na požadované otáčky.

Na obrázku (Obr. 2.6) můžeme vidět náhradní schéma pro jednu fázi synchronního motoru s permanentními magnety.



Obr. 2.6: Náhradní schéma jedné fáze motoru s permanentními magnety

Podmínkou pro toto schéma a následující vztahy jsou tyto zjednodušující předpoklady:

- průběh magnetické indukce ve vzduchové mezeře a tedy i indukovaného napětí je sinusový, přičemž je obecně uvažován motor s vyniklými póly, t.j. s různou magnetickou vodivostí v podélném a příčném směru
- parametry (R, L) stroje jsou konstantní a stejné ve všech třech fázích
- ztráty v železe jsou zanedbány
- tlumící vinutí na rotoru není provedeno a rovněž se zanedbávají tlumící účinky materiálu rotoru
- nulový vodič není připojen

K napájení je využit FM, který napájí motor fázovým napětím U_S. Odpor R_S je odpor jedné fáze statorového vinutí, L_S je náhradní indukčnost reakce kotvy a E je napětí indukované magnetickým tokem ϕ_F permanentních magnetů. Velikost fázoru indukovaného napětí E je úměrná úhlové rychlosti rotoru ω.

Rovnice pro spřažené magnetické toky Ψ_d a Ψ_q:

$$\Psi_d = L_d.i_d + \Psi_f$$
$$\Psi_q = L_q.i_q$$

Ψ_d, Ψ_q .... spřažené magnetické toky

L_d , L_q.... indukčnost vinutí statoru

Ψ_f..... magnetický tok PM

i_d, i_q ..... statorový proud

Rovnice pro statorová napětí u_d a u_q:

$$u_d = R_s.i_d + \frac{d\Psi_d}{dt} - \omega.\Psi_q$$

$$u_q = R_s.i_q + \frac{d\Psi_q}{dt} + \omega.\Psi_d$$

u_d, u_q ... statorová napětí

R_s ..... odpor fázového vinutí

Momentová rovnice:

$$m = \frac{3}{2}.p_p.[\Psi_f + (L_d - L_q).i_d].i_q$$

m ..... elektromagnetický moment
p_p ..... počet polpárů vinutí motoru

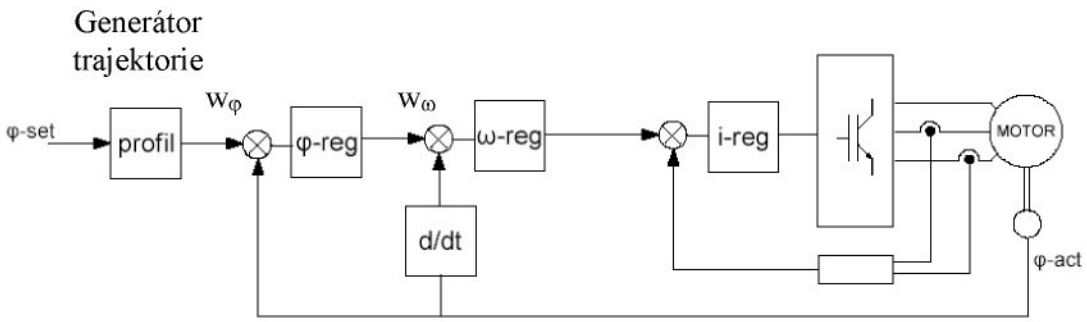
Frekvenční měnič obsahuje dva důležité parametry, které umožňují realizovat tyto rovnice přímo v jeho řídicí elektronice. Těmito parametry je přímo tokotvorná a momentotvorná složka proudu, pomocí nichž je prakticky motor řízen. Více o principu funkce synchronních motorů včetně odvození výše uvedených rovnic a způsobech řízení těchto motorů se lze dočíst v [1].
Proti asynchronním motorům mají synchronní servomotory tyto výhody:

- větší účinnost (mají nepatrné ztráty v rotoru),
- menší nároky na chlazení (v důsledku menších ztrát),
- vyšší hustotu výkonu na jednotku objemu,
- snadnou regulaci otáček a polohy na základě měření provozních elektrických veličin,
- spolehlivou kontrolu polohy.

2.4 Synchronní servopohony

Řídící struktura servopohonu se synchronním motorem má několik úrovní. Blokové schéma regulační struktury FM je uvedeno na obrázku (Obr 2.7). Nejnižší úrovní je regulace proudu a realizace pulsně šířkové modulace. Tato úroveň je náročná na rychlost regulace a v některých měničích je vykonávána analogovou

řídicí elektronikou. Struktura regulace momentu je odvozena z principů vektorového řízení. V převážné části rozsahu otáček je motor řízen tak, aby vyvíjel požadovaný moment. Žádanou hodnotu úhlu natočení φ_{set}, popř. otáček zadává generátor trajektorie. Tento blok vypočítává průběh okamžité žádané polohy a rychlosti při vykonávání pohybu z výchozí do cílové polohy na základě uživatelem zadaných parametrů trajektorie. Mezi synchronním motorem a FM je polohová zpětná vazba, tvořena snímačem polohy. Je napojena do porovnávacího členu, kde je porovnávána s žádanou hodnotou polohy w_{φ}, kterou udává generátoru trajektorie. Derivace aktuální polohy pak vstupuje do porovnávacího členu, který ji porovná s žádanou hodnotou rychlosti w_{ω}. Servopohonu je zpravidla nadřazen řídicí systém, který servopohon ovládá. Tento řídicí systém zajišťuje start pohonu nebo jeho vypnutí, generuje žádané hodnoty rychlosti, polohy a zrychlení či zpomalení.



Obr. 2.7: Blokové schéma regulační struktury FM

Některé typy řídicích jednotek servopohonů umožňují i autonomní funkci pro provádění jednoduchých polohovacích operací.

Komunikace s okolními zařízeními je zajištěna buď pomocí zmiňovaných analogových a logických vstupních a výstupních signálů nebo prostřednictvím standardní průmyslové sběrnice.

Momentové charakteristiky frekvenčně řízených synchronních pohonů jsou podobné charakteristikám cize buzených stejnosměrných strojů. Závislost momentu $M = f(n)$ je téměř konstantní.

3 Servopohon Siemens Masterdrives Motion Control

Měniče Siemens Masterdrives Motion Control (viz. Obr. 3.1) obsahují regulační strukturu, která je určena zejména pro pohony s vysokými požadavky na přesnost a dynamiku průběhu regulované veličiny (poloha, otáčky, moment), tedy pro aplikace s velkým momentovým přetížením a zrychlením, zejména při rozběhu a brzdění pohonu (servomotory). Umožňuje jak skalární tak i vektorové řízení. Skalární řízení má oproti vektorovému řízení otevřenou rychlostní smyčku. Pro přesné řízení jsou využívány polohové servomechanizmy, u kterých se snímá kromě rychlosti také poloha hřídele motoru.

Obr. 3.1: FM Masterdrives Motion Control od společnosti Siemens

Vlastnosti:

- Napájecí napětí: 3 x 380V-3 x 480V; 50/60Hz
- Výkonové řady: Compact Plus: 0,55kW - 18,5kW

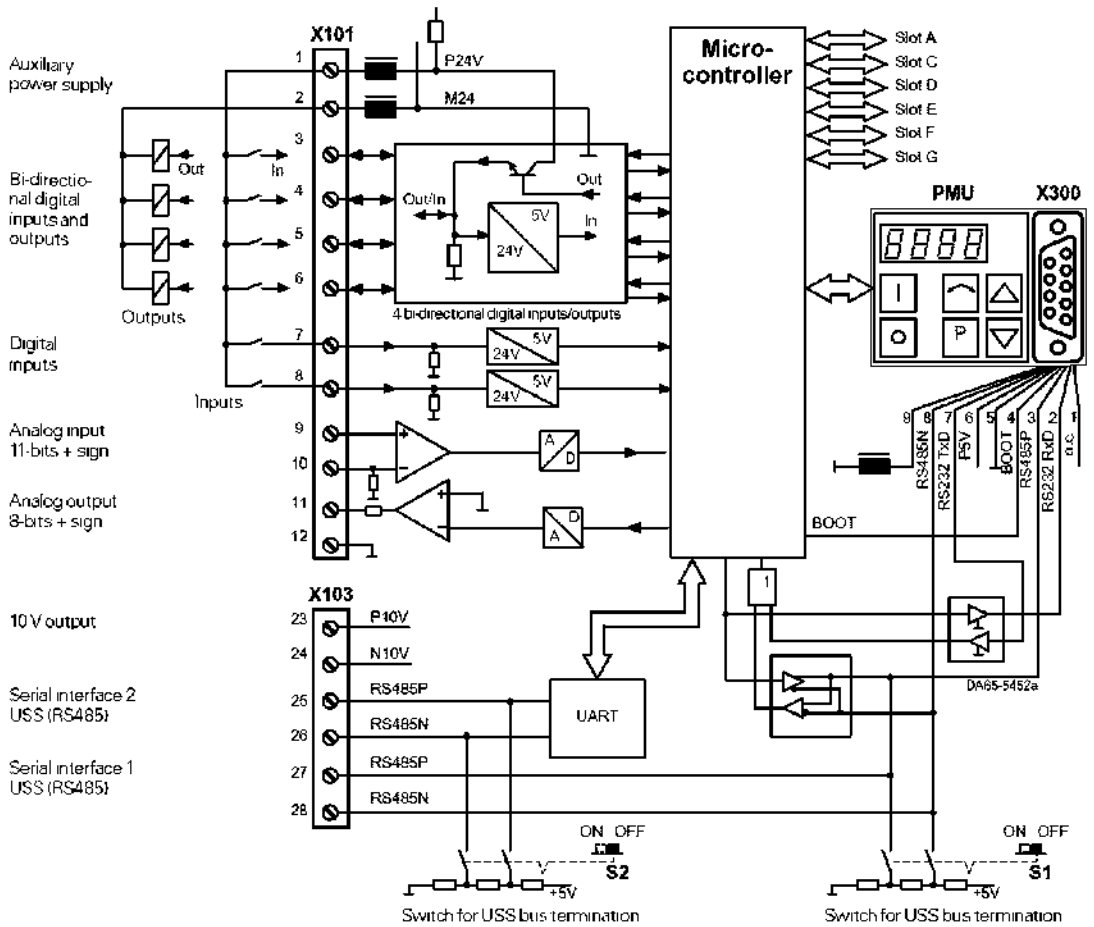
  Compact: 2,2kW - 37,5kW

  Vestavné: 45kW - 250kW

Jednotlivé výkonové řady se liší nejen výkonovým spektrem, ale také v možnostech rozšíření o některé přídavné moduly. Výhodou těchto měničů je vysoká modularita, možnost napájení 1 nebo 4 kvadrantovým napájecím modulem. Snímání otáček a polohy lze realizovat resolverem, optickým sin/cos snímačem, absolutním snímačem nebo inkrementálním snímačem.

3.1 Řídicí elektronika

Elektronika měniče je napájena z vnitřního zdroje stejnosměrného napětí 24 V. Zdroj napětí je připojen na stejnosměrný meziobvod měniče. V elektronickém modulu je realizována proudová, rychlostní i polohová regulační smyčka a do zpětné vazby lze zapojit inkrementální i absolutní snímače nebo resolvery. Všechny funkce měniče lze nastavit a v nejjednodušším případě měnič ovládat pomocí standardního ovládacího panelu (PMU), což pro složitější aplikace nepostačuje. Pro snazší nastavení měniče lze použít sériovou linku RS 232/485 nebo případně pomocí rozšiřující desky CBP průmyslovou sběrnici Profibus, která umožňuje řízení měniče nadřazeným PLC automatem nebo PC. Pro tyto účely je měnič vybaven až 6 rozšiřujícími pozicemi pro přídavné komunikační, rozšiřující a snímačové desky.



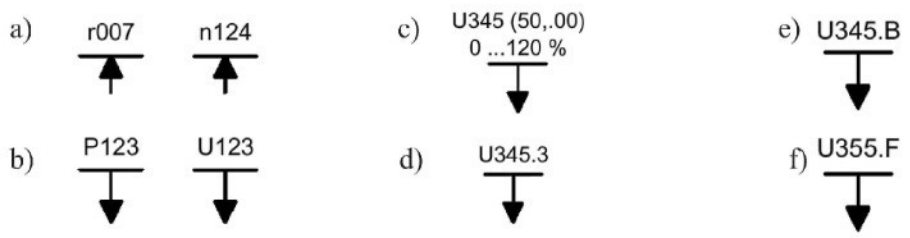
Obr. 3.2: Blokové schéma řídicích vstupů a výstupů, komunikačních linek a PMU ovládacího panelu, převzato z [7]

3.2 Programování funkcí a BICO technologie

Jádrem celého měniče je signálový mikroprocesor, který zajišťuje načítání digitálních a analogových vstupů a aktuální polohy. V dalším kroku vypočítá požadovaný akční zásah do motoru na základě algoritmu. Ten je dán regulační strukturou, kterou je nutné popsat programem. Nakonec nastaví digitální a analogové výstupy a jednotku PWM.

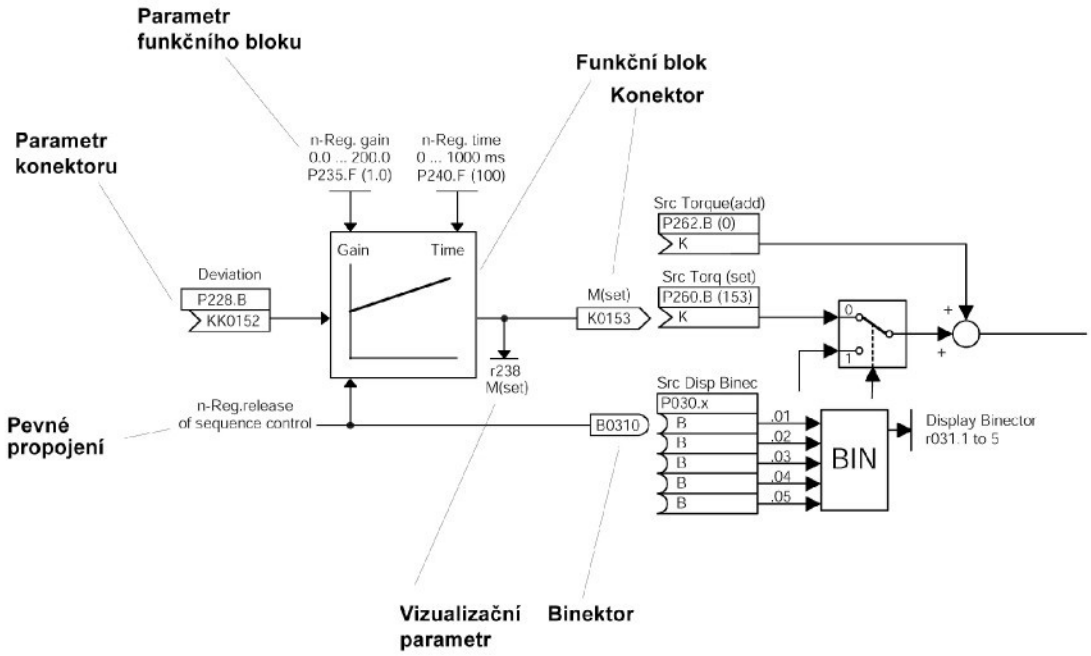
Přizpůsobení funkcí měniče na konkrétní podmínky aplikace se provádí prostřednictvím programu Drive monitor, který je volně ke stažení. Tento program nám umožňuje propojit konektory s parametry dle našich potřeb při programování. Tato propojení pak určují chování měniče.

