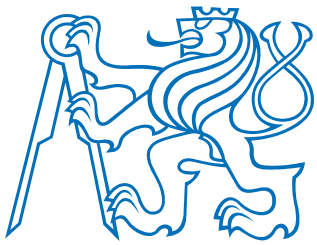
České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra řídicí techniky



Diplomová práce

Synchronizace servopohonů

Praha, 2005 Jan Pšenička

České vysoké učení technické v Praze - Fakulta elektrotechnická

Katedra řídicí techniky *Školní rok:* 2003/2004

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Jan Pšenička

Obor: Technická kybernetika

Název tématu: Synchronizace servopohonů

Zásady pro vypracování:

1. Navrhněte mechanický model s několika servopohony, který by demonstroval jejich součinnost a synchronizaci. Model realizujte.
2. Navrhněte a realizujte řízení tohoto modelu pomocí průmyslových automatů s komunikací Profibus DP-V2.
3. Realizujte vzdálený přístup k modelu přes Internet.

Seznam odborné literatury: Dodá vedoucí práce.

Vedoucí diplomové práce: Ing. Pavel Burget

Datum zadání diplomové práce: prosinec 2003

Termín odevzdání diplomové práce: květen 2005

L.S.

doc. Ing. Michael Šebek, DrSc.
vedoucí katedry

prof. Ing. Vladimír Kučera, DrSc.
děkan

V Praze dne 25.03.2004

PROHLÁŠENÍ

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

Nemám závažný 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 .....……. .....……

Jan Pšenička

PODĚKOVÁNÍ

Na tomto místě bych rád poděkoval všem, kteří mi při vytváření této práce pomáhali nalézt správnou cestu. Především vedoucímu diplomové práce Ing. Pavlu Burgetovi, bez jehož pomoci, rad a odborných připomínek by tato práce nemohla vzniknout.

Také bych chtěl poděkovat svým rodičům a přátelům za podporu, které se mi od nich dostávalo po celou dobu studia.

ABSTRAKT

Diplomová práce se zabývá návrhem a realizací mechanického modelu se čtyřmi synchronními motory, na nichž je vidět jejich vzájemná součinnost a synchronizace. Ovládání každého motoru zajišťuje servozesilovač (ACOPOS), který je řízen programovatelným logickým automatem (Power Panel PP220) s využitím moderní komunikace v reálném čase Ethernet Powerlink. Vizualizace řízeného modelu je realizována pomocí dotykové obrazovky Power Panelu a webových stránek (vzdáleného přístupu přes internet).

ABSTRACT

The main goal of this work is to design and realize a mechanical model controlled by synchronous motors. We show their synchronization and cooperation. Controlling of each motor is provided by servo drive (ACOPOS) that is controlled by programmable logical controller - PLC (Power Panel PP220) via modern real-time communication Ethernet Powerlink. The touch screen of the PLC and the web pages are used to control the model and to visualize the actual data.

OBSAH

1. ÚVOD

V současné době dochází k velkému rozvoji moderních technologií, počítačových i průmyslových sítí a jejich propojení s internetem. Rozvíjí se také oblast monitorování a řízení.

Cílem této diplomové práce je navržení a vytvoření mechanického modelu se čtyřmi synchronními motory, na nichž bude vidět jejich vzájemná součinnost a synchronizace. Vlastní pohyb každého motoru zajišťuje servozesilovač (ACOPOS) pomocí pulsně šířkové modulace napájecího napětí řízeného motoru. Servozesilovače jsou řízeny programovatelným logickým automatem (Power Panel PP220) s využitím moderní komunikace v reálném čase Ethernet Powerlink s dobou komunikačního cyklu 400 µs. Všechny použité řídicí komponenty jsou od firmy B+R automatizace Brno.

Dalším cílem je naprogramování ukázkové řídicí a vizualizační aplikace. K softwarovému polohovému řízení motorů jsou použity dvě metody. První spočívá v řízení servozesilovače z PLC pomocí speciálních příkazů tzv. akcí. Tyto akce slouží k inicializaci servozesilovače, nastavení parametrů regulátoru, jeho zapínání, nastavení mezních hodnot, ovládání motorů pomocí základních pohybů atd. Druhá metoda spočívá v řízení motoru pomocí vačkového automatu a vačkových profilů. Vačkový automat i profily se zpracovávají v servozesilovači. Vačkový profil je realizován pomocí trajektorie polohy, rychlosti a zrychlení v závislosti na čase.

Vizualizace je realizována pomocí dotykové obrazovky, která je součástí Power Panelu PP220 a pomocí webových stránek, které běží na webovém serveru Power Panelu.

Navržený model bude v budoucnu využíván studenty na cvičení z předmětu Řídicí systémy. Studenti se seznámí s možnostmi polohového řízení motorů. Dále se naučí v prostředí Automation Studio naprogramovat řídicí aplikaci pro PLC i servozesilovače a vytvořit vizualizaci pro operátorský panel.

Každá kapitola této práce se zabývá určitou problematikou související s realizací modelu, jeho řízením a vizualizací.

Druhá kapitola popisuje důležité součásti sady prostředků pro vývoj řídicích systémů firmy B+R automatizace, v jejíž terminologii je tato sada označována jako automatizační software. Těmi jsou vývojové prostředí pro aplikace (Automation Studio), komunikace mezi řídicími systémy a jednotlivými zařízeními (Automation Net), operační systém řídicích systémů (Automation Runtime) a jednotlivé řídicí systémy (Automation Targets).

Ve třetí kapitole se zabývám strukturou polohového řízení motorů pomocí servozesilovače (ACOPOS), který je ovládán pomocí PLC. Dále zde popisuji vlastnosti a způsob řízení servozesilovače a také komunikaci Ethernet Powerlink.

Čtvrtá kapitola se zabývá možnostmi softwarové realizace samotného polohového řízení pomocí speciálních řídicích příkazů tzv. akcí (nc-actions), vačkového automatu (CAM profile automat), který používá vačkové profily (CAM profile) nebo pomocí funkcí knihovny PLCopen.

V páté kapitole popisuji realizovaný model, jeho funkci, zapojení a jednotlivé komponenty.

Šestá kapitola popisuje nejdůležitější kroky při tvorbě aplikace (vytvoření projektu, hardwarovou konfiguraci, nastavení komunikace a další) a nastavení nezbytných parametrů. Dále popisuje strukturu řídicího programu, jeho jednotlivé součásti, algoritmy a softwarové řízení modelu.

Sedmá kapitola se zabývá vizualizací řízeného modelu pomocí operátorského panelu, vzdáleného přístupu z webových stránek a VNC (Virtual Network Computing).

Poslední osmá kapitola obsahuje zhodnocení výsledků diplomové práce.

2. AUTOMATIZAČNÍ SOFTWARE

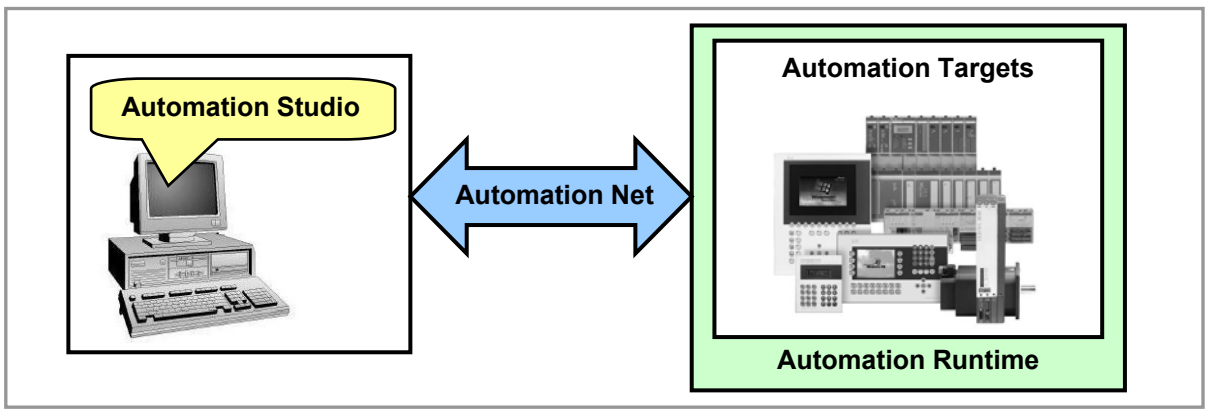
Vytvářené automatizační úkoly požadují vhodný programovací nástroj, který umožňuje v co nejkratším čase vytvořit rostoucí počet automatizačních projektů. Automatizační projekty často začínají velmi jednoduchým řídicím úkolem a později se vyvíjejí do komplexnějších, které umožňují více úkonů (vizualizace, polohové řízení, regulace, komunikace po více rozhraních a další). V případě zařízení B+R jsou všechny vytvořené součásti projektu kompatibilní pro různé typy řídicích procesorů, protože procesory obsahují kompatibilní operační systém. Toto je zřetelná výhoda pro uživatele. Pokud například změníme typ řídicího systému, vyžaduje tato změna malé nebo dokonce žádné úpravy v naprogramovaném zdrojovém kódu.

Automatizační software [2] obsahuje programovací a diagnostické nástroje pro každou fázi vývoje automatizačního projektu. Dále kombinuje všechny softwarové balíčky potřebné pro konfiguraci, programování, komunikaci a vykonávání všech řídicích součástí – řídicí systémy, polohovací systémy a vizualizační (panelové) systémy.

Části nutné pro vytvoření automatizačních aplikací lze popsat pomocí klíčových výrazů, které jsou shrnuty v tab. 2.1 a jejich princip je ukázán na obr. 2.1.

Výraz	*Význam*
Automation Studio	Vývojové prostředí pro konfiguraci, programování a diagnostiku řídicí aplikace
Automation Net	Komunikace mezi řídicími systémy – nahrání/stáhnutí řídicího programu, přenos dat z / do / mezi řídicími systémy
Automation Runtime	Operační systém zajišťující vykonávání řídicích programů
Automation Targets	Řídicí jednotky

Tab. 2.1 – Přehled jednotlivých částí automatizační software



Obr. 2.1 – Obecné propojení jednotlivých částí automatizačního softwaru

2.1. Programovací software – „Automation Studio”

Automation Studio [3,4] je prostředí sloužící k tvorbě automatizačních aplikací (řídicí, vizualizační i komunikační části). Umožňuje jednoduché a účinné uspořádání a programování automatizačních zařízení za použití grafického uživatelského rozhraní běžícího v MS Windows.

Všechny dostupné řídicí systémy (Automation Targets), v nichž běží operační systém (Automation Runtime), jsou propojeny automatizační sítí (Automation Net). Celá tato sestava je součástí prostředí Automation Studio.

2.1.1. Vývojové prostředí

Vývojové prostředí Automation Studio slouží k snadné konfiguraci a programování automatizačních úloh. Grafické zobrazení konfigurace hardwaru a softwaru nabízí jasný přehled o projektu.

Hardwarovou konfiguraci buď sami definujeme, nebo je cílový hardware systémem automaticky rozpoznán a podporován.

Všechny proměnné jsou v programu zpřístupněny využitím symbolických jmen. Vztah mezi symbolickým jménem a hardwarem (vstup, výstup, vnitřní proměnné) je definován v deklaraci proměnných. Při deklaraci proměnných definujeme tyto parametry: jméno proměnné, datový typ, rozsah, oblast, inicializační hodnotu, vlastníka knihoven a popis.

Programování ve vývojovém prostředí Automation Studio odpovídá standardu IEC 61131–3. Programování lze provádět ve všech standardních jazycích: kontaktní schémata (LAD), jazyk mnemokódů (IL), strukturovaný text (ST) a sekvenční strukturované diagramy (SFC). Prostředí dále nabízí možnost použít jazyky Automation Basic (AB) nebo ANSI–C. Barevná syntaxe programových hlavních řídicích slov zjednoduší tvorbu a analýzu kódu programu pro všechny jazyky.

Automation studio má v sobě zakomponované knihovny, které obsahují široké spektrum standardních funkčních bloků pro různé druhy operací (od jednoduchých logických operací a matematických operací až ke komunikačním protokolům a složitým řídicím algoritmům). Pomocí správce knihoven lze vytvářet a spravovat vlastní funkční bloky.

Řídicí aplikaci můžeme rozdělit do více úloh, které odpovídají procesům z pohledu operačního systému a které mají určité vlastnosti, označované jako třídy úloh. U jednotlivých tříd úloh definujeme prioritu vykonávání, dobu komunikačního cyklu a jazyk, ve kterém jsou úlohy dané třídy psány.

Program každé úlohy se dělí na inicializační část, která se provede jen po restartovaní, spuštění systému a na cyklickou část, která se opakuje po definovaném časovém intervalu. Jednotlivé úlohy řeší určitou část aplikace a lze je psát v libovolném jazyce, který je optimální pro danou problematiku.

Řídicí aplikace dále obsahuje: datové, systémové a speciální (advanced) objekty. Datové objekty se používají jako tabulky, do kterých je možno zapisovat. Systémové objekty se vytvářejí automaticky při překladu programu a jsou využívány k volání potřebných knihoven. Speciální objekty obsahují data reprezentující vizualizaci, parametry polohového řízení, vačkové profily a další.

Automation Studio též obsahuje nástroje pro monitorování, vyhodnocování proměnných a ladění programu (za podmínky běhu procesoru).

2.1.2. Polohovací komponenty

Polohovací komponenty jsou softwarový balík, jehož zásluhou se servozesilovače (ACOPOS) stávají nedílnou součástí celé řídicí aplikace. Komunikace mezi řídicím systémem a servozesilovačem je zajišťována operačním systémem a vlastní řízená osa vystupuje v aplikaci jako datová struktura, jejíž jednotlivé položky přesně popisují vlastnosti osy – od vstupních a výstupních signálů, přes nastavení vnitřních regulátorů až po okamžité hodnoty veškerých známých veličin (požadované i skutečné okamžité proudy, rychlosti, zrychlení, polohy, odchylky, teploty atd.), včetně možnosti využít vzdálený servozesilovač pro lokální jednoduché logické řízení s vlastními technologickými vstupy a výstupy.

Editor parametrů umožňuje upravovat, kopírovat a zobrazovat inicializační parametry os. Parametry jsou přehledně rozděleny podle typu ve stromové struktuře.

Zcela samostatnou kategorií je víceosé řízení. Zde nabízí Automation Studio grafický editor a řídicí příkazy (CAM profil automat), pro definice závislostí trajektorií os typu master–slave včetně integrovaných technologických funkcí typu elektronické převodovky, letmé pily, polohové vačky a kompenzační převodovky pro přechod mezi jednotlivými částmi definované trajektorie. Jde–li o aplikaci, kdy není možné předem definovat závislosti mezi jednotlivými osami, stále zůstává možnost pracovat s celým víceosým systémem, kde osy tvoří objekty programu, a definovat závislosti os za běhu aplikace. Tak je možné vytvořit řešení i velmi komplikovaných záležitostí bez omezení daných běžnými systémy CNC, ovšem při zachování přesných polohovacích algoritmů.

Zkušební prostředí „Test“ slouží pomocí příkazů (nc-actions) k nastavení a otestování parametrů (limitní, řídicí, pohybové a další) servozesilovače, k získání aktuálních dat a k vyzkoušení pohybů reálných os.

Prostředí zároveň nabízí možnost sledovat on–line práci pohonů pomocí elektronického osciloskopu (Data Tracer), jenž pracuje s daty vzorkovanými přímo servozesilovačem ACOPOS. Je to výkonný nástroj pro nastavování a diagnostiku pohonů. Na základě takto získaných průběhů je naladění patřičných pohybových parametrů dosti jednoduchou záležitostí, přičemž pro práci v tomto testovacím režimu není třeba napsat jediný řádek programového kódu a pouze na základě vlastností produktů B+R (PLC, motorů, automatizační sítě a prostředí Automation Studia) je možné získat potřebné údaje, které se automaticky využijí v dané aplikaci. Osciloskop umožňuje současné zobrazení až 10 grafů. Uživatel má možnost definovat způsob spouštění osciloskopu (např. od začátku pohybu). K přesnému vyhodnocování slouží i funkce lupy umožňující změnu měřítka, dále měřicí kurzory a možnosti zobrazení maximálních a průměrných hodnot.

Polohovací komponenty tedy zajišťují integrované řešení pro vývoj, konfiguraci a diagnostiku aplikací s polohovým řízením, převodovkami a elektronickými vačkami. V jednom projektu lze spravovat až 255 os. Tyto osy jsou nedílnou součásti projektu, ve kterém jsou editovány a zadávány jejich parametry.

2.1.3. Vizualizační komponenta

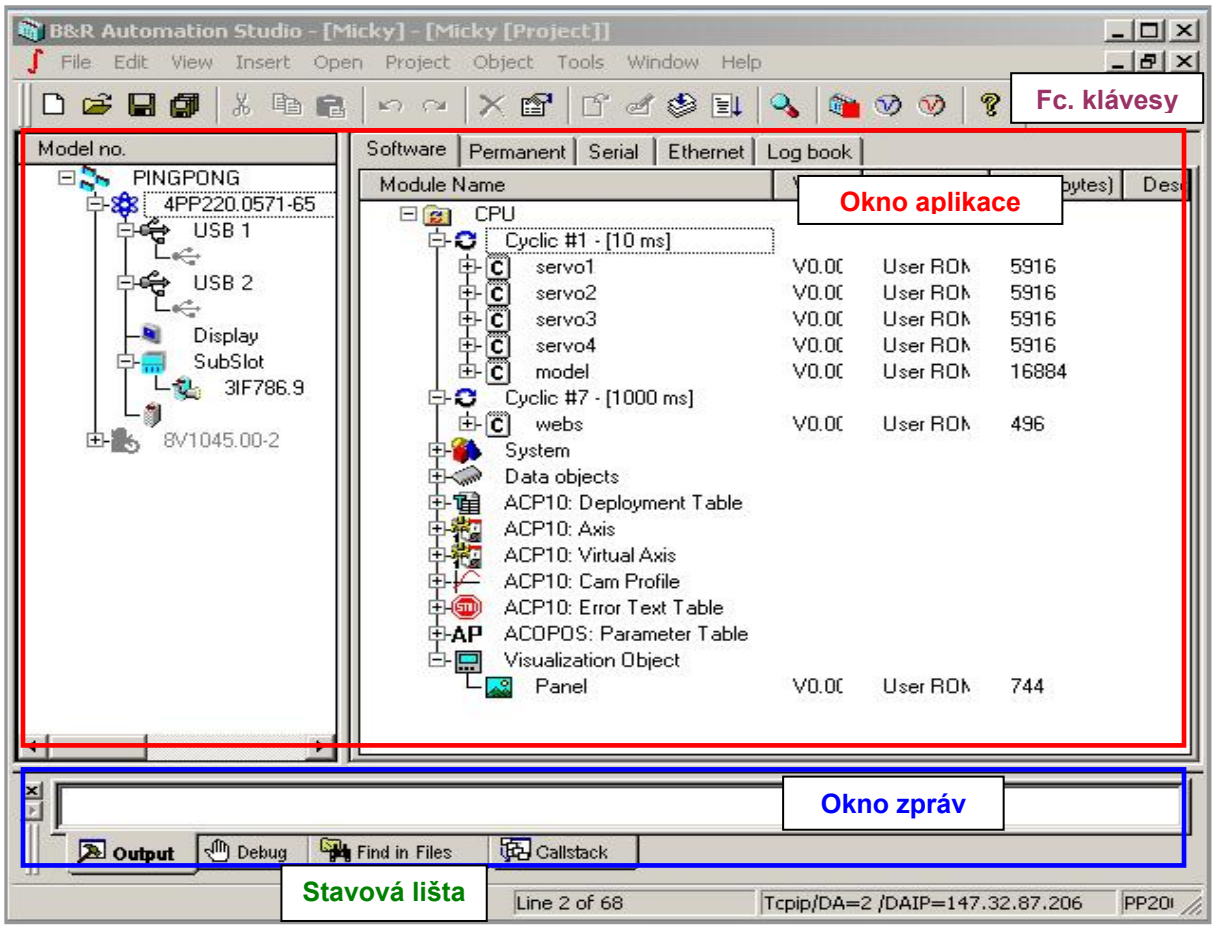
Pomocí vizualizační komponenty můžeme vytvářet procesní obrazovky a další vizualizační prvky současně s řídicí částí automatizačního projektu. Tak dosáhneme optimální interakce mezi vizuálními a logickými prvky. Společná datová oblast pro řízení, vizualizaci a pohony odstraňuje potřebu opakovaného zadávání.

Editor vizualizačních obrazovek s plnou podporou grafiky se automaticky adaptuje na používaný hardware. Na obrazovku, která podporuje 256 barev, lze vkládat tyto základní prvky: vstupní a výstupní pole, tlačítka, rastrové obrázky, grafické prvky s dynamickou podporou změn atributů a další.

Vizualizační komponenta umožňuje zaznamenávat změny procesů, které zpracovává v systému alarmů. Alarmy lze jednotlivě konfigurovat.

2.1.4. Struktura Automation Studia

Automation studio se skládá z několika částí (viz. obr. 2.2): funkčních kláves (otevření a uložení projektu, překlad programu, nahrání do řídicího systému, zapnutí stavu online a restart řídicího systému), okna aplikace (hardwarová konfigurace a objekty jednotlivých komponent), okna zpráv (informuje o chybách při tvorbě SW) a stavové lišty (zobrazuje typ procesoru, typ komunikace a její stav).



Obr. 2.2 – Struktura Automation Studia

Okno aplikace:

Hardwarová konfigurace (levá strana) je využívána k přehledné prezentaci použitých komponent. Pravá strana se automaticky přizpůsobí požadavkům levé strany. To znamená, klikneme–li na prvek v hardwarové konfiguraci, zobrazí se nám odlišné položky v pravé části (např. pro procesor to je: software, parametry, komunikace a výpis hlášení z procesoru, ale pro jiné komponenty se to liší). V jednotlivých položkách potom můžeme vkládat automatizační úkoly a systémové objekty, definovat potřebné parametry, nastavit parametry komunikace atd.

2.1.5. Monitorování, vyhodnocování a ladění

Automation Studio poskytuje řadu možností k monitorování, vyhodnocování a ladění. Používá k tomu Online Monitor procesních proměnných, Watch, Tracer, Line Coverage, Test, Debugger a System log book .

Online Monitor

V aplikačním okně ukazuje u jednotlivých objektů projektu jejich umístění (systém, projekt), umístění (user Rom, system Rom) a stav (run, use). Dále umožňuje sledování hodnot proměnných přímo v kódu aplikace (popřípadě i jejich změnu ve vybraných jazycích).

Line Coverage

Pomocí barevného označení řádků (částí) v kódu programu ukazuje, jaká část aplikace se vykonává.

Watch

Umožňuje zobrazovat hodnoty vybraných proměnných ve vybraném kódu (tzn. binárně, dekadicky atd.) v editačním okně a měnit jejich hodnotu.

Tracer

Tato komponenta graficky zaznamenává změnu hodnot proměnných v časovém intervalu. Tento displej dovoluje tato naměřená data dále zpracovávat (nalezení maximální/minimální hodnoty, změřit čas mezi dvěmi provedenými stavy atd.).

Test

Test je speciální prostředí, které slouží k nastavení a otestování parametrů servozesilovače, jeho ovládání a získávání aktuálních dat pomocí akcí (nc-actions).

Debugger

Překladač kontroluje správnost zdrojového kódu a informuje o chybách, které vypíše v okně zpráv a zobrazí nám její popis i pozici. Dále umožňuje krokování programu, vstupováním do procedur i provedení pouze jednoho cyklu.

System log book

Informuje o fatálních a kritických chybách vzniklých za běhu aplikace, dále o stavech procesu a komunikace. Vyskytující se hlášení se zobrazují ve výstupním okně, kde je zobrazen jejich kód, popis a čas vzniku.

Ukázka monitorovacích, vyhodnocovacích a ladicích součástí je v příloze 2.

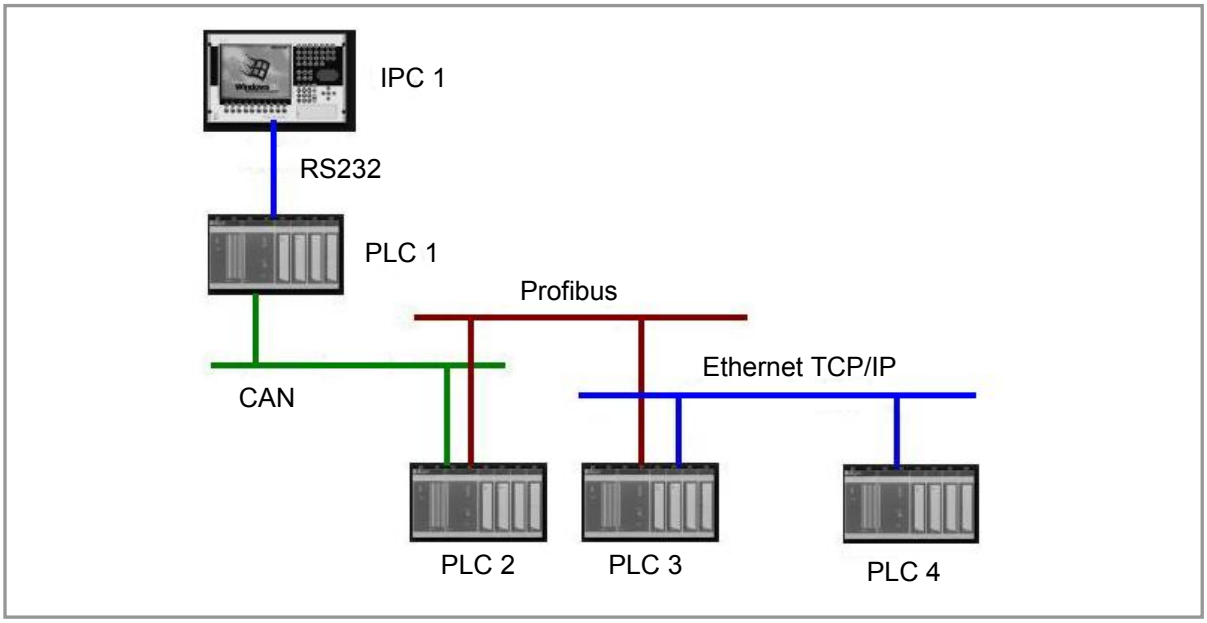
2.2. Automatizační síť – „Automation Net“

Automatizační síť slouží ke komunikaci a výměně dat mezi jednotlivými řídicími systémy (Automation Targets), Automation Studiem a dalšími I/O moduly.

Automatizační síť používají všechny součásti systému. Každá stanice v síti musí být schopna vyměňovat a zpracovávat programové objekty a procesní data nezávisle na cílovém operačním systému (Windows, operační systém řídicích systémů), používaném protokolu (INA2000, Net2000, Mininet a další) a přenosovém médiu (Ethernet Powerlink, Ethernet TCP/IP, Remote IO, Profibus, sériového rozhraní a CAN) .

Mnoho automatizačních řešení využívá distribuovanou inteligenci. Úlohy jsou rozloženy do několika automatizačních systémů, kde každý má unikátní adresu.

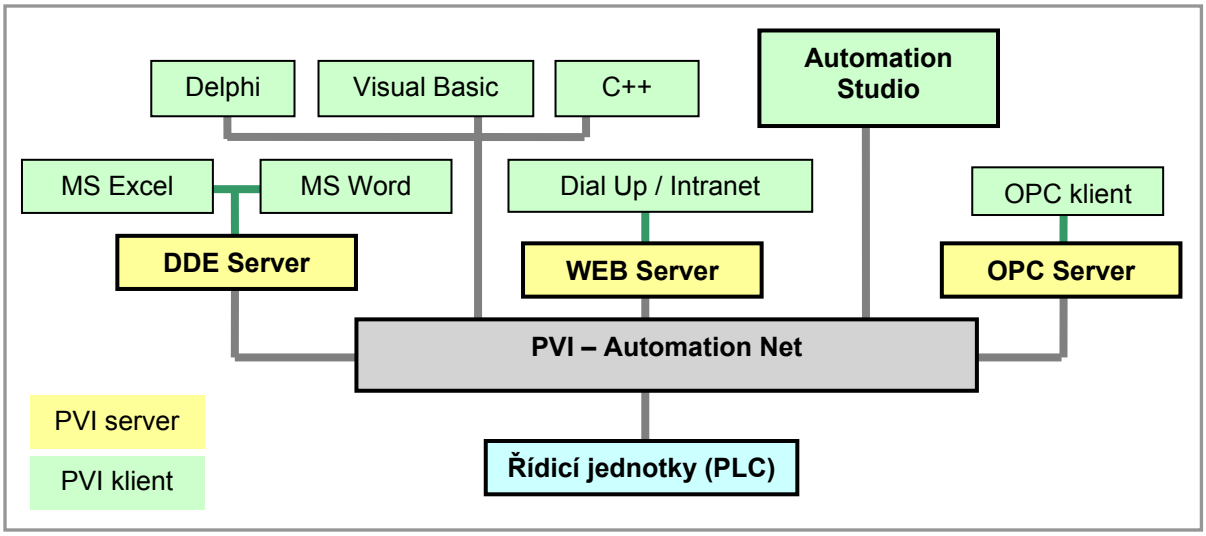
Spojení mezi jednotlivými automatizačními systémy může být potom realizováno pomocí odlišných typů automatizační sítě. Ukázka propojení distribuovaného řešení řízení a komunikace je na obr. 2.3.



Obr. 2.3 – Distribuovaná inteligence v automatizační síti

2.2.1. Rozhraní pro vizualizaci procesů – „PVI“

Rozhraní pro vizualizaci procesů (PVI) je komponenta automatizační sítě, která organizuje spojení z průmyslového prostředí do rozhraní pro Windows programy. Pomocí PVI je možné spojit odlišné architektury za pomocí DDE serveru, OPC serveru nebo webového serveru do jednoho řídicího celku. Dále je možné přenášet data do programů jako je MS excel, word ale i C++ atd. Možnosti propojení těchto architektur je ukázáno na obr. 2.4.



Obr. 2.4 – PVI rozhraní

Princip PVI klient/server umožňuje vzdálené operace PVI aplikací. To znamená, že PVI aplikace pracuje jako PVI klient, který může přenášet procesní proměnné a programy z/do PVI serveru pomocí sítě, modemového spojení nebo dalších komunikačních médií.