Každý parametr je jednoznačně definován svým označením a názvem. Označení parametru se skládají z písmene a třímístného čísla. Velká písmena **P**, **U**, **H** a **L** označují parametry, jejichž hodnotu lze měnit, parametry označené malými písmeny **r**, **n**, **d** a **c** pak označují parametry, jejichž hodnotu lze jen číst (vizualizační parametry). Základní symboly pro parametry ve funkčním diagramu MC jsou uvedeny na obrázku (obr. 3.3). Parametry sloužící k zobrazení hodnoty jednotlivých signálů se označují prefixem **r xxx** nebo **n xxx**, kde **xxx** je číslo příslušného parametru (viz. obr. 3.3 a)). Parametry pro nastavování hodnot jednotlivých signálů ve formátu **P xxx** nebo **U xxx** (viz. obr. 3.3 b)). Parametr typu **P** resp. **U** sloužící pro nastavování, jehož přednastavená hodnota od výrobce odpovídá 50.00% a rozmezí (0...120%), ve kterém lze tuto hodnotu parametru měnit je uveden na obrázku (Obr. 3.3 c), d)). Vedle názvu parametrů a jejich čísel mají mnohé parametry také index. Pomocí indexů je možné přiřadit jednomu parametru se stejným číslem více hodnot. U nastavovacích parametrům jsou někdy použity také indexy (viz. Obr. 3.3 e), f)), pro volbu sady Binektorů v BICO technologii či funkčních datových bloků. Tyto indexy umožňují rychlou změnu regulační struktury měniče odpovídající vybrané sadě. Všechny parametry jsou detailně popsány v [7].



Obr. 3.3: Základní symboly pro parametry ve funkčním diagramu MC

Propojování jednotlivých bloků logického obvodu je realizováno tzv. BICO technologií. Tato technologie obsahuje tři základní druhy konektorů, které jsou uvedeny na obrázku (Obr. 3.4). Binektory určené pro signály logických úrovní „0“ a „1“ označené písmenem **B**, konektory s označením **K** jsou určené pro přenos informace o velikosti *word* a konečně konektory s označením **KK** pro informace o velikosti *doubleword*. Pomocí těchto konektorů lze různě strukturovat zapojení regulační struktury uvnitř měniče podle potřebného typu realizovaného pohonu.



Obr. 3.4: Symboly používané v BICO technologii, převzato z [7]

3.3 Ovládací rozhraní, diagnostika a komunikace

Hlavními řídícími vstupy je tlačítko pro zapnutí nebo vypnutí měniče. Dalšími vstupy pro řízení resp. vizualizaci jsou k dispozici analogové či digitální vstupy resp. výstupy. Pro rychlou diagnostiku slouží display umístěný na panelu PMU. Diagnostika pohonu zahrnuje možnost zobrazovat provozní údaje a rovněž rozsáhlý soubor softwarových i obvodových ochran motoru i měniče před přetížením a zničením. Na displeji jsou zobrazeny rovněž čísla aktuálních výstrah či chyb.

Měniče umožňují ovládání buď z vestavěného terminálu nebo pomocí komunikačních rozhraní Profibus, CANbus nebo po sériové lince s protokolem USS z nadřazených PC nebo PLC. Popřípadě pro vzájemnou komunikaci mezi měniči je využita optická komunikační linka SIMOLINK.

3.4 Možnosti a funkce

Základní programové vybavení (firmware) FM dovoluje řešit značně složité úlohy v závislosti na požadavcích ovládané technologie a může být rozšířen o další dodatečné funkce (Options). Všechny základní regulační struktury daného firmwaru jsou doplněny tzv. volně programovatelnými bloky (free blocks). K propojování těchto bloků nebo pro jejich modifikace slouží technika BICO. S její pomocí lze řešit různé modifikace použitých regulačních struktur (viz Kap. 3.2).

V rámci základního softwaru je zde začleněn blok „Simple positioner" (SP) pro realizaci polohového servopohonu. Jsou to jednoduché polohovací funkce nadřazené regulátoru polohy. SP je tvořen třemi základními bloky, které lze použít individuálně nebo mohou být libovolně propojené mezi sebou technikou BICO. SP nabízí tyto tři základní polohovací módy:

Setup mode

Jde o servisní režim. Tato funkce umožňuje při respektování nastavené maximální rychlosti, maximálního zrychlení a směru pohybu popojíždění pohonem. To znamená, že při aktivaci příslušného signálu se pohon rozjede definovanou rychlostí, zrychlením a směrem a po deaktivaci se hřídel motoru zastaví, opět definovaným zpomalením.

Homing mode

Tento mód se používá v případě použití klasického inkrementálního enkodéru, kde není žádná vazba mezi měřením polohy enkodérem a mechanickou polohou stroje při inicializaci řídící jednotky servopohonu. Tento mód se využívá pro nalezení vztažného bodu např. u delších dopravníků nebo při přesném polohování. Lze použít následující režimy:

→ Letmý homing – tento mód je založen na nulování polohy ve „vztažném bodě" udávaným čidlem polohy, od kterého bude opět poloha měřena.

→ Go to home – mód pro aktivní vyhledávání polohy u lineárních os. Rozsah pohybu osy je omezen koncovými spínači. Po aktivaci tohoto režimu se osa pohybuje určeným směrem tak dlouho, dokud nepřejede čidlo nulové polohy nebo dokud nenarazí na koncový spínač. V takovém případě dojde k reverzaci osy a vyhledávání nulového bodu pokračuje v druhém směru.

Positioning mode

V tomto módu určuji cílovou polohu, rychlost, zrychlení a zpomalení pohybu, generování trajektorie již zajišťuje měnič sám.

Lze použít absolutní nebo relativní polohování. Při absolutním polohování se cílová poloha zadává absolutně vzhledem k nulové poloze pohonu. Relativní polohování se vyznačuje pouze zadáním přírůstku polohy vzhledem k aktuální poloze pohonu. Směr pohybu je určen znaménkem zadaného přírůstku polohy.

Tento mód podporuje pohyb jak po lineární tak po rotační ose. Omezení rozsahu je prováděno pro obě osy pohybu odlišně. Omezení pro rotační osu je dáno osovým cyklem, tzn. že každé otáčce odpovídá daný počet jednotek polohy (LU). Pokud zadáme například cílovou polohu 5000 LU a osový cyklus je 4000 LU, pak pohon najede do polohy 1000 LU. V případě lineární osy se využívají tzv. „software limit switch". Jedná se o „vnitřní" programovatelné koncové spínače, které omezují rozsah pohybu pohonu.

3.5 Doporučené servomotory

Doporučené servomotory používané v součinnosti s měniči MC jsou synchronní servomotory s PM do 10^2 kW (např. typy 1FT6, 1FK6 od firmy Siemens) a asynchronní servomotory (např. typ 1PH7 od firmy Siemens) nebo lineární motory. Synchronní servomotory mají velkou momentovou přetížitelnost, vysoké maximální otáčky a minimální moment setrvačnosti, zajišťující výbornou dynamiku pohonu.
Hodnoty pasivních parametrů servomotorů (např. R, L) se zadávají do měniče ručně, pro motory Siemens jsou pevně uloženy v paměti řídicí jednotky (CUMC) měniče a jejich volba se provádí zadáním typového čísla motoru.

4 OPC Technologie

OPC technologie je světovým standardem výměny dat v průmyslové automatizaci. Umožňuje propojení osobních PC k automatizačnímu zařízení. V případě OPC musí aplikace nebo servery, které spolu chtějí komunikovat, implementovat tzv. OPC rozhraní. Servery implementující toto rozhraní se nazývají OPC servery, klienti OPC klienti. OPC server umožní kterékoliv klientské aplikaci přístup k datům serveru nebo k zařízení, se kterým OPC server komunikuje. OPC server je aplikace ekvivalentní komunikačnímu ovladači (driver) pro datovou komunikaci a umožňuje současné připojení libovolnému počtu OPC klientů, kteří tato data ze serveru získávají. OPC server vyhodnocuje čtená data. Pokud dojde ke změně sledované veličiny server o tom dá vědět svým klientům. Pokud se naopak hodnoty sledovaných veličin nemění, nezatěžuje OPC server síť ani klienty zbytečnými komunikacemi. Komunikace s OPC serverem je tedy založena na událostech a ne na periodickém vyhodnocování změn. Klienti mohou posílat pomocí OPC serveru do zařízení i povely.

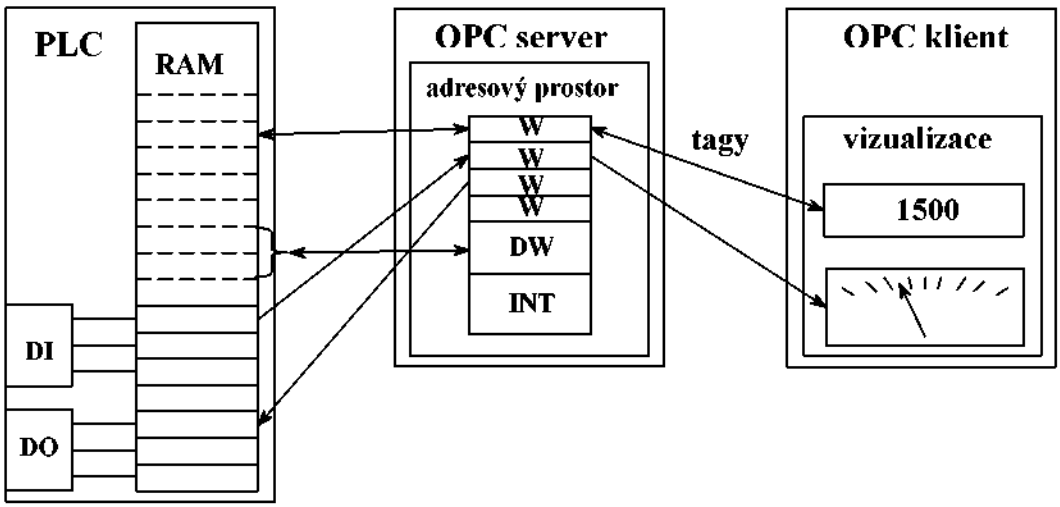
Existuje mnoho klientských aplikací, které vyžadují data z datových zdrojů a přístup k těmto datům přes vyvinuté ovladače. Ke každému zařízení musí být vyvinut ovladač pro jeho klientskou aplikaci. Datové zdroje mohou být reprezentovány buď přímo průmyslovými zařízeními, jako jsou PLC apod., databázemi řídících aplikací např. SCADA systémů nebo OPC serverem.

OPC je založeno na COM technologii, která umožňuje jedné aplikaci využívat rozhraní dalších aplikací k získávání informací a různých funkcí. Rozhraní je jediný způsob, jak může klient přistupovat k členským datům a funkcím objektu a to pomocí ukazatele na dané rozhraní.

OPC poskytuje několik výhod, výrobci hardwaru mohou vyvinout pouze jednu sadu ovladačů pro vývojáře koncových aplikací. Vývojáři softwaru nebudou muset přepisovat aplikace kvůli změnám vlastností nebo aktualizacím hardwaru, jednoduše použijí novější ovladač se stejným rozhraním jako měl ten původní. Další výhodou je, že může několik klientských aplikací současně přistupovat k datovým zdrojům.

4.1 Adresový prostor OPC serveru

Data jsou umístěna v adresovém prostoru OPC serveru. OPC klienti přistupují k těmto datovým položkám prostřednictvím tzv. tagů, což jsou ukazatele do adresového prostoru OPC serveru. Jednotlivé tagy jsou obrazem dat umístěných v systému PLC, PC serveru nebo databázi. Možnosti přístupu k jednotlivým datovým, či paměťovým prostorům ilustruje obrázek (Obr. 4.1). Směr komunikace mezi OPC klientem a OPC serverem závisí na typu datového elementu umístěného v paměti PLC. Pokud přistupujeme z OPC klienta do paměti RAM PLC, tak je tato komunikace obousměrná. OPC server může z výstupů číst a také do nich zapisovat, naopak ze vstupů může pouze číst. Od toho se odvíjí i směr komunikace mezi OPC klientem a OPC serverem, který funguje jen jako zprostředkovatel dat z PLC. Na jedno místo v paměťovém prostoru nebo na jeden digitální vstup či výstup může odkazovat i více tagů (i různého datového formátu).



Obr. 4.1: Přístupy k adresovému prostoru pomocí tagů

5 Popis pracoviště a součinnosti jednotlivých prvků

Pohon v naší úloze tvoří frekvenční měnič *Siemens Masterdrives Motion Control* s označením 6SE7016 – 1EA51 – Z, který má parametry (viz. Tab. 5.1). Další součástí pohonu je servomotor 1FT6062-6AF71-3EA0 s parametry uvedenými v tabulce (Tab. 5.2).

Vstupní napětí	U_1 = 380..480 V, trojfázové
Vstupní frekvence	f = 50/60 Hz
Výstupní napětí 3AC	U = 0-380..480 V
Výstupní frekvence	f‘ = 0..400 Hz
Jmenovitý výstupní proud	I_{Un} = 6,1 A
Výkonové ztráty	P_z = 0,15 W
Jmenovitý proud v DC meziobvodu	I_{DCn} = 7,3 A
Hmotnost	m = 9,5 kg

Tab. 5.1: Parametry použitého FM

Jmenovité otáčky	n_n = 3000 ot./min.
Maximální otáčky	n_{max} = 4800 ot./min.
Jmenovitý výkon	P_n = 1,5 kW
Jmenovitý moment	M_n = 4,7 N.m
Klidový moment	M_0 = 6,0 N.m
Jmenovitý proud	I_n = 3,4 A
Klidový proud	I_0 = 4,0 A
Moment setrvačnosti	J = 0,85.10-3 kg.m^2
Hmotnost	m = 9,5 kg

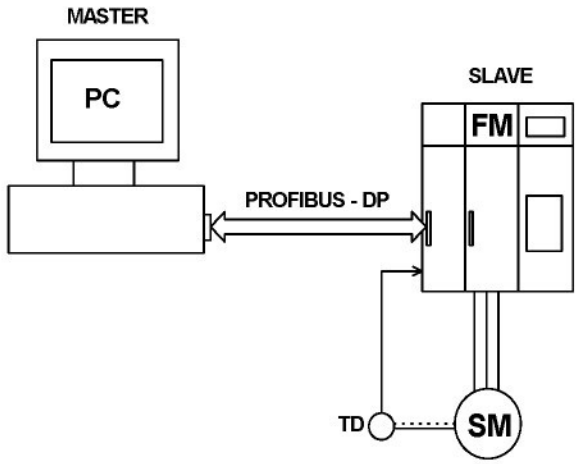
Tab. 5.2: Parametry použitého servomotoru

Nadřazený řídící systém pohonu tvoří osobní počítač (dále označován jako Server PC), vybaven komunikační kartou CP 5611 pro připojení sběrnice Profibus a běžnou síťovou kartou pro Ethernet. Nutným softwarovým vybavením tohoto PC je OPC server „OPC.SimaticNET“, který poskytuje firma Siemens. Pro konfiguraci pak slouží programy „HW–Config“ a „NET Pro“, které jsou součástí programového balíku Step 7. HW–Config je součástí programu „SIMATIC Manager“, je určen pro specifikaci zařízení použitých v projektu. Zde slouží k definování hardwarové struktury Master OPC serveru a komunikačních linek napojených na Slave zařízení.

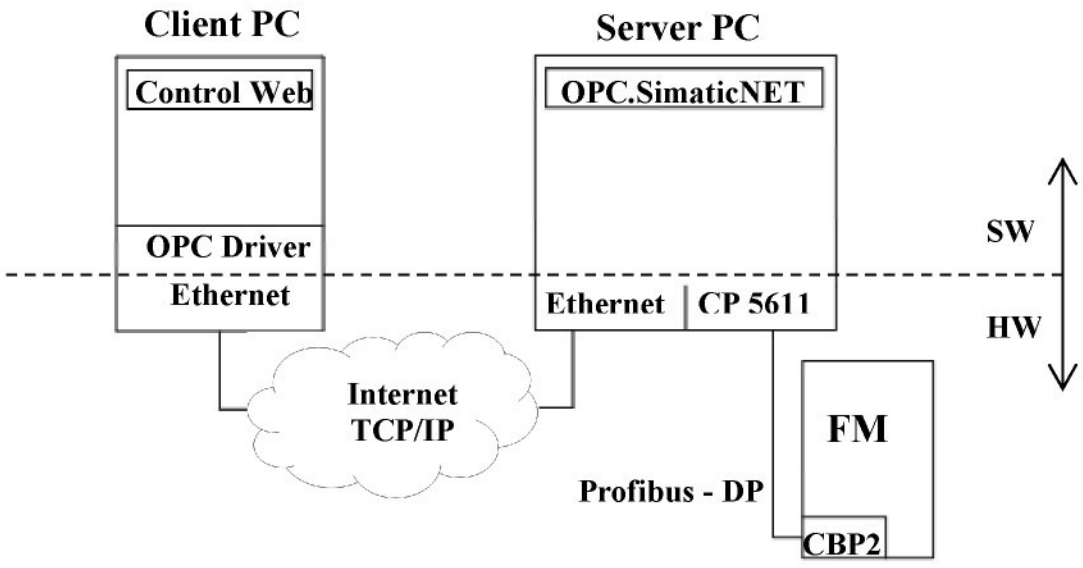
Program NET Pro slouží ke konfiguraci sítě a k zpřehlednění komunikačních linek mezi jednotlivými zařízeními.

Pomocí programu „Drive monitor“ napojujeme příslušné konektory na parametry uvnitř měniče (např. P734.01 = K30), (viz. Kap. 6.5 a 3.2). Pro ověření funkčnosti OPC komunikace je vhodný jednoduchý OPC klient, firma Siemens nabízí aplikaci „OPC Scout“.

Na straně klienta (Client PC) pak stačí jen běžná síťová karta pro Ethernet. Jako OPC klient zde vystupuje aplikace v prostředí Control Web 5 s příslušným OPC ovladačem. Klientská aplikace může být spuštěna i na Server PC. Hierarchie řízení a komunikačních rozhranní je uvedena na obrázku (Obr. 5.1). Uspořádání software a hardware včetně komunikačních rozhraní ilustruje obrázek (Obr. 5.2).



Obr. 5.1: Hierarchie řízení a komunikačních rozhranní



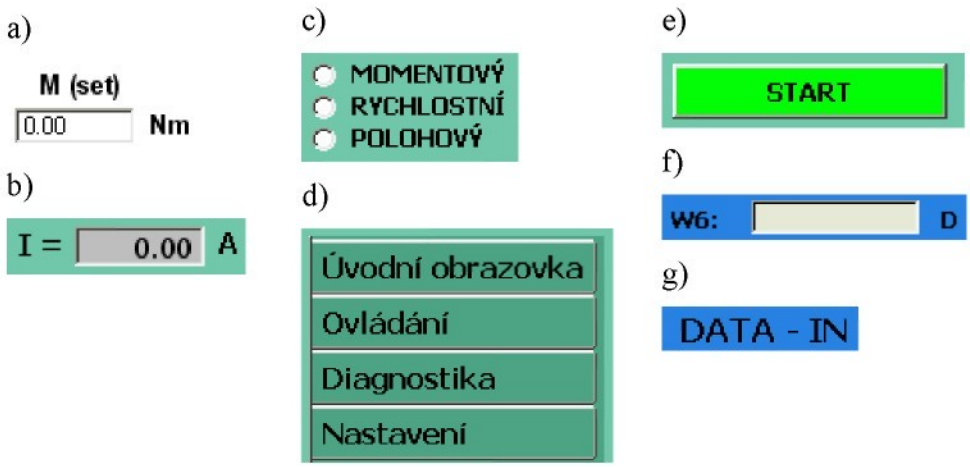
Obr. 5.2: Uspořádání SW a HW včetně komunikačních rozhranní

6 Aplikace v prostředí Control Web 5

Vývojové prostředí Control Web slouží k návrhu aplikací pro vizualizaci různých technologických procesů. Obsahuje všechny komponenty nutné pro tvorbu vizualizačních aplikací – zobrazovací a ovládací prvky, i specializované přístroje (např. alarmy a archivy). Umožňuje programovat jak datově řízené aplikace, tak i náročné aplikace v reálném čase pro řízení procesů. Struktura programu je tvořena přístroji a jim přidruženými funkcemi. Každý přístroj vlastní své událostní procedury, pomocí kterých je možné sestavit velmi kvalitní program, v případě nedostatku funkcí nabízených prostředím CW 5 si uživatel může mnohé vytvořit i sám s využitím povolených programovacích prostředků.

6.1 Základní informace o použitých přístrojích

Přístroje se dělí dle funkce na zobrazovací, ovládací a řídící, popisové, aj. Dělí se do dvou hlavních skupin, na přístroje viditelné a neviditelné. Neviditelný přístroj *Program* byl využit pro deklarování funkcí, které jsou volány z viditelných přístrojů. Mezi námi využité viditelné přístroje patří především *Panel*, *Control (a)*, *Meter (b)*, *Multiswitch (c)*, *Tabswitch (d)*, *Switch (e)*, *Stringdisplay (f)*, *Table*, *Label (g)* a *Draw*. Vzhled některých přístrojů ilustruje obrázek (Obr. 6.1).



Obr. 6.1: Vzhled některých přístrojů použitých v naší aplikaci
a) *Control*, b) *Meter*, c) *Multiswitch*, d) *Tabswitch*, e) *Switch*, f) *Stringdisplay*, g) *Label*.

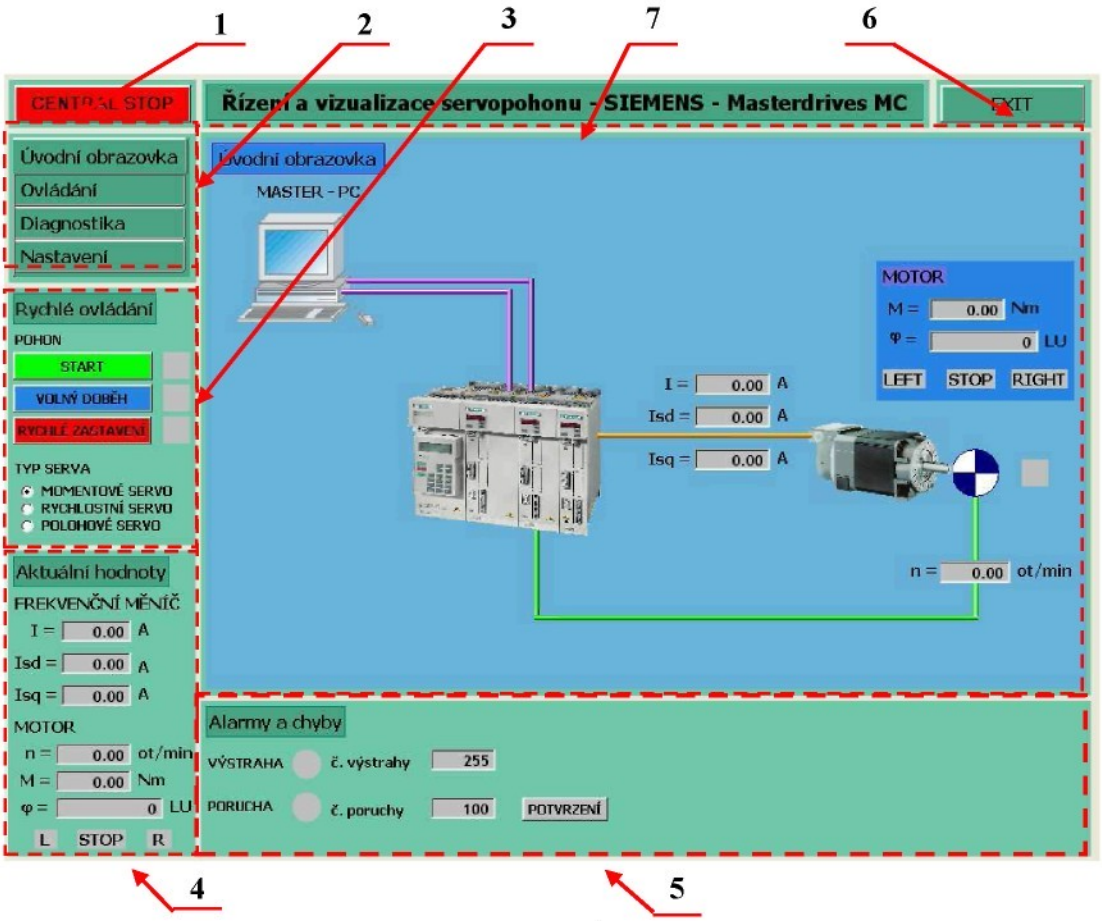
Přístroj *Panel* slouží k umisťování jednotlivých přístrojů například v případě potřeby jejich současného zobrazení. *Control* je přístroj sloužící k zadávání hodnot proměnných, *Meter* naopak umožňuje jejich zobrazení. *Multiswitch* a *Tabswitch* jsou

přístroje, které umožňují výběr jedné z více položek, v naší aplikaci byly například tyto přístroje využity pro přepínání mezi *Panely*. *Switch* je přístroj s booleovským výstupem buď TRUE při sepnutí nebo FALSE při rozepnutí. Přístroj *Stringdisplay* je určen pro zobrazení hodnoty typu *string*, v aplikaci byl použit například pro zobrazení hodnot v hexadecimální soustavě, kdy byla použita převodní funkce *str(proměnná, 16)*. *Table* je běžná tabulka, která byla využita při diagnostice chyb. Přístroj pro vložení různých popisků se nazývá *Label*. Nakonec bychom mohli zmínit přístroj *Draw*, který byl v naší aplikaci použit především v panelech *Ovládání* pro grafickou ilustraci regulačních struktur.

6.2 Vzhled aplikace

Aplikace je tvořena co nejnázorněji, aby posloužila výukovým účelům. Je strukturována do jednotlivých panelů, které jsou přepínány pomocí přepínačů *Tabswitch*. Hlavní panel s níže popsanými částmi je uveden na obrázku (Obr 6.2).

Panel obsahuje tlačítko *CENTRAL STOP (1)*, které umožňuje kdykoliv vypnout pohon. Pohon je možné opět nastartovat až po uvolnění tohoto tlačítka. Pod ním je umístěn hlavní přepínač panelů *(2)*, ve kterém je možné vybrat základní panely aplikace, *Úvodní obrazovku*, *Ovládání*, *Diagnostiku* a v poslední pozici panel pro různá *Nastavení*. Níže jsou umístěna tlačítka pro *Rychlé ovládání (3)* pohonu, která umožňují start pohonu, volný doběh či rychlé zastavení a také možnost volby *Typu servopohonu*. Ten může být momentový, rychlostní nebo polohový. *Aktuální hodnoty* z FM a motoru jsou zobrazeny v levém dolním rohu hlavního panelu *(4)*. Je zde informace o aktuální hodnotě celkového proudu, který teče z měniče do motoru, a jeho složkách – tokotvorné I_{sd} a momentotvorné I_{sq}. Dále jsou tu hodnoty aktuální rychlosti, momentu a polohy a signalizace směru otáčení hřídele motoru. Vedle je umístěn panel pro zobrazení *Alarmů a chyb (5)*, který signalizuje chybu či výstrahu a její číslo. Vzniklou chybu lze potvrdit pomocí tlačítka *POTVRZENÍ*. V pravém horním rohu hlavního panelu je umístěno tlačítko *EXIT (6)* pro ukončení aplikace. Uprostřed jsou zobrazovány panely, vybrané hlavním přepínačem panelů, o kterých se dozvíte v následující kapitole. Na obrázku (Obr 6.2) je tímto panelem *Úvodní obrazovka (7)*.



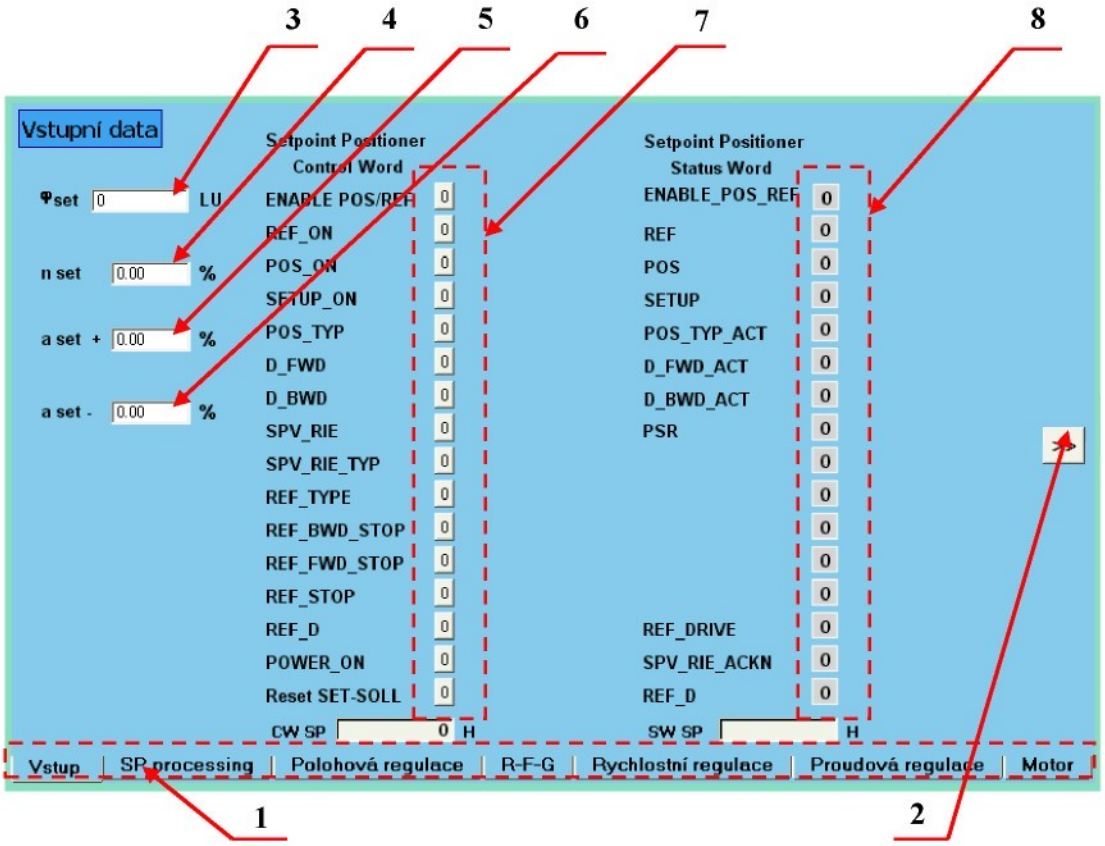
Obr. 6.2: *Hlavní panel* s *Úvodní obrazovkou*

6.3 Základní panely aplikace

Na *Úvodní obrazovce (7)* – (Obr. 6.2 uprostřed) je k dispozici náhled celého zařízení se zobrazením důležitých parametrů jednotlivých komponent – řídícího počítače a servopohonu.