2.2.2. Porovnání parametrů automatizačních sítí

V závislosti na použité komunikaci rozlišujeme následující vlastnosti: rychlost, vzdálenost, maximální počet připojených stanic, zda umožňuje komunikaci v reálném čase a zapojení. Přehled vlastností je shrnut v tab. 2.2.

Typ	*Automatizační síť*	*Přenos I / O*	*Maximální rychlost*	*Maximální vzdálenost*	*Maximální počet stanic*	*Reálný čas*	*Topologie*
Ethernet Powerlink	ANO	ANO	100 MBaud	100 m	253	ANO	Hvězda, sběrnice
Ethernet TCP/IP a UDP/IP	ANO	NE	10 Mbaud	100 / 185 m	dle provedení	NE	Hvězda, sběrnice
Expanze	X)	ANO	16 Bit Bus	2 m	4 – 9	ANO	Point to point
Remote I/O	X)	ANO	2 Mbaud	1200 m	32	ANO	Sběrnice
Profibus	ANO	NE	500 kBaud	1200 m	32	ANO	Sběrnice
Sériové připojení	ANO	NE	Dle: RS232, 422, 485	15 – 1200 m	2 nebo více	NE	Point to point, Sběrnice
CAN	ANO	ANO	500 kBaud	1000 m	64	ANO	Sběrnice

X) Optimalizováno pro I/O přenos

Tab. 2.2 – Porovnání automatizačních sítí

2.3. Operační systém – „Automation Runtime“

Operační systém reprezentuje vrstvu mezi Automation Studiem a jednotlivými řídicími systémy komunikujících pomocí automatizační sítě.

Programy a data vytvořené v Automation Studiu jsou pomocí automatizační sítě nahrány do operačního systému řídicí jednotky, ve kterém se dále zpracovávají. Operační systém poskytuje vytvořeným programům možnost mít na všech cílových řídicích systémech jednotné provozní prostředí. To umožňuje jednotné programování a provoz na libovolném zařízení.

Operační systém řídicího systému pracuje jako nastavitelný, deterministický, víceúlohový systém reálného času. Požadovanou dobu cyklu pro zpracování všech dat programu lze rozčlenit do několika tříd úloh a zajistit tím, aby byly všechny úlohy zpracovány v požadovaném času.

2.3.1. Víceúlohové zpracování – „Multitasking"

Na víceúlohových systémech se procesorový výkon dělí pro vykonání více úloh. Tyto úlohy se zpracovávají paralelně. Úloha je nezávislá část programu reprezentující mnohé procesy (řídicí algoritmy, digitální a analogová spojení, kontrolu, vyhodnocení měření, podmínky a další).

Dělení úloh nám poskytne následující výhody:

1. Strukturované uspořádání aplikace
2. Vytvoření úlohy v nejvhodnějším programovacím jazyce
3. Programování a testování jednotlivých funkcí samostatně
4. Jednoduchou údržbu jednotlivých úloh

U víceúlohových systémů se setkáme s následujícími požadavky pro PLC aplikace:

1. Paralelní zpracování vícenásobných řídicích úkolů
2. Deterministické víceúlohové zpracování (s přesnými časovými periodami)
3. Různé a nastavitelné časy cyklu s jejich monitorováním
4. Identický I/O obraz pro každou třídu úloh

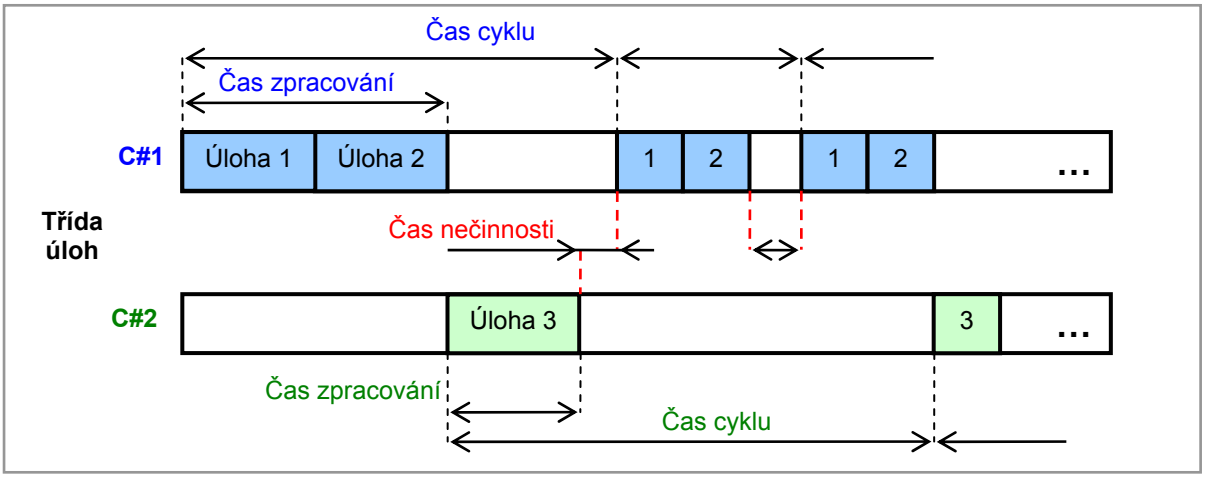
Čas cyklu je poskytnut pro vykonání všech úloh v dané třídě úloh. Při zpracování se rozlišuje doba zpracování dat a doba nečinnosti (všechny úlohy byly zpracovány v požadovaném času a procesor nemá co vykonávat).

Plánovač úloh aktivuje příslušné úlohy na začátku jejich třídy. Úlohy se v příslušné třídě cyklicky vykonávají dle času cyklu. Jednotlivé třídy se zpracovávají v daném pořadí podle jejich priority a času.

Na začátku každého cyklu se provádějí následující kontroly:

1. Jestli jsou úlohy ukončené uvnitř nastaveného času třídy úloh
2. Jestli nový vstupní obraz je plně dosažitelný
3. Jestli byl poslaný výstupní obraz

Na obr. 2.5 je ukázáno víceúlohové zpracování tří úloh (úloha 1,2 a 3), které jsou rozděleny do dvou tříd (C#1 a C#2).



Obr. 2.5 – Víceúlohové zpracování

2.3.2. Třída úloh – „Task classes“

Třída úloh obsahuje všechny řídicí úlohy se stejným časem cyklu. Každý procesor má určitý počet těchto tříd a doba cyklu každé třídy se dá definovat. Třída úloh se dělí na dva typy a to na časové úlohy a na cyklické úlohy.

Časová úloha je určena pro velmi rychlé a časově kritické části aplikace. Časové úlohy jsou aktivované hardwarovým časovačem a můžou být volány a sledovaný ve velmi krátkých intervalech (od 1 ms).

Cyklické úlohy jsou vykonávány periodicky. Čas cyklu je definován uživatelem a je monitorován systémovým manažerem, který zajistí jeho řádné vykonání. Dále je definován čas tolerance, který nám uvádí čas, do kterého musí být úloha vykonána po spuštění.

2.3.2.1. Pravidla pro vykonávání úloh

Třídy úloh se vykonávají podle následujících pravidel:

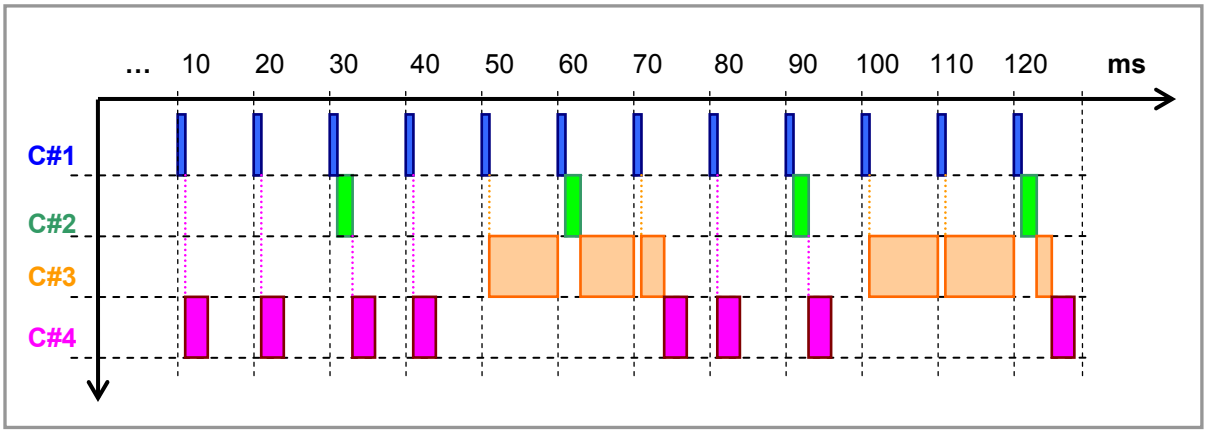
1. Vždy se jako první vykoná systémový manager
2. Dále se vykonávají úlohy s větší prioritou, to znamená jako první C#1, dále C#2 atd.
3. Následující třída se spustí pokud je čas po skončení třídy předchozí
4. Každá třída má definováno do jakého maximálního času se musí celá vykonat, pokud to nestihne, vyvolá se chybové hlášení
5. Třída, která nemá definovaný cyklický čas, se vykonává pokaždé, jsou li vykonány předchozí třídy a je čas na vykonání této třídy

2.3.2.2. *Vykonávání více tříd*

Zadání příkladu pro vykonávání více tříd úloh je v tab. 2.3. a diagram vykonávání je znázorněn na obr. 2.6. U každé třídy úloh známe dobu vykonávání všech jejich úloh a dobu cyklu.

Třída úloh	Cyklický čas [ms]	Doba vykonávání úloh [ms]
C#1	10	0,8
C#2	30	1,6
C#3	50	20,0
C#4	Dle možností systému	3,2

Tab. 2.3 – Zadání příkladu pro vykonávání více tříd úloh



Obr. 2.6 – Vykonávání čtyř tříd

2.3.2.3. *Vytížení procesoru*

Vytíženost procesoru je závislá na době zpracování všech úloh systému.

Pro názornost je vytíženost procesoru ukázána na příkladu. Doba trvání třídy úloh je 9 ms. Jednou úlohu přiřadíme do cyklické třídy C#1 s cyklicky časem 10 ms a jednou do třídy C#3 s cyklickým časem 100 ms.

Třída úloh	*Cyklický čas [ms]*	*Doba zpracování [ms]*	*Doba nečinnosti [ms]*	*Vytíženost [%]*
C#1	10	9	1	**90**
C#3	100	9	91	**9**

Tab. 2.4 – Vytížení procesoru

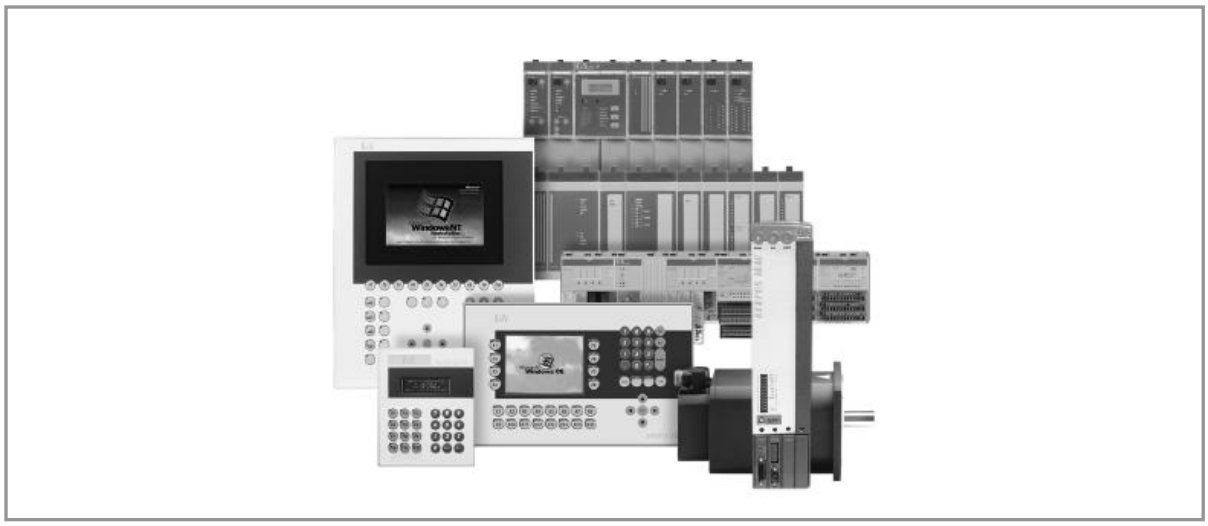
Z tabulky 2.4 vyplývá, že vytíženost procesoru je závislá na cyklickém času jednotlivých tříd a tím pádem je nutno tento čas volit v závislosti na časové náročnosti všech úloh v příslušné třídě.

2.4. Řídicí jednotky – „Automation Target“

Řídicí jednotky (viz. obr. 2.7) jsou hardwarová zařízení, na kterých běží operační systém. Všechny řídicí procesy a automatizační úlohy mohou být řízený programem běžícím v řídicí jednotce.

Řídicí jednotky jsou složeny z různých komponent (vstupní a výstupní moduly, centrální řídicí jednotka, komunikační moduly a interface moduly), které se dají libovolně kombinovat.

Jednotlivé řídicí jednotky se dělí do následujících skupin: řídicí systémy (PLC), polohovací systémy (digitální servozesilovače) a vizualizační panelové systémy.



Obr. 2.7 – Řídicí jednotky

Řídicí systémy jsou automatizační systémy (PLC), které poskytují nové úrovně výkonu, funkčnosti a ovládací bezpečnosti. Tyto PLC systémy pokryjí veškeré požadavky na řízení, od jednoduchého logického řízení až po komplexní automatizační systémy.

Polohovací systémy slouží k polohovému řízení motorů. Řídicí polohové funkce jsou plně včleněny do řídicího systému (modulový servozesilovač s řídicím procesorem). Použitím vačkových profilů řízení je umožněno aplikacím, aby byly snadno konfigurovatelné pro co nejvíce os.

Vizualizační panelové systémy jsou modulární operátorské panely s klávesnicí, displejem (i dotykovým) a procesorem sloužící k řízení i vizualizaci aplikací. Jednotlivé systémy se dělí podle komponent, které obsahují.

3. POLOHOVÉ ŘÍZENÍ

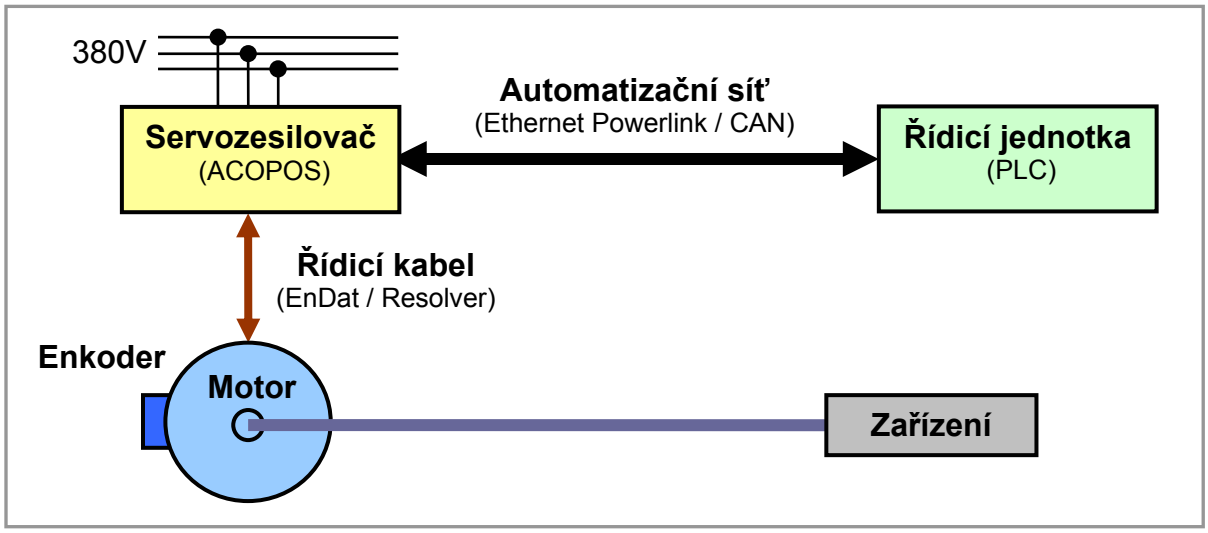
Součástí mnoha automatizačních úloh, nejen z oblasti aplikací CNC, je řízení jedné či více os. V reálném světě je tato problematika často úzce provázána s celkovým řízením strojů či technologií a vzniká zde prostor pro tvorbu aplikací nebo produktů integrujících obě oblasti.

Požadavky na taková řízení jsou naprosto zřejmé – na jedné straně to je komfortní práce se zařízením typu PLC se všemi jeho výhodami (téměř libovolné řídicí algoritmy, rozsáhlá paměť, široké komunikační a vizualizační možnosti), naproti tomu precizní (nejlépe digitální) servozesilovač, vnášející do celého řešení vysokou dynamiku a přesnost. Použití servozesilovačů řady ACOPOS přináší i do nejnáročnějších automatizačních úloh zcela nový rozměr – od asynchronního motoru, přes synchronní motory, lineární motory, řídicí systém až po obslužný panel je vše sloučeno do jednoho celku. Spojujícím prvkem není pouze výrobce či nálepka na obalu, ale samotná vnitřní struktura systému.

Pro přenos informací, dat a povelů z jedné části do druhé, mezi řídicím systémem a servozesilovačem, využijeme komunikačních možností servozesilovače, který umožňuje komunikace Ethernet Powerlink a CAN.

Datová sběrnice CAN je vhodná pro ovládání malých zařízení s menším počtem řízených os (motorů). Oproti tomu komunikace Ethernet Powerlink se používá k ovládání zařízení s větším počtem os a tam, kde potřebujeme využít vysoké požadavky na dynamiku pohybových systémů.

Řešení polohového řízení pomocí servozesilovače, který ovládá motor na základě řídicího algoritmu v PLC je znázorněno na obr. 3.1.



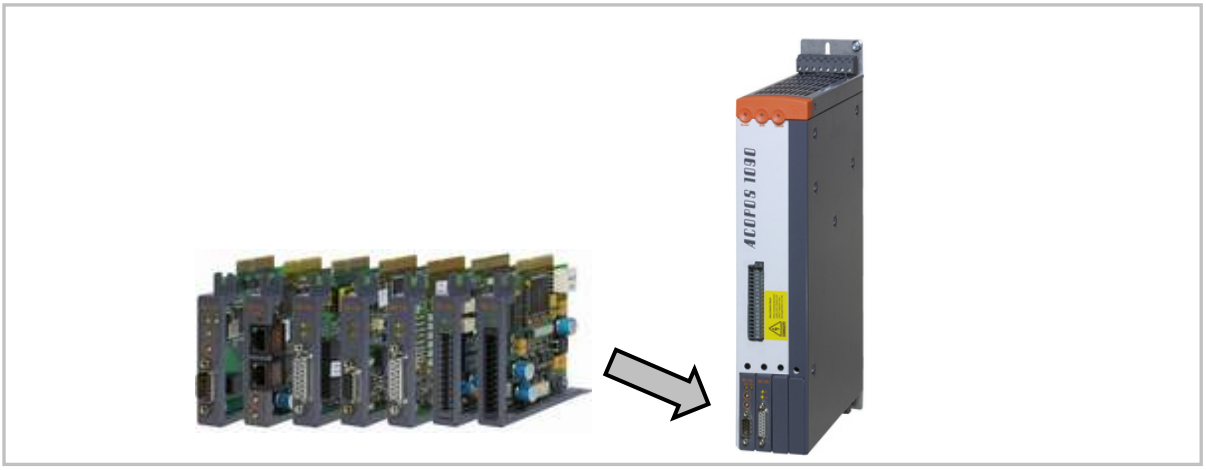
Obr. 3.1 – Polohové řízení

Řídicí systém naprogramovaný ve vývojovém prostředí Automation Studio je obsluhován z operátorského panelu či průmyslového PC. Řídí veškeré pohybové záležitosti všech zapojených pohonů – vše propojeno do jediné komunikační sítě, s jednotnou diagnostikou, při dokonalé synchronizaci a maximálním výkonu, tj. bez ztráty špičkových vlastností kterékoliv z komponent.

3.1. Servozesilovač – „ACOPOS”

Digitální servozesilovače ACOPOS [1] (viz. obr. 3.2) jsou určeny k řízení synchronních i asynchronních motorů s napájecím napětím 3 × 380 V v proudové, rychlostní i polohové smyčce. Komunikační cyklus pro řízení motoru je 400 µs.

Jedná se o digitální servozesilovače s velkým výpočetním výkonem, přesnými řídicími algoritmy a špičkovými parametry výkonové části.



Obr. 3.2 – Servozesilovač

Integrovanou součástí servozesilovačů jsou brzdné rezistory i odrušovací filtr (EMC), samozřejmostí je možnost propojení stejnosměrných částí více servozesilovačů pomocí sběrnice DCbus, která umožňuje maximálně efektivní využití dodávané energie.

Výborné řídicí vlastnosti jsou zajištěny použitím signálového procesoru (DSP) a promyšleným řídicím systémem. DSP poskytuje dostatečný výkon pro výpočet veškerých parametrů pohybu a v neposlední řadě také neustále sleduje zatížení výkonových částí servozesilovače a podle něj řídí jeho funkci. Konkrétně to znamená, že na základě údajů z čidel teploty je v reálném čase pořizován matematický teplotní model, na jehož základě je následně určeno skutečné výkonové zatížení jednotlivých částí.

Servozesilovač je schopen ve špičkách dodávat relativně vysoké proudy a tedy i výkony (proud v rozsahu od 1.0 – 128.0 A, výkon v rozsahu 0.5 – 64.0 kW) bez

nebezpečí vlastního poškození a bez omezení nějakou teoretickou maximální hodnotou, které by ve svém důsledku znamenalo omezení dynamiky celého pohybu.

Hlavním určením DSP je ovšem řízení připojeného motoru. V servozesilovači je realizována jak proudová, tak i rychlostní a polohová řídicí smyčka. Řídicí část umožňuje realizovat proudovou smyčku s periodou 50 μs, rychlostní s periodou 200 μs a polohovou s periodou 400 μs.

Vlastní pohyb je řízen pomocí pulsně šířkové modulace napájecího napětí motorů se základní frekvencí 20 kHz, která zajišťuje přesnou realizaci vypočtených akčních zásahů.

Zpětná vazba je uskutečněna pomocí snímačů polohy (endat, rezolver, inkrementální nebo absolutní snímač polohy). Součástí těchto snímačů je i tzv. elektronický typový štítek, na kterém jsou zaznamenána všechna významná mechanická a elektronická data motoru, která umí servozesilovač (ACOPOS) využít. Odpadá tedy nutnost „seznámit“ servozesilovač s parametry připojeného motoru.

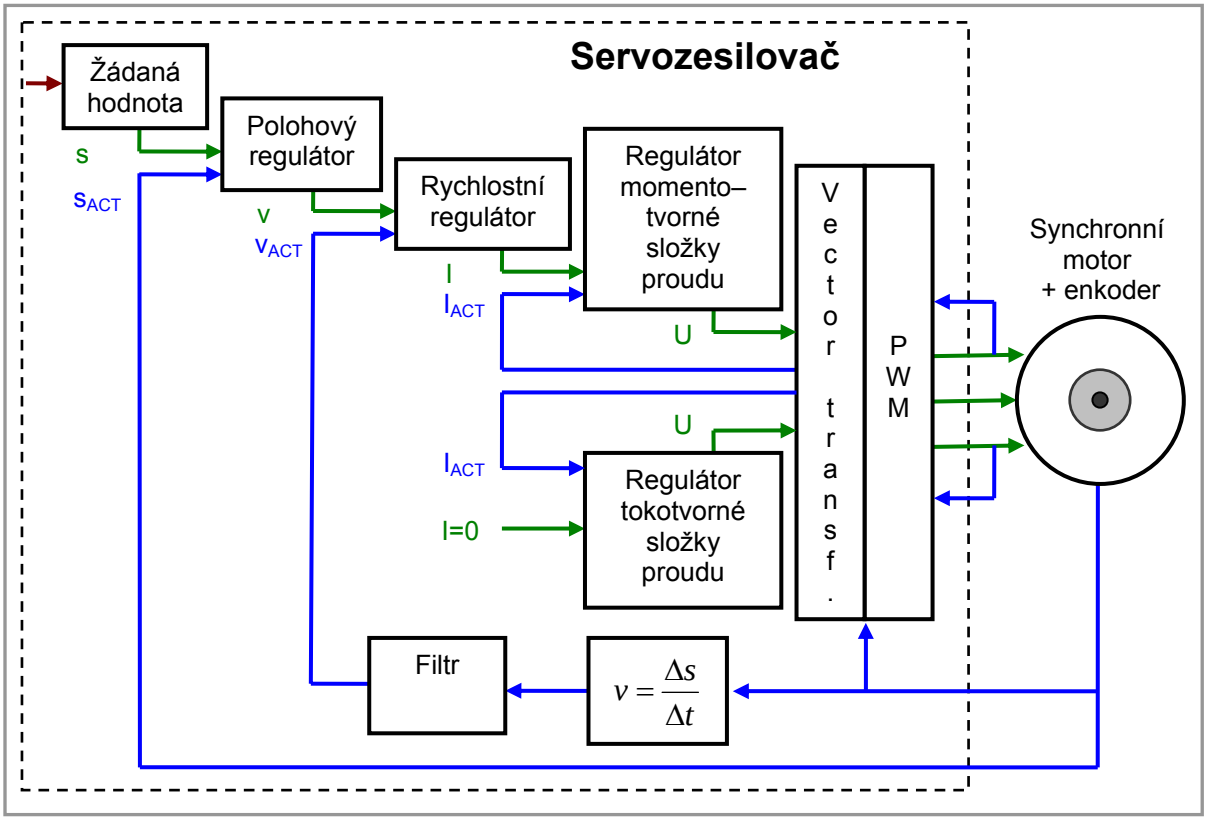
Budoucí konfigurace servozesilovačů je realizována pomocí zásuvných modulů. Zásuvné moduly jsou k dispozici pro vytvoření síťového spojení s dalšími servozesilovači, řídicími a vizualizačními systémy, ale i s kodéry, senzory a akčními členy. K dispozici je také CPU modul, který nahrazuje řídicí systém potřebný pro řízení servozesilovače.

Servozesilovače mohou být použity k řízení v různých konfiguracích závislých na síťovém typu a požadavcích aplikace. Následující řídicí funkce je možné použít pro všechny konfigurační příklady: elektronické převodovky, elektronické kompenzační převodovky, příčné řezačky, elektronické vačkové profily, letmé pily, CNC a další.

3.1.1. Řídicí smyčka

Servozesilovač obsahuje řídicí, měřicí ale i vyhodnocovací prvky. Jednotlivé regulační obvody polohy, rychlosti i proudu jsou kombinovány použitím oddělených regulačních smyček. To umožňuje snadnější definici jednotlivých parametrů.

Kvalita řízení je zajištěna použitím krátkého času cyklu pro jednotlivé regulační smyčky (proudové, rychlostní a polohové). Zapojení regulační řídicí smyčky je na obr. 3.3.



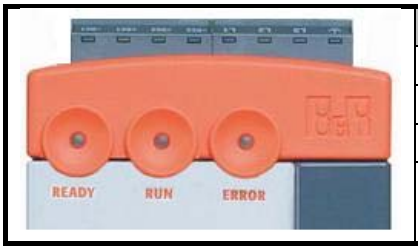
Obr. 3.3 – Zapojení regulační smyčky servozesilovače

Polohový PI nebo prediktivní regulátor s kladnou zpětnou vazbou přemění požadovanou hodnotu polohy na odpovídající rychlost. Rychlostní PI regulátor konvertuje žádanou hodnotu z polohového regulátoru na odpovídající proud. Proudové regulátory slouží k vektorovému řízení motoru (regulace otáček se provádí změnou momentu). Regulovány jsou momentotvorná a tokotvorná složka proudu. Regulátor momentotvorné složky reguluje proud pro změnu momentu. Naproti tomu regulátor tokotvorné složky je signálovým procesorem počítán na maximální využití magnetického obvodu elektrického stroje.

Nastavení polohového a rychlostního regulátoru se provádí ručně pomocí parametrů. Proudový regulátor se nastaví automaticky pro motory od fy. B+R nebo ručně pro motory od jiných výrobců.

3.1.2. Indikace stavu servozesilovače

K indikaci stavu servozesilovače slouží led diody na čelním panelu. Tyto diody signalizují jeho připravenost, chod a případné poruchy. Popis diod signalizujících stav servozesilovače je popsán v tabulce 3.1.



Dioda	*Popis*	*Barva*
1. (z leva)	Připravenost	Zelená
2.	Chod	Oranžová
3.	Porucha	Červená
Pokud nesvítí ani jedna dioda, není napájena řídicí jednotka servozesilovače pomocí 24 V.		

Tab. 3.1 – Popis diod signalizujících stav servozesilovače

Připravenost

Svítí, pokud je servozesilovač připraven k činnosti (je napájen 380 V, je nahrán operační systém z PLC a neexistují žádné stálé nebo dočasné chyby).

Chod

Svítí, pokud je zapnut regulátor servozesilovače a neexistují žádné stálé nebo dočasné chyby.

Porucha

Svítí, pokud existuje v servozesilovači stálá nebo dočasná chyba. Po odstranění chyby dioda automaticky zhasne.

Příklad stálých poruch: nespojena nebo vadná zpětná vazba z motoru, nízké napájecí napětí, nespojený nebo vadný snímač teploty motoru, vnitřní chyba na zařízení a další.

Příklady dočasných poruch: napájecí stejnosměrné napětí 24 V není v tolerovaném rozsahu, vnitřní ovládací napětí stejnosměrného napětí překračuje povolený rozsah, přehřátí vnitřních obvodů servozesilovače, přehřátí motoru, nefunkční komunikace CAN nebo Powerlink (špatný kabel, špatně nastavené adresy na komunikačních kartách) a další.

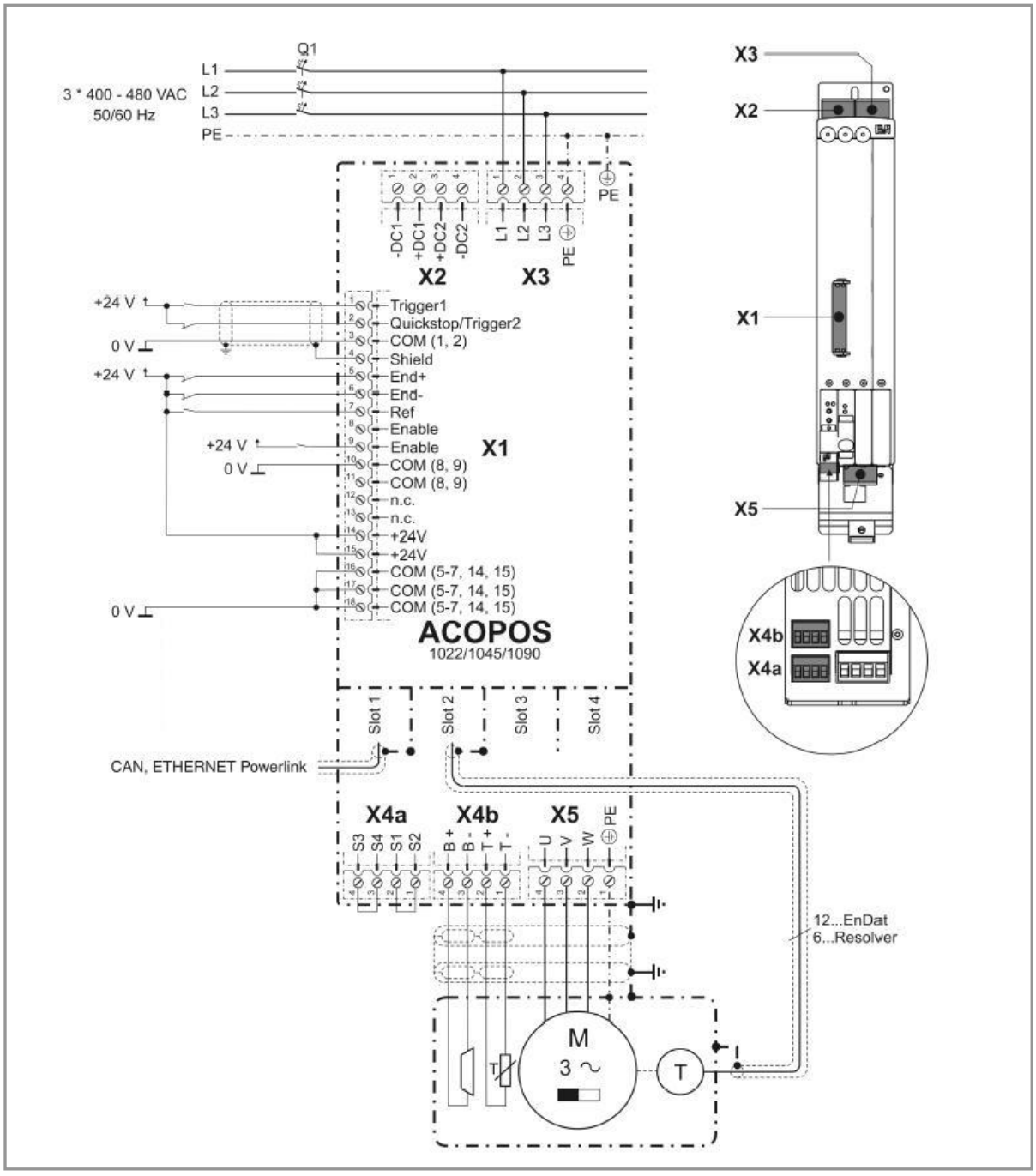
3.1.3. Principielní zapojení konektorů

V tab. 3.2 je popis a označení konektorů a slotů pro moduly servozesilovače.

Konektor	**Význam**
X1	Konektor pro koncové spínače/tlačítka a napájení řídicí jednotky
X2	Konektor vlastního zdroje +24 V
X3	Napájecí konektor 380 V servozesilovače
X4a	Konektor pro možnost externího napájení +24 V brzdy motoru
X4b	Konektor pro snímač teploty a el. brzdu motoru
X5	Konektor pro napájení motoru
Slot	**Význam**
Slot 1–4	Volné sloty pro rozšiřující moduly (komunikační, řídicí, digitální I/O atd.)

Tab. 3.2 – Seznam konektorů a slotů pro moduly servozesilovače

Význam jednotlivých svorek všech konektorů je popsán v příloze 3. Na obr. 3.4 je znázorněno rozmístění těchto konektorů a jejich principielní zapojení. V praxi je možno použít libovolné zapojení napájení i typů spínačů. Obrázek a popis potřebných řídicích kabelu motoru je v příloze 4 a 5.



Obr. 3.4 – Schématické zapojení servozesilovače

3.2. Ethernet Powerlink

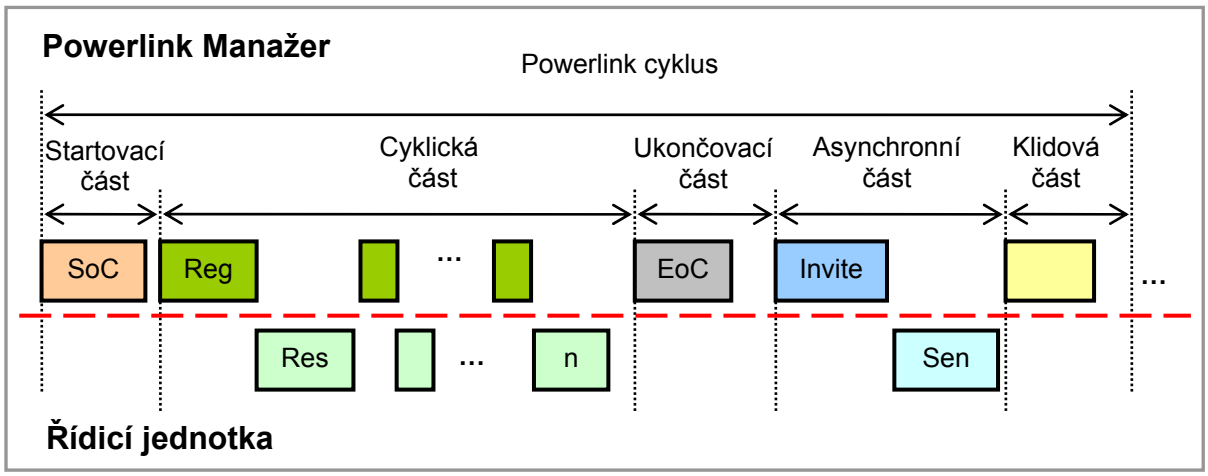
Ethernet Powerlink [6,14] je moderní protokol založený na standardu Fast Ethernetu. ETHERNET Powerlink je striktně deterministický, izochronní protokol pro přenos dat v reálném čase, který umožňuje synchronizovat jednotlivé účastníky a cyklicky s nimi komunikovat. Má pevnou dobu cyklu a minimální rozptyl synchronizačních značek. Z toho důvodu je výhodný pro typy úloh jako je řízení víceosých mechanismů.

Ethernet Powerlink využívá k řízení přístupu na síť mechanismus označený jako „Slot Communication Network Management“. Tento mechanismus je modifikací metody master-slave a spojuje v sobě výhodu velmi vysoké přenosové rychlosti s optimálním vytížením sítě při minimálním rozptylu v časování telegramů. V režimu „master-slave“ řídí komunikaci na síti řídicí zařízení - master. Ten dokáže zabránit kolizím, k nimž by jinak mohlo dojít, kdyby se dvě zařízení pokusila vysílat současně, avšak nevýhodou je, že zařízení nemohou komunikovat přímo, ale vždy jen prostřednictvím nadřízené stanice. U metody Slot Communication Network Management je tomu jinak. Každé zařízení v síti má přesně vymezenou dobu k vysílání, kdy smí vysílat data kterékoliv stanici, několika stanicím nebo i všem stanicím v síti. Tím je zaručeno, že vysílá právě jen jedna stanice. Ke kolizím nemůže dojít a šířka přenosového pásma sběrnice je optimálně využita.

Ethernet Powerlink musí kromě rychlé cyklické výměny dat zabezpečit i acyklickou komunikaci, která však cyklickou nesmí nijak ovlivnit. Pro použití v automatizaci je také nezbytné vyřešit synchronizaci vstupně-výstupních dat a řídicích systémů v celém automatizovaném celku. V rychlé cyklické výměně dat je nejkratší nastavitelná doba cyklu sběrnice 400 µs (to navíc odpovídá periodě regulátoru polohy servozesilovače B+R). V této periodě může být plně obslouženo až osm zařízení. V případě, že to nestačí, je nutné doby cyklu prodloužit: v 800 µs cyklu lze obsloužit 28 zařízení, v cyklu 1,2 ms jich může být až 63. To vše za předpokladu, že každá stanice posílá přibližně 100 bytů užitečných dat.

3.2.1. Komunikační cyklus

Protokol pracuje izochronně. To znamená, že výměna dat mezi stanicemi probíhá periodicky, v pevně definovaných časových intervalech (Powerlink cyklus). Čas cyklu se může nakonfigurovat libovolně. Jeden Powerlink cyklus obsahuje následující časové intervaly: startovací část, cyklickou část, ukončovací část, asynchronní část a klidovou část. Ukázka jednoho cyklu komunikace je na obr. 3.5.

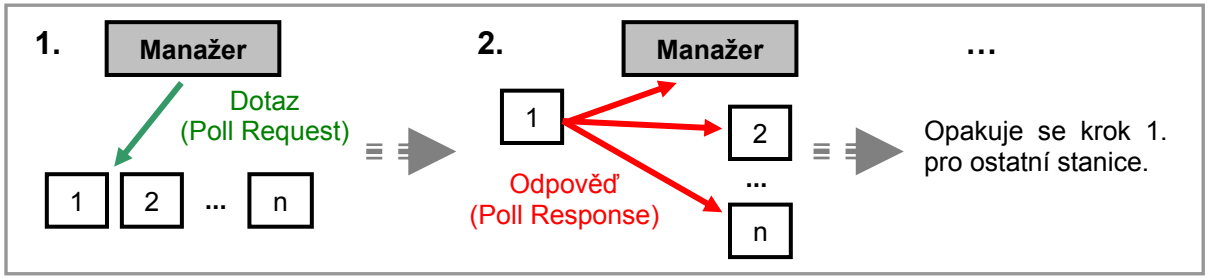


Obr. 3.5 – Jeden komunikační cyklus Ethernet Powerlinku

Powerlink cyklus začíná tzv. startovací částí (SoC), kterou používá master (manažer) řídicí jednotka k synchronizaci všech účastníků komunikace. Pouze tato část vytváří časové řízení. Potom probíhá cyklická komunikace kde manažer pošle dotaz (poll request) k první stanici, která pošle na tento dotaz odpověď (poll response) a tak to pokračuje i pro další stanice. Všechny tzv. rychlé stanice jsou takto adresovány v každém Powerlink cyklu. Cyklickou komunikaci ukončí opět manažer tzv. ukončovací částí (EoC). Následuje asynchronní komunikace. Po skončení asynchronního přenosu následuje doba klidu, tj. doba mezi dokončenou asynchronní dobou a začátkem dalšího cyklu. Během tohoto času všichni síťoví účastníci čekají na začátek následujícího cyklu. Doba klidu může být i nulová.

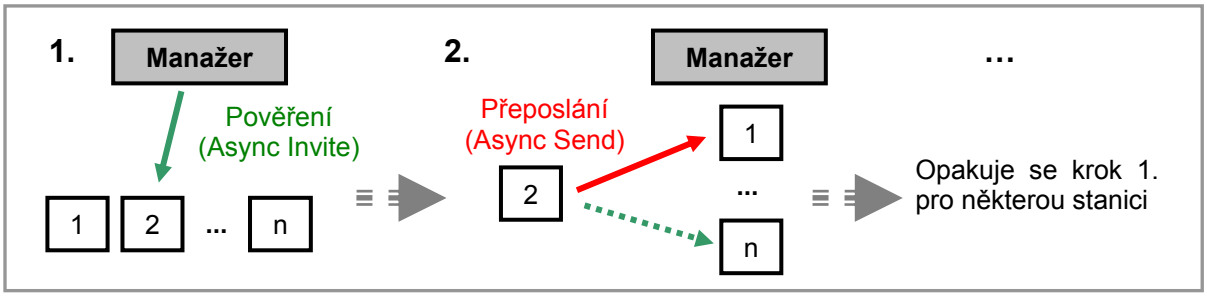
Jedna stanice musí být master tzv. Powerlink manažer. Ostatní dosažitelné stanice v síti musí být nakonfigurovány v manažeru a jsou pasivní. To znamená, že sami od sebe neposílají data, ale čekají na povel od manažera, který může být aktivní i sám.

Během cyklické části (viz. obr. 3.6) každá řídicí jednotka přijímá data z manažeru v Powerlink cyklu a opětovně posílá data zpět. Toto se provádí postupně od první stanice k poslední. Dotaz může být přijat pouze stanicí s příslušnou adresou, avšak odpověď je poslána všem (broadcast), to znamená, že je možné tato data sledovat (číst) ostatními pasivními stanicemi. Výměna dat tedy pracuje jako model vydavatel / čtenář (Publisher/Subscriber).



Obr. 3.6 – Ukázka cyklické komunikace

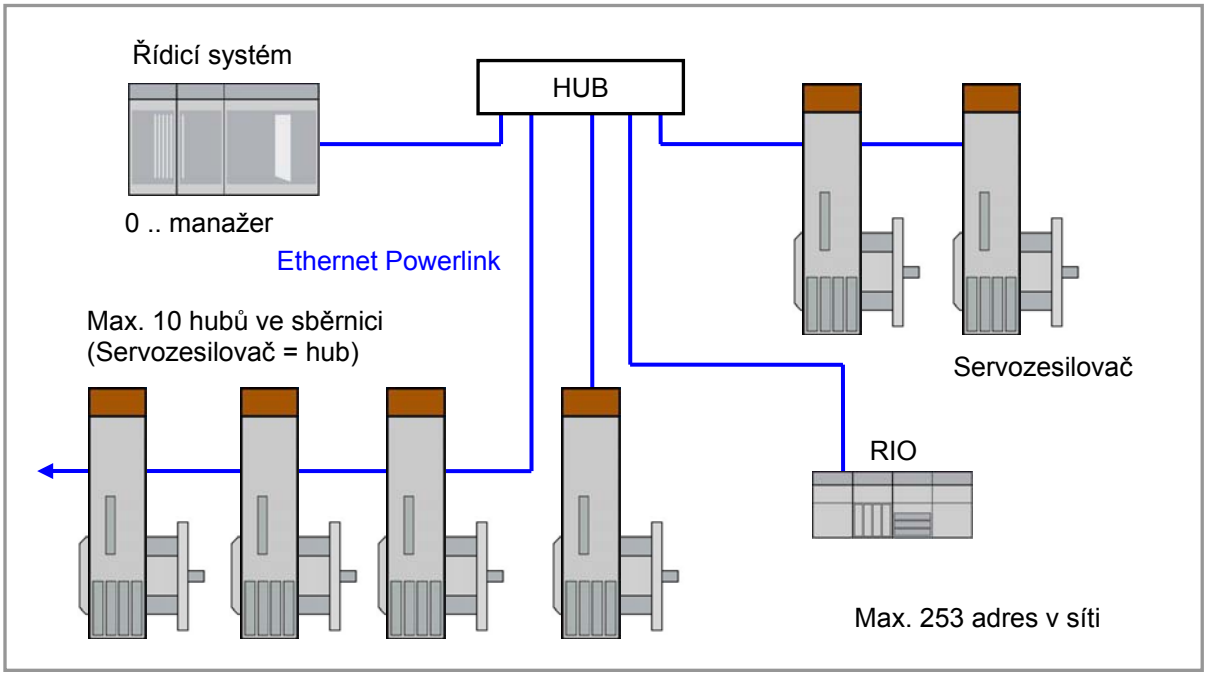
Acyklická data (viz. obr. 3.7) jsou vyměňována mezi stanicemi ve zbývajícím času až do dalšího cyklu. Manažer kontroluje přiřazení vysílacího oprávnění. Pokud chce stanice poslat asynchronní data, musí o tom manažera informovat při odpovědi na dotaz v cyklické části. Manažer vybere stanici z těch, které poslaly žádost o vysílání (včetně sebe) a pošle ji pověření (Invite). Stanice pak může poslat asynchronní data (Send). Vysílací rámec pak může být předán další stanici.



Obr. 3.7 – Ukázka asynchronní komunikace

3.2.2. Zapojení

Komunikaci Ethernet Powerlink lze zapojit jako sběrnici nebo do hvězdy. Maximální počet účastníků je 253, protože některé adresy jsou u řídicích systémů určeny ke speciálním účelům (adresa 00 znamená bootovací režim, ale také master stanici při řízení servozesilovačů, adresa FF znamená diagnostický režim). Ukázka zapojení je na obr. 3.8.



Obr. 3.8 – Možnosti propojení komunikace Ethernet Powerlink

4. SW REALIZACE POLOHOVÉHO ŘÍZENÍ

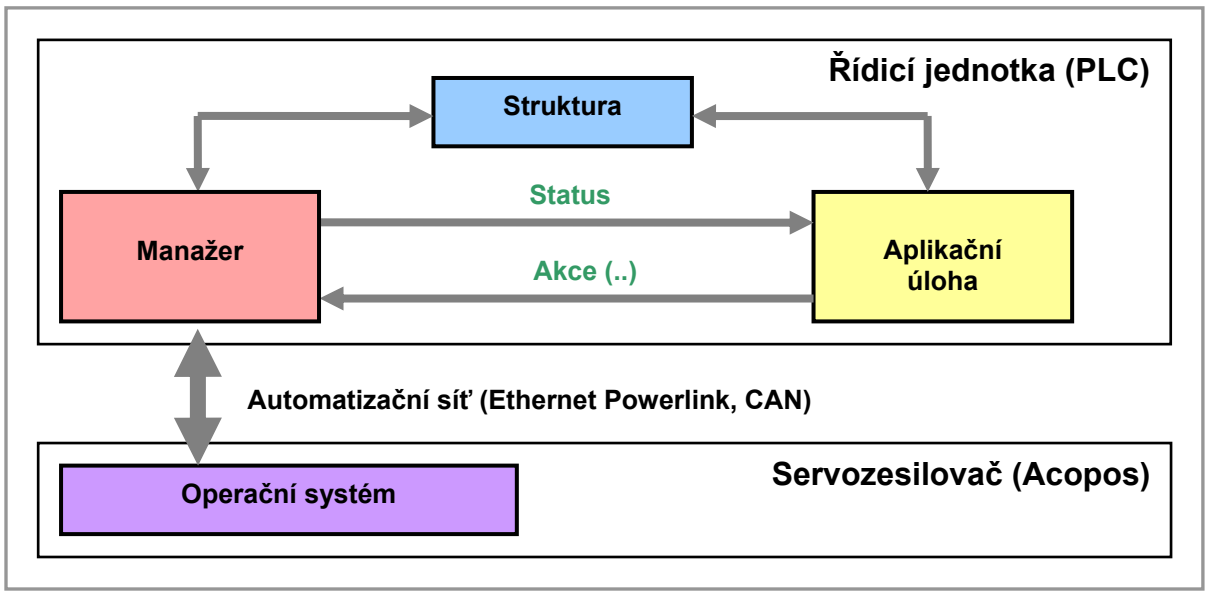
K řízení pohybu a nastavování polohy (motion control) se v průmyslové automatizaci využívá velmi mnoho systémů a softwarových řešení.

Servozesilovače firmy B+R podporují tyto možnosti softwarového řešení: řízení pomocí speciálních řídicích příkazů tzv. akcí (nc-action), řízení vačkovými automaty (CAM profile automat), které používají vačkové profily (CAM profile), ale také pomocí funkčních bloků mezinárodního standardu polohového řízení „PLCopen Motion Control“. Všechny tyto možnosti jsou součástí polohovacího softwaru typu ACP10.

Vlastní řízená osa (servozesilovač) zde vystupuje jako datová struktura, jejíž jednotlivé položky přesně popisují vlastnosti osy – od vstupních a výstupních signálů, přes nastavení vnitřních regulátorů až po okamžité hodnoty veškerých známých veličin (požadované i skutečné okamžité proudy, rychlosti, zrychlení, polohy, odchylky, teploty atd.).

4.1. Řízení pomocí akcí – „nc-actions“

Samotný proces polohového řízení pomocí akcí [7] (viz. obr. 4.1) se vykonává v aplikační úloze v nadřazeném řídicím systému (PLC).



Obr. 4.1 – Struktura řízení pomocí akcí

Řídicí (aplikační) úloha je použita k ovládání objektů (reálná a virtuální osa atd.) použitím datových struktur a akcí, které jsou poskytované manažerem. Manažer

řídicího systému komunikuje přes automatizační síť s operačním systémem servozesilovače.

Manažer má na starost posílání řídicích příkazů (akcí) z aplikační úlohy v PLC do servozesilovače, dále výměnu parametrizačních dat (parametry regulátoru, motoru, enkoderu a další), procesních dat (teplota, proud, výkon, rychlost a další) a nakonec získává informace o stavu servozesilovače (varování a poruchy).

Akce se používají k inicializaci servozesilovače, nastavení parametrů regulátoru, jeho zapínání, nastavení mezních hodnot, ovládání motorů pomocí základních pohybů atd.

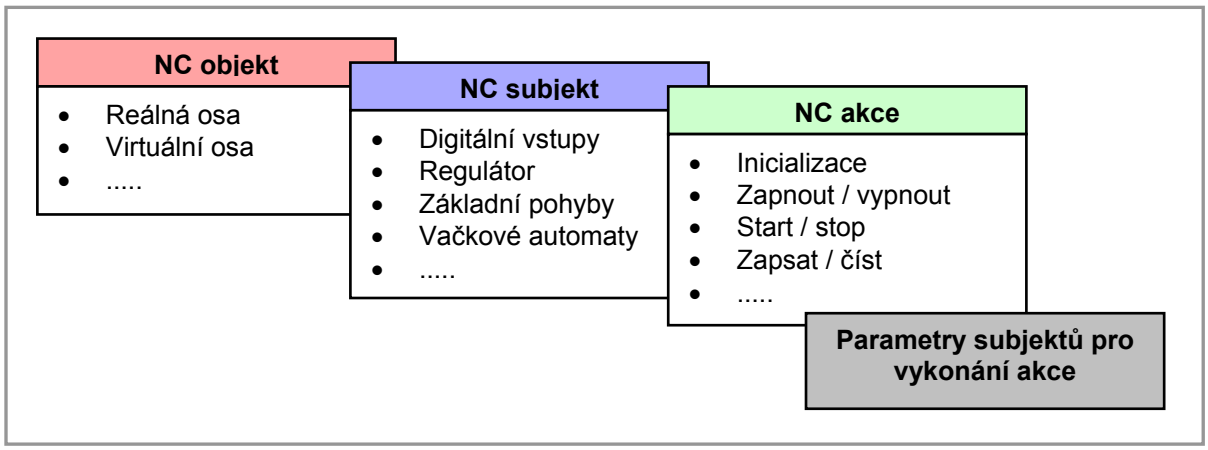
Operační systém servozesilovače má na starost: dosažení požadované hodnoty pohybů na základě získaných parametrů z řídicí jednotky, vyhodnocení aktuálních hodnot z enkoderu (celkový proud, teplota, aktuální poloha a rychlost a další) a nakonec ovládání řídicí logiky pro splnění požadovaných úkolů.

4.1.1. Datová struktura řízené osy

Objekty jsou datové typy, obsahující parametry přesně popisující druh řízené osy, která může být definována jako reálná osa, virtuální osa a další. Každý objekt je rozdělen do subjektů (digitální vstupy, základní pohyby, nastavení regulátoru, inicializace proměnných a další), ve kterých jsou uloženy jeho parametry dle jejich charakteru. Všechny subjekty obsahují určité akce.

Všechny objekty, subjekty, akce a parametry jsou definovány pomocí unikátních symbolických jmen.

Přehled struktury objektů, subjektů a akcí je na obr. 4.2.



Obr. 4.2 – Datová struktura

Význam a deklarace vybraných objektů jsou popsány v tab. 4.1.

Objekt	*Symbolické jméno*	*Datový typ*	*Popis*
Skutečná osa	ncAxis	ACP10axis	Používá se při skutečném řízení servozesilovače
Virtuální osa	ncV_axis	…	Používá se pro synchronizaci pohybu více servozesilovačů (tj. řídicí osa)
…	…	…	…

Tab. 4.1 – Výpis objektů

Přesný obsah jednotlivých datových struktur závisí na typu objektu, protože skutečná osa vyžaduje více informací než například virtuální osa.

Význam a deklarace vybraných subjektů jsou popsány v tab. 4.2.

Subjekt	*Symbol. jméno*	*Popis*
Globální	ncGLOBAL	Globální inicializace všech definovaných parametrů
Digitální vstupy	ncDIG_IN	Parametr, na jehož základě dojde k vyhodnocení vstupního napětí (0 nebo 24 V DC) a podle toho nastavení vstupu do log 0 nebo 1
Limitní hodnoty	ncLIMITS	Meze pro pohyb motorů (maximální, minimální rychlost a zrychlení, softwarové konce v positivním nebo negativním směru atd.)
Regulátor	ncCONTROLLER	Nastavení parametrů regulátorů servozesilovače (integrační, derivační složka, zpoždění atd.)
Počáteční poloha	ncHOMING	Nastavení motoru do počáteční polohy
Základní pohyby	ncPOS_MOVE	Pohyb v positivním směru otáčení motoru
	ncABS_MOVE	Pohyb na danou pozici
	ncREL_MOVE	Pohyb o danou vzdálenost
Vačkové profily	ncCAM_PROF	Nastavení parametrů vačkových profilů
Vačkové automaty	ncAUTOMAT	Nastavení parametrů vačkového automatu a jeho ovládání
…	…	…

Tab. 4.2 – Výpis subjektů

Význam a deklarace vybraných akcí je popsán v tab. 4.3.

Akce	*Symbol. jméno*	*Popis*
Inicializace	ncINIT	Inicializace nastavených parametrů subjektu
Zapnutí	ncSWITCH_ON	Zapnutí (regulátoru, simulace atd.)
Vypnutí	ncSWITCH_OFF	Vypnutí (regulátoru, simulace atd.)
Start	ncSTART	Spuštění (pohybu atd.)
…	…	…

Tab. 4.3 – Výpis akcí

Podrobný popis všech parametrů od významu po nastavení je popsán v *Position Introduction* [7].

4.1.2. Řídicí funkce

Polohové procedury jsou naprogramovány v aplikační úloze. Procesní a parametrizační data jsou definována ve struktuře (nebo se dají předefinovat přímo v úloze) a následně jsou poslána do servozesilovače příslušným příkazem (akcí).

Příkazy pro řízení servozesilovače jsou spouštěny v aplikační úloze pomocí funkce *„ncaction(objekt, subjekt, akce)“*. Předtím, než budeme volat akce, je potřeba vytvořit paměťové místo pro objekt v manažeru pomocí funkce *„ncalloc(..)* nebo *ncaccess(..)“.*

Pro obě funkce definujeme její vstupní parametry a po provedení každé funkce dostáváme zpětné hlášení pomocí výstupních parametrů.

4.1.2.1. Funkce ncalloc a ncaccess

Funkce ncalloc() a ncaccess() vytvoří v manažeru paměťové místo pro objekt (servozesilovač). Ncalloc se používá pokud je servozesilovač definován v hardwarovém stromě a ncaccess pokud je definován ve speciálním objektu „Deployment Table“. Po tomto vytvoření jsou objekty periodicky ovládány pomocí manažeru. Z výkonových důvodů je vhodné funkce volat v inicializační části aplikační úlohy. Popis parametrů a způsob volání funkcí je v tab. 4.4 a 4.5.

<table>
<tr><th colspan="5">Volání funkce:</th></tr>
<tr><td colspan="5">status = ncalloc(komunikace, adresa, typ, kanál, objekt)</td></tr>
<tr><th colspan="5">Vstupní parametry:</th></tr>
<tr><th>Proměnná</th><th>Typ</th><th colspan="3">Popis</th></tr>
<tr><td rowspan="4">Komunikace</td><td rowspan="4">UINT</td><td colspan="3">Podle typu komunikace, zápis ve tvaru ncACP10MAN + konstanta</td></tr>
<tr><td>Typ</td><td>Konstanta</td><td>Hodnota</td></tr>
<tr><td>CAN</td><td>ncCAN_IF</td><td>0 x 00</td></tr>
<tr><td>Power Link</td><td>ncPOWERLINK_IF</td><td>0 x 01</td></tr>
<tr><td>Adresa</td><td>UINT</td><td colspan="3">Adresa modulu</td></tr>
<tr><td>Typ objektu</td><td>UINT</td><td colspan="3">ID typu objektu, pro objekt: ncAXIS = 1</td></tr>
<tr><td>Kanál</td><td>UINT</td><td colspan="3">Číslo kanálu, pro objekt ncAxis = 1</td></tr>
<tr><th colspan="5">Výstupní parametry:</th></tr>
<tr><td>Objekt</td><td>UDINT</td><td colspan="3">Objekt (Ukazatel na strukturu)</td></tr>
<tr><td>Status</td><td>UINT</td><td colspan="3">Status funkce ncOK pokud se funkce provedla úspěšně nebo číslo poruchy</td></tr>
</table>

Tab. 4.4 – Funkce ncalloc

Příklad volání funkce:

```
status=ncalloc(ncACP10MAN+ncPOWERLINK_IF,3,ncAXIS,1,(UDINT)&ax_obj);
```

Volání funkce:		
status = ncaccess(sw_id, jméno, objekt)		
Vstupní parametry:		
Proměnná	**Typ**	**Popis**
Sw_id	UINT	Softwarové ID - ncACP10MAN
Jméno	UINT	Jméno objektu definované v Deployment table
Výstupní parametry:		
Objekt	UDINT	Objekt (Ukazatel na strukturu)
Status	UINT	Status funkce ncOK pokud se funkce provedla úspěšně nebo číslo poruchy

Tab. 4.5 – Funkce ncaccess

Příklad volání funkce:
status=ncaccess1 (ncACP10MAN,"osa1",&ax_obj);

4.1.2.2. Funkce ncaction

Funkce ncaction() slouží k volání akcí pro subjekty daného objektu. Popis parametrů a způsob volání funkce je v tab. 4.6.