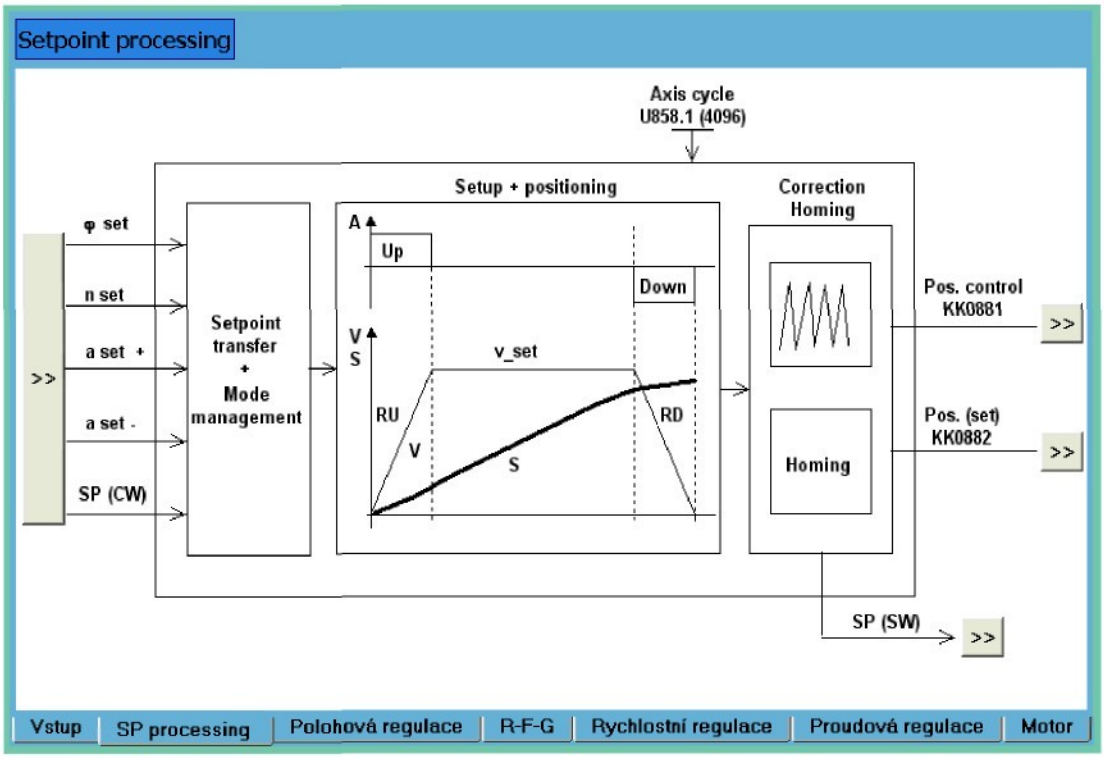
Při výběru položky *Ovládání* (Obr. 6.3) se zobrazí panel, pod kterým je přepínač *(1)* pro vizualizaci jednotlivých částí regulační struktury měniče s grafickým vyobrazením a příslušnými editačními a monitorovacími přístroji. Pro jednoduché přepínání mezi těmito panely slouží tlačítka s dvojitou šipkou *(2)*, která nás přepínají mezi navazujícími panely. Všechna grafická vyobrazení v následujících panelech jsou pouze ilustrativní.

Jako první je zde panel se *Vstupními daty* (Obr. 6.3) pro blok Simple Positioner polohového servopohonu. Nastavuje se tu žádaná poloha *(3)*, rychlost *(4)* a zrychlení *(5)* či zpomalení *(6)*. Dále je zde možné nastavovat resp. zobrazovat jednotlivé bity řídícího *(7)* resp. stavového *(8)* slova bloku Simple Positioner.

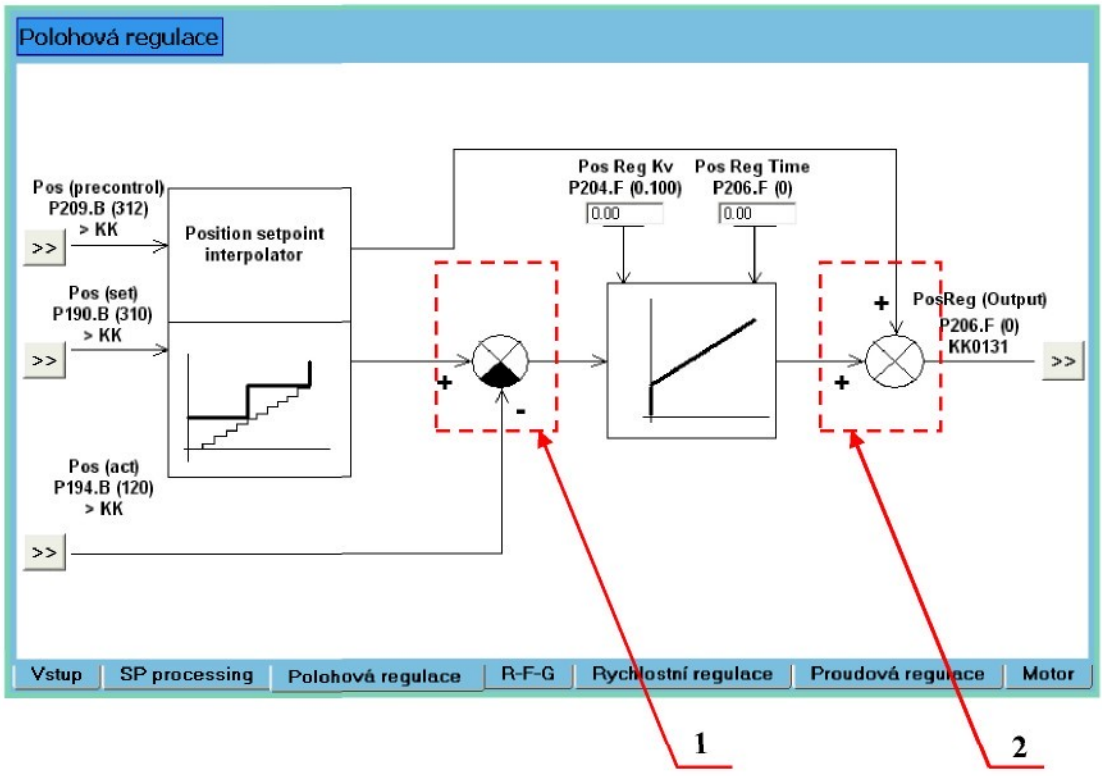


Obr. 6.3: Panel *Vstupní data*

Na následujícím obrázku (Obr. 6.4) je vyobrazen panel *Setpoint processing* se základní strukturou bloku Setpoint positioner.



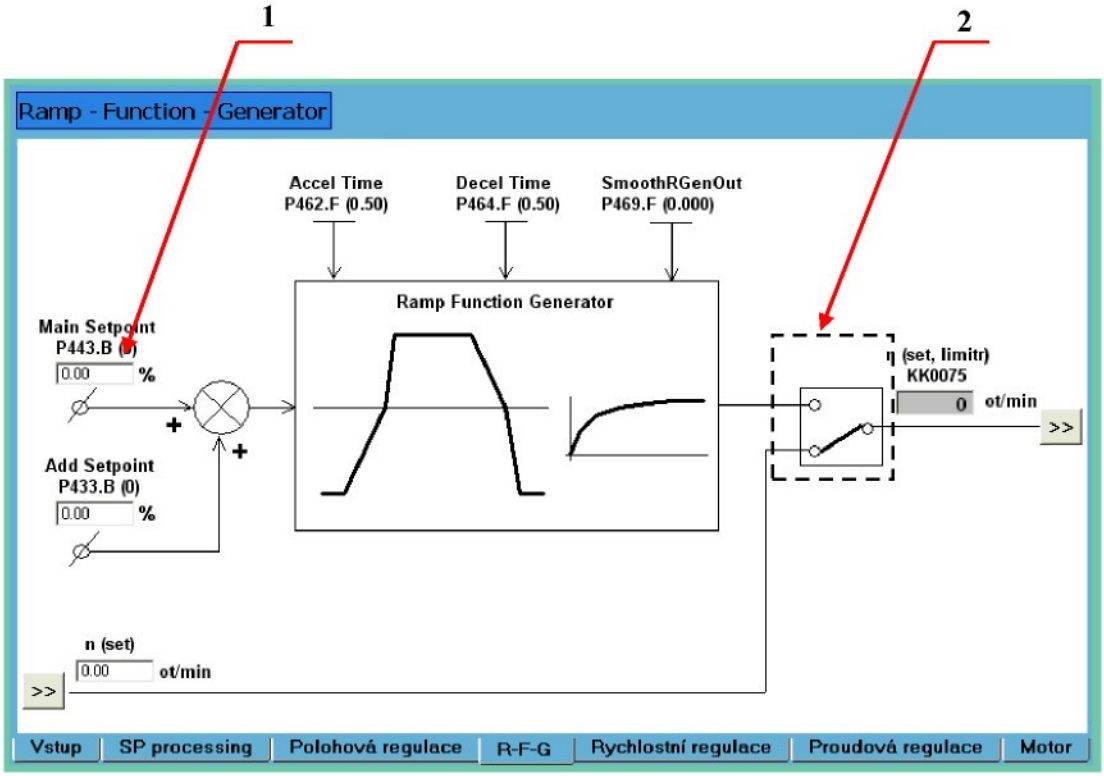
Obr. 6.4: Panel *Setpoint processing* a základní struktura bloku Setpoint positioner



Obr. 6.5: Panel *Polohohová regulace*

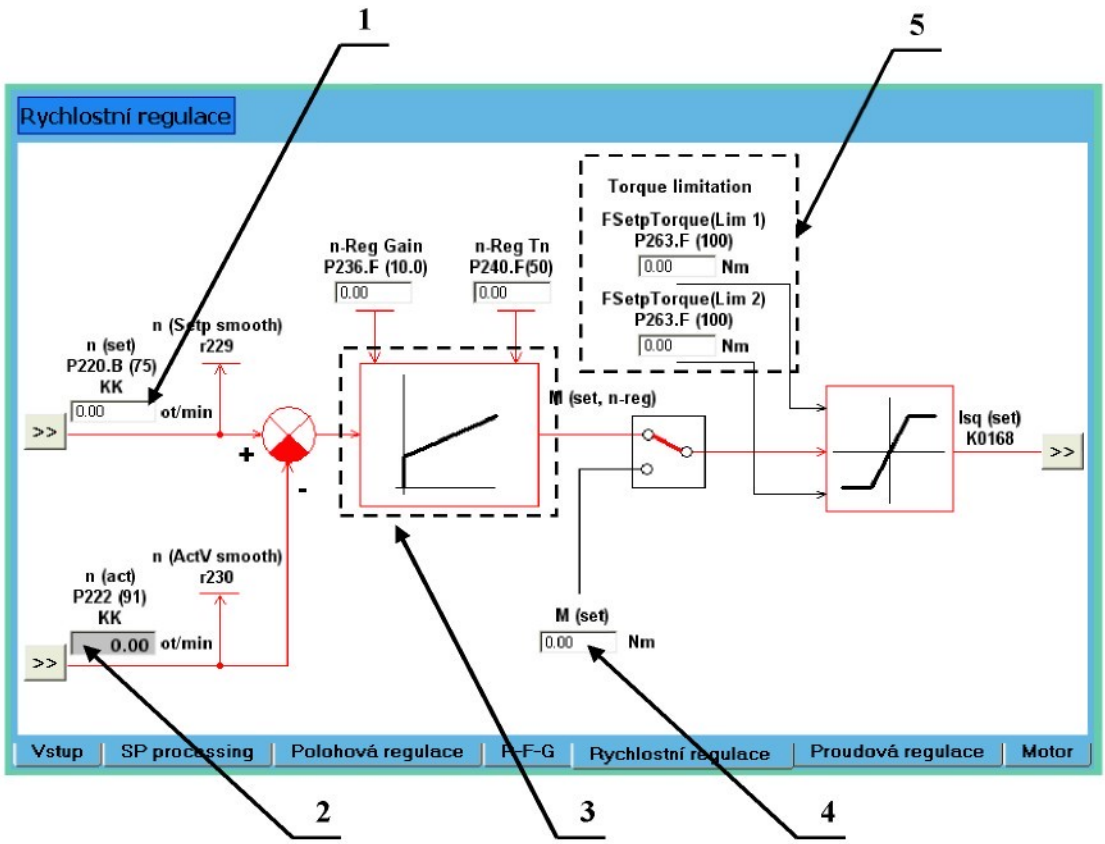
Na panelu *Polohová regulace* (Obr. 6.5) je znázorněno porovnávání žádané polohy s aktuální polohou *(1)*. Dále je tu naznačen součtový člen *(2)*, „porovnávající“ hodnotu z výstupu regulátoru polohy s hodnotou z polohového interpolátoru.

Struktura panelu *Rampový funkční generátor* (RFG) je zřejmá z obrázku (Obr. 6.6). Tento panel umožňuje zadání žádané hodnoty rychlosti *(1)*, která vstupuje do bloku RFG. Za tímto blokem dochází k rozhodování pomocí přepínače *(2)*, který „volí“ odkud se bude číst žádaná rychlost. Ta může být čtena buď z RFG nebo z výstupu regulace polohy.



Obr. 6.6: Panel *Rampový funkční generátor*

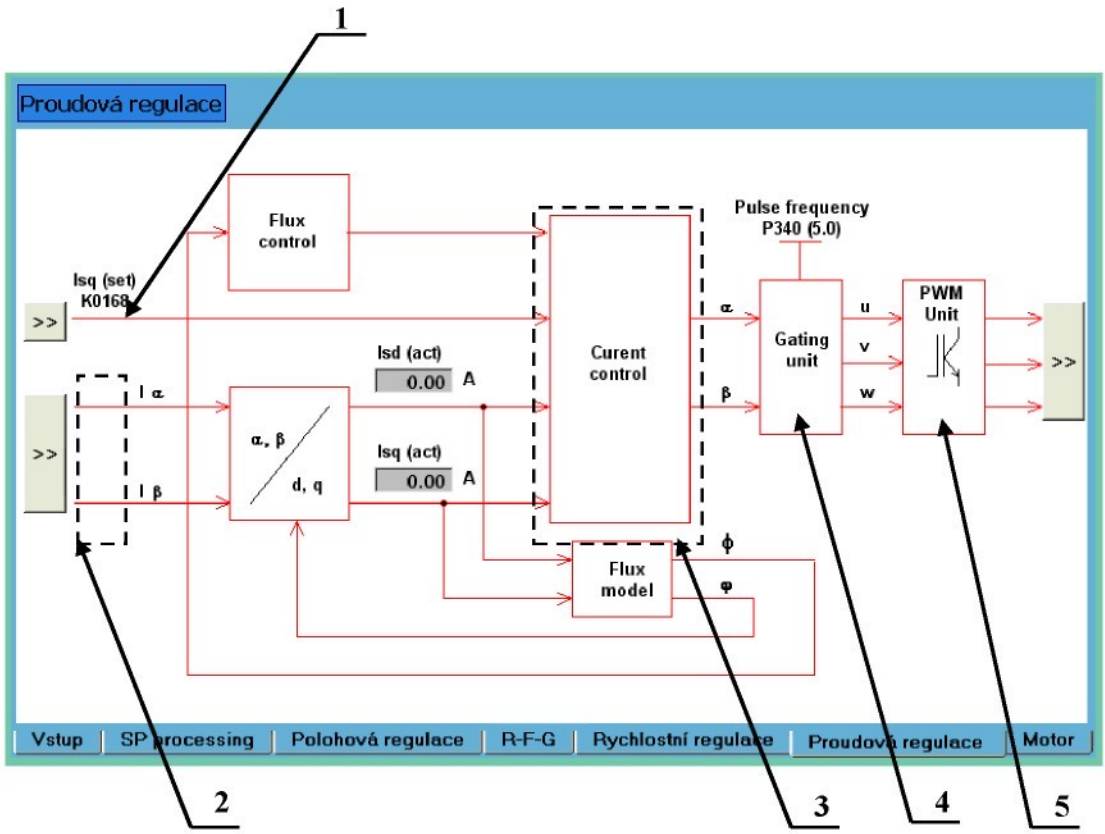
Další panel *Rychlostní regulace* (Obr. 6.7) umožňuje nastavení žádané hodnoty rychlosti *(1)* v případě volby rychlostního servopohonu a ilustruje její porovnání s aktuální hodnotou *(2)*. Je zde také naznačen přepínač s možností výběru čtení momentu buď z výstupu rychlostního regulátoru *(3)* nebo přímo z přístroje control *(4)* (pro momentový servopohon). Momentové omezení, lze zadat dvěma parametry *(5)*.