Volání funkce:		
status = ncaction(objekt, subjekt, akce)		
Vstupní parametry:		
Proměnná	**Typ**	**Popis**
Objekt	UDINT	Objekt z funkce ncalloc() nebo ncaccess
Subjekt	UINT	Subjekt (regulátor, pohyby atd.)
Akce	UINT	Požadovaná akce (zapni, zastav atd.)
Výstupní parametry:		
Status	UINT	Status funkce (stav nebo číslo poruchy)

Tab. 4.6 – Funkce ncaction

Pro použití funkce ncaction je třeba věnovat pozornost následujícím bodům:

1. Akce je poslána do manažeru úspěšně, jestliže se vrátí potvrzení „status == ncOK". Pokud se tak nestane, neměla by aplikace povolit další krok.
2. Status == ncACTIVE znamená, že akce (úloha) z manažeru je vykonávána servozesilovačem. V takovém případě by aplikace měla zůstávat v tomto kroku.
3. Všechny další hodnoty pro status signalizují jednu z možných chyb.
4. Status == ncOK znamená, že příkaz byl úspěšně poslán do manažeru, ale neznamená to, že příkaz byl poslaný do servozesilovače, nebo že příkaz byl vykonaný.

Příklad zapnutí regulátoru:
Volaná funkce: status = ncaction(ax_obj,ncCONTROLLER,ncSWITCH_ON)
Status: status = ncOK

4.1.3. Manažer

Manažer představuje rozšířený operační systém, který vytváří spojení mezi řídicí aplikací běžící v PLC a operačním systémem servozesilovače. Pro řízení servozesilovače používá systémové objekty (acp10man – manažer a acp10sys1 – operační systém), které jsou dostupné v PLC projektu. Manažer dále obsahuje technickou specifickou část standardních funkcí ncalloc() a ncaction(). Při bootování manažer inicializuje oblast paměti obsahující struktury všech objektů.

Během cyklické operace manažer ovládá následující práce: přenos parametrizačních dat a příkazů do servozesilovače, čtení stavových dat ze servozesilovače a jejich zápis do struktury a přenos operačního systému do servozesilovače.

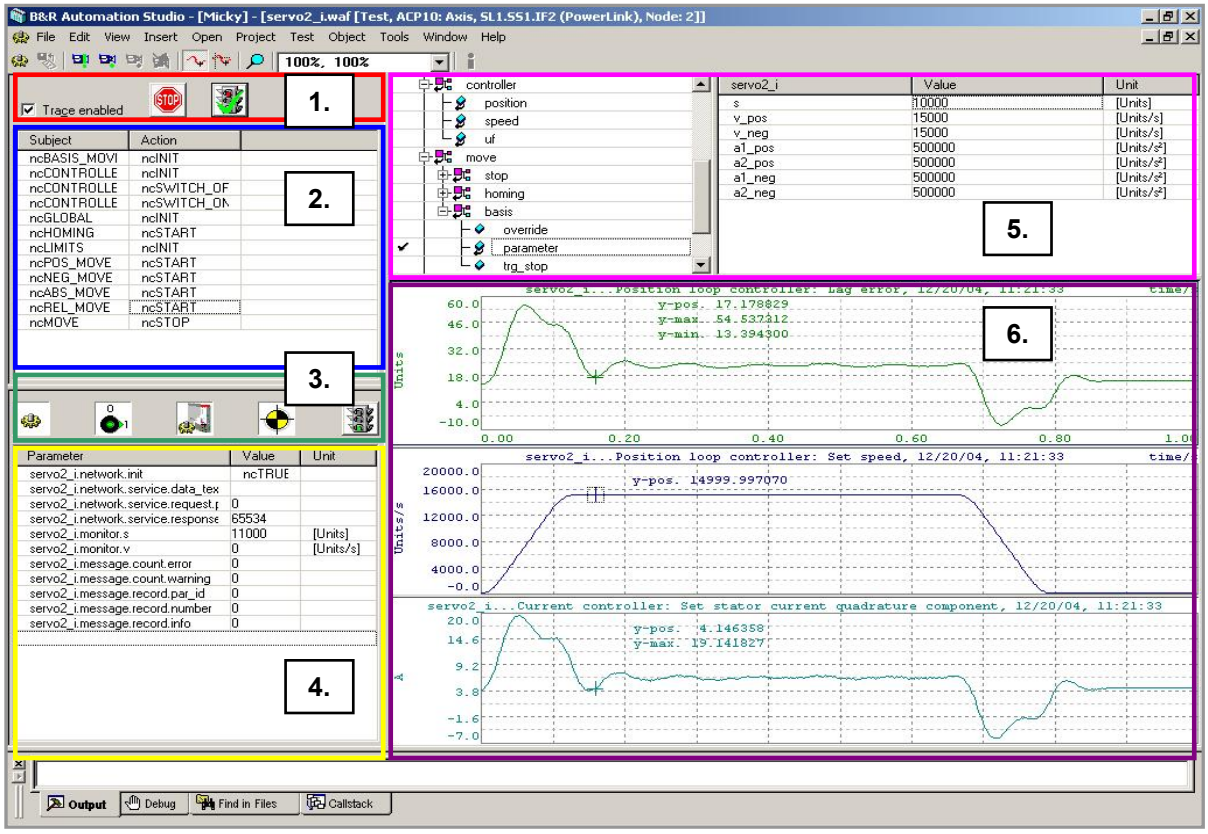
Dále zpracovává objekty v třídě úloh, která je definována v konfiguraci jako „Třída úloh pro manažer“. Proto manažer instaluje oddělenou úlohu, která je volaná na začátku této třídy.

4.1.4. Ladicí prostředí Test

Test je speciální prostředí, které slouží k nastavení parametrů servozesilovače, odzkoušení jednotlivých příkazů tzv. akcí (nastavení regulátoru, pohyby a další) a získávání aktuálních informací (poruchy, stav spojení a další). Dále umožňuje pomocí funkce trace zobrazit až 10 různých průběhů. Data můžeme snímat neustále, po začátku nebo ukončení pohybu. K lepšímu vyhodnocení slouží funkce lupy, měřicí kurzory a možnost zobrazení průměrných, maximálních a dalších hodnot.

Okno ladicího prostředí „Test“ můžeme rozdělit do několika část,(viz. obr.4.3):

1. Trace enable (povolení snímání dat pro každou vybranou akci), tlačítko zastavení pohybu a tlačítko kvitování poruchy
2. Okno pro vložení a volání základních akcí
3. Ikony signalizující stav servozesilovače – zleva: směr pohybu (positivní/negativní), stav regulátoru (zapnut/vypnut), simulační režim (zapnut/vypnut), zda bylo úspěšně provedeno nastavení motoru do počáteční polohy (homing) a zda obsahuje poruchu (po kliknutí na ni se zobrazí popis aktuální poruchy)
4. Vybraná monitorovaná data (stav komunikace, aktuální rychlost a další)
5. Struktura parametrů ovládané osy (tzv. struktura objektu)
6. Okno vybraných průběhů (lag error, žádaná rychlost, žádaný proud a další, např. aktuální poloha atd.)



Obr. 4.3 – Prostředí Test

4.2. Řízení pomocí vačkového automatu – „CAM profile automat“

Vačkový profilový automat (viz. obr. 4.4) je sekvenční automat s parametrovatelným nastavením, které umožní elektronicky propojit několik mechanických os. Vzájemné závislosti mezi osami se definují pomocí vhodně volených polynomů (neboli profilů). Ty mohou být statické, nebo se mohou počítat až za běhu stroje. Lze tak např. realizovat tyto technologické funkce: elektronická převodovka, elektronická kompenzační převodovka, letmý střih, definovaný vačkový profil a další.

Automat je reprezentován datovou strukturou, která obsahuje potřebné parametry pro řízení osy (servozesilovače a motoru).

Kromě reálné osy, jejíž výstup jde přímo do motoru, může v servozesilovači běžet ještě virtuální osa, která je funkčně rovnocenná, jen její výstup lze libovolně využít. Obě osy lze libovolně spojovat a to jak v rámci jednoho servozesilovače, tak i mezi více servozesilovači.

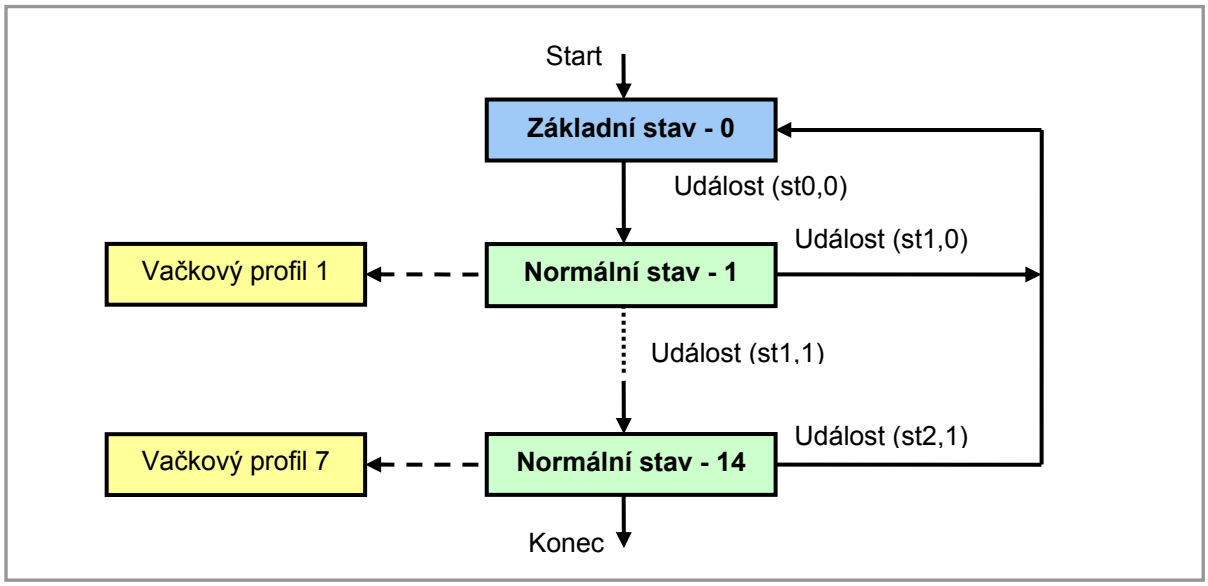
Vačkový automat se může skládat maximálně z patnácti stavů. Obsahuje jeden základní a 14 normálních stavů, ve kterých lze definovat vačkové profily.

Kromě několika předdefinovaných profilů (jako je např. závislost 1:1), lze do servozesilovače nahrát až 10 uživatelských vačkových profilů. Můžeme tedy mít některé stavy s pevně danými uživatelskými profily nebo s polynomy, které se spočítají sami podle aktuálních dat, když se servozesilovač do tohoto stavu dostane

V některých aplikací však není nutno tyto profily vůbec používat, a lze si vystačit pouze s předdefinovaným profilem 1:1. Jiný profil můžeme nastavit, pokud požadujeme, aby slave byl dvakrát rychlejší než master. Dále můžeme použít vhodně zvolené kompenzační převodovky, které se počítají dynamicky za běhu stroje.

Pro vačkový automat je potřeba nadefinovat jednotlivé stavy, pro které definujeme parametry pro master a slave osu, události pro přechod mezi jednotlivými stavy, kompenzační parametry, typ vačkového profilu, který se bude vykonávat v aktuálním stavu, následující stav a další.

Pro každý stav můžeme nadefinovat několik odlišných událostí, kde každá specifikuje určitý přechod do dalšího stavu. Pokud je aktivní nějaká událost a není definovaná v aktuálním stavu je ignorována.



Obr. 4.4 – Vačkový profilový automat

Před spuštěním vačkového automatu je potřeba provést nahrání všech vačkových profilů, sekvence a parametrů automatu do servozesilovače pomocí *knihovních funkcí pro řízení servozesilovače* [5]. Pro prací s vačkovými automaty se používají akce ncCAM_PROF, ncDOWNLOAD a další.

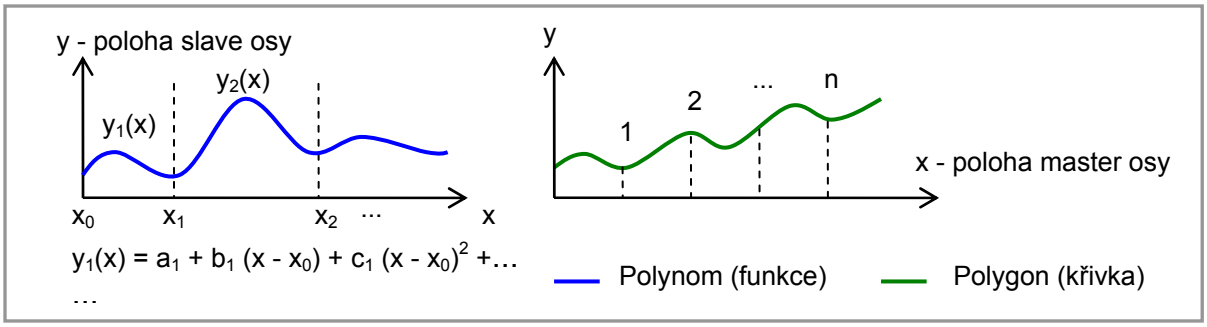
Řízení pomocí vačkového automatu a vačkových profilů neřeší nastavení regulátoru a dalších potřebných parametrů servozesilovače. K tomu je zapotřebí již dříve zmíněných akcí.

Během chodu vačkového automatu nelze měnit všechny jeho parametry, ale pouze některé. Po spuštění automatu běží vše v servozesilovači (viz. obr. 4.6) a není zatěžována řídící jednotka. To redukuje zatížení na uživatelském programu a dovolí velmi rychlé, událostmi řízené vačkové přechody z jednoho stavu do dalšího.

Řízení pohonů s využitím vačkového profilu je realizováno pomocí trajektorie pohybu, rychlosti a zrychlení pro každou osu (reálnou i virtuální).

Trajektorie (viz. obr. 4.5) může být definována pomocí funkce nebo křivky. Trajektorie definovaná pomocí funkce se může skládat z 64 částí, které mohou být maximálně šestého řádu ($y = a + b\,x + c\,x^2 + d\,x^3 + e\,x^4 + f\,x^5 + g\,x^6$). Trajektorie realizovaná křivkami se skládá ze stejně dlouhých úseků.

Při realizaci řízení pomocí více profilů, musí být přechod mezi jednotlivými stavy spojitý, jinak může dojit ke skokům pohybu a tím i k výpadku řízení vznikem poruchy.



Obr. 4.5 – Trajektorie vačkového profilu vytvořená pomocí křivky a funkce

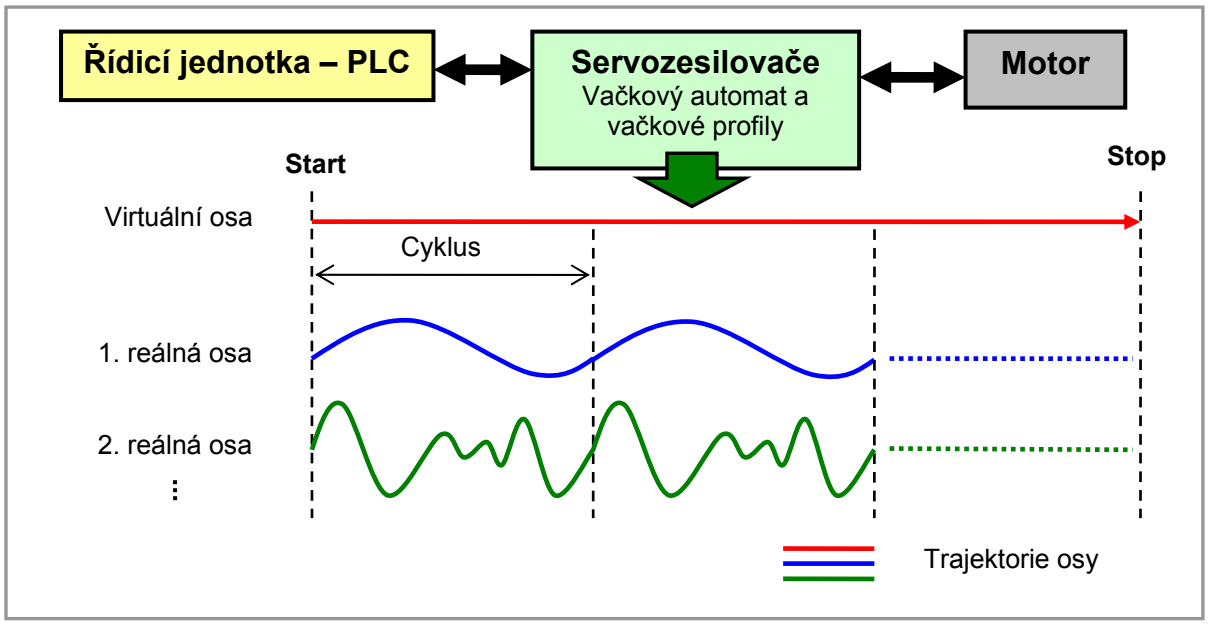
Jeden řízený motor pomocí servozesilovače představuje jednu reálnou osu. Jednotlivé slave osy jsou závislé na nadřazené master ose, která může být virtuální, ale i reálná. Je možné nadefinovat velké množství závislostí master-slave pro všechny osy, např. virtuální závislá na reálné a naopak, ale i reálná závislá na reálné.

Trajektorie (křivky) jsou v PLC reprezentovány speciálním objektem typu „NC CAM profile – ACP10“, tento objekt je zapotřebí vložit do aplikace. Trajektorie se vytváří pomocí vačkového editoru, který je součástí Automation Studia.

4.2.1. Řízení a synchronizování reálných os osou virtuální

V našem případě jsou reálné osy závislé na nadřazené virtuální master ose (viz. obr. 4.6), která reprezentuje čas. Reálné osy se potom synchronizují pomocí časového cyklu jejich trajektorií. Všechny nadefinované trajektorie reálných os se opakují stále dokola v definovaném časovém cyklu dokud je nezastavíme.

Při zastavení pohybů reálných os se musí zastavit i virtuální osa, jinak osy nebudou pracovat synchronně. Pokud zastavíme pouze jednu reálnou osu, a potom ji opět spustíme, bude tato osa posunuta o ztracený čas.



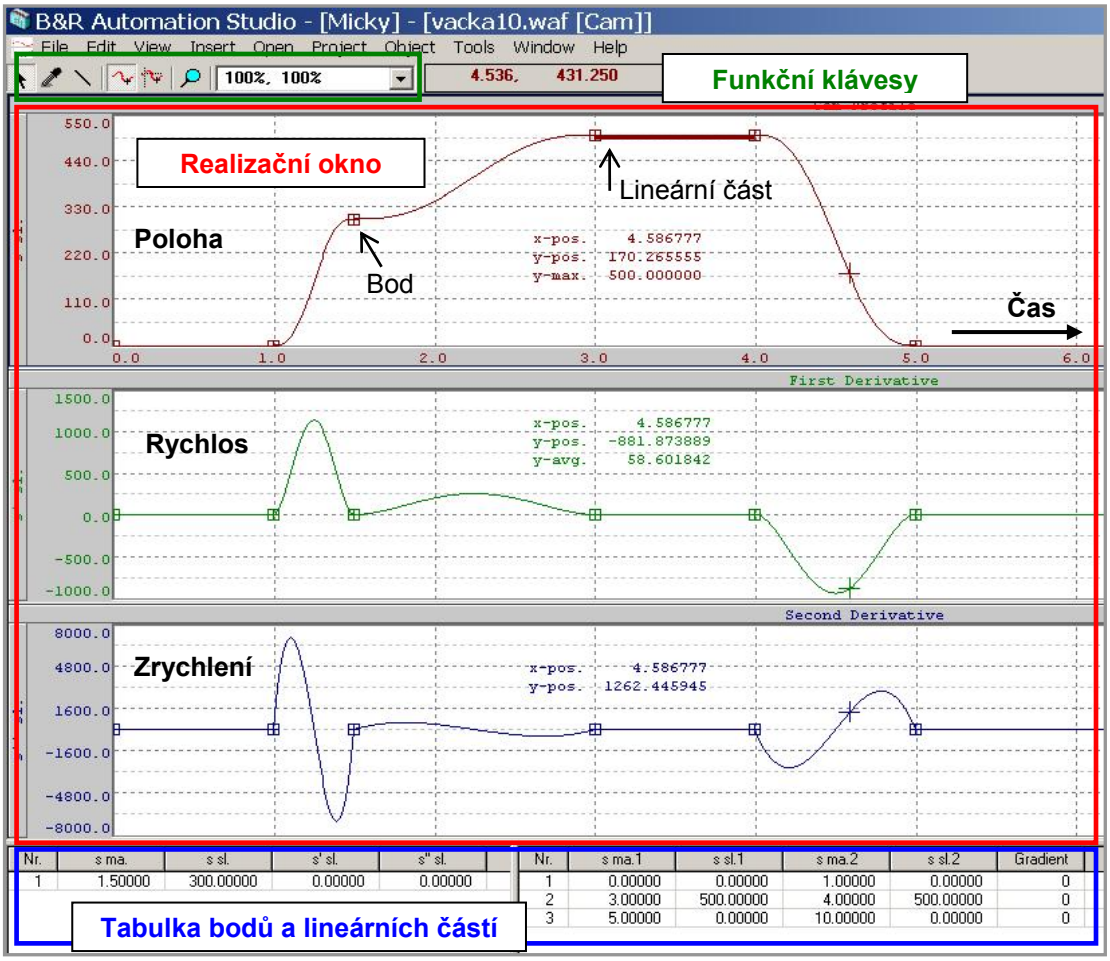
Obr. 4.6 – Řízení pomocí vačkových profilů

Virtuální osa nemusí mít nadefinovanou trajektorii. Jelikož se jedná o řídicí osu zajišťující součinnost ostatních reálných os, stačí ji jenom spustit pomocí akce positivního pohybu. Nastavení parametrů se provádí stejně jako pro strukturu reálné osy. Není–li virtuální osa v pohyby, nejsou v pohybu ani reálné osy.

4.2.2. Prostředí pro realizaci vačkových profilů

Vývojové prostředí pro vytváření vačkových profilů se skládá z několika částí (viz. obr. 4.7): titulek, menu, funkční klávesy (pohyb kurzoru po trajektorii, vložení bodu, vložení lineární části, zobrazení hodnot a odchylek a lupa), realizační okno (trajektorie polohy, rychlosti a zrychlení), okno dat (levá půlka obsahuje pozice bodů a pravá část pozice lineárních částí).

Trajektorie polohy se vytváří pomocí bodu nebo lineárních částí, rychlost a zrychlení se vytváří automaticky v závislosti na tom, jakou vzdálenost chceme vykonat v závislosti na čase. Při definování trajektorií reálných os je třeba dát pozor na časovou hodnotu jednoho dílku časové osy. Jeho velikost je závislá na rychlosti pohybu master osy (např. virtuální osa).



Obr. 4.7 – Prostředí pro realizaci vačkových profilů

Příklad:

Rychlost master osy (např. virtuální) je 1 unit/s, potom 1 dílek definované trajektorie slave osy (reálné) je 1 s. Bude–li rychlost 10 units/s potom 1 dílek bude 0,1 s.

4.3. Řízení pomocí funkčních bloků knihovny PLCopen

Standardizace v oblasti softwarových řešení pro řízení pohybu je důležitá pro snižování celkových nákladů na techniku pohonů. Organizace PLCopen [16] proto definovala knihovnu funkčních bloků pro řízení pohybu a na jejím základě mezinárodní standard PLCopen Motion Control Library, vycházející ze standardu pro programování průmyslových řídicích systémů IEC 61131-3. Standard PLCopen Motion Control Library je publikován a je volně přístupný.

Mezi jazyky definovanými standardem IEC 61131-3 (kontaktní schémata - LAD, strukturovaný text - ST, Automation Basic – AB nebo ANSI-C.) využívá princip knihovny objektů jazyk funkčních bloků (Function Blocks – FB). Tím, že poskytuje

skrytá a zapouzdřená data použitelná v nejrůznějších architekturách řízení, umožňuje vytvářet aplikační programy téměř nezávislé na hardwaru, a tudíž opakovaně použitelné na různých výpočetních základnách.

Úplná specifikace standardu PLCopen Motion Control Library obsahuje především vlastní knihovnu funkčních bloků pro řízení jedné osy a koordinované řízení několika os, stavový diagram řízené osy a příklady použití funkčních bloků při programování.

Jednotlivé funkční bloky jsou rozděleny do oblastí: administrativních funkcí, jednoosých a víceosých pohybů a speciálních technologických funkcí. PLCopen také umožňuje využití funkcí knihovny „Smart Process Technology“.

Administrativní funkční bloky se používají k ovládání řídicí jednotky servozesilovače, pro čtení chyb a parametrů, zápis dat atd.

Pohyb jednoosých zařízení pokryjí základní polohové řídicí funkce, kterými jsou nastavení do počáteční polohy, absolutní a relativní pohyb, ale také neomezený pohyb v daném směru a další. Pro víceosé řízení podporuje PLCopen funkce převodovky, vačkových profilů, fázového posunu a další. U těchto funkcí můžeme použít závislosti „master-slave“.

Technologické funkce (např. rotační nůžky) jsou poskytovány jako „open source“ a vychází ze základních funkčních bloků. Pokud uživateli nevyhovuje jejich funkce má možnost úpravy podle vlastní potřeby.

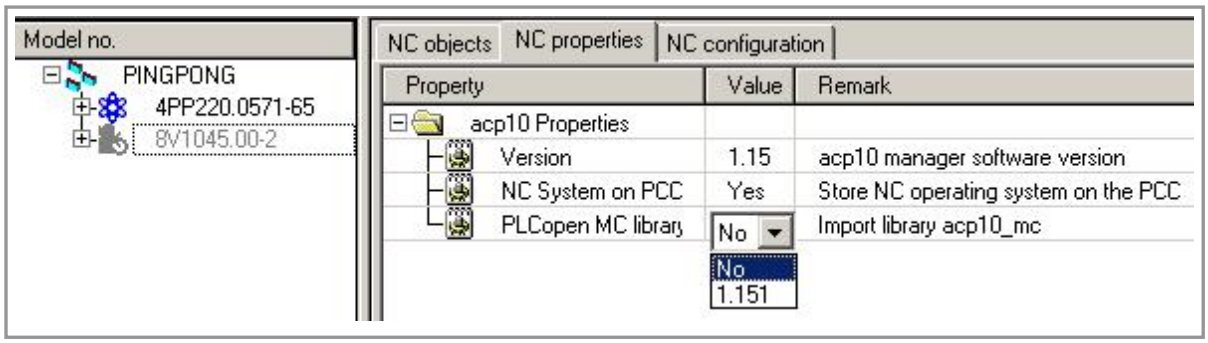
Smart Process Technology je volně konfigurovatelná knihovna technologických funkcí obsažená ve firmwaru polohových řídicích produktů (servozesilovačů). Její využití přináší vlivem krátkých reakčních dob až dvojnásobné zvýšení produktivity strojů. K přednostem funkcí obsažených v SPT patří zejména: velká rychlost a přesnost pohybu zásluhou synchronního zpracování a reakční doby 400 µs, možnost použití menšího počtu snímačů (popř. žádného) díky využití nepřímých technologických veličin, nové možnosti řízení výrobních procesů a další.

Funkční bloky obsahují vstupy a výstupy s přiřazenými názvy a definovanými typy dat. V každém funkčním bloku je obsažen kód určující a provádějící jeho funkci a promítající ji do odpovídajícího okolního prostředí (řízení pohybu).

4.3.1. Princip řízení pomocí PLCopen

Definice řízení pomocí funkčních bloků knihovny PLCopen se provádí při hardwarové konfiguraci servozesilovače v jeho parametrizační části (viz. obr. 4.8). Dále je zapotřebí v projektu vložit knihovnu funkcí PLCopen (acp10_mc) do seznamu knihoven. Na obr. 4.9 je ukázka zápisu funkce pohybu v definovaném směru (MoveVelocity) v jazyce ANSI-C.

Princip řízení je stejný jako při řízení servozesilovače pomocí akcí. Řídicí funkce knihovny PLCopen se odkazují na datovou strukturu řízené osy (servozesilovač) a volají se v aplikační úloze. Datovou strukturu osy je zapotřebí vytvořit pomocí funkce ncaccess nebo ncalloc (viz. kapitola 4.1.2).



Obr. 4.8 – Nastavení řízení pomocí PLCopen

```
AxisNcAccess = ncaccess(ncACP10MAN, "servo2", &pOsa);

MC_MoveVelocity.Axis         = pOsa;
MC_MoveVelocity.Execute      = Execute;
MC_MoveVelocity.Velocity     = Velocity;
MC_MoveVelocity.Acceleration = Acceleration;
MC_MoveVelocity.Deceleration = Deceleration;
MC_MoveVelocity.Direction    = Direction;
MC_MoveVelocity(&MC_MoveVelocity);
InVelocity       = MC_MoveVelocity.InVelocity;
Error            = MC_MoveVelocity.Error;
ErrorID          = MC  MoveVelocitv.ErrorID:
```

Obr. 4.9 – Ukázka funkce (PLCopen) pohybu v daném směru (MoveVelocity)

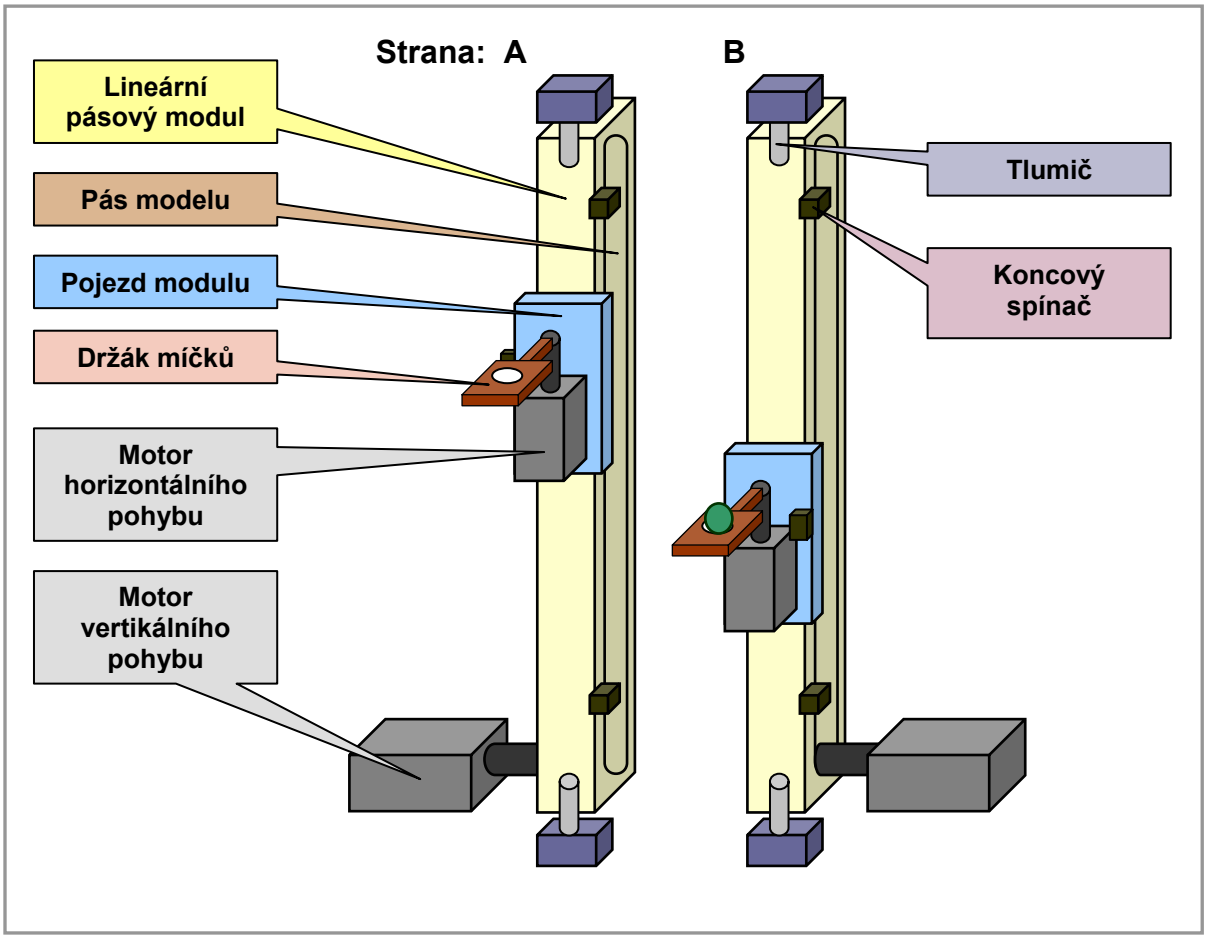
Řízení servozesilovače pomocí funkčních bloků knihovny PLCopen není možné používat ve stejném projektu jako řídicí příkazy tzv. akce (nc-actions).

5. REALIZOVANÝ MODEL

Kapitola popisuje realizovaný model, od jeho návrhu až po samotnou realizaci. Model jsme nazvali „Model míčků“ a je umístěn v laboratoři K09 na Katedře řídicí techniky FEL ČVUT Praha.

5.1. Řízený model

Na obr. 5.1 je zjednodušeně znázorněn realizovaný model. Fotografie skutečného modelu je v příloze 1.



Obr. 5.1 – Řízený model

K ocelové konstrukci jsou přichyceny dva lineární pásové moduly MLR 10-110 firmy Rexroth Bosch, které obsahují pojezd. Každý modul je poháněn jedním velkým synchronním motorem (B+R 8MSA5L). Tyto moduly zajišťují svislý pohyb „nahoru – dolů“. Na každém pojezdu je dále připevněn další menší motor (B+R 8MSA3L), na který je přidělán držák míčku. Pomocí těchto motorů je realizován horizontální pohyb „do stran“.

Při návrhu mechanické části modelu musíme zajistit dostatečný prostor pro rychlé pohyby (zrychlení, zpomalení a konstantní rychlost) k vyhození a chycení míčků. Také musíme zajistit bezpečný prostor pro zastavení pohybu při selhání řízení. Aktivní délka pro rychlé pohyby rychlostí 5 m/s se zrychlením 50 m/s^2 je 220 cm a pro bezpečné zastavení je 30 cm na každé straně modulu.

Pro zajištění bezpečného provozu je každý lineární modul osazen dvěma koncovými spínači (nahoře i dole). Při přejetí koncového spínače dochází pomocí servozesilovače k zastavení motorů. Jeden spínač je též umístěn na každém pojezdu.

Pro ochranu lineárního modulu před jeho zničením nárazem pojezdem, je na každé straně přidělán bezpečnostní tlumič (fy. ACE).

Při výpadku napájení 380 V by nemuselo dojít k zastavení motoru pomocí servozesilovače. Motor by se potom pohyboval nekontrolovaně.

Proto byly použity dva napájecí zdroje 24 V. První zdroj PS102 je napájen z jedné fáze 380 V a napájí pouze tlačítko rychlého vypnutí (quickstop) pohybu motorů pomocí servozesilovače, k povelu pro zastavení dojde pokud není napětí 380 V nebo pokud sepneme tlačítko. Druhý zdroj PS320 je napájen z DC sběrnice servozesilovačů a napájí PLC, řídicí svorkovnice (X1) všech servozesilovačů a ostatní spínače. Tento zdroj zajistí dostatečně dlouhou dodávku energie řídicím jednotkám a ostatním komponentám po výpadku 380 V a zajistí tak bezpečné zastavení motorů.

Zapojení modelu je realizováno tak, aby splňovalo požadavky řídicí kategorie 3 podle normy EN 954-1.

5.2. Struktura řízení modelu

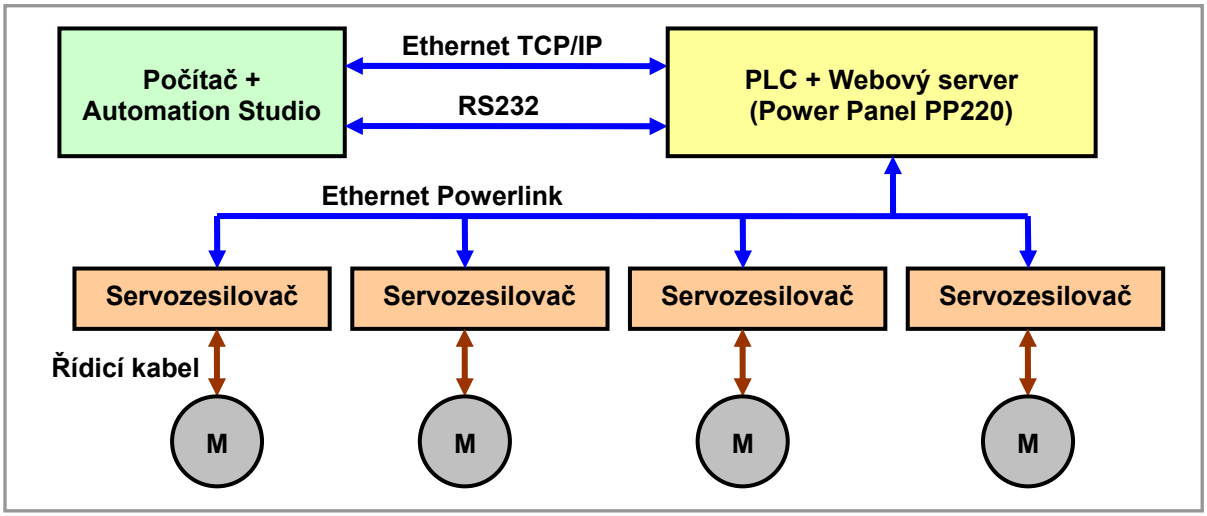
Polohové řízení modelu je realizováno pomocí synchronních motorů. Ovládání každého motoru zajišťuje servozesilovač (Acopos), který je řízen průmyslovým automatem (Power Panel PP220) s využitím komunikace Ethernet Powerlink.

Průmyslový automat (Power Panel PP220) ovládá servozesilovač pomocí speciálních řídicích příkazů tzv. akcí nebo pomocí definovaného vačkového profilu (viz. kapitola 4), který je volán na základě stavu vačkového automatu. Přenos dat je realizován pomocí real–time komunikace Ethernet Powerlink (viz. kapitola 3.2). Servozesilovač ovládá motor pomocí pulsně šířkové modulace napájecího napětí motoru se základní frekvencí 20 kHz.

Komunikace mezi počítačem a PLC je realizována pomocí komunikací RS232 a Ethernet TCP/IP. Pomocí těchto komunikací můžeme provést upload/download

programu z/do PLC, dále můžeme pomocí Automation Studia monitorovat, vyhodnocovat a ladit náš program. V poslední řadě můžeme i model ovládat z webových stránek.

Strukturu polohového řízení a ovládání modelu můžeme popsat dle obr. 5.2.



Obr. 5.2 – Struktura řízení a komunikace modelu

5.3. Komponenty modelu

5.3.1. PLC – Power Panel PP220

Pro řízení modelu a vizualizaci dat bylo použito PLC - Power Panel PP220 [8] (viz. obr. 5.3). Power Panel se skládá z řídicí jednotky (procesor, paměť a další) rozšiřitelné o speciální moduly a z dotykové obrazovky. Programování PLC se provádí pomocí Automation Studia (viz. kapitola 2.1). Základní parametry PLC jsou uvedeny v tab. 5.1.

Pro komunikaci se servozesilovači byl do slotu PLC vložen komunikační modul Ethernet Powerlink IF786 (viz. příloha 7).

Obr. 5.3 – PLC (Power Panel PP220)

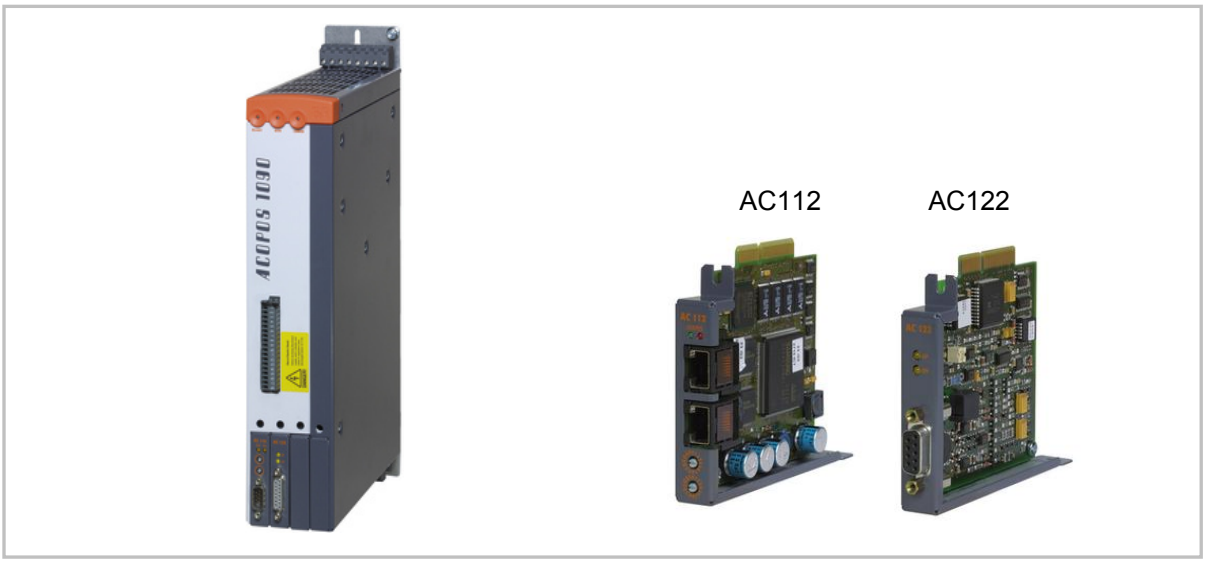
Typ	4PP220.0571-65 Rev.K0
Napájení	24 VDC
Procesor	Geode SC2200 266 MHz, MMX kompatibilní
Paměti	Flash 2MB, hlavní paměť 64MB DRam, sdílené 4MB grafické paměti
Porty	Ethernet 10/100MB, RS232, 2*USB, volný slot pro komunikační moduly (CAN, Ethernet Powerlink a další)
Obrazovka	LCD, 256 barev, QVGA 320*240pixels, 5,7in

Tab. 5.1 – Parametry PLC

5.3.2. Servozesilovače – 1022 a 1090

Pro ovládání motorů byly použity dva typy servozesilovačů (viz. obr. 5.4). Pro motor 8MSA3L je potřeba servozesilovač typu 1022 a pro motor 8MSA5L typ 1090. Jednotlivé servozesilovače se liší maximálním výkonem, kterým mohou ovládat motor. Důležité parametry jsou popsány v tab. 5.2.

Všechny servozesilovače obsahují ve svých slotech dva komunikační moduly (viz. příloha 8 a 9), první pro komunikaci Ethernet Powerlink (AC112) a druhý pro získání aktuální pozice z motoru (Rezolver - AC122).



Obr. 5.4 – Servozesilovač a jeho komunikační moduly

Typ	***Výkonové hlavní vedení***			***Řízení motoru***	
	Napětí [V]	***Startovací proud [A]***	***Max. výkon [kVA]***	***Proud fáze [A]***	***Špičkový proud [A]***
1022	3*380	4	3	2,2	14
1090		7	10	8,8	24

Tab. 5.2 – Parametry servozesilovačů

5.3.3. Synchronní motory

Pro ovládání modelu byly vybrány dva typy třífázových synchronních motorů (viz. obr. 5.5). Ty se používají pro aplikace, které požadují znamenité dynamické vlastnosti, precizní řízení polohy, ale i minimální velikost a hmotnost motorů. Porovnání charakteristik závislosti momentu na rychlosti otáčení jednotlivých typů motorů je v příloze 10.

Obr. 5.5 – Synchronní motor

Pro vertikální pohyb byly vybrány dva motory typu 8MSA5L opatřeny elektrickou brzdou a pro horizontální pohyb dva motory typu 8MSA3L bez elektrické brzdy. Důležité parametry obou motorů jsou uvedeny v tab. 5.3.

Jako zpětná vazba u každého motoru pro získání informací o aktuální pozici je použit enkoder typu rezolver.

Typ motoru	*Rychlost [1/min]*		*Moment [Nm]*		*Výkon [kW]*	*Hmotnost [kg]*	*Servo–zesilovač*
	Jmen.	Max.	Jmen.	Max.	Jmen.		
8MSA3L.R0-30	3000	12000	2,15	10	0,68	3,2	1022
8MSA5L.R0-B6	3000	9000	11	40,5	3,46	12,1	1090

Jmen. – jmenovitý, Max. – maximální

Tab. 5.3 – Parametry motorů

5.3.3.1. Výpočet parametrů motoru vertikálního pohybu

Data pro výpočet parametrů motoru vertikálního pohybu (viz. tab.5.4):

Hmotnost vertikální pohyblivé hmoty	m	15,6 kg
Maximální požadované zrychlení	a	50 m/s^2
Maximální požadovaná rychlost	v	5 m/s
Průměr řemenu lineárního modulu MLR10–110	d	92 mm

Tab. 5.4 – Zadané parametry pro výpočet motoru vertikálního pohybu

Výpočet A:

Maximální možná síla v řemenu (F_{Max}) musí být menší než maximální síla přípustná pro lineární modul MLR10–110 ($F_{MaxModulu}$):

$$F_{Max} = 1{,}2 * m * a = 1{,}2 * 15{,}6 * 50 = 936N$$
$$\left(F_{Max} = 936N\right) < \left(F_{MaxModulu} = 1740N\right)$$

Maximální krouticí moment (M_{KMax}) nutný pro realizaci pohybu musí být menší než maximální přípustný moment pro lineární modul MLR10–110 ($M_{KMaxModulu}$):

$$M_K = F * r = m * a * r = 780 * 0.046 = 35{,}9$$
$$M_{KMax} = 1{,}2 * M_K = 43Nm$$
$$\left(M_{KMax} = 43N\right) < \left(M_{KMaxModulu} = 80N\right)$$

Doba chodu motoru při rozběhu hmot:

$$v = a * t \quad t = v/a = 5/50 = 0{,}1s$$

Maximální nutné otáčky motoru:

$$n = v/s = v/\pi * D = v/\pi * 0{,}092 = 17{,}3s^{-1} = 1038\,\mathrm{min}^{-1}$$
$$t = v/a = 5/50 = 0{,}1s$$

Úhlová rychlost motoru:

$$\varepsilon = 2\pi * n/t = 2\pi * 17{,}3/0{,}1 = 1086s^{-2}\left(rad\right)$$

Výpočet B:

Přibližná kinetická energie (W_K) nutná pro urychlení posuvných hmot během jedné otáčky a krouticí moment (M_K) na hřídeli motoru:

$$W_K \doteq 0{,}5 * m * v^2 = 0{,}5 * 15{,}6 * 5^2 = 195J$$
$$M_K = W_K/2\pi = 195/2\pi = 31Nm$$

Kinetická energie na konci rozběhu (W_{Kr}) a moment nutný k rozběhu motoru – rotoru (M_{Kr}) během 1 otáčky na rychlost n=17,3s^{-1}:

$$W_K \doteq 0{,}5 * \varpi^2 * I_{Servopohonu+Spojky} = 0{,}5 * \left(2\pi * 17{,}3\right) * 0{,}005 = 29{,}5J$$
$$M_{Kr} = W_K/2\pi = 29{,}5/2\pi = 4{,}7Nm$$

Celkový moment (M_{KC}), který je nutno vyvinout je potom přibližně:

$$M_{KC} = M_K + M_{Kr} = 31 + 4{,}7 = 35{,}7Nm$$

Uvedené výpočty A a B jsou dostačující pro návrh motoru z hlediska jeho konstrukce. Obě vypočtené hodnoty nepřesahují povolená maxima pro lineární modul.

Je nutno počítat s motorem schopným rozběhu z n = 0 na n = 1038 min^{-1} za čas t = 0,1 s. Po celou dobu rozběhu je požadován krouticí moment M_{KMax} = 43 Nm. Konkrétní typ motoru je nutno volit s ohledem na jeho výkonovou charakteristiku a moment setrvačnosti.

Z hlediska pevnosti hřídele lineárního modulu je nutné volit hřídel bez drážky pro pero.

5.3.3.2. Výpočet parametrů motoru otočného ramene

Data pro výpočet parametrů motoru horizontálního pohybu (viz. tab.5.5):

Úhlová rychlost	ε	155 s^{-2}
Maximální hmotnost koule	m	0,5 kg
Poloměr ramene držáku koule	r	0,15 m

Tab. 5.5 – Zadané parametry pro výpočet motoru horizontálního pohybu

Výpočet:

Moment setrvačnosti (I) koule na rameni:

$$I = m * r^2 = 0{,}5 * 0{,}15^2 = 0{,}011 kgm^2$$

Celkový moment setrvačnosti (hřídel, spojka) lze odhadnout dle konstrukčního uspořádání na cca.:

$$I_{Max} = 2 * I = 2 * 0{,}011 = 0{,}022 kgm^2$$

Potom krouticí moment pro rozběh rotujících hmot je:

$$M_{KMax} = I_{Max} * \varepsilon = 0{,}022 * 155 = 3{,}41 Nm$$
$$\left(M_{KMax} = 3{,}41\right) < \left(M_{KMaxMotor} = 10\right)$$

Maximální krouticí moment (M_{KMax}) nutný pro realizaci pohybu musí být menší jak maximální moment pohonu horizontálního pohybu ($M_{KMaxMotor}$).

Krouticí moment motoru bude dle konstrukce a realizace otočného ramene a hmotnosti koule maximálně 3,41 Nm. S tímto momentem je nutno počítat již při rozběhu motoru z n = 0 na n = max.

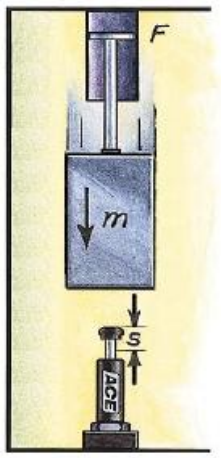
5.3.4. Tlumič

Pro ochranu lineárního modulu před zničením nárazem, byl použit tlumič SCS33-25 firmy ACE (viz. obr. 5.6), který jsem vybral na základě spočítaných parametrů a konzultaci s firmou BIBUS s.r.o, která tlumič dodala a provedla kontrolní výpočet pomocí programu poskytovaného výrobcem.

Obr. 5.6 – Tlumič

5.3.4.1. Výpočet tlumiče

Pro výpočet typu tlumiče je potřeba znát základní údaje (viz. tab. 5.6)



Zadané parametry		
Hmotnost	m	16 kg
Síla	F	800 N
Rychlost	v	5 m/s
Počet nárazů	c	1 1/h
Zdvih tlumiče (dle katalogu)	s	0,025 m
Spočítané parametry (dle vzorců / program ACE)		
Kinetická energie	W_1	200 /200 Nm
Poháněcí síla	W_2	24 / 22 Nm
Celková energie	W_3	224 / 222 Nm
Celková energie za čas	W_4	224 / 222 Nm/h
Efektivní hmotnost	m_e	17,92 / 17,8 kg

Tab. 5.6 – Výpočet parametrů tlumiče

Vzorce pro výpočet parametrů tlumiče:

Kinetická energie:

$$W_1 = \tfrac{1}{2} m v^2 = \tfrac{1}{2} * 16 * 5^2 = 200 Nm$$

Poháněcí síla:

$$W_2 = (F + mg) * s = (800 + 160) * 0.025 = 24 Nm$$

Celková energie

$$W_3 = W_1 + W_2 = 200 + 24 = 224 Nm$$

Celková energie za čas

$$W_4 = W_3 * c = 224 * 1 = 224 Nm / h$$

Efektivní hmotnost

$$m_e = {2 * W_3}/{v^2} = {2 * 224}/{5^2} = 17{,}92 kg$$

5.3.5. Napájení

Pro napájení servozesilovačů byl použit zdroj 3*380 V jištěný 32 A pojistkou vyvedený ze stolového rozvaděče.

Jedna fáze napájí zdroj napětí PS102 (100-240VAC/24VAC,2A), který napájí pouze tlačítko rychlého vypnutí pohybu motorů.

Pro napájení řídicí jednotky servozesilovačů, PLC a spínačů byl vybrán třífázový zdroj PS320 [9] (400-500VDC/24VAC,20A), který je napájen z DC sběrnice servozesilovačů a dostatečně pokryje potřebný odběr. Tento zdroj zajistí dostatečně dlouhou dodávku energie řídicím jednotkám po výpadku 380 V.

DC sběrnice všech servozesilovačů je propojená (viz. příloha 11) a umožňuje využití brzdící a hnací energie jednotlivých motorů. Vzniklá energie se vrací do sběrnice a externí napájecí zdroj ji může využít k vytvoření 24 VDC.

Obr. 5.7 – Napájecí zdroje

5.3.5.1. Výpočet proudového odběru ze zdroje 24 V

K výpočtu proudového odběru je zapotřebí znát odběr jednotlivých komponent modelu. Přehled odběru všech komponent je v tab. 5.7.

Výpočet potřebného proudu pro servozesilovač:

$$I_{24VDC}[A] = I_{24VDC\max}[A] - \frac{1.1}{24V * k}\left(22W - \sum P_{MODULU}[W]\right)$$

$$P_{MODULU} = P_{AC112} + P_{AC122} = 2.5 + 1.2$$

$$I_{24VDC}[A] = 2.5 - \frac{1.1}{24 * 0.64}\left[22 - (2.5 + 1.2)\right] = 1.19A$$

Hodnoty $I_{24VDC\max}$ a k jsou odlišné pro různé typy servozesilovačů, ale pro typ 1022 a 1090 jsou shodné. P_{AC112} a P_{AC122} je výkon komunikačních modulů servozesilovače (Ethernet Powerlink modul a rezolver modul).

Zařízení	*Max. proud [A]*	*Počet*	*Celkem [A]*
Servozesilovač	1.19	4	4.76
PLC	0.7	1	0.7
Elektrická brzda motoru	1	2	2
Maximální proudový odběr všech komponent			**7.46**

Tab. 5.7 – Proudový odběr modulů ze zdroje 24 V

Vypočtený maximální proud potřebný pro komponenty modelu je 7.46 A. Při chodu všech řídicích komponent byl naměřen skutečný odběr proudu 5.8 A.

5.3.5.2. Proudový odběr ze zdroje 380 V

Proudový odběr motorů je závislý na aktuálním stavu každého motoru. Záleží na tom, zda motor vykonává pohyb nebo je v klidu. Též záleží, jestli se motor rozjíždí nebo se otáčí konstantní rychlostí. Přehled proudového odběru jednotlivých motorů je v tab. 5.8.

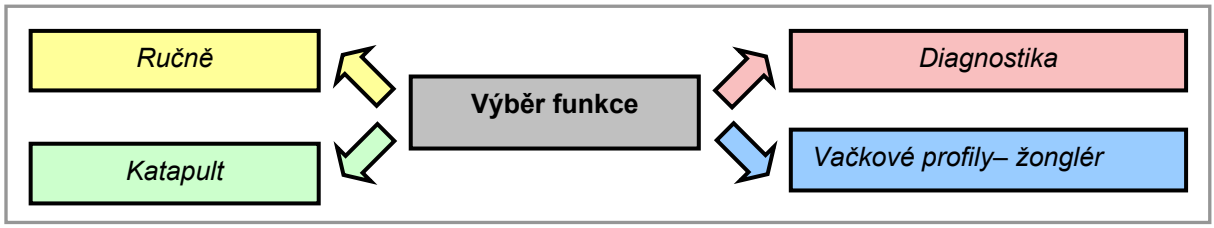
Motor	*Proud [A]*		
	Jmenovitý	*Mezní*	*Maximální pulsní*
8MSA3L	1,62	1,8	7,7
8MSA5L	7,3	8,68	43,2

Tab. 5.8 – Proudový odběr motorů

5.4. Funkce modelu

Řídicí a vizualizační aplikace nabízí čtyři možnosti ovládání modelu. Ruční ovládání pomocí základních pohybů a funkce katapult, které jsou realizované pouze pomocí řídicích akcí (nc-actions). Dále řízení pomocí definovaných vačkových profilů, které vykonávají i funkci žonglér. Nakonec diagnostika stavů servozesilovačů a jejich ovládání, které se provádí opět pomocí řídicích akcí.

Diagram ovládání je znázorněn na obr. 5.8. Z něj vyplývá, že můžeme vždy ovládat jen jednu vybranou část.

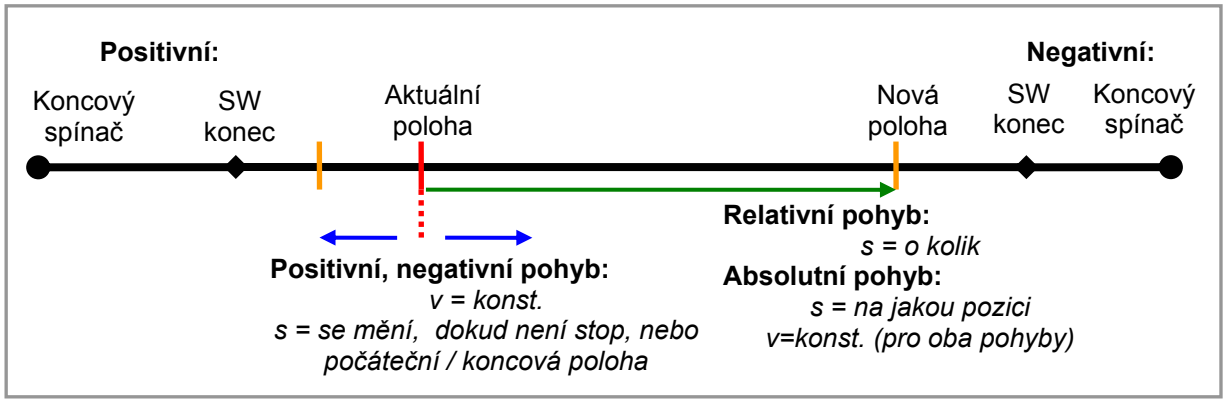


Obr. 5.8 – Funkce modelu

5.4.1. Ruční ovládání

Model ovládáme pomocí čtyř základních pohybů (viz. obr. 5.9), kterými jsou positivní a negativní pohyb (motor se otáčí ve vybraném směru nastavenou rychlosti), absolutní pohyb (motor dojede na určitou pozici danou rychlostí) a relativní pohyb (motor urazí danou vzdálenost danou rychlosti). Dále je k dispozici stop, který ihned zastaví vykonávaný pohyb.

Pokud positivní resp. negativní pohyb neukončíme stopnutím, tak se motor zastaví na positivním resp. negativním softwarovém konci (tj. definované mezi možného pohybu) a vznikne chyba. Pokud ji nepotvrdíme, motor se znovu nerozjede. Pokud špatně nadefinujeme hodnotu softwarového konce, zastaví se motor na koncovém positivním resp. negativním spínači. Při zadání vzdálenosti mimo meze pohybu, pro relativní resp. absolutní pohyb, nedojde k rozjetí motoru.



Obr. 5.9 – Ruční ovládání

K softwarové realizaci ručního řízení byly použity řídicí akce.

5.4.2. Katapult

Funkce katapult (viz. obr 5.10) ovládá model automaticky a provede jeden cyklus, který se vykoná po spuštění. Samotná funkce se skládá ze dvou částí. První z nich spočívá ve vyhození míčku ve svislém směru skrz překážku a jeho chycení v maximální výšce. Druhá spočívá v upuštění míčku a v jeho opětovném chycení v nadefinované výšce. Během obou částí se držák míčku vyhne překážce.

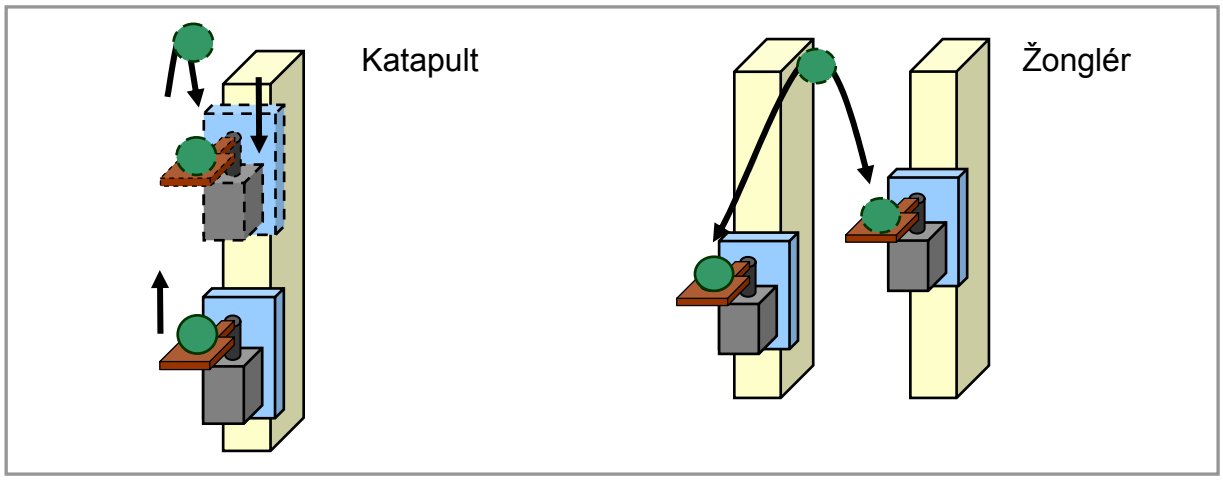
Všechny pohyby motoru jsou realizovány pomocí základního relativního pohybu, kdy motor urazí danou vzdálenost danou rychlostí. K softwarové realizaci této funkce byly použity řídicí akce.