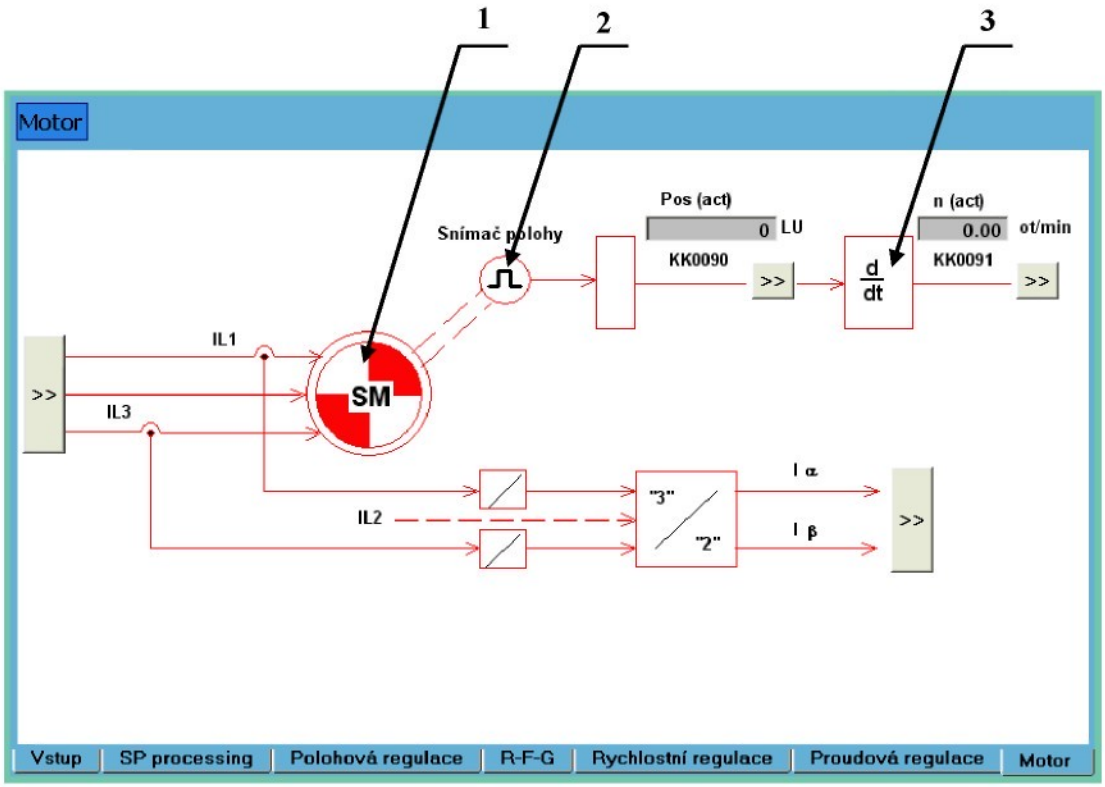
Obr. 6.7: Panel *Rychlostní regulace*

Vstupem dalšího panelu, *Proudové regulace* (Obr. 6.8) je momentotvorná složka proudu *(1)* z panelu momentové regulace a zpětná proudová vazba z jednotlivých fází motoru *(2)*. Do modulu proudového řízení *(3)* pak vstupují přepočítané a žádané hodnoty tokotvorné a momentotvorné složky proudu. Další navazující blok řídicí jednotka *(4)* ovládá blok tranzistorů IGBT, kde je realizována PWM *(5)*.

Na posledním panelu ovládání jménem *Motor* (Obr. 6.9) můžeme vidět synchronní motor *(1)* s vyobrazeným snímačem aktuální polohy *(2)* a derivačním blokem *(3)* pro získání aktuální rychlosti.

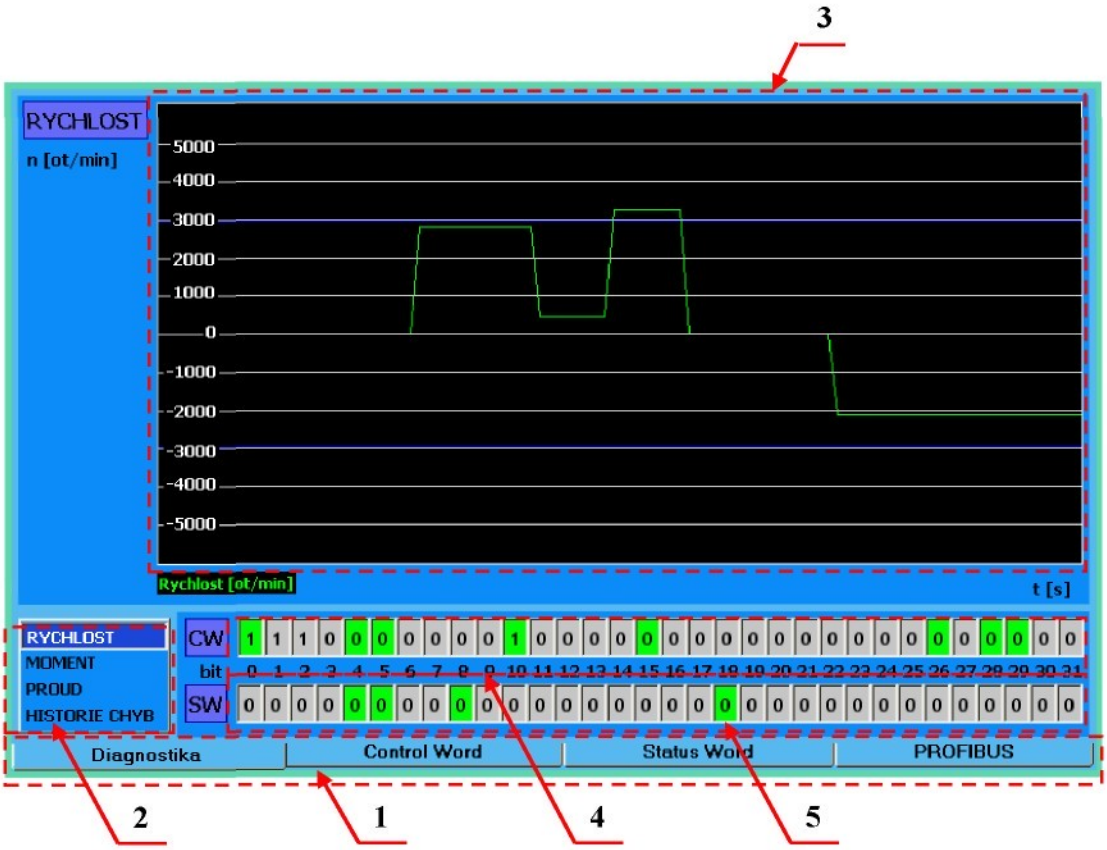


Obr. 6.8: Panel *Proudová regulace*



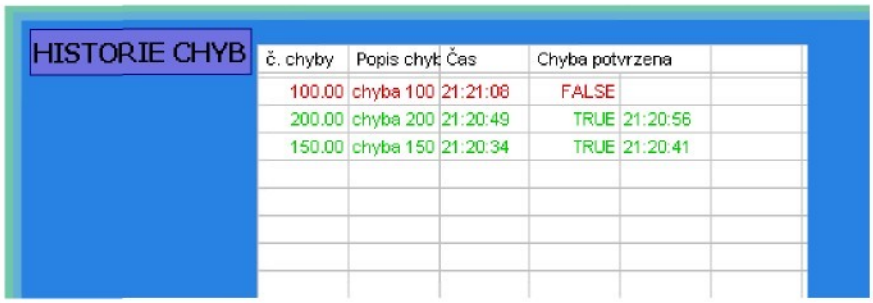
Obr. 6.9: Panel *Motor*

Důležité informace o stavu pohonu lze nalézt na panelu *Diagnostika,* na jehož spodní části najdeme přepínač mezi panely *Diagnostika*, *Control word*, *Status word* a *Profibus (1)*. Nad přepínačem je bitové zobrazení řídícího *(4)* a stavového *(5)* slova.



Obr. 6.10: Panel *Diagnostika*

Pokud vybereme panel *Diagnostika* (Obr. 6.10) objeví se v přístroji *Multiswitch (2)* tyto volby: zobrazení průběhu *Rychlosti (3)*, *Momentu* nebo jednotlivých *Proudů* v čase a také tabulka *Historie chyb* (Obr. 6.11), kde je sledován vznik resp. potvrzení chyby a informace o čase kdy k tomu došlo. Nechybí zde ani údaj o názvu a čísle vzniklé chyby.



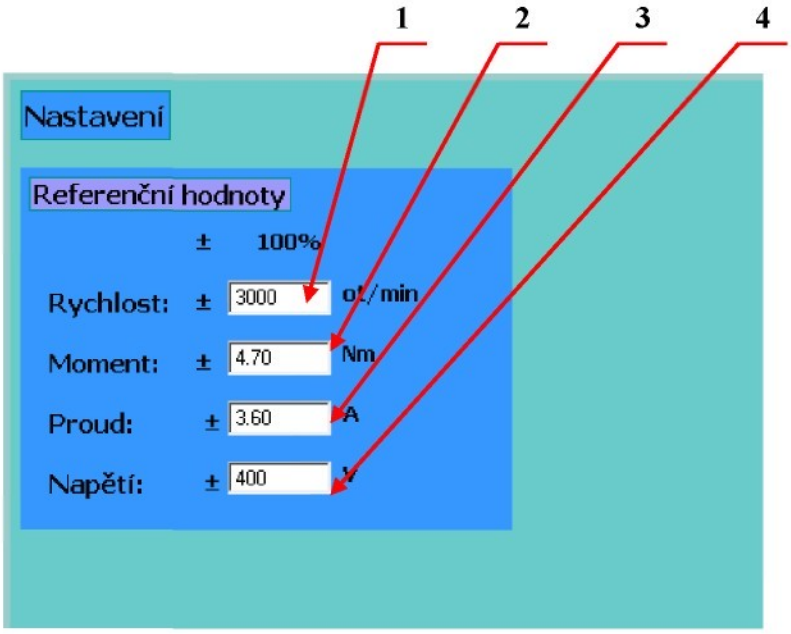
Obr. 6.11: Výřez z panelu *Historie chyb*

Obr. 6.12: Panel *Control Word*

V panelu *Control word* (Obr. 6.12) jsou zobrazeny jednotlivé bity řídícího slova *(1)*, které jsou doplněny o jejich význam *(2)*. Dále je zde umístěno tlačítko *(3)*, které umožní nastavit hodnotu jednotlivých bitů *(4)* pomocí jednotlivých *Switchů*. V panelu *Status word* jsou sledovány jednotlivé bity slova stavového. Pro jednotlivé *wordy* těchto 32 bitových slov je zobrazeno jak v panelu *CW* i *SW* také hexadecimální vyjádření *(5)*.

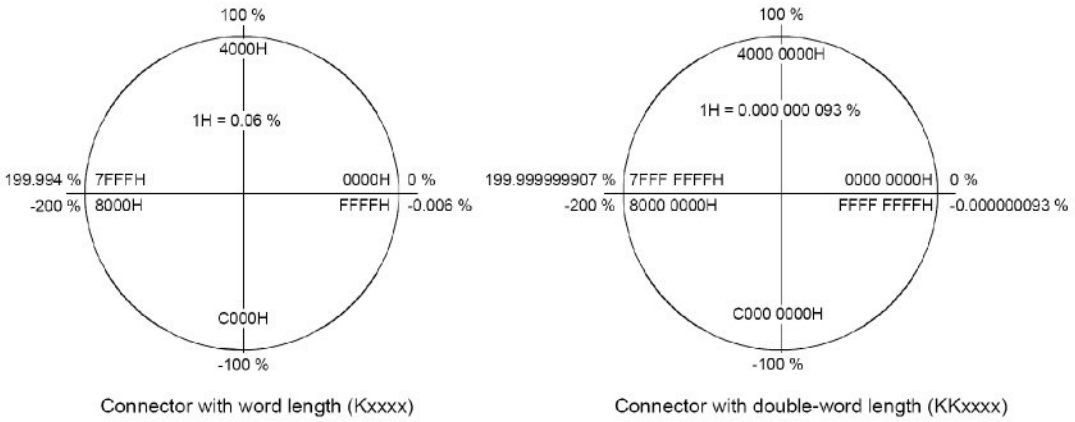
Poslední položkou je panel *Profibus* (Obr. 6.13), kde jsou zobrazena vstupní (IN) a výstupní (OUT) data z aplikace. Jde o data, která jsou posílána jednotlivými komunikačními kanály (viz. Kap. 6.5), jejich vyjádření je v dekadické *(1)* i hexadecimální *(2)* podobě.

Obr. 6.13: Panel *Profibus*



Obr. 6.14: Panel *Nastavení*

V panelu *Nastavení* (Obr. 6.14) je možné navolit referenční hodnoty pro rychlost *(1)*, moment *(2)*, proud *(3)* a napětí *(4)*. Tyto hodnoty jsou používány při převodech z procentuálního vyjádření na fyzikální veličinu a zpět. Uvnitř měniče jsou totiž všechna data vyjádřena dekadickými čísly ve formátu *word* (rozsah od nuly do 2^{16}) nebo ve formátu *doubleword* (do 2^{32}) a tomu přímo odpovídají hodnoty v procentech dle následujícího obrázku (Obr 6.15). Proto je nutné stanovit referenční hodnotu příslušné veličiny, která odpovídá 100%.



Obr. 6.15: Převody mezi procenty a jejich dekadickým vyjádřením, převzato z [7]

6.4 Způsoby řízení aplikace

Vývojové prostředí Control Web 5 dovoluje programátorovi vytvářet aplikace dvojím způsobem. První možnost představují tzv. datově řízené aplikace. Chování aplikace je do značné míry určeno nevizuálním jádrem aplikace. Jádro samo zajišťuje vazby mezi daty a přístroji. Pokud se například změní hodnota vstupního kanálu aplikace, přístroj *Meter* „sám od sebe“ zobrazí aktuální hodnotu (jádro aplikace přístroj *Meter* samo aktivuje).

Druhou volbu tvoří tzv. aplikace reálného času. Chování aplikace je opět řízeno jádrem aplikace, ovšem vazby mezi daty a přístroji (resp. jejich aktivaci) musí programátor ošetřit sám. Změna hodnoty vstupního kanálu se na přístroji *Meter* neprojeví, dokud tento přístroj není aktivován. Aktivaci zde musí zajistit programátor sám (např. časovačem nebo jako důsledek aktivace jiného objektu v aplikaci). Tento typ aplikací klade na jejich tvůrce vyšší nároky, na druhou stranu dovoluje vytvářet aplikace, jejichž chování (časování jednotlivých přístrojů) může být časově velmi provázané a lze je použít pro řízení časově kritických procesů.