5.4.3. Vačkové profily – žonglér

Vačkové profily umožňují ovládat všechny motory pomocí vybraného předem definovaného vačkového profilu (CAM profile), u kterého definujeme trajektorii

polohy, rychlosti a zrychlení v závislosti na čase. Profily se stále opakují podle stavu vačkového automatu, dokud je uživatel nezastaví.

Funkce žonglér (viz. obr. 5.10) spočívá v přehazování míčků z jedné strany modulu na druhou. Bylo vytvořeno několik profilů s různým provedením přehazování (pouze z jedné strany na druhou jedním míčkem, přehazování dvou míčků navzájem atd.). Součinnost motorů je dána pouze těmito průběhy.



Obr. 5.10 – Funkce katapult a žonglér

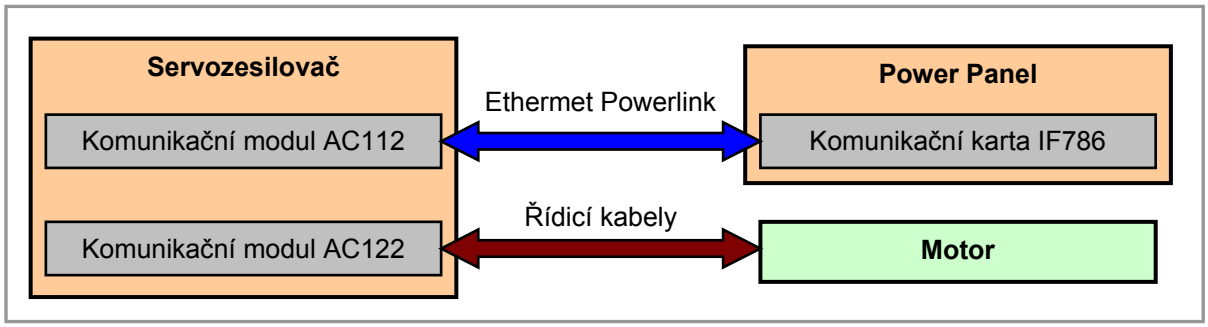
5.4.4. Diagnostika

Diagnostika slouží k získávání informací o stavu všech servozesilovačů (počet a význam chyb, informace i provedení inicializace, stavu regulátoru atd.) a k jejich ovládání (kvitování chyb, zapnutí / vypnutí regulátoru a provedení nastavení do počáteční polohy). K ovládání servozesilovačů se používají řídicí příkazy tzv.akce.

5.5. Realizace

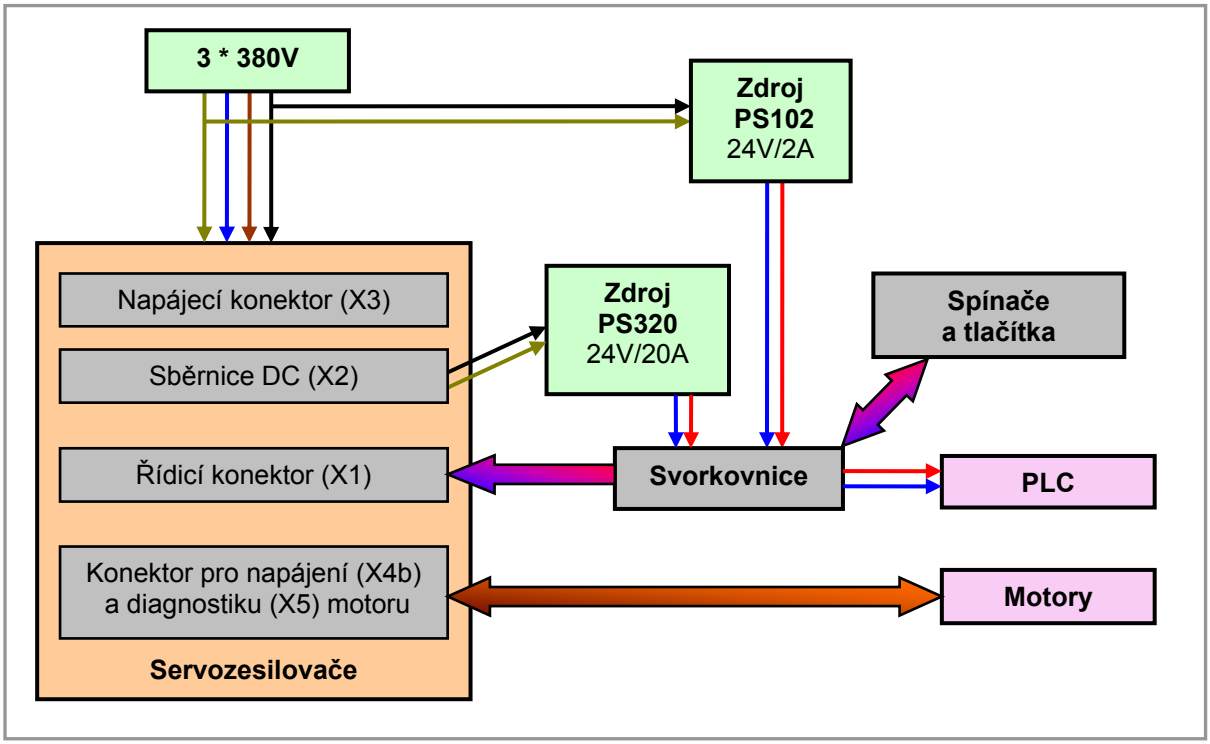
5.5.1. Zapojení kabelů

Na obr. 5.11 je schematické zapojení řídicích a komunikačních kabelů mezi PLC, servozesilovači a motory, popis kabelů je v příloze 4, 5 a 6.



Obr. 5.11 – Schématické zapojení komunikačních/řídicích kabelů

Realizace zapojení kabelů modelu se dá zjednodušeně popsat blokovým schématem dle obr. 5.12.



Obr. 5.12 – Schématické zapojení kabelů

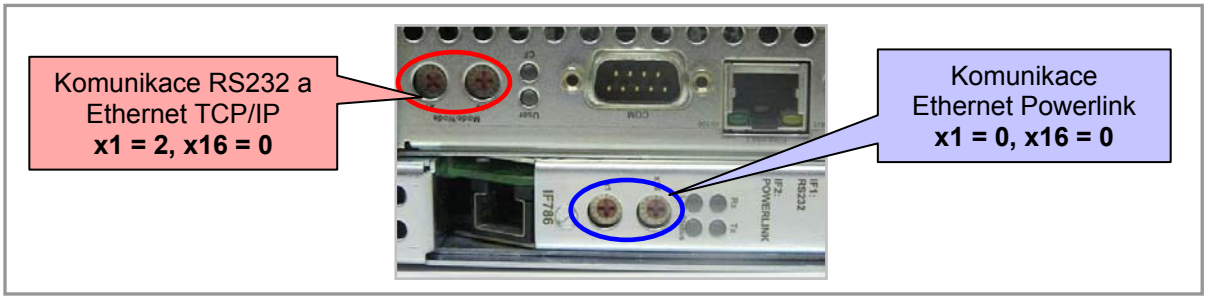
Kompletní zapojení tlačítek, spínačů, konektorů svorkovnice a ostatních komponent je popsáno v přílohách a také na přiloženém CD.

Příloha č. 12 obsahuje popis a rozmístění jednotlivých svorkovnic na DIN liště. Příloha č. 13 obsahuje popis všech kontaktů wago svorkovnic. V příloze č. 14 je zapojení napájení modelu. V příloze č. 15 je zapojení řídicího konektoru (X1) servozesilovačů a v příloze č.16 zapojení všech spínačů a tlačítka.

5.5.2. Nastavení komunikačních adres zařízení

5.5.2.1. PLC

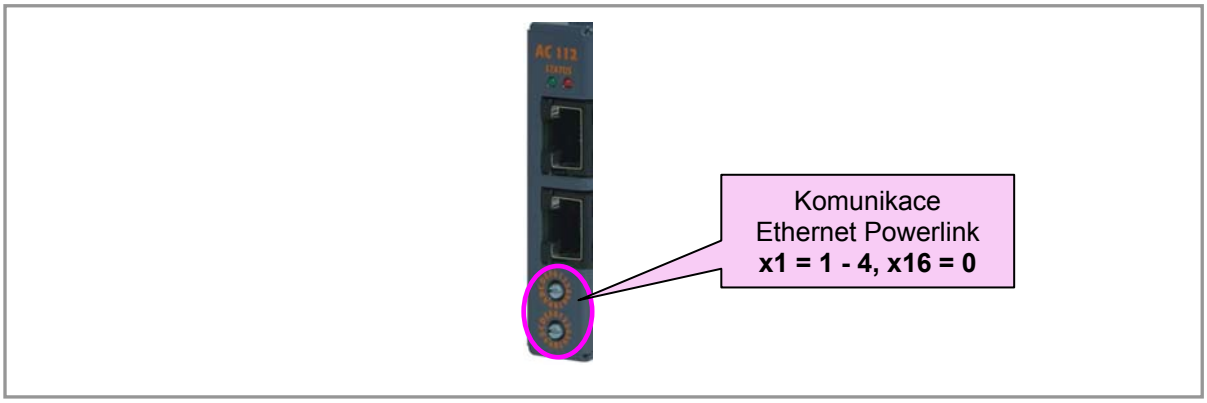
U PLC – Power Panel PP220 je třeba nastavit dva druhy adres (viz. obr. 5.13). První adresa (node number) je pro komunikaci s počítačem po RS232 a Ethernet TCP/IP. Na PLC se nastavuje na hodnotu 2 a v počítači v Automation Studiu se nastaví na 1. Druhý typ adresy se nastavuje na komunikační kartě IF786 a slouží pro komunikaci se servozesilovači po Ethernet Powerlinku. Jelikož je PLC master, musí mít adresu nastavenou na hodnotu 0 (servozesilovače mají potom adresy 1,2,3 a 4)



Obr. 5.13 – Nastavení komunikačních adres PLC

5.5.2.2. Servozesilovač

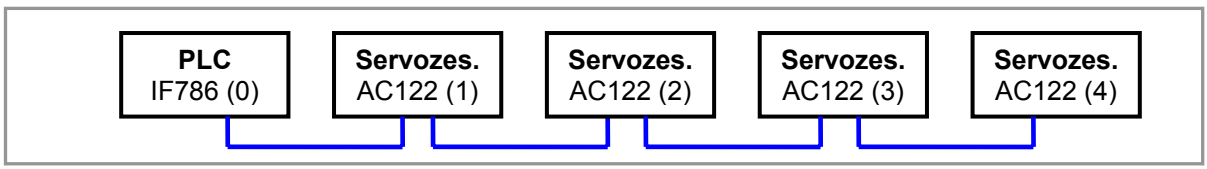
U každého servozesilovače je třeba nastavit adresu na komunikační Ethernet Powerlinkové kartě AC112 (viz. obr. 5.14). Adresy musí být unikátní a budou mít hodnoty 1,2,3 a 4.



Obr. 5.14 – Nastavení adresy na kartě ACPOSu AC112

5.5.2.3. Realizace propojení komunikace Ethernet Powerlink

Propojení komunikace Ethernet Powerlink je realizováno jako sběrnice (viz. obr 5.15), každý servozesilovač slouží jako hub.



Obr. 5.15 – Propojení komunikace Ethernet Powerlink

6. ŘÍZENÍ

V této části jsou popsány kroky k vytvoření řídicího programu pro řídicí systém Power Panel v prostředí Automation Studio firmy B+R automatizace, od vytvoření hardwarové konfigurace, nastavení komunikace, vložení součástí potřebných pro tvorbu samotného softwaru až po nahrání do řídicí jednotky.

Dále je zde popsána struktura řídicího programu. Protože se vytvořený program skládá z různých částí: systémových a speciálních objektů a jednotlivých úloh (úlohy obsahují řídicí algoritmus pro funkce modelu), je zde uveden jejich význam, popis a nastavení.

Nakonec je zde uveden popis řídicích algoritmů a výpočty parametrů pro vykonávání funkcí modelu.

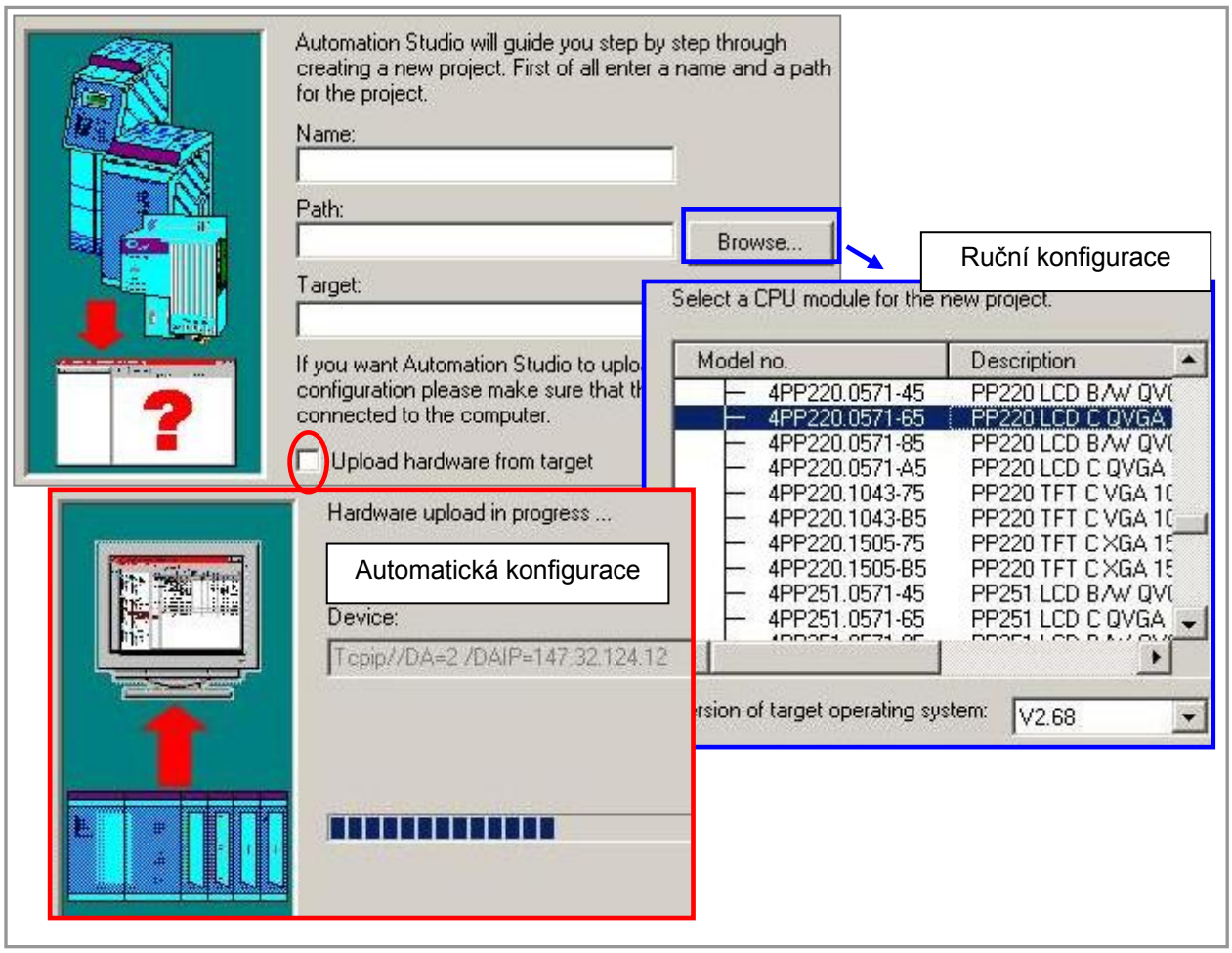
6.1. Postup vytvoření řídicího programu a nastavení potřebných parametrů

Vytvoření řídicího programu lze shrnout do několika základních na sebe navazujících kroků:

1. Vytvoření nového projektu
2. Hardwarová konfigurace
3. Nastavení komunikace
4. Tvorba řídicí aplikace
    - Vložení funkčních knihoven
    - Deklarace proměnných
    - Vložení potřebných objektů
    - Vytvoření řídicích tříd a jejich úloh
    - Vlastní naprogramování algoritmů, řízení a ovládání
5. Přeložení programu
6. Nahrání operačního systému do řídicí jednotky
7. Nahrání programu do řídicí jednotky
8. Testování a ladění

6.1.1. Vytvoření nového projektu

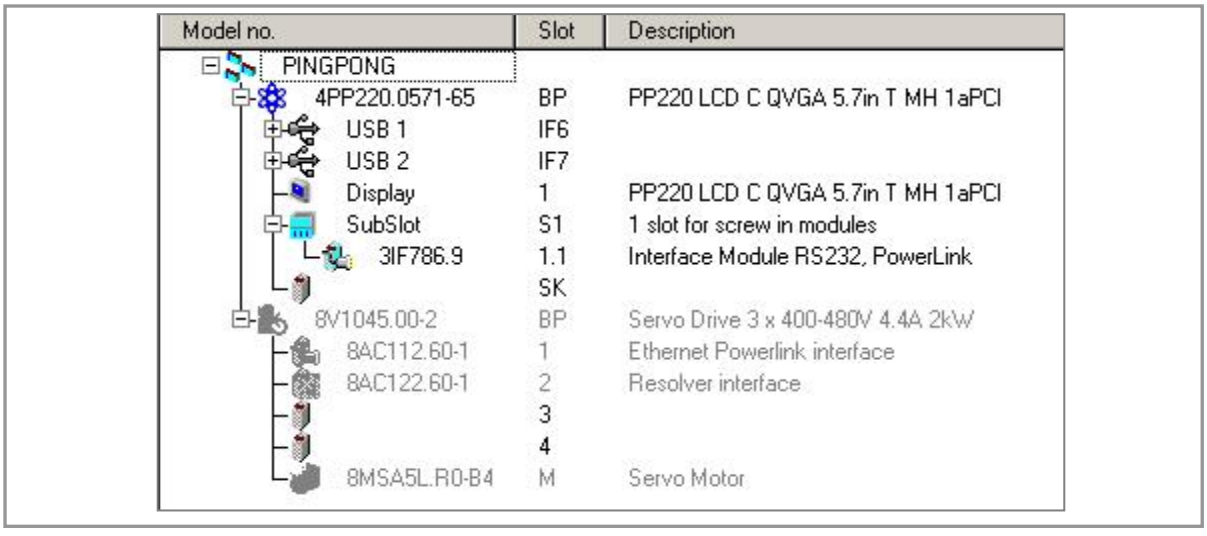
Při vytváření nového projektu definujeme typ řídicího systému (viz. obr. 6.1), který vybereme ručně ze seznamu nabízených systému nebo Automation Studio automaticky rozpozná typ řídicí jednotky, pokud jsme připojeni (online) k řídicímu systému pomocí předdefinované aktivní komunikace (nejčastěji RS232). Dále definujeme verzi operačního systému, který se dá později aktualizovat.



Obr. 6.1 – Vytvoření projektu a hardwarové konfigurace řídicího systému

6.1.2. Hardwarová konfigurace

V hardwarové konfigurací, která je v levé straně hlavního okna, se zobrazují řídicí systémy a další jednotky v grafické stromové struktuře (viz. obr. 6.2).

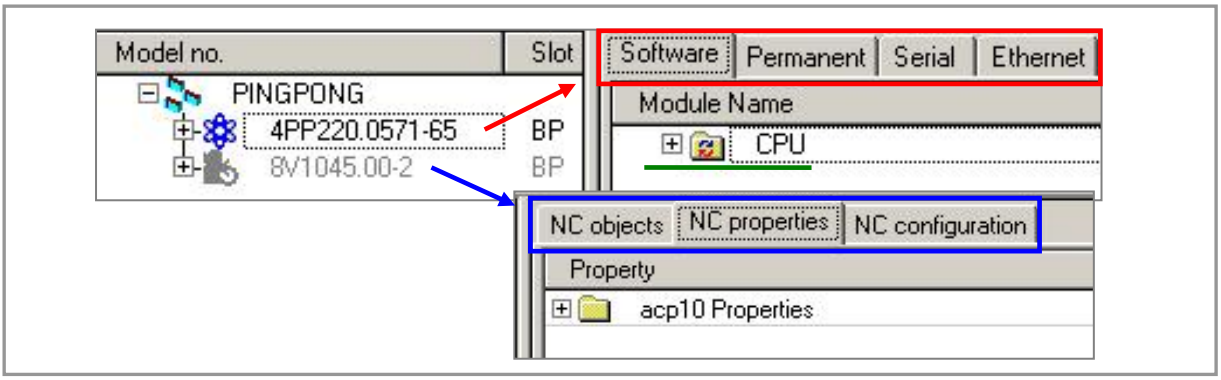


Obr. 6.2 – Hardwarová konfigurace

Tento hardwarový strom je možno vytvořit ručně uživatelem bez spojení s PLC a dalšími systémy (offline) nebo automaticky s aktivním spojením s PLC a dalšími systémy (online).

Moduly jsou seřazeny podle pozic (slotů) a jsou zobrazovány podle typu (procesor, digitální vstupy a výstupy atd.). Detailní informace o řídicím systému, od instalace po popis, jsou zobrazeny v integrované hardwarové dokumentaci.

U všech vložených komponent je dále potřeba nastavit jejich parametry, které se zobrazí v pravé straně okna aplikace po kliknutí na vybranou komponentu (viz. obr. 6.3) a také systémové vlastnosti procesoru řídicího systému (CPU – Edit → Properties).



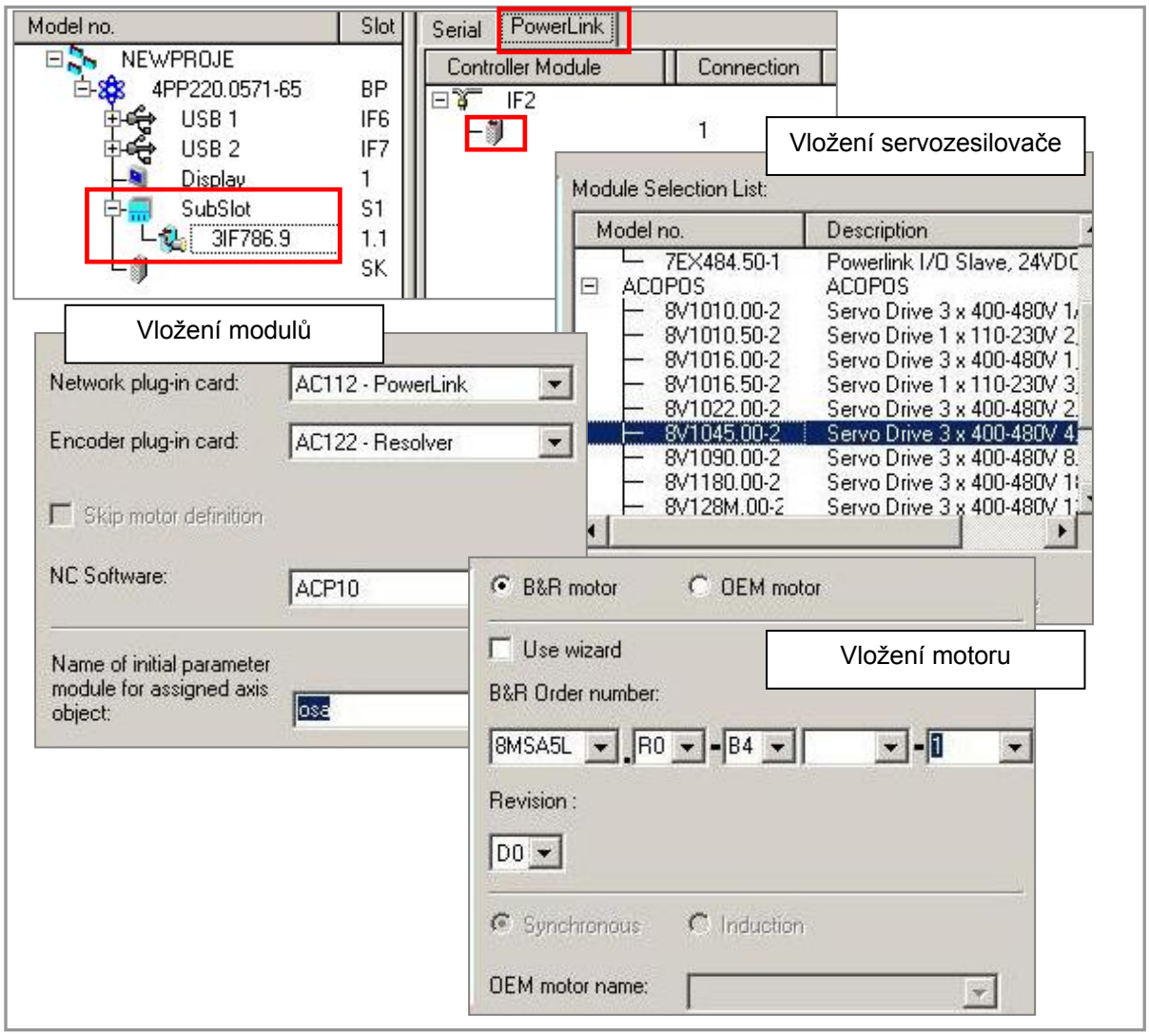
Obr. 6.3 – Parametry komponent hardwarové konfigurace

6.1.2.1. Popis konfigurace řízeného modelu

Hardwarovou konfigurací řídicího systém PP220 (4PP220.0571–65), který obsahuje komunikační kartu 3IF786.9 (Ethernet Powerlink) jsme definovali při vytvoření projektu.

Konfigurace servozesilovače (viz. obr. 6.4) 8V1045.00–2 se provádí později kliknutím na komunikační kartu 3IF786, vybráním záložky Powerlink a vložením servozesilovače do volného slotu.

Při konfiguraci servozesilovače se dále definují jeho moduly (AC112 komunikace Ethernet Powerlink a AC122 rezolver), typ řídicího softwaru (ACP10), název datové struktury servozesilovače (osa) a nakonec řízený motor.



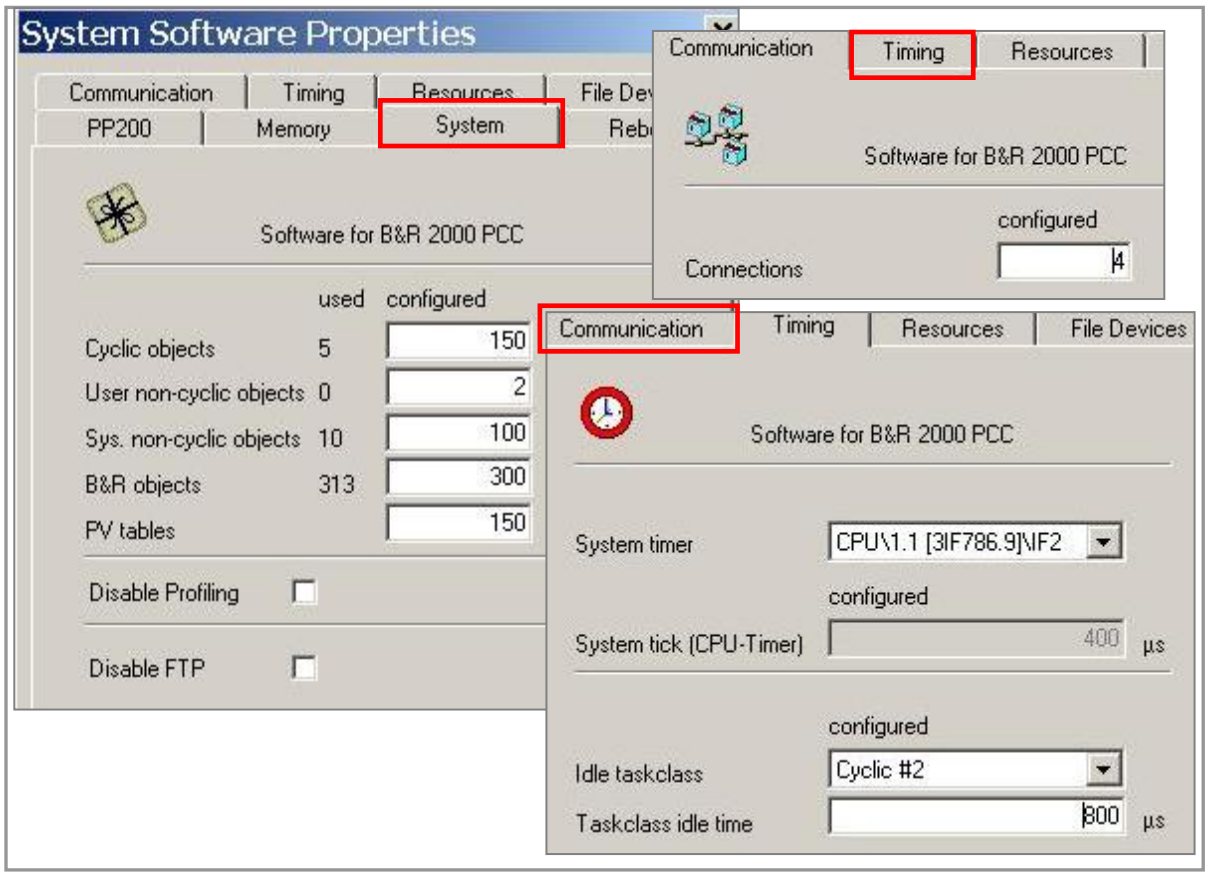
Obr. 6.4 – Konfigurace servozesilovače

Protože jsem hardwarovou konfiguraci ostatních servozesilovačů definoval pomocí speciálního objektu „NC Deployment Table“ a konfiguraci parametrů modulů servozesilovače a motoru ve speciálním objektu „ACOPOS: Parameter Table“, slouží takto vložený servozesilovač a motor pouze k definování systémových parametrů (např. verze softwaru manažeru). Popis a význam parametrů těchto objektů je rozvedený níže. Vložený servozesilovač ve stromové konfiguraci je nutno zablokovat (disable).