Naše aplikace byla tvořena nejprve jako datově řízená, ale později se ukázalo, že je to překážkou. Například při změně dvou proměnných, se kterými daný přístroj pracoval, docházelo k dvojí aktivaci tohoto přístroje, což se ukázalo jako neřešitelný problém. Aplikace totiž obsahuje poměrně složité vazby mezi více přístroji a datovými elementy. Správné chování bylo možné zajistit pouze použitím aplikace reálného času a programově definovaným aktivováním požadovaných elementů.

6.5 Komunikace a komunikační telegramy

Pro komunikaci s vnějšími zařízeními využívá aplikace tzv. *kanály*. Pro ty je nutné nastavení ovladačů, v naší aplikaci je využit ovladač OPC klient pro Control Web. Ovladač zajišťuje spojení mezi vstupně/výstupním zařízením a aplikací v prostředí Control Web. Realizace definic kanálů v programu viz. příloha B, odst. 3.

Každý použitý ovladač obsahuje mapovací soubor (*.dmf) (viz. Obr. 6.16), kde je uvedeno u každé položky příslušné číslo *kanálu*, jeho typ a směr. Blok těchto položek je uvozen klíčovým slovem *begin* a ukončen slovem *end*.

```
begin
1 longint bidirectional
2 longint bidirectional
3 longint bidirectional
    ...
end.
```

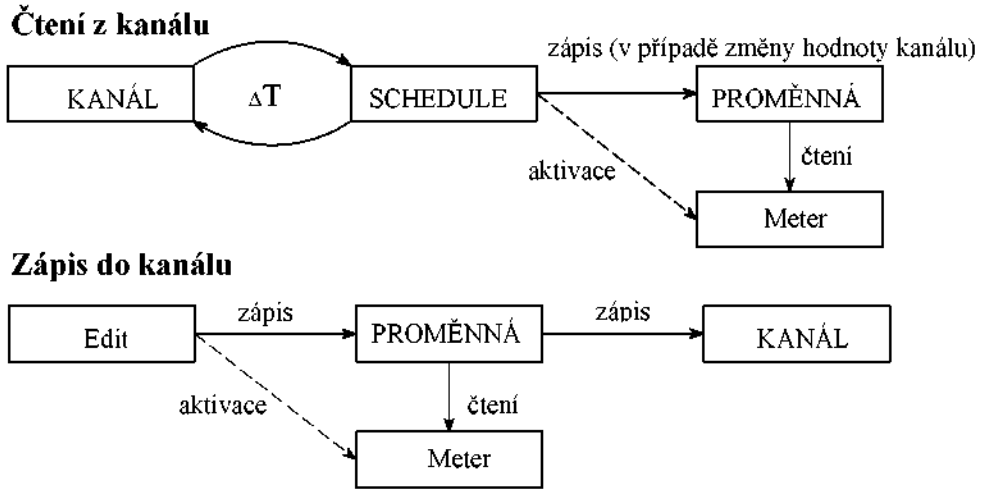
Obr. 6.16: Příklad struktury mapovacího souboru

```
[comment]
OPCDRV PAR soubor vytvořen konfigurátorem OPC ovladače

[server]
CLSID={B6EACB30-42D5-11D0-9517-0020AFAA4B3C}
sync=true
wait_running=true
host=\\PRI

[channels]
1 = DP:[CP 5611]SLAVE003_IW8,1
2 = DP:[CP 5611]SLAVE003_IW10,1
3 = DP:[CP 5611]SLAVE003_IW12,1
    ...
```

Obr. 6.17: Příklad struktury parametrického souboru

Dále musí ovladač obsahovat parametrický soubor (*.par) (viz. Obr. 6.17) pro konfiguraci ovladače. V něm je uveden identifikátor třídy OPC serveru uvozený slovem *[server] a CLSID*, název hostitelského PC kde vizualizační aplikace běží (*host=\\PRI)*, a nakonec jsou zde uvedena čísla jednotlivých *kanálů* s parametry komunikace. Tyto parametry určují propojení *kanálů* s příslušnými tagy (viz. Kap. 4.1), ze kterých je zřejmá také struktura tohoto datového prostoru. Pro naši aplikaci jsme využili 20 vstupně-výstupních *kanálů*, kde prvních 10 využíváme pro vstup a druhých 10 pro výstup. Úplný výpis těchto souborů je k dispozici v příloze A.

Každý vstupní *kanál* je napojen na tzv. *schedule*, což jsou plánované datové elementy. Probíhá zde periodické vyhodnocování *kanálů*. *Schedule* využívají událostní procedury *OnScalarNumericalDataChange* (viz. Příloha B, odst. 5), která v případě změny hodnoty datového elementu načte *kanál* do příslušné vstupní proměnné a současně aktivuje příslušné přístroje příkazem *send*. Plánované datové elementy byly využity z důvodu snížení zatížení procesoru, který by musel po časových krocích číst jednotlivé kanály a současně aktivovat i všechny přístroje, které by s danou proměnnou pracovaly. Pokud zvolíme možnost využít plánované datové elementy, tak se do proměnné načítá kanál pouze v případě změny jeho hodnoty. Samozřejmě je na tomto principu postavena i aktivace přístrojů, což podstatně ušetří práci procesoru. Výstupní kanály jsou napojeny přímo na výstupní proměnné a jsou aktivovány pouze jejich změnou. Schéma, popisující přenosy dat mezi kanály a proměnnými s aktivací příslušných přístrojů, je uvedeno na obrázku (Obr. 6.18). Programové napojení *schedule* na *kanály* umístěno v příloze B, odst. 4.



Obr. 6.18: Práce s kanály při jejich čtení, a zápisu do nich

V projektu jsou použity tři regulační struktury: momentový, rychlostní a polohový servopohon. Každá z těchto regulačních struktur proto požaduje trochu odlišné řídící a stavové veličiny. Naše aplikace využívá jeden typ komunikačního telegramu, význam datových elementů telegramu závisí na zvoleném typu regulace (typu servopohonu).

Pohon může být nakonfigurován na jeden ze standardních telegramů (označují se PPO1 až PPO5). Vzájemně se liší počtem programovacích (PKW) a procesních (PZD) slov. PKW umožňují měnit a číst parametry (tím i „programovat" chování měniče, viz Kap. 3.2), PZD dokáží pouze přenášet data mezi PC a pohonem potřebná k jeho řízení. Námi zvolený typ telegramu PPO5 je nejuniverzálnější, má 10 procesních slov (PZD) typu *word*. Volba typu telegramu se provádí pomocí programu HW-Config.

Každý typ servopohonu vyžaduje různé modifikace obsahu řídícího telegramu jdoucího z řídícího PC do pohonu. Oproti tomu přijímaný, stavový telegram je pro všechny servopohony shodný. Stavovým telegramem (viz. Tab. 6.1) je přenášeno v prvních dvou slovech stavové slovo (nižší word *SW1* ve slově W1 a vyšší *SW2* ve slově W2). Další v řadě jsou skutečná hodnota rychlosti *n(act)*, jednotlivé složky proudů *Isq(act)*, *Isd(act)* a skutečná hodnota momentu *M(act)*. Skutečná hodnota polohy *s(act)* je rozdělena do slov W7 a W8, kde vyšší word je ve slově W7. Zprávy o chybách *Flt* popř. výstrahách *Wrn* přicházejí společně ve slově W9, kde vyšší byte reprezentuje číslo chyby a nižší číslo výstrahy. Na poslední pozici stavového telegramu je umístěno stavové slovo od SP *SW3*, které podrobněji definuje tabulka (Tab. 6.2). Čtvrtý sloupec popisuje propojení konektorů s parametry uvnitř měniče.

PZD	Název	Význam	Propojení
W 1	*SW1*	Stavové slovo SW1	P734.01 = K30
W 2	*SW2*	Stavové slovo SW2	P734.02 = K31
W 3	*n (act)*	Skutečná hodnota rychlosti	P734.03 = KK91
W 4	*Isq (act)*	Skutečná hodnota Isq	P734.04 = K184
W 5	*Isd (act)*	Skutečná hodnota Isd	P734.05 = K182
W 6	*M (act)*	Skutečná hodnota momentu	P734.06 = K241
W 7	*s (act)*	Skutečná hodnota polohy – High	P734.07 = KK120
W 8	*s (act)*	Skutečná hodnota polohy – Low	P734.08 = KK120
W 9	*Flt / Wrn*	Číslo chyby/ číslo varování	P734.09 = K250
W 10	*SW3*	Stavové slovo bloku Simple Positioner	P734.10 = K887

Tab. 6.1: Univerzální stavový komunikační telegram

Bit	Název signálu	Význam
0	*ENABLE POS/REF*	Povolení bloku SP
1	*REF_ON*	Aktivace režimu Homing
2	*POS_ON*	Aktivace režimu Positioning
3	*SETUP_ON*	Aktivace režimu Setup
4	*POS_TYPE*	Volba typu polohování (0 = absolutní, 1 = relativní)
5	*D_FWD*	Kladný směr pohybu (pro režim Setup a polohování rotační osy)
6	*D_BWD*	Záporný směr pohybu (pro režim Setup a polohování rotační osy)
7	*SPV_RIE*	Přenos vstupních dat do bloku SP (aktivní na náběžnou hranu)
8	*SPV_RIE_TYPE*	Typ přenosu vstupních dat do bloku SP (1 = trvalý přenos, 0 = přenos na náběžnou hranu)
9	*REF_TYPE*	Volba typu homingu (0 = letmý homing, 1 = Go to home position)

Tab. 6.2: Využívané bity řídícího slova Simple positioneru a jejich význam

V řídícím telegramu se jednotlivé položky liší s výjimkou řídícího slova, které je vždy rozloženo do prvních dvou *wordů* telegramu, a to na *CW1* a *CW2*. Z řídícího slova využíváme bity 0, 1, 2, 7 a 27, a proto jsou napojeny jen parametry příslušné těmto bitům. Jejich význam je popsán v tabulce (Tab. 6.3).

Bit		Význam
0	CW1	zapnutí (vypnutí) pohonu
1	CW1	volný doběh
2	CW1	rychlé zastavení
7	CW1	nulování poruchy (potvrzení chyby či výstrahy)
27	CW2	podřízený pohon (pro momentový servopohon)

Tab. 6.3: Využívané bity řídícího slova a jejich význam

V případě, že chceme řídit pohon z MASTER zařízení je nutné vždy nastavit bit č.10 řídícího slova na hodnotu logická „1“. V opačném případě by měnič neakceptoval přijímaná data.

Pro momentový servopohon jsou přenášeny tyto žádané hodnoty veličin. Ve třetím resp. čtvrtém slově kladné *M(lim+)* resp. záporné *M(lim-)* momentové omezení a ve slově W5 žádaná hodnota momentu *M(set)*. Řídící telegramy pro momentový servopohon jsou v tabulce (Tab. 6.4).

PZD	Název	Význam	Propojení
W 1	*CW1*	Řídící slovo CW1 (bit č. 10 = log. „1“) (viz Tab. 6.4)	P554 = B3100 P557 = B3101 P560 = B3102 P567 = B3107
W 2	*CW2*	Řídící slovo CW2 (viz Tab. 6.4)	P587 = B3211
W 3	*M (lim+)*	Momentové omezení kladné	P265 = K3003
W 4	*M (lim-)*	Momentové omezení záporné	P266 = K3004
W 5	*M (set)*	Žádaná hodnota momentu	P261 = K3005
W 6			
W 7			
W 8			
W 9			
W 10			

Tab. 6.4: Řídící komunikační telegram pro momentový servopohon

Pro rychlostní servopohon jsou přenášeny tyto žádané hodnoty veličin. Ve třetím resp. čtvrtém slově je opět kladné *M(lim+)* resp. záporné *M(lim-)* momentové omezení a ve slově W6 žádaná hodnota rychlosti *n(set)*. Řídící telegram pro rychlostní servopohon znázorňuje tabulka (Tab. 6.5).

PZD	Název	Význam	Propojení
W 1	*CW1*	Řídící slovo CW1	P554 = B3100 P557 = B3101 P560 = B3102 P567 = B3107
W 2	*CW2*	Řídící slovo CW2	P587 = B3211
W 3	*M (lim+)*	Momentové omezení kladné	P265 = K3003
W 4	*M (lim-)*	Momentové omezení záporné	P266 = K3004
W 5			
W 6	*n (set)*	Žádaná hodnota otáček	P443 = K3006
W 7			
W 8			
W 9			
W 10			

Tab. 6.5: Řídící komunikační telegram pro rychlostní servopohon

Pro polohový servopohon jsou přenášeny tyto žádané hodnoty veličin. Ve třetím resp. čtvrtém slově je opět kladné resp. záporné momentové omezení a slovo W5 představuje řídící slovo ze Simple positioneru *CW3*. Ve slově W6 je žádaná

hodnota rychlosti *n(set)*, v dalších slovech je zrychlení *a(set+)*, zpomalení *a(set-)*, a nakonec slova W9 a W10 představují opět váhově rozloženou žádanou hodnotu

PZD	Název	Význam	Propojení
W 1	*CW1*	Řídící slovo CW 1	P554 = B3100 P557 = B3101 P560 = B3102 P567 = B3107
W 2	*CW2*	Řídící slovo CW 2	P587 = B3211
W 3			
W 4			
W 5	*CW3*	Řídící slovo bloku Simple Positioner (viz Tab. 6.3)	U866.1 = B3500 U866.2 = B3501 U866.3 = B3502 U866.4 = B3503 U866.5 = B3504 U866.6 = B3505 U866.7 = B3506 U866.8 = B3507 U866.9 = B3508 U866.10 = B3509
W 6	*n (set)*	Žádaná hodnota rychlosti	U868 = K3006
W 7	*a (set+)*	Žádaná hodnota zrychlení	U869.1 = K3007
W 8	*a (set-)*	Žádaná hodnota zpomalení	U869.2 = K3008
W 9	*s (set)*	Žádaná hodnota polohy – High	U867 = KK3039
W 10	*s (set)*	Žádaná hodnota polohy – Low	

Tab. 6.6 Řídící komunikační telegram pro polohový servopohon

polohy *s(set)*. Strukturu řídícího telegramu pro polohový servopohon objasňuje tabulka (Tab. 6.6).

6.6 Datové elementy

Pod datovým elementem si lze představit proměnnou, konstantu či kanál. Každý datový element musí být předem deklarován, je nutné zadat jeho jméno, typ atd. Datové elementy v prostředí Control Web lze sdružovat do sekcí podle svého významu. Nejprve se zaměřím na sekce *proměnných* použité v naší aplikaci.

V první sekci *Control word* jsou jednotlivé bity řídícího slova a bit pro tlačítko *CENTRAL STOP*. Další sekce *Komunikace* obsahuje proměnné které reprezentují hodnoty posílané jednotlivými kanály s výjimkou *CW* a *SW*, pro něž je speciální sekce proměnných. V sekci *CW_SW* jsou jednotlivé *wordy* řídícího a stavového

slova, které jsou zapisovány resp. čteny z jednotlivých kanálů. V sekci diagnostika jsou proměnné pro *Multiswitch*, ve kterém se nastavují tyto binární proměnné a na základě jejich hodnoty se zobrazí příslušný diagnostický panel. Proměnné týkající se vzniklých či potvrzených chyb jsou v sekci *Chyby*. Proměnné nutné pro správné přepínání ovládacích panelů jsou umístěny v sekci *Ovládání*. Animace a zobrazení aktivních cest, *Meterů* či *Controlů* pro zvolený typ servopohonu v ovládacích panelech jsou zajištěny proměnnými v sekci *Typ servopohonu*. V sekci *Nastavení* jsou referenční hodnoty otáček, momentu, proudu a napětí potřebné pro přepočty žádané hodnoty proměnné v základních jednotkách na hodnotu, která bude posílána kanálem do měniče či naopak. V sekci *Data* jsou umístěny pomocné proměnné pro správný chod *Meterů* a *Controlů*, které zobrazují různá vyjádření veličin. V sekci *Control_word_SP* jsou jednotlivé bity řídícího slova pro polohový setpoint. Sekce *Upgrade* slouží k dočasnému zablokování některých funkcí aplikace, které zatím nebyly implementovány.

Dalšími datovými elementy jsou *konstanty*. V aplikaci je jedna sekce *konstant* a jsou v ní dekadická vyjádření rozsahu datového typu *word* a *doubleword*. V dalších sekcích vystupují *kanály* a *schedule*, které již byly zmíněny v kapitole (Kap. 6.5).

6.7 Funkce

Hlavním a důležitým základem pro správný chod aplikace jsou funkce. Jejich jednoduchost a univerzálnost musí být v rovnováze s požadavky na danou aplikaci. Proto je důležité tvořit funkce tak, aby zbytečně nezatěžovaly procesor.

V prostředí Control Web může být funkce implementována pouze v přístrojích. Prostředí nám nabízí dva typy přístrojů, viditelné a neviditelné. Pokud potřebujeme funkci, která bude volána vícekrát nezávisle na sobě, implementujeme ji do neviditelného přístroje jménem *Program*. Danou funkci můžeme kdykoliv zavolat pomocí nativního vyjádření *Program.název_funkce(parametr)* z libovolné událostní procedury viditelného přístroje. Pokud nám však stačí použít funkci jen lokálně, stačí ji implementovat přímo ve viditelném přístroji.

Program poskytuje mnoho nadefinovaných událostních procedur lišících se dle použitého přístroje. Umožňuje nám také vytvořit si procedury vlastní, podle našich potřeb. Z nich bych v první řadě zmínil funkce umístěné v neviditelném bloku *Program*. V prvním neviditelném přístroji *Setup*, tohoto bloku, je implementována

funkce která se spouští pouze při startu aplikace a provádí inicializaci některých proměnných (viz. Příloha B, odst. 1).

V dalším přístroji jménem *Program* jsou níže popsané procedury. Tyto procedury umožňují přívětivější zadávání žádaných hodnot, které mají rozměr v procentech nebo v konkrétních fyzikálních jednotkách (viz. závěr Kap. 6.3) a přepočítávají vstupní data na celočíselné dekadické vyjádření (v rozsahu 16 nebo 32 bitů). Procedury jsou umístěny v bloku *Program*, protože jsou využívány v mnoha přístrojích. Díky tomu není nutné je programovat v každém přístroji zvlášť, tyto procedury jsou danými přístroji pouze volány. Zde jsou tyto procedury vypsány jen pomocí matematických výrazů, zdrojový kód se nachází v příloze B, odst. 2.

První z nich je procedura sloužící pro převod z dekadické hodnoty typu *word* či *doubleword* na procenta a zpět.

```
procedure K2P( K : cardinal ): real;   (Word na procenta )
```

$K \in \langle 0;2*256) \Rightarrow K2P = \frac{K}{256}.100$

$K = 2*256 \Rightarrow K2P = -\frac{K}{256}.100$

$K \in \langle 2*256;4*256) \Rightarrow K2P = \left(\left(\frac{K}{2*256}.100\right) - 200\right)*2$

```
procedure KK2P( KK : longcard ): real;   (DWORD na procenta )
```

$KK \in \langle 0;2*65536) \Rightarrow KK2P = \frac{KK}{65536}.100$

$KK = 2*65536 \Rightarrow KK2P = -\frac{KK}{65536}.100$

$KK \in \langle 2*65536;4*65536) \Rightarrow KK2P = \left(\left(\frac{KK}{2*65536}.100\right) - 200\right)*2$

```
procedure P2K( P : real ): longint;   (Procenta na WORD )
```

$P \in \langle 0;200) \Rightarrow P2K = round\left(\frac{P*256}{100}\right)$

$P = 200 \Rightarrow P2K = round\left(\frac{P*256}{100}\right)$

$P \in \langle -200;0) \Rightarrow P2K = round\left(\left(\frac{\frac{P}{2}+200}{100}\right)*2*256\right)$

procedure P2KK(P : real): longint; (Procenta na DWORD)

P ∈ <0;200) => $P2KK = round\left(\frac{P*65536}{100}\right)$

P = 200 => $P2KK = round\left(\frac{P*65536}{100}\right)$

P ∈ <-200;0) => $P2KK = round\left(\left(\frac{\frac{P}{2}+200}{100}\right)*2*65536\right)$

Následující procedura slouží pro převod hodnoty signálu z procent na fyzikální veličinu a zpět.

procedure P2Value(P, RefValue : real): real; (% na Fyz. vel.)

$$P2Value = \frac{P*RefValue}{100}$$

procedure Value2P(P, RefValue : real): real; (Fyz. vel. na %)

$$Value2P = \frac{P*100}{RefValue}$$

A nakonec procedura sloužící k získání nižšího nebo vyššího *bytu* z proměnné typu *word* resp. nižšího či vyššího *wordu* z proměnné typu *doubleword*.

procedure KHigh(K : cardinal): real; (WORD na HighBYTE)

$$KHigh = \frac{65280 \text{ and } K}{256}$$

procedure KLow(K : cardinal): real; (WORD na LowBYTE)

$$KLow = 255 \text{ and } K$$

procedure KKHigh(KK : longcard): real; (DWORD na HighWORD)

KK >= 0 => $KKHigh = \frac{4294901760 \text{ and } KK}{65536}$

KK < 0 => $KKHigh = -\frac{4294901760 \text{ and } KK}{65536}$

procedure KKLow(KK : longcard): real; (DWORD na LowWORD)

$$KKLow = 65535 \text{ and } KK$$

V jednotlivých viditelných přístrojích jsou implementovány níže uvedené typy funkcí. Jsou to například bitové operace, přepínací funkce mezi panely, funkce pro diagnostiku chyb, atd...

Funkce pro nastavení resp. nulování jednoho bitu čísla:

`bitor(s1,s2:longcard):longcard` resp. `bitand(s1,s2:longcard):longcard`
Například pro nastavení bitu 0 wordu 1 řídícího slova je použita funkce:

```
sek_v_CW_SW.v_CW1 := bitor(1, sek_v_CW_SW.v_CW1).
```

Pro nulování bitu 0 wordu 1 řídícího slova je použita funkce:

```
sek_v_CW_SW.v_CW1 := bitand(65531, sek_v_CW_SW.v_CW1).
```

Podrobněji o těchto funkcích v příloze B, odst. 6.

Použitím zmíněných funkcí je zabráněno změně ostatních bitů řídícího slova při nastavení či nulování určitého bitu tohoto slova. Funkce je využita například u tlačítek *Start pohonu*, *Volný doběh* a *Rychlé zastavení*.

Další procedura je implementována v panelu *Ovládání* (viz Příloha B, odst. 7). Jejím vstupem jsou proměnné reprezentující jednotlivé panely v sekci panelů *Ovládání*. Proměnné jsou nastavovány stiskem tlačítka se symbolem ''>>'', který aktivuje přepínač ovládacích panelů. Ten přepne na požadovanou záložku prezentovanou danou proměnnou pomocí nativní procedury *SetSelIndex (Index: longcard)*, umístěné v událostní proceduře *OnActivate*. Index definuje pořadové číslo záložky.

Zajímavá funkce je realizována v tabulce *Historie chyb* (viz. Příloha B, odst. 8). Při výskytu definované chyby se do tabulky vypíše do řádku její číslo, popis, čas vzniku a boolovská hodnota, která říká zda chyba již byla potvrzena či ne. V případě potvrzení se změní barva celého řádku z původní červené na zelenou a přidá se čas potvrzení. V případě příchodu další chyby se posune řádek a vše se opakuje. Řádek je vkládán pomocí nativní procedury *InsertRow(Row:longcard)* v událostní proceduře *OnActivate*. V tabulce je tedy možné sledovat časový běh chybových hlášení či výstrah.

7 Postup při uvedení do provozu

V této kapitole si popíšeme pár základních kroků, které je nutno provést, aby celá aplikace správně fungovala. Postup pro uvedení do provozu se skládá ze dvou hlavních částí. První z nich je prvotní uvedení do provozu, při kterém se fyzicky propojí stanoviště a nainstaluje se nezbytný software. V druhé části, opakovaném uvedení do provozu, se již jedná o kontrolu propojení a nastavení konkrétních programů. Tyto dvě části jsou podrobně popsány níže:

Prvotní uvedení do provozu:

1. Fyzické propojení mezi Server PC a pohonem Motion Control sběrnicí Profibus. Nastavení terminačních odporů na koncích této sběrnice.
2. Fyzické propojení mezi Client PC a Server PC sítí Ethernet.
3. Instalace softwaru DriveMonitor na libovolný PC.
4. Instalace softwaru Step 7 na Server PC.
5. Instalace softwarového balíku SimaticNET na Server PC a následná instalace OPC serveru.
6. Instalace testovacího OPC klienta na Server i Client PC.
7. Instalace programu Control Web na Client PC.

Opakované uvedení do provozu:

1. Kontrola fyzického propojení mezi Server PC a pohonem MC.
2. Kontrola fyzického propojení mezi Client PC a Server PC.
3. Nahrání parametrické sady, pro zvolený typ servopohonu, do měniče.

   Parametrické sady jsou v následujících souborech:

   Momentový servopohon: ***BP_MC_torque.dnl***

   Rychlostní servopohon: ***BP_MC_speed.dnl***

   Polohový servopohon: ***BP_MC_position.dnl***

   Adresář na CD: ***Parametrické sady***

   Použití: Nahrání pomocí programu DriveMonitor. Uvedené soubory obsahují kompletní parametrickou sadu (tj. hodnoty všech parametrů měniče). K dispozici jsou i skriptové soubory, jejichž spuštěním v programu Drive

Monitor se změní nastavení pouze těch parametrů, které jsou nutné pro komunikaci prostřednictvím Profibus-DP podle zadaných obsahů telegramů (viz. Kap. 6.5).

4. Nahrání konfigurace do OPC serveru pomocí programu Step 7,

   Soubor archivu: ***Stastn_1.zip***

   Adresář na CD: ***Konfigurace HW a sítě***

   Použití: Soubory projektu se extrahují ze ZIP archivu v prostředí Step 7 (nástroj Simatic Manager) pomocí volby File/Retrieve... Hardwarová konfigurace počítače s OPC serverem se nahrává pomocí nástroje HWConfig, struktura sítě a komunikačních sběrnic se nahrává programem NetPro.

5. Ověření funkčnosti OPC serveru pomocí testovacího OPC klienta. V případě, že komunikace se vzdáleným PC nefunguje, může být problém buď v nastavení firewallu OS Windows XP nebo v nastavení DCOM modelu na obou PC. Příslušné podklady pro odstranění těchto problémů naleznete v [3] nebo [9] popř. v [10].

6. Spuštění souboru vizualizace v programu Control Web 5,

   Soubor: ***Rizeni a vizualizace pohonu.cw***

   Adresář na CD: ***Řízení a vizualizace pohonu v prostředí Control Web***

   Použití: Otevřeme soubor *Rizeni a vizualizace pohonu.cw* v programu Control Web a můžeme hned začít pracovat s řízením a vizualizací pohonu.

8 Závěr

Cílem bakalářské práce bylo navrhnout program, pro řízení a vizualizaci servopohonu, v programovém prostředí Control Web 5. Ke komunikaci mezi operátorským počítačem (Client PC) a pohonem byla použita OPC technologie v součinnosti s průmyslovou komunikační sběrnicí Profibus a sběrnicí Ethernet. Aplikace má sloužit k výukovým účelům a proto má být co nejnázornější a nejpřehlednější.

Servopohon tvoří synchronní servomotor s permanentními magnety, typového označení 1FT6, který je řízen frekvenčním měničem Siemens Masterdrives Motion Control. Nadřazený řídící systém tvoří PC s OPC serverem (Server PC), který pomocí vizualizační aplikace tento pohon řídí. Vytvořená aplikace může být spuštěna na Server PC, nebo i vzdáleně na Client PC. Ke komunikaci mezi Server PC a pohonem byla využita průmyslová sběrnice Profibus, komunikace mezi Server PC a Client PC je realizována standardní sítí Ethernet.

K tomu aby bylo možné tuto aplikaci sestavit bylo nutné získaní teoretických znalostí o pohonech obecně a konkrétních informací o námi využívaných zařízeních. Dalším krokem bylo seznámení se základními programovacími zásadami pro tvorbu aplikací v prostředí Control Web 5. Toto prostředí je určeno pro návrh vizualizačních aplikací pro různá technická odvětví. Programátor kompletuje aplikaci tvořenou přístroji, které jsou vzájemně provázány funkcemi a procedurami. Lze tak vytvořit jakoukoli vizualizační a řídící aplikaci, komunikující s vnějším prostředím pomocí komunikačních kanálů.

Vytvořený program umožňuje základní řízení ve třech regulačních úrovních, momentové, rychlostní a polohové. Pro každý typ regulace bylo nutné vhodně navrhnout strukturu komunikačních telegramů pro přenášená řídící a stavová slova. Vizualizační aplikace je uspořádána do přehledných panelů, což umožňuje uživateli zcela intuitivní ovládání a demonstruje tak základních vlastností pohonu, jeho režimy a vnitřní struktury.

Aplikace splnila svým rozsahem všechny požadavky zadání. V průběhu její tvorby se ukázaly podněty pro případné rozšíření a zároveň zdokonalení. Vhodné by bylo rozšíření aplikace o možnost programování měniče nastavováním jeho parametrů přímo z aplikace a možnost exportu dat z grafů.

Doufám že tato práce bude přínosem ke zkvalitnění výuky, a myslím si, že je zajímavou ukázkou toho jak lze v praxi využít vizualizační software pro řízení servopohonu. Nyní nezbývá něž Vám popřát mnoho úspěchů při používání vizualizačního programu a mnoho zajímavých poznatků získaných při čtení tohoto textu.

Použitá literatura:

[1] J. Pavelka, Z. Čeřovský, J. Javůrek: Elektrické pohony, skripta ČVUT, 2001, ISBN 80-01-02314-1

[2] Dokumentace Control Web 5, soubor elektronické nápovědy

[3] OPC Overview Version 1.0, October 27, 1998, OPC Foundation

[4] OPC Common Definition and Interfaces Version 1.0, October 27, 1998, OPC Foundation

[5] W. Moller-Nehring, W. Bohrer, Universal Serial Interface Protocol, Specification, Siemens A.G., 1994

[6] Simovert MASTERDRIVES MotionControl, Návod k obsluze a údržbě, Siemens A.G., 1997

[7] Simovert MASTERDRIVES MotionControl, Compendium, Siemens A.G., 1999 (viz přiložené CD)

[8] Control Web 2000, Průvodce systémem pro tvorbu a nasazení aplikací reálného času

[9] K. H. Deiretsbacher, J. Luth, R. Mody: Using OPC via DCOM with Microsoft Windows XP SP 2, OPC Foundation, 2004, (viz přiložené CD).