6.1.2.2. Nastavení vlastností procesoru řídicího systému

Při nastavení parametrů procesoru (viz. obr. 6.5) řídicího systému je potřeba definovat tyto parametry: systémové (maximální počet vložených jednotlivých typů objektů), časování (počet spojení), komunikační (podle čeho bude systémový časovač synchronizovat, třídu úloh, kde bude komunikace zpracovávána, její cyklický čas) a další.

Nastavení se provádí výběrem modulu procesoru (CPU) ve skupině software řídicího systému PP220 → Properties



Obr. 6.5 – Nastavení parametrů procesoru řídicího systému

6.1.3. Konfigurace komunikace

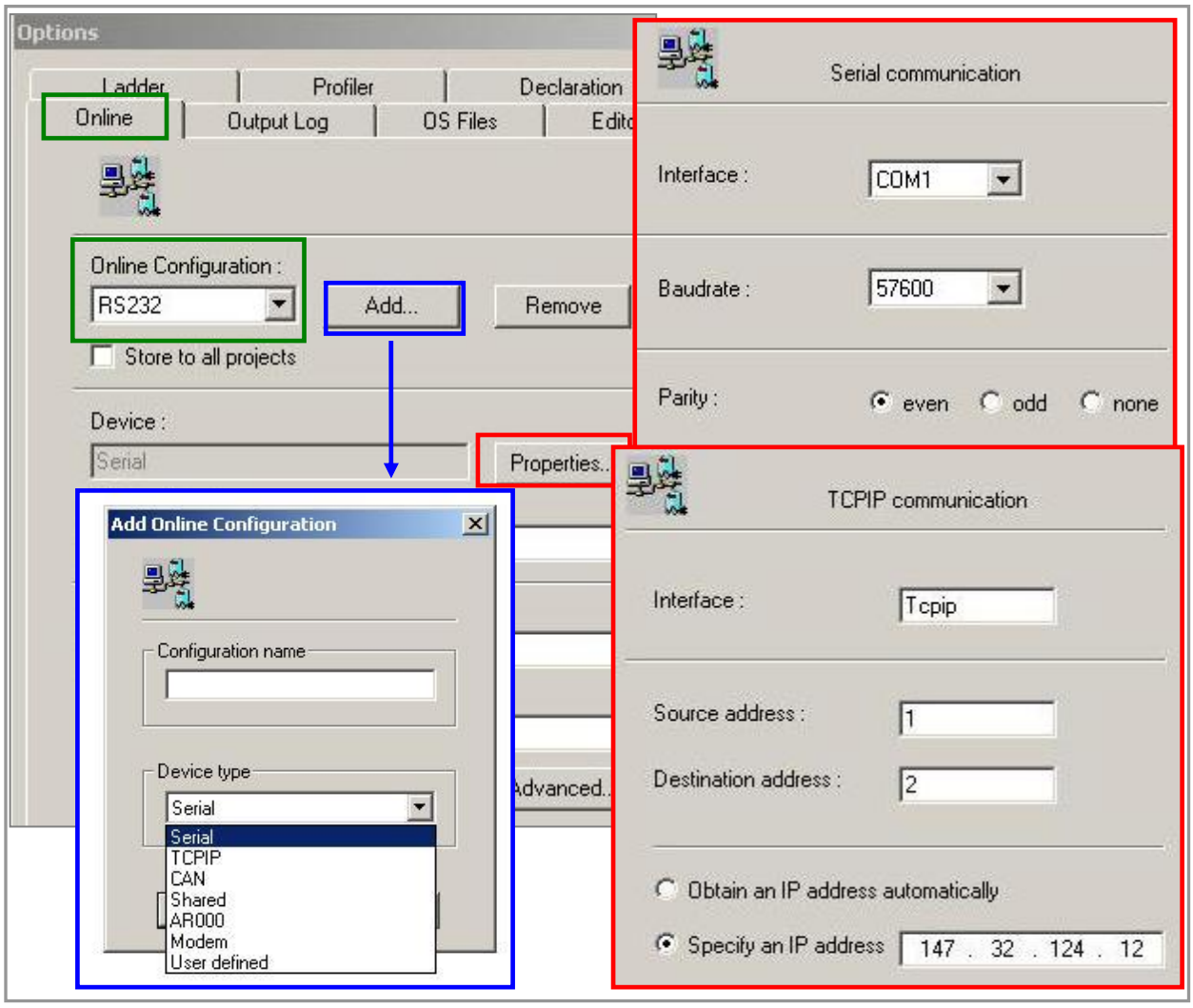
Automation studio může komunikovat s řídicími systémy pomocí libovolné komunikace (RS232, Ethernet atd.). Dále je třeba mít přístup do řídicího systému z dalších aplikací (webových stránek, které běží na webovém serveru systému, přístup na ftp systému atd.). Řídicí systém může komunikovat i s jinými dostupnými systémy (servozesilovače, vstupní a výstupní moduly atd.) např. pomocí komunikací Ethernet Powerlink, CAN a dalších.

Proto je zapotřebí provést konfiguraci na straně počítače, ale i konfiguraci na straně řídicího systému pro všechny typy komunikací.

6.1.3.1. Konfigurace komunikace mezi řídicím systémem a Automation Studio

Nastavení komunikace mezi Automation Studiem a řízeným řídicím systémem (viz. obr. 6.6) se provádí tímto postupem: „Tools → Options → Záložka Online → Online Configuration – vybraná komunikace → Properties“.

Pro sériovou komunikaci je zapotřebí nastavit: port (COM1), rychlost a paritu. Pro komunikaci Ethernet TCP/IP se nastavuje: adresa počítače (source address=1), adresa řídicího systému (destination address=2) a IP adresa.

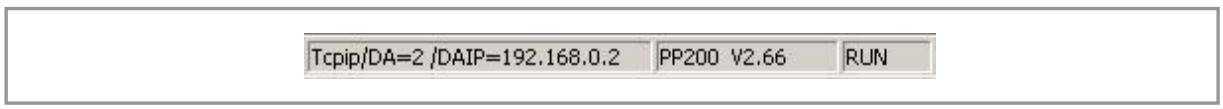


Obr. 6.6 – Konfigurace komunikace mezi řídicím systémem a Automation Studio

Výběr aktuální nastavené komunikace pro online spojení s řídicím systémem se provádí výběrem ze seznamu v „Online Configuration“.

Neexistuje–li potřebná komunikace, je ji zapotřebí vytvořit pomocí tlačítka „Add“ a nadefinovat (zadat jméno nové komunikace – například Ethernet a vybrat její typ – TCPIP), potom nastavit potřebné parametry jako se definují při jejím nastavení.

Informace o aktuálním typu komunikace a jejím stavu (viz. obr. 6.7) mezi Automation Studio a řídicím systémem se nachází na stavové liště (dolní pravý roh).

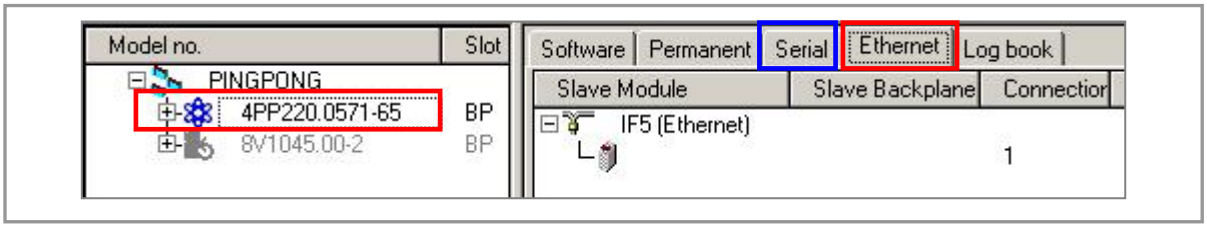


Obr. 6.7 – Aktuální komunikace mezi řídicím systémem a Automation Studio

6.1.3.2. Nastavení komunikace řídicího systému

Nastavení komunikace řídicího systému se provádí v parametrech příslušné komunikace (viz. obr. 6.8). Tyto parametry se potom nahrávají do řídicího systému.

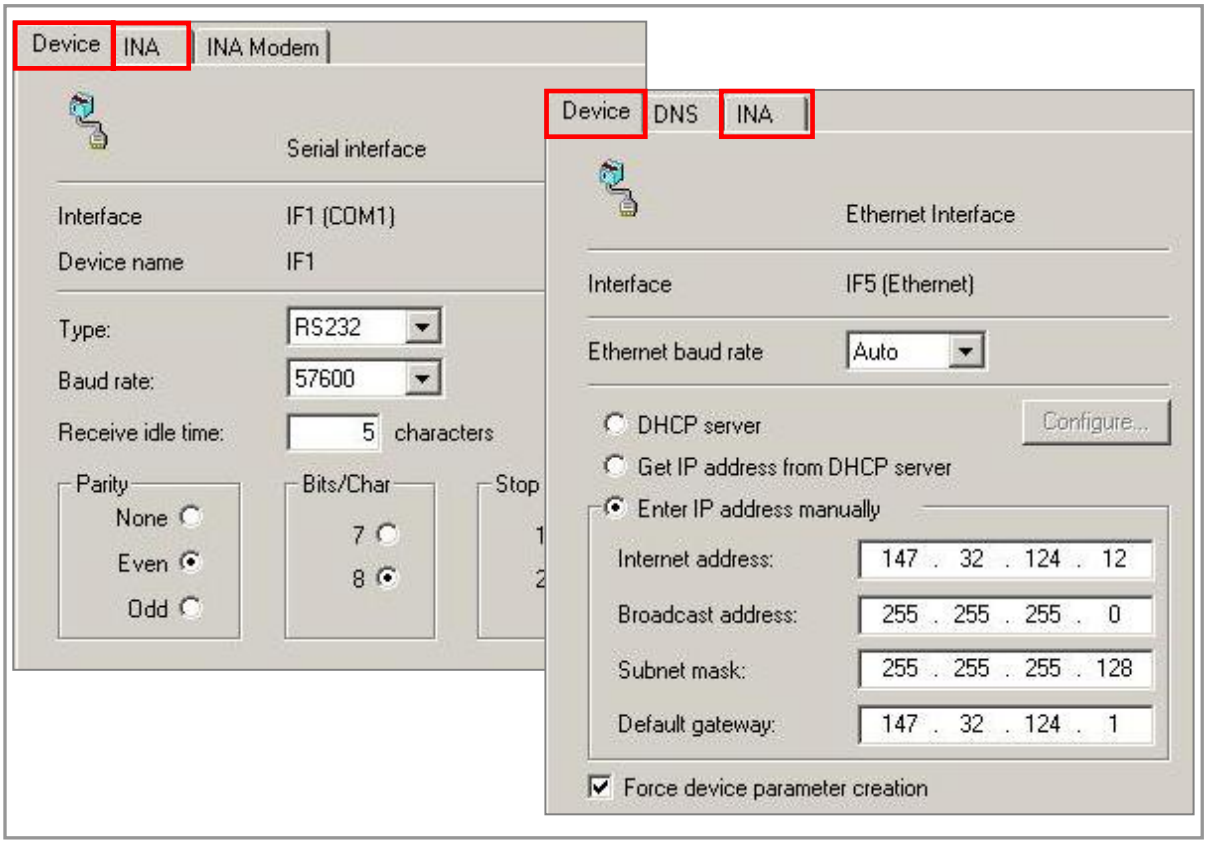
Vybereme procesor (4PP220) → Záložka Serial / Ethernet → vybereme IF → Properties → nastavíme parametry komunikace.



Obr. 6.8 – Nastavení komunikace řídicího systému

Parametry se nastaví v záložce Device (viz. obr. 6.9), dále je zapotřebí v záložce INA zaškrtnout „Active INA communication“ (povolení online komunikace).

Parametry se nastavují stejné jako při nastavování komunikace mezi řídicím systémem a Automation Studio. Pro sériovou komunikace se nastaví: typ, rychlost, počet bitů, počet stop bitů a parita. Pro komunikaci Ethernet TCP/IP se nastavuje IP adresa ručně nebo se nechá automaticky detekovat z DHCP serveru.

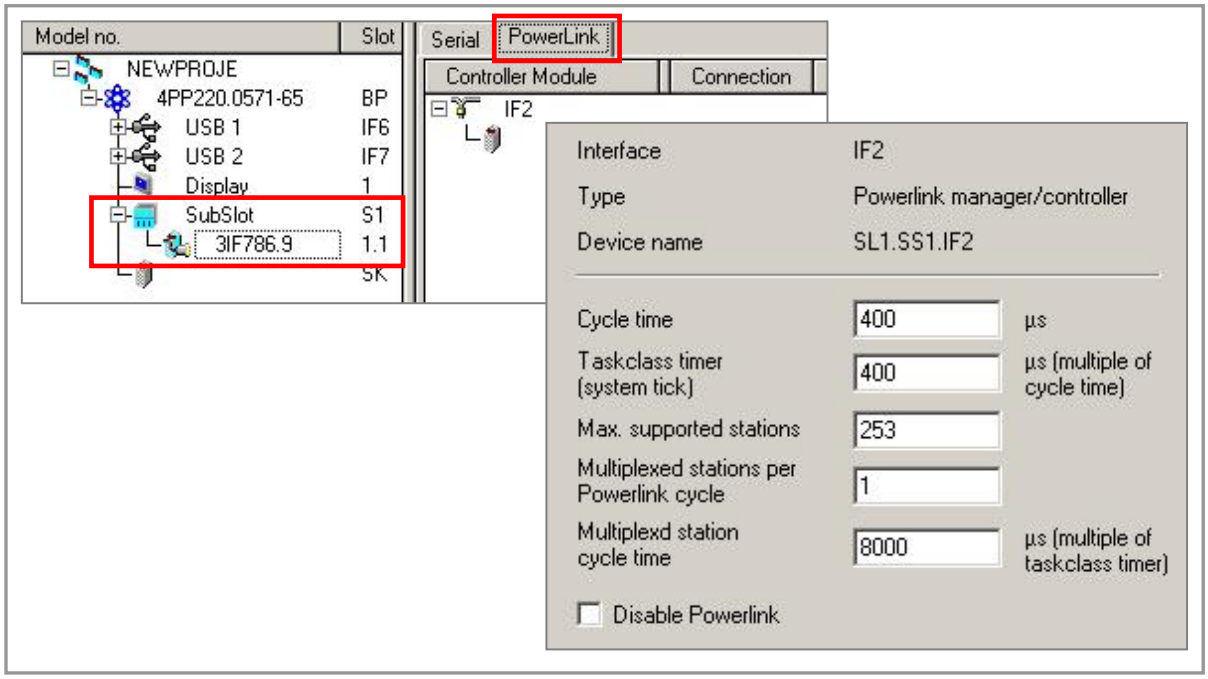


Obr. 6.9 – Parametry komunikace řídicího systému

Po tomto nastavení parametrů komunikace řídicího systému je nutné přeložit program a poté provést nahrání do procesoru. Je potřeba mít vybranou aktivní komunikaci (při prvním nastavení RS232). Po nahrání je možné přepnout komunikaci z RS232 na Ethernet TCP/IP nebo kteroukoliv nově vytvořenou komunikaci.

6.1.3.3. Nastavení komunikace Ethernet Powerlink

Nastavení parametrů komunikace (viz. obr. 6.10) se provádí v tabulce vlastností. Vybereme komunikační kartu v hardwarové konfiguraci, potom zvolíme záložku Powerlink a nakonec vybereme properties na IF2.



Obr. 6.10 – Nastavení komunikace Ethernet Powerlink

6.1.4. Tvorba řídicí aplikace

6.1.4.1. Vložení knihoven

Jednotlivé knihovny obsahují definice funkcí, které používáme v programu (časovače, práce s texty nebo komunikací atd.). Knihovny, které se do programu nevloží automaticky pří jeho vytvoření, je potřeba vložit ručně: „Open → Library Manager → Insert → Library".

Při změně verze systémových a dalších objektů je zapotřebí zkontrolovat verzi příslušných knihoven.

Pokud používáme v programu funkce knihovny (např. libacp10cz.a), musíme ji vložit do řídicí úlohy, která ji využívá: „Insert → Fille" (platí pro jazyk ANSI-C).

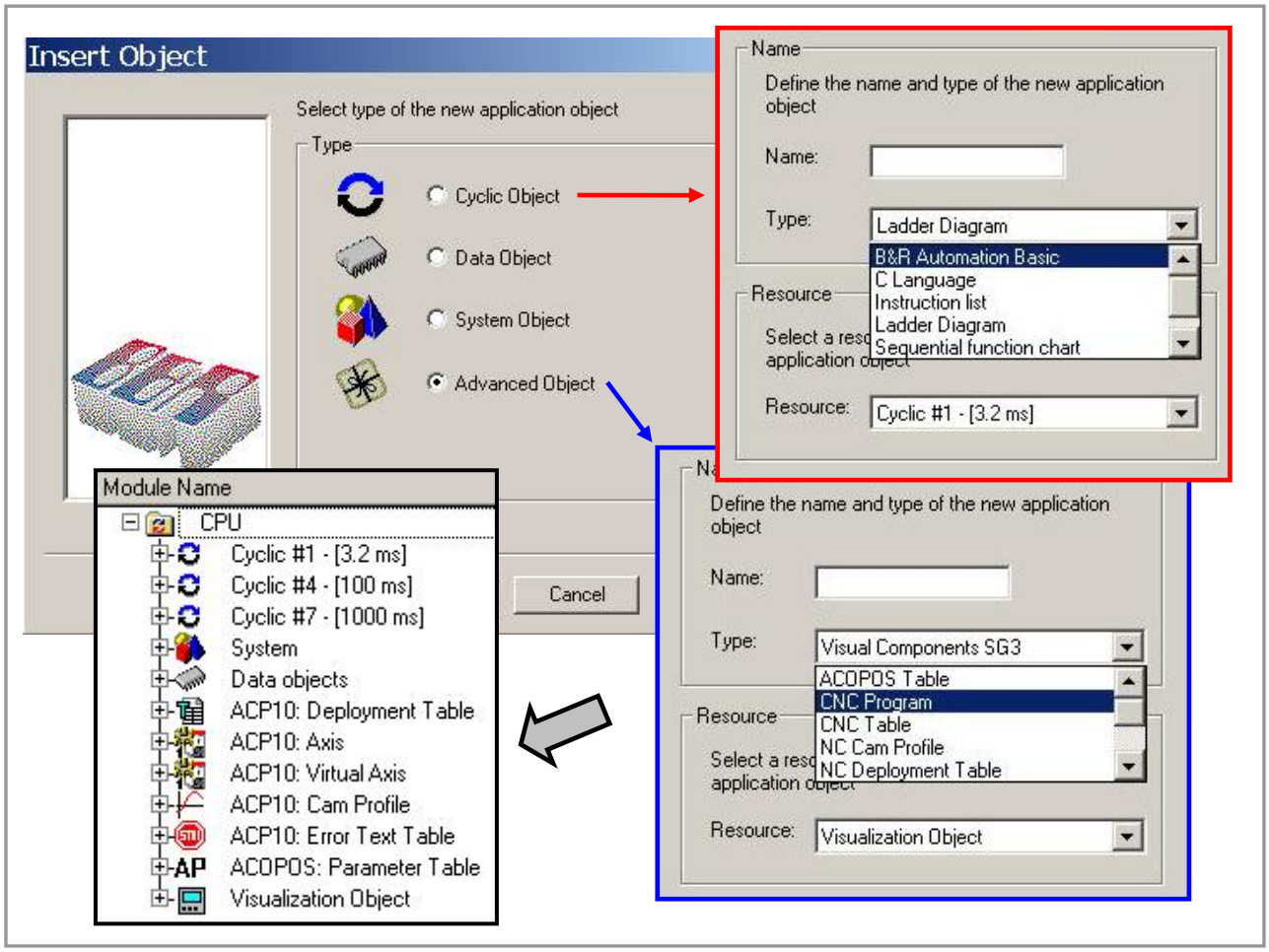
6.1.4.2. Deklarace proměnných

Deklarace proměnných dané úlohy (Tasku) se liší podle toho, v jakém jazyce je úloha naprogramována. Pro jazyk ANSI–C se deklarace provádí přímo v kódu, pro další jazyky se provádí v seznamu proměnných úlohy – „Open → Declaration, nebo pravé tlačítko na úloze → Declaration“.

6.1.4.3. Vložení objektů, vytvoření tříd úloh a úloh

Do řídicí aplikace musí být vloženy řídicí úlohy, které jsou součástí třídy úloh a je v nich napsán kód aplikace, datové objekty, speciální objekty obsahující data potřebná k realizaci jednotlivých částí řízení a vizualizace (parametrizační tabulka, vačkové profily, struktura dat reálných os – servozesilovačů, vizualizační panel atd.) a nakonec se do programu při jeho překladu automaticky vloží systémové objekty. Výjimkou je webový server, který se musí vložit ručně (viz. Kapitola 7.2).

Vložení všech objektů (viz. obr. 6.11) se provádí stejně v aplikačním okně v záložce software řídicího systému: „Insert → Object → Cyclic, Data, System a Advanced object“.



Obr. 6.11 – Vložení objektů

Úlohy jsou součástí tříd úloh, které se liší podle času cyklu opakování. Pro jednotlivé třídy definujeme dobu, do které se musí vykonat všechny úlohy dané třídy. Pro každou úlohu definujeme jazyk, ve kterém je vytvořena.

6.1.5. Přeložení programu

Při překladu zkontrolujeme správnost kódu celého programu. Výsledky překladu se nám zobrazí v okně zpráv. Pokud existuje určitá chyba, jsme informování o jejim významu a pozici.

Přeložení programu se provádí: „Project → Build / Build All, nebo pomocí funkční klávesy“.

6.1.6. Nahrání operačního systému do řídicí jednotky

Pří instalaci operačního systému do řídicího systému je nutné dodržet tyto kroky:

1. Adresa (Node) na PLC musí být v poloze 0 – tzv. Bootovací režim
2. První instalace se provádí z Automation studia → Project → Service → Transfer operating system
3. Pokud chceme jen aktualizovat stávající verzi operačního systému, tak z Automation Studia → Project → Change OS version
4. Spuštění procesoru v run režimu

6.1.7. Nahrání programu do řídicí jednotky

Nahrání potřebných dat z Automation Studia do řídicího systému se provádí: „automaticky po přeložení programu nebo Project → Transfer To Target nebo pomocí funkční klávesy Transfer“.

6.1.8. Testování a ladění

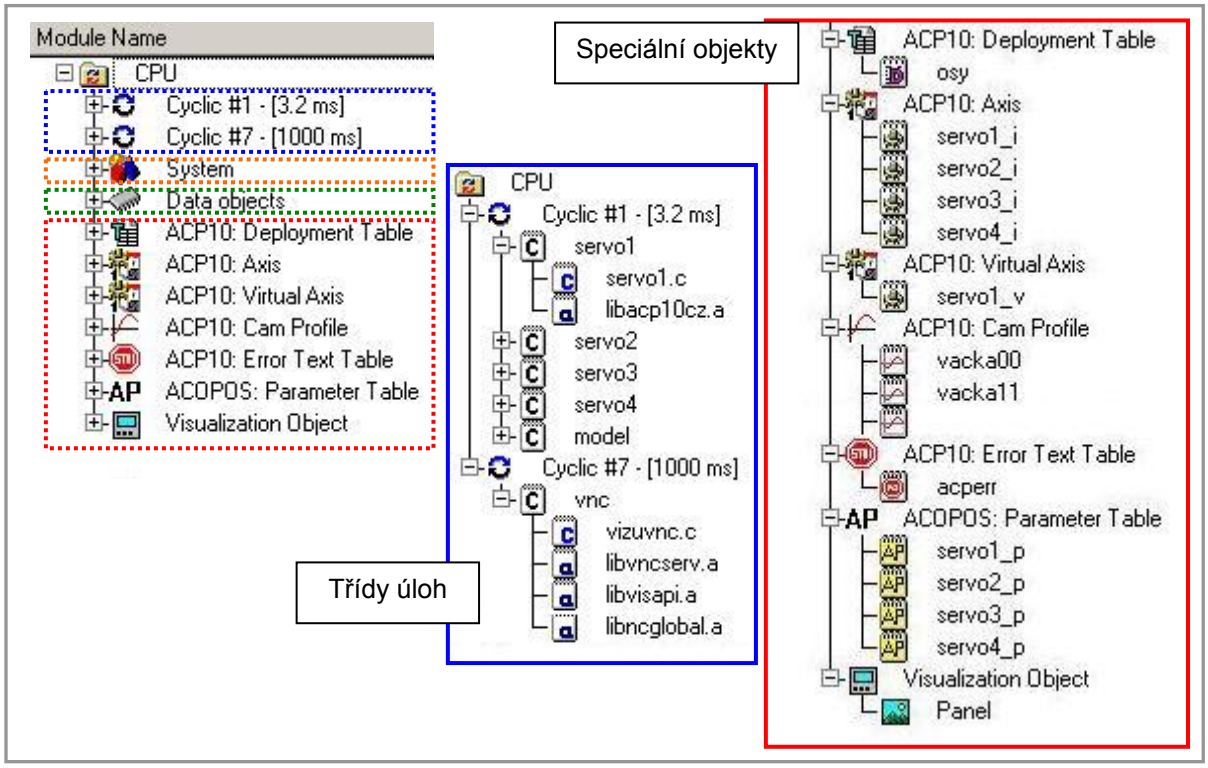
Testování a ladění se provádí pomocí již dříve zmíněných komponent (viz. kapitola 2.1.5).

6.2. Struktura řídicího programu a popis objektů

Pro ovládání modelu byly naprogramovány dvě aplikace. První realizovala řízení modelu pomocí jednotlivých funkcí (ruční ovládání, funkce katapult, žonglér a diagnostika) a vizualizaci modelu. Druhá aplikace demonstrovala řízení pomocí funkcí knihovny PLCopen.

Struktura obou řídicích aplikací je podobná a skládá se z více objektových částí: cyklické třídy, která obsahuje jednotlivé řídicí úlohy, systémových, datových a speciálních objektů.

Ukázka struktury aplikace řízení modelu pomocí funkcí je na obr. 6.12.



Obr. 6.12 – Struktura programu pro ovládání modelu pomocí funkcí

Kvůli přehlednosti, snadnějšímu testování, ladění a odlehčení procesoru byl celý řídicí program rozdělen do více tříd úloh s rozdílnými časy cyklu. Do aplikace byly také vloženy potřebné speciální objekty, jejich popis a význam je uveden v tab. 6.1. Nakonec se do programu vložily při jeho překladu systémové objekty, pokud se tak nestalo, bylo zapotřebí tento objekt vložit ručně (např. webový server).

Typ objektu	*Označení*	*Význam objektu*
ACP10: Deployment Table	osy	Hardwarová konfigurace servozesilovačů (adresa, typ atd.)
ACP10: Axis	servo1_i, …. servo4_i	Parametry reálné osy (servozesilovače)
ACP10: Virtual Axis	servo1_v	Parametry virtuální osy
ACP10: CAM Profile	vacka00, ….	Vačkové profily (trajektorie rychlosti, polohy a zrychleni)
ACP10: Error Text Table	acperr	Seznam poruch (jejich kód a popis)
ACOPOS: Parameter Table	servo1_p, …. servo4_p	Hardwarová konfigurační tabulka motoru, rezolveru a dalších parametrů
Visualization Object	Panel	Vizualizace operátorského panelu

Tab. 6.1 – Seznam vložených speciálních objektů

6.2.1. Řídicí úlohy programu pro řízení modelu pomocí funkcí

Každá z tříd úloh programu pro řízení modelu (ručně, katapult a žonglér) má na starost určitou problematiku související s jeho ovládáním a vizualizací. Všechny úlohy jednotlivých tříd jsou naprogramovány v jazyce ANSI–C. Do jednotlivých úloh bylo zapotřebí vložit potřebné knihovny (*.a) a popřípadě i hlavičky (*.h).

První třída (Cyclic#1) s cyklickým časem 3,2 ms, který odpovídá osmi násobku doby cyklu komunikace Ethernet Powerlink, obsahuje řídicí úlohy pojmenované servo1,2,3 a 4 a model.

Úloha servo1,2,3, a 4 obsahuje soubor servoX.c, ve kterém je napsán řídicí program pro daný servozesilovač, a knihovnu *libacp10cz.a*, která obsahuje funkce potřebné k polohovému řízení pomocí speciálních řídicích příkazů (nc-actions) a vačkového automatu (CAM profile automat). Řídicí program každé úlohy má na starost vytvoření datové struktury daného servozesilovače (reálné osy), jeho inicializaci a řízení. Dále obsahuje řídicí algoritmus pro nahrání vačkového profilu a vačkového automatu do servozesilovače, jeho spuštění a vypnutí. V souboru servo1 je navíc ovládána virtuální osa.

Úloha model obsahuje jednotlivé soubory (*.c), kde každý zajišťuje určitou řídicí logiku modelu. Dále obsahuje hlavičku „promenne.h“, ve které jsou nadefinovány všechny lokální a globální proměnné této úlohy a knihovny libacp10cz.a a *libstandard.a*, která obsahuje funkce pro práci s proměnnými typu string.

Řídicí program v *souboru hlavni.c* vyhodnocuje zvolenou funkci z operátorského panelu nebo webových stránek na základě definovaných parametrů. Dle navolené funkce (diagnostika, ručně, katapult a žonglér) volá příslušnou funkci (c–soubor), ve které je řídicí část programu. Řídicí data pro ovládání servozesilovače jsou zapisována do jeho datové struktury, která byla vytvořena v *úloze servoX*.

Řídicí program v *souboru diag.c* získává a zobrazuje informace o stavu jednoho vybraného servozesilovače a motoru (informace o úspěšném provedení inicializace servozesilovače, stavu řídicí jednotky, zda bylo provedeno nastavení motoru do počáteční polohy a popis existujících chyb) a též zajišťuje jejich ovládání (zapnutí/vypnutí řídicí jednotky servozesilovače, nastavení motoru do počáteční polohy, potvrzení vzniklých poruch).

Řídicí program v *souboru rucne.c* ovládá jeden vybraný servozesilovač a motor pomocí základních pohybů (positivní, negativní, absolutní, relativní a zastavení) na základě zadané rychlosti a polohy.

Řídicí program v *souboru katapult.c* ovládá vybranou stranu modelu (servozesilovače 1 a 2 nebo servozesilovače 3 a 4) pomocí řídicího algoritmu funkce katapult.