[10] www.opcfoundation.com

[11] Webové stránky časopisu Automa, www.automa.cz

[12] Webové stránky Vysoké školy VŠB – Technická univerzita Ostrava, Fakulta elektrotechniky a informatiky www.fei.vsb.cz

Příloha A - Úplný výpis parametrického a mapovacího souboru

Parametrický soubor *OPC_01.par* :

```
begin

   1 longint bidirectional
   2 longint bidirectional
   3 longint bidirectional
   4 longint bidirectional
   5 longint bidirectional
   6 longint bidirectional
   7 longint bidirectional
   8 longint bidirectional
   9 longint bidirectional
  10 longint bidirectional
  11 longint bidirectional
  12 longint bidirectional
  13 longint bidirectional
  14 longint bidirectional
  15 longint bidirectional
  16 longint bidirectional
  17 longint bidirectional
  18 longint bidirectional
  19 longint bidirectional
  20 longint bidirectional

end.
```

Mapovací soubor *OPC_01.dmf* :

```
[comment]
OPCDRV PAR soubor vytvořen konfigurátorem OPC ovladače

[server]
CLSID={B6EACB30-42D5-11D0-9517-0020AFAA4B3C}
sync=true
wait_running=true
host=\\PRI

[channels]
1 = DP:[CP 5611]SLAVE003_IW8,1
2 = DP:[CP 5611]SLAVE003_IW10,1
3 = DP:[CP 5611]SLAVE003_IW12,1
4 = DP:[CP 5611]SLAVE003_IW14,1
5 = DP:[CP 5611]SLAVE003_IW16,1
6 = DP:[CP 5611]SLAVE003_IW18,1
7 = DP:[CP 5611]SLAVE003_IW20,1
8 = DP:[CP 5611]SLAVE003_IW22,1
9 = DP:[CP 5611]SLAVE003_IW24,1
10 = DP:[CP 5611]SLAVE003_IW26,1

11 = DP:[CP 5611]SLAVE003_QW8,1
12 = DP:[CP 5611]SLAVE003_QW10,1
13 = DP:[CP 5611]SLAVE003_QW12,1
14 = DP:[CP 5611]SLAVE003_QW14,1
15 = DP:[CP 5611]SLAVE003_QW16,1
16 = DP:[CP 5611]SLAVE003_QW18,1
17 = DP:[CP 5611]SLAVE003_QW20,1
18 = DP:[CP 5611]SLAVE003_QW22,1
19 = DP:[CP 5611]SLAVE003_QW24,1
20 = DP:[CP 5611]SLAVE003_QW26,1
```

Příloha B - Výpis procedur a některých částí programu

Procedura implementovaná v programovém bloku *Setup*, slouží k inicializaci referenčních proměnných a některých bitů řídícího slova:

```
timer = infinite;
%časovač nastaven tak, aby se provedla aktivace
 přístroje a následně i procedura pouze jednou.

procedure OnActivate();
begin
 sek_v_Nastavení.v_Rychlost_max:=3000;
 sek_v_Nastavení.v_Moment_max:=4.7;
 sek_v_Nastavení.v_Proud_max:=3.6;
 sek_v_Nastavení.v_Napětí_max:=400;
 sek_v_CW_SW.v_CW1:=bitor(2,sek_v_CW_SW.v_CW1);
 sek_v_Control_Word.v_cw_bit1:=false;
 sek_v_CW_SW.v_CW1:=bitor(4,sek_v_CW_SW.v_CW1);
 sek_v_Control_Word.v_cw_bit2:=false;
 sek_v_CW_SW.v_CW1:=bitor(1024,sek_v_CW_SW.v_CW1);
 sek_v_Control_Word.v_cw_bit10:=true;
 send table_2;
 send rychlost_max;
 send moment_max;
 send proud_max;
 send napětí_max;
 send button_Volný_doběh;
 send button_Rychlé_zastavení;
 send switch_cw1;
 send switch_cw2;
 send switch_cw10;
end_procedure;
```

Procedury implementované v programovém bloku *Program*, jsou tu uvedeny převodní funkce:

```
procedure K2P( K : cardinal ): real;(*Word na procenta*)
begin
if (0 <= K) and (K < (2*sek_c_Nastavení.c_K))
 then return (K/sek_c_Nastavení.c_K*100);
elsif (K = 2*sek_c_Nastavení.c_K)
 then return -(K/sek_c_Nastavení.c_K*100);
elsif (K > 2*sek_c_Nastavení.c_K) and (K <
       4*sek_c_Nastavení.c_K)
 then return
      ((((K)/(2*sek_c_Nastavení.c_K))*100)-200)*2);
end;
end_procedure;

procedure KK2P( KK : longcard ): real;(*DWORD na procenta*)
begin
if (0 <= KK) and (KK < (2*sek_c_Nastavení.c_KK))
 then return (KK/sek_c_Nastavení.c_KK*100);
elsif (KK = 2*sek_c_Nastavení.c_KK)
 then return -(KK/sek_c_Nastavení.c_KK*100);
elsif (KK > 2*sek_c_Nastavení.c_KK) and (KK <
      4*sek_c_Nastavení.c_KK)
 then return
      ((((KK)/(2*sek_c_Nastavení.c_KK))*100)-200)*2);
end;
end_procedure;

procedure P2K( P : real ): longint;(*Procenta na WORD*)
begin
if (0<=P) and (P<200)
 then  return round (P*sek_c_Nastavení.c_K/100);
elsif P=200
 then  return round (P*sek_c_Nastavení.c_K/100);
elsif (-200<=P) and (P<0)
 then  return round
       (((P/2)+200)/100)*2*sek_c_Nastavení.c_K);
end;
end_procedure;

procedure P2KK( P : real ): longint;(*Procenta na DWORD*)
begin
if (0<=P) and (P<200)
 then  return round (P*sek_c_Nastavení.c_KK/100);
elsif P=200
 then  return round (P*sek_c_Nastavení.c_KK/100);
elsif (-200<=P) and (P<0)
 then return round
      ((((P/2)+200)/100)*2*sek_c_Nastavení.c_KK);
end;
end procedure;
```

```
procedure P2Value( P, RefValue : real ): real;
(*Procenta na Fyz. veličinu*)
begin
 return P*RefValue/100;
end_procedure;

procedure Value2P( P, RefValue : real ): real;
(*Fyz. Veličina na Procenta*)
begin
 return P*100/RefValue;
end_procedure;

procedure KHigh( K : cardinal ): real;
(*WORD na HighBYTE *)
begin
 return (bitand(65280,K))/256;
end_procedure;

procedure KLow( K : cardinal ): real;
(*WORD na LowBYTE *)
begin
 return bitand(255,K);
end_procedure;

procedure KKHigh( KK : longcard ): real;
(*DWORD na HighWORD *)
begin
if KK >= 0
 then return (bitand(4294901760,KK))/65536
else return -(bitand(4294901760,KK))/65536;
end;
end_procedure;

procedure KKLow( KK : longcard ): real;
(*DWORD na LowWORD *)
begin
 return bitand(65535,KK);
end_procedure;
```

Níže je uvedena definice vstupních komunikačních kanálů, je zde definován datový typ, ovladač, index kanálu a směr komunikace:

```
channel sek_ch_In {comment = 'Procesní data z pohonu'};
ch_In_1:longint {driver=OPC_01; driver_index = 1;
direction = bidirectional};
ch_In_2:longint {driver = OPC_01; driver_index = 2;
direction = bidirectional};
ch_In_3:longint {driver = OPC_01; driver_index = 3;
direction = bidirectional};
ch_In_4:longint {driver = OPC_01; driver_index = 4;
direction = bidirectional};
ch_In_5:longint {driver = OPC_01; driver_index = 5;
direction = bidirectional};
ch_In_6:longint {driver = OPC_01;driver_index = 6;
direction = bidirectional};
ch_In_7:longint {driver = OPC_01;driver_index = 7;
direction = bidirectional};
ch_In_8:longint {driver = OPC_01;driver_index = 8;
direction = bidirectional};
ch_In_9:longint {driver = OPC_01;driver_index = 9;
direction = bidirectional};
ch_In_10:longint {driver = OPC_01;driver_index = 10;
direction=bidirectional};
end_channel;
```

```
channel sek_ch_Out {comment = 'Procesní data vstupující do
pohonu'};
ch_In_1:longint {driver=OPC_01; driver_index = 11;
direction = bidirectional};
ch_In_2:longint {driver = OPC_01; driver_index = 12;
direction = bidirectional};
ch_In_3:longint {driver = OPC_01; driver_index = 13;
direction = bidirectional};
ch_In_4:longint {driver = OPC_01; driver_index = 14;
direction = bidirectional};
ch_In_5:longint {driver = OPC_01; driver_index = 15;
direction = bidirectional};
ch_In_6:longint {driver = OPC_01;driver_index = 16;
direction = bidirectional};
ch_In_7:longint {driver = OPC_01;driver_index = 17;
direction = bidirectional};
ch_In_8:longint {driver = OPC_01;driver_index = 18;
direction = bidirectional};
ch_In_9:longint {driver = OPC_01;driver_index = 19;
direction = bidirectional};
ch_In_10:longint {driver = OPC_01;driver_index = 20;
direction=bidirectional};
end_channel;
```

Následující blok programu realizuje propojení plánovaných datových elementů *schedule* s příslušnými *kanály*. Perioda čtení je nastavena na 1 sekundu.

```
schedule sek_sch_In {timer = 1; condition = true};
    sch_In_1 = sek_ch_In.ch_In_1;
    sch_In_2 = sek_ch_In.ch_In_2;
    sch_In_3 = sek_ch_In.ch_In_3;
    sch_In_4 = sek_ch_In.ch_In_4;
    sch_In_5 = sek_ch_In.ch_In_5;
    sch_In_6 = sek_ch_In.ch_In_6;
    sch_In_7 = sek_ch_In.ch_In_7;
    sch_In_8 = sek_ch_In.ch_In_8;
    sch_In_9 = sek_ch_In.ch_In_9;
    sch_In_10 = sek_ch_In.ch_In_10;
```

Příklad použití procedury *OnScalarNumericalDataChange*, využívané plánovanými datovými elementy *schedule*.

```
procedure OnScalarNumericalDataChange(var DataElement:real );
begin
  sek_v_CW_SW.v_SW1 = sch_In_1;
  % napojení proměnné na schedule
    send sw_0; %aktivace přístrojů
    send sw_1;        .
    send sw_2;        .
    send sw_3;        .
    send sw_4;
    send sw_5;
    send sw_6;
    send sw_7;
    send sw_8;
    send sw_9;
    send sw_10;
    send sw_11;
    send sw_12;
    send sw_13;
    send sw_14;
    send sw_15;
    send button_pohon_ON_OFF;
    send sig_pohon_v_chodu;
    send sig_motor_zastaven;
    send sig_porucha;
    send sig_výstraha;
    send left_1;
    send stop_1;
    send right_1;
          ........
```

Procedura *OnOutput* implementovaná v tlačítku *Start pohonu*. Je zde provedeno testování zda je tlačítko sepnuté a zároveň je-li tlačítko *CENTRAL STOP* v poloze FALSE. V případě splnění podmínky se provede nastavení bitu pro start pohonu obsaženého v řídícím slově. Dále je vykonáno kopírování nového řídícího slova do kanálu a aktivace příslušných přístrojů.

```
procedure OnOutput( Output : boolean );
begin
if Output and not sek_v_Control_Word.v_central_stop
 then sek_v_CW_SW.v_CW1:=bitor(1,sek_v_CW_SW.v_CW1)
elsif not Output and not sek_v_Control_Word.v_central_stop
 then sek_v_CW_SW.v_CW1:=bitand(65534,sek_v_CW_SW.v_CW1);
end;
sek_ch_Out.ch_Out_1 := sek_v_CW_SW.v_CW1;
send string_display_12;
send multi_switch_1;
end_procedure;
```

Zobrazování panelů v sekci *Ovládání* zajišťuje následující procedura umístěná v přístroji *Multiswitch - Přepínač_panelů_3*. Ten dle logické proměnné, která je výstupem *(output =)* tlačítka pro přepínání mezi panely („>>"), provede zobrazení příslušného panelu pomocí nativní procedury *SetSelIndex(1)*.

```
switch_Isq_set_1
output = sek_v_Ovládání.v_Panel_Proudová_regulace;
logic = set_true_on_press;
receivers = Přepínač_panelů_3;

Přepínač_panelů_3
procedure OnActivate();
begin
if v_Panel_Vstup
 then Přepínač_panelů_3.SetSelIndex(1);
elsif v_Panel_Setpoint_processing
 then Přepínač_panelů_3.SetSelIndex(2);
elsif v_Panel_Polohová_regulace
 then Přepínač_panelů_3.SetSelIndex(3);
elsif v_RFG
 then Přepínač_panelů_3.SetSelIndex(4);
elsif v_Panel_Rychlostní_regulace
 then Přepínač_panelů_3.SetSelIndex(5);
elsif v_Panel_Proudová_regulace
 then Přepínač_panelů_3.SetSelIndex(6);
elsif v_Panel_Motor
 then Přepínač_panelů_3.SetSelIndex(7);
end;
end_procedure;
```

Následující procedura popisuje chování tabulky *Historie chyb při její aktivaci*, je zde vidět testování *bitu* stavového slova, který říká zda nastala chyba nebo porucha či nikoliv. A zároveň je testována proměnná, která představuje číslo vzniklé chyby. V případě vzniku jakékoliv chyby či poruchy se vkládá nový řádek do tabulky... Zde je uvedena procedura provádějící testování a výpis definované chyby č. 100, 150, či 200 do tabulky:

```
procedure OnActivate();
begin
if ((sek_v_Chyby.v_ch_H = 100) or
    (sek_v_Chyby.v_ch_H = 150) or
    (sek_v_Chyby.v_ch_H = 200)) and
    (bitget(sek_v_CW_SW.v_SW1,3) = 1)
% bitget je funkce pro získání určitého bitu dekadického
  čísla
 then table_1.InsertRow(1);        % vložení řádku
if (sek_v_Chyby.v_ch_H = 100)
 then sek_v_Chyby.v_ch_Text := 'chyba 100';
elsif (sek_v_Chyby.v_ch_H = 150)
 then sek_v_Chyby.v_ch_Text := 'chyba 150';
elsif (sek_v_Chyby.v_ch_H = 200)
 then sek_v_Chyby.v_ch_Text := 'chyba 200';
 end;
sek_v_Chyby.v_ch_H1:= sek_v_Chyby.v_ch_H;
 SetRealValue(0,1,1,sek_v_Chyby.v_ch_H,false,false);
 SetStringValue(0,1,2,sek_v_Chyby.v_ch_Text,false,false);
 SetStringValue(0,1,3,date.NowToString(),false,false);
 SetTextColor(0,1,1,1,5,200,0,0);
 SetBooleanValue(0,1,4,false,false,false);
end;
```

Příloha C - Obsah doprovodného CD

V této příloze je stručně naznačena struktura adresářů a souborů uložených na přiloženém CD.

Adresář	Soubor(y)
	Read me.doc - základní informace o CD
Doprovodný text BP	*Desky a titulní stránka.pdf* *Hlavní text.pdf*
Řízení a vizualizace pohonu v prostředí Control Web 5	*Rizeni a vizualizace pohonu.zip*
DriveMonitor - parametrické sady	*BP_MC_torque.dnl* (Momentový servopohon) *BP_MC_speed.dnl* (Rychlostní servopohon) *BP_MC_position.dnl* (Polohový servopohon)
Step 7 - konfigurace HW a sítě	*Stastn_1.zip*
Instalace OPC Matrikon	*opc-simulation-server-manual.pdf* *matrikon_opc_explorer_sim_server.exe*
Simovert MASTERDRIVES MotionControl, Compendium	*mc16_kompend_e.pdf*
DCOM	*Using OPC via DCOM with Microsoft Windows XP SP 2.pdf*