Ovládání modelu pomocí funkce ručně, katapult a diagnostika je realizováno pomocí speciálních řídicích příkazů tzv. akcí (nc-actions) a ovládání pomocí vačkových profilů. Funkce žonglér je realizována vačkovým automatem a vačkovými profily. Podrobný popis jednotlivých parametrů vačkového automatu je v přiloženém souboru „*LibAcp10cz.chm*“.

Druhá třída (Cyclic#7) s cyklickým časem 1000 ms obsahuje *řídicí úlohu vnc*, která obsahuje *soubor vizuvnc.c*. Tato část programu zajišťuje spuštění VNC serveru pro možnost vzdálené vizualizace a ovládání pomocí VNC prohlížeče, který se spustí na počítači zapojeného do sítě internet. Řídicí úloha obsahuje knihovnu *libvncserv.a*, ve které jsou funkce pro zapnutí a vypnutí serveru.

6.2.1.1. Řídicí algoritmus funkce katapult

Řídicí funkce *„katapult“* (úloha katapult.c) se skládá ze dvou algoritmů. První řídí servozesilovač a motor zajišťující pohyb nahoru – dolů, a druhý do stran. Podle zadaných dat se ovládá vybraná strana modelu (levá nebo pravá).

Kroky algoritmu nahoru – dolů (ovládání modulu)

1. Inicializace a nastaveni počátečních parametrů
2. Čekání na povel start
3. Nastavení motorů do počáteční polohy
4. Vyhození míčku a jeho chycení
5. Upuštění míčku a jeho chycení
6. Konec funkce a zpět na krok 1

Kroky algoritmu doleva – doprava (vyhnutí překážce)

1. Inicializace a nastaveni parametrů
2. Podle aktuální pozice motoru vykonávající pohyb nahoru – dolů se provádí vyhnutí překážce pohybem do strany
3. Zpět na krok 1

6.2.1.2. Řídicí algoritmus vačkových profilů - funkce žonglér

Řídicí funkce *„vačkové profily – žonglér“* (úloha zonlger.c) je realizována dvěmi algoritmy. První zajišťuje nahrání vačkových profilů do servozesilovače a druhý spuštění čtyř reálných a virtuální osy.

Kroky algoritmu nahrání profilů:

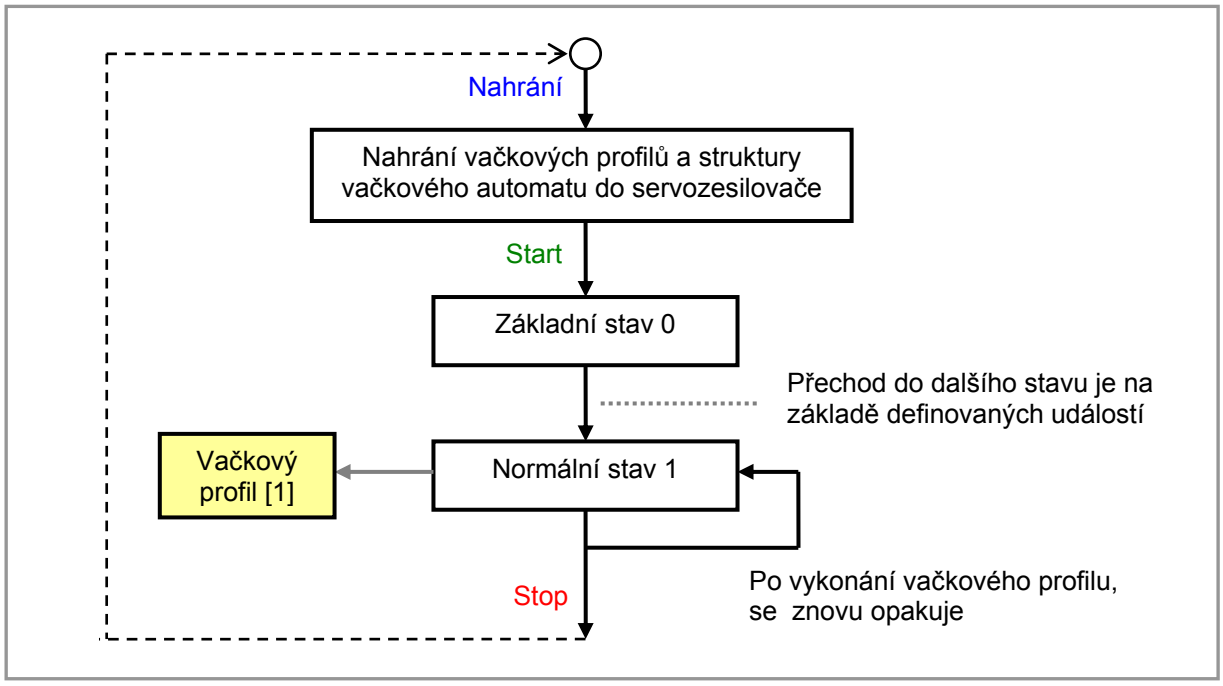
1. Vyhodnocení zadávaných profilů z panelu nebo webových stránek
2. Nahrání vačkových profilů z PLC do servozesilovače

Kroky algoritmu řízení os pomocí vačkového profilu:

1. Nastavení počátečních parametrů pro čtyři reálné a virtuální osu
2. Čekání na povel start
3. Nastavení motorů do počáteční polohy
4. Spuštění pohybu reálných a virtuální osy (vačkového automatu)
5. Čekání na povel stop (ukončení pohybu – automatu)
6. Opakuje se krok 1

6.2.1.3. Popis řízení pomocí vačkového automatu

Struktura vačkového automatu (viz. obr. 6.13) je stejná pro všechny servozesilovače. Obsahuje jeden základní stav, kde se provedou nastaveni a jeden normální stav, který spouští vačkový profil.



Obr. 6.13 – Struktura vačkového automatu

Před samotným spuštěním vačkového automatu je potřeba provést nahrání vačkových profilů a struktury vačkového automatu do servozesilovače. Pro vačkový automat je potřeba nadefinovat jednotlivé stavy, pro které definujeme parametry pro master a slave osu, události pro přechod mezi jednotlivými stavy, typ vačkového profilu, který se bude vykonávat v aktuálním stavu a následující stav (viz. kap. 6.5.2).

Všechny vačkové profily jednotlivých os jsou závislé pouze na virtuální ose, která reprezentuje čas. Profily os jsou synchronizovány časem po jejich spuštění. Každý vačkový profil se vykonává určitou dobu, po jejím uplynutí se profil začne vykonávat znova od začátku dokud automat nezastavíme.

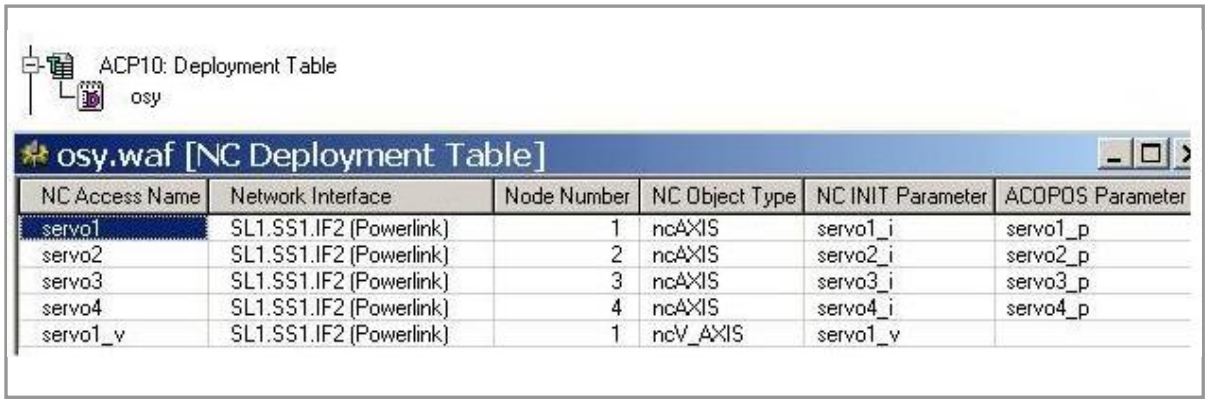
6.2.2. Řídicí úloha demonstrující řízení pomocí PLCopen

Program obsahuje jednu třídu úloh (Cyclic#1) s cyklickým časem 10,4 ms. Vložená úloha *PLCopen* je naprogramované v jazyce Automation Basic (AB) a realizuje ovládání pouze jednoho servozesilovače pomocí funkcí knihovny PLCopen.

Servozesilovač můžeme zapnout a inicializovat. Dále můžeme řízený motor nastavit do počáteční polohy a ovládat ho pomocí funkce „pohyb v daném směru“ (MoveVelocity) a funkce stop. Ovládání nelze provádět z operátorského panelu, ale pouze z Automation Studia. Do manažeru knihoven (library manager) museli být vloženy *knihovny libacp10_mc a libncglobal*. První obsahuje řídicí funkce knihovny PLCopen a druhá funkce potřebné k vytvoření datové struktury řízeného servozesilovače.

6.2.3. Konfigurační tabulka objektů – „Deployment table“

Objekt „ACP10:Deployment table“ (viz. obr. 6.14) je použit k vytvoření řídicí konfigurace. V této tabulce definujeme softwarové objekty (reálná a virtuální osa) pro specifické hardwarové objekty (servozesilovač). Nastavujeme inicializační parametry příslušného objektu (servozesilovače), kterými jsou unikátní jméno, komunikační rozhraní, adresa, typ objektu, jméno struktury parametrů a jméno hardwarové parametrizační tabulky servozesilovače.

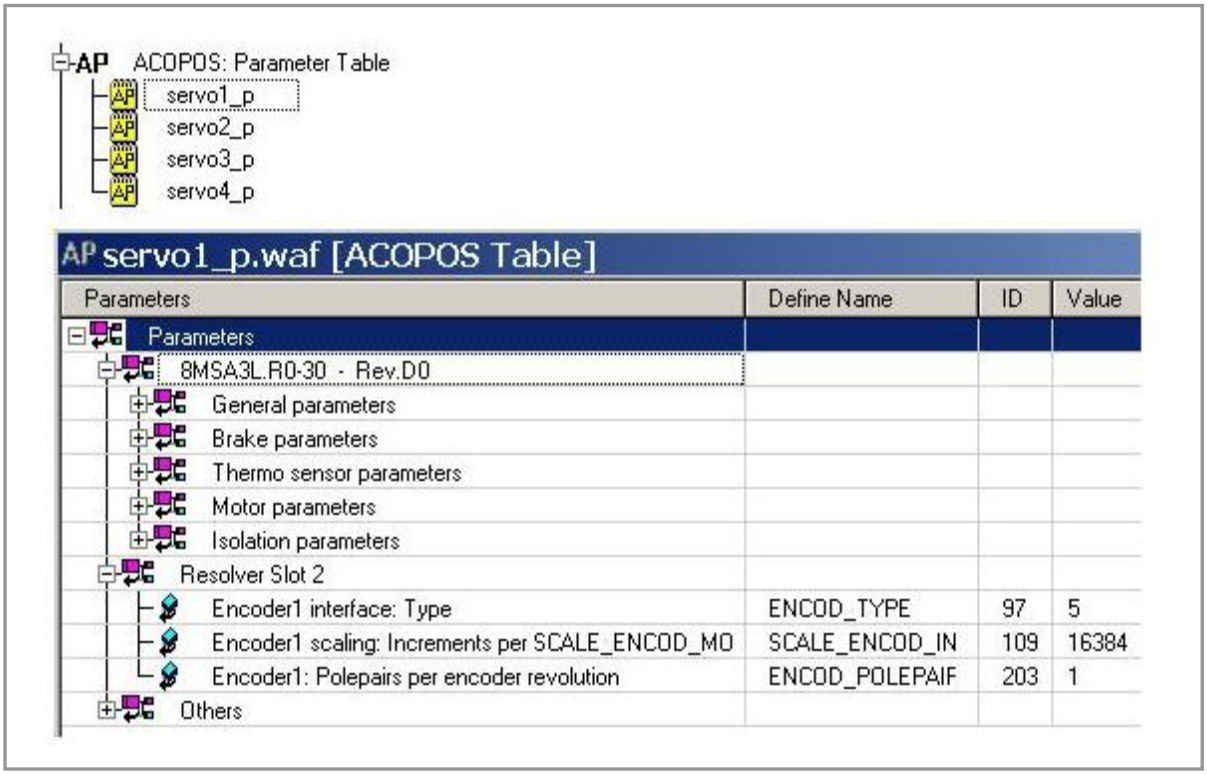


NC Access Name	Network Interface	Node Number	NC Object Type	NC INIT Parameter	ACOPOS Parameter
servo1	SL1.SS1.IF2 (Powerlink)	1	ncAXIS	servo1_i	servo1_p
servo2	SL1.SS1.IF2 (Powerlink)	2	ncAXIS	servo2_i	servo2_p
servo3	SL1.SS1.IF2 (Powerlink)	3	ncAXIS	servo3_i	servo3_p
servo4	SL1.SS1.IF2 (Powerlink)	4	ncAXIS	servo4_i	servo4_p
servo1_v	SL1.SS1.IF2 (Powerlink)	1	ncV_AXIS	servo1_v	

Obr. 6.14 – Konfigurační tabulka objektů

6.2.4. Hardwarová parametrizační tabulka – „Parameter Table“

Objekt „ACOPOS:Parameter Table“ (viz. obr. 6.15) je použit k vytvoření hardwarové konfigurace servozesilovačů.



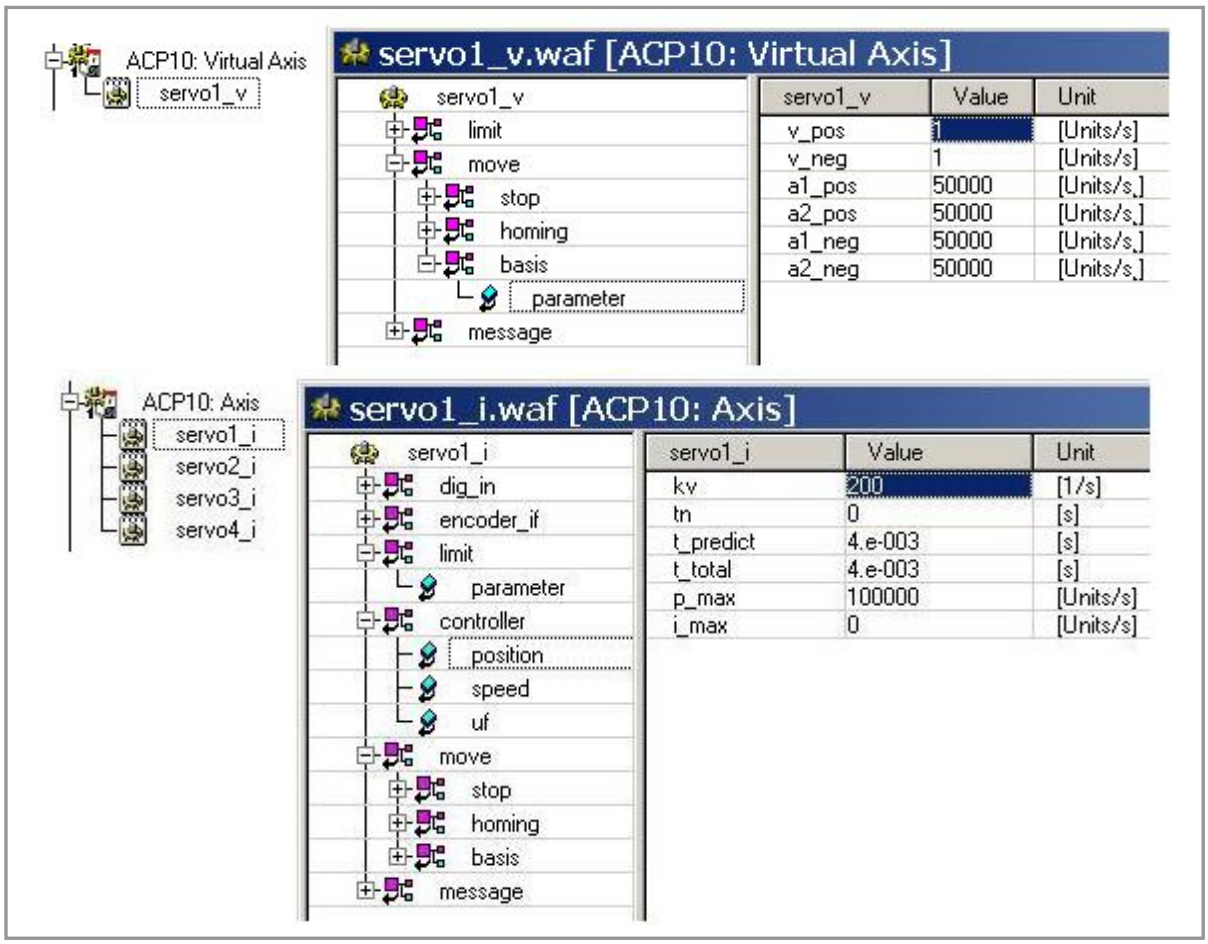
Obr. 6.15 – Hardwarová parametrizační tabulka

Pro každý servozesilovač je zapotřebí jedné tabulky, ve které definujeme parametry příslušného řízeného motoru, modulu rezolveru (AC122) a další parametry potřebné k řízení motoru (např. parametry filtru rušivých frekvencí, master stanici pro řízení pomocí vačkových profilů - automatu a další).

Zjištění rušivých frekvencí a potřebných parametrů frekvenčního pásma se provádí pomocí nástroje Trace. Musíme změřit charakteristiku skutečného proudu a potom přidat další graf (Add New Chart), pro který definujme v jeho vlastnostech vzorec pro osu y (Formula y) ve tvaru *FFT(‘graf proudu, který vložíme pomocí tlačítka - insert into Y’)*.

6.2.5. Struktura reálné a virtuální osy – „ Axis a Virtual Axis”

Objekt „ACP10:Axis” (ACP10:Virtual Axis) obsahuje parametry reálné (virtuální) osy (viz. obr. 6.16) od nastavení vstupních kontaktů, přes nastavení regulátorů, až po parametry pohybu. Jednotlivé parametry jsou rozděleny do skupin (subjektů) dle významu. Struktura parametrů obou os se liší, protože pro virtuální osu není potřeba nastavovat tolik parametrů jako pro osu reálnou.

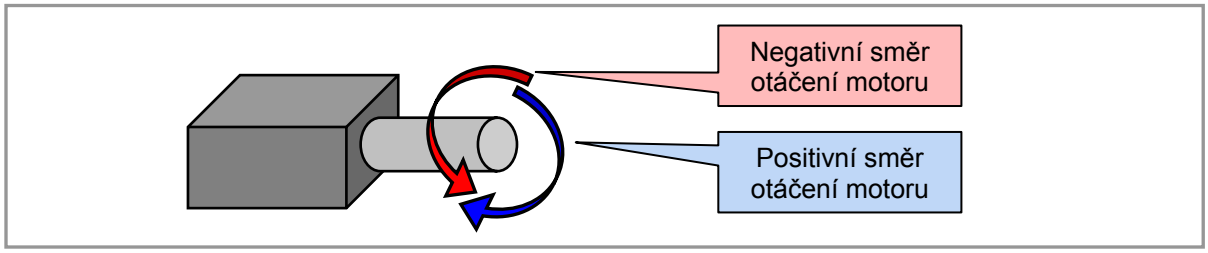


Obr. 6.16 – Struktura virtuální a reálné osy

Jelikož je zapotřebí nastavit velké množství parametrů jednotlivých skupin, budou zde popsány a vysvětleny jen ty podstatné. Parametry struktury se inicializují po spuštění PLC a servozesilovačů. Všechny parametry se dají měnit i z běžící aplikace pomocí dříve zmíněných akcí. Pro *odladění nastavených parametrů* a ověření jejich funkce slouží testovací prostředí „Test“.

6.2.5.1. Směr otáčení motoru

Při nastavování parametrů servozesilovače (limitní hodnoty, koncové polohy atd.) a při ovládání motorů, je třeba dát pozor na směr otáčení motoru, protože rozlišujeme positivní a negativní směr otáčení motoru (viz. obr 6.17).

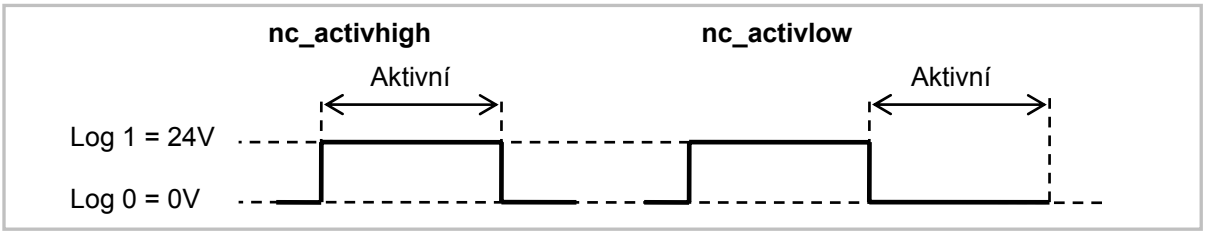


Obr. 6.17 – Popis směrů otáčení motoru

6.2.5.2. Digitální vstupy

Skupina digitálních vstupů obsahuje dva vypínací spínače (trigger1, trigger2 – quickstop), positivní a negativní hardwarový konec a referenční spínač, pro které nastavujeme parametr úrovně spínaní.

Nastavený parametr „*nc_activlow nebo nc_activhigh*“ určuje, zda je spínač aktivní při 0 nebo 24V (je–li spínač aktivní provede se příslušná akce). Dále můžeme nastavit parametr „*nc_activ+nc_quickstop*“. Parametr qickstop určuje, zda se akce zastavení provede plynule nebo okamžitě. Úrovně spínaní jsou ukázány na obr. 6.18.



Obr. 6.18 – Úrovně spínání digitálních vstupů

Positivní nebo negativní koncový spínač slouží k zastavení motoru při jeho aktivaci. Jelikož tento spínač rozlišuje směr otáčení motoru, dojde k zastavení na positivním konci, vykonával–li motor pohyb v positivním směru, při negativním pohybu k zastavení nedojde.

Referenční spínač se používá k nastavení motoru do počáteční polohy a vypínací spínače (trigger) se používají k rychlému odstavení řídicí jednotky servozesilovače nebo k zastavení motorů.

Pro zajištění bezpečného chodu řízeného modelu byla pro všechny spínače nastavena úroveň spínání na 0 V (ncactivlow), která zareaguje na výpadek napájení.

6.2.5.3. Enkoder

Pro enkoder nastavujeme směr počítání polohy a kolik jednotek (units) bude na daný počet otáček.

6.2.5.4. Limitní hodnoty

Ve skupině limitních hodnot nastavujeme maximální hodnotu rychlosti a zrychlení pohybu v positivním resp. negativním směru, hodnotu positivního a negativního softwarového konce, plynulý náběh atd.

Aby se motor vertikálního pohybu rozjel z 0 na 5 m/s bez velkých počátečních překmitů, musel jsem nastavit parametr plynulého náběhu (t_{JOLT}) na 0.1s.

6.2.5.5. Regulátor

Ve skupině regulátoru nastavujeme polohový , rychlostní a proudový regulátor. Pro polohový regulátor (PI nebo predikční) nastavujeme proporcionální a integrační složku, čas predikce, celkový čas a maximální hodnoty složek. Pro rychlostní (PI regulátor) nastavujeme pouze proporcionální a integrační složku a dobu filtrování. U proudového nastavujeme jen typ motoru (lineární nebo kvadratický), ostatní parametry se nastaví automaticky.

Nejprve se nastavuje rychlostní regulátor s tím, že zesílení polohového je nulové. Ale v našem případě musíme mít zadanou hodnotu polohového regulátoru pro motor vertikálního směru alespoň 50, protože by jinak pojezd ujížděl dolů.

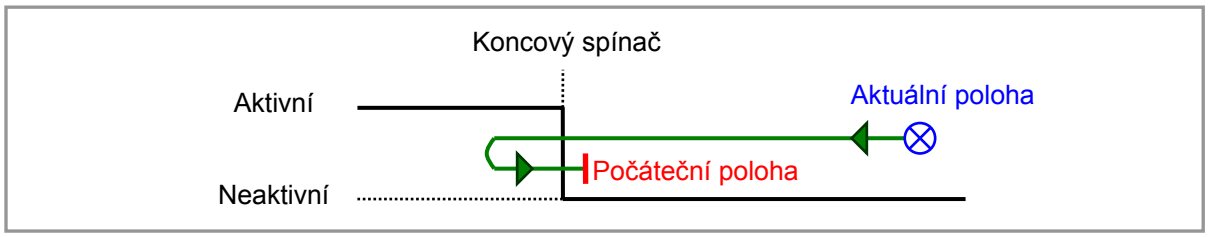
6.2.5.6. Parametry pohybů

Ve skupině pohybů definujeme parametry pro nastavení motoru do počáteční polohy: rychlost vykonávání před a po najetí na nastavovací spínač, zrychlení pohybu, režim nastavení (najetí na koncový spínač nebo referenční spínač, přímé nastavení kdy aktuální poloha je žádaná a další), směr vykonání nastavení před a po najetí na spínač a zda použijeme referenční puls.

Dále nastavujeme parametry pro vykonání standardních pohybů (rychlost, dráha, zrychlení a brzdění), kterými jsou pohyb v positivním a negativním směru a relativní nebo absolutní pohyb.

Nastavení řízeného motoru do počáteční polohy (homing):

Pro všechny motory řízeného modelu bylo provedeno nastavení do počáteční polohy metodou najetí na koncový spínač (viz. obr. 6.19). Parametry směru a typu spínače se liší, protože typ otáčení (positivní, negativní) není stejný pro všechny motory. Nastavení se provede při změně stavu spínače ze stavu neaktivní do stavu aktivní (ncOpen – ncClose).



Obr. 6.19 – Nastavení motoru do počáteční polohy

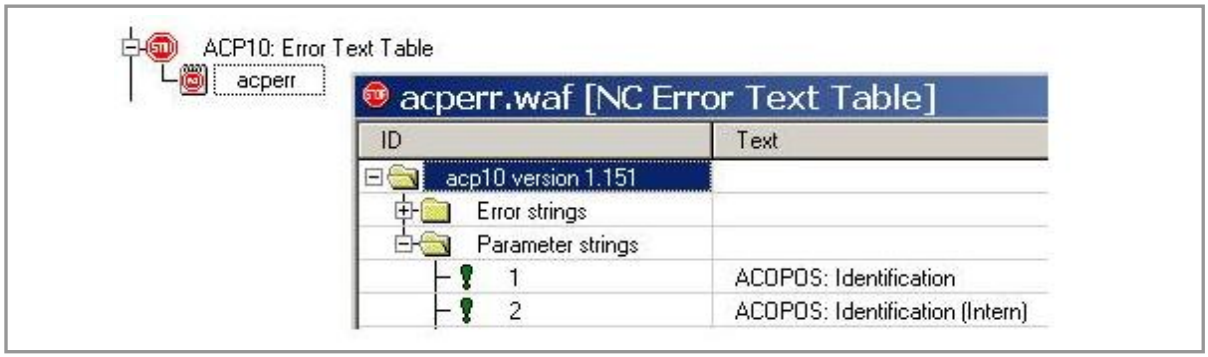
6.2.6. Vizualizační objekt – „Visualization object“

Vizualizační objekt (viz. obr. 6.20) obsahuje veškeré informace a parametry vizualizačních obrazovek - jejich vstupních a výstupních polí, dynamickém nastavení textů a grafických prvků, vlastnosti tlačítek, nastavení seznamu alarmů atd.

Obr. 6.20 – Objekt pro vizualizaci

6.2.7. Tabulka poruch – „ Error Text Table”

Objekt „ACP10:Error Text Table“ (viz. obr. 6.21) obsahuje seznam poruch a parametrizačních informací, kde pro každý záznam máme jeho identifikační číslo a popis. Tyto informace lze použit v aplikaci (např. zobrazit na operátorském panelu) pro identifikaci stavu systému.

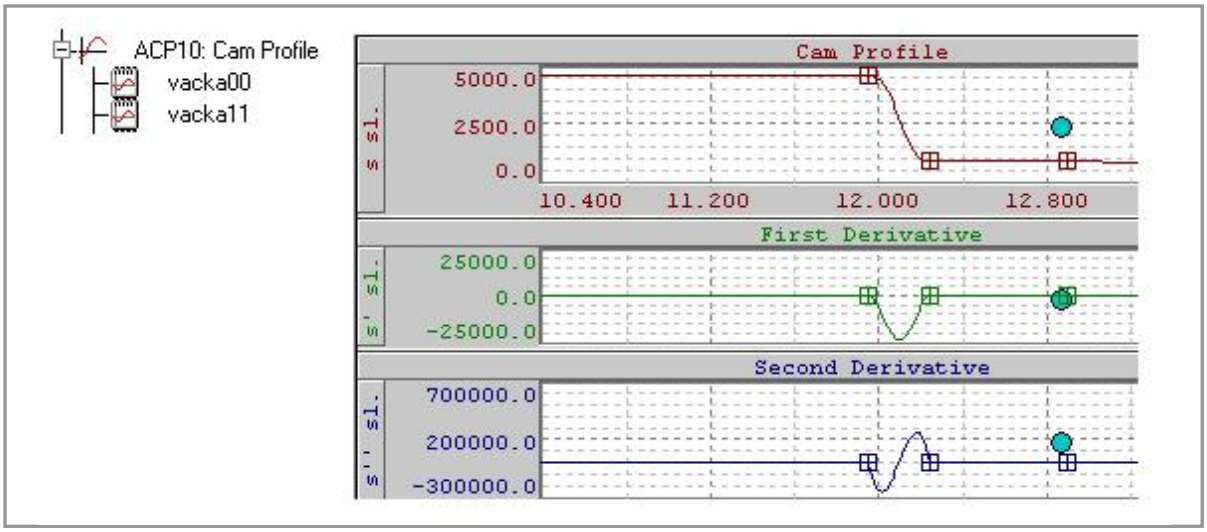


Obr. 6.21 – Objekt seznamu poruch servozesilovače

6.2.8. Vačkový profil – „CAM profile“

Objekt „ACP10:CAM profile“ obsahuje vytvořené trajektorie (rychlosti, polohy a zrychlení) pro řízení pomocí vačkových profilů (viz. obr. 6.22).

Těchto profilů může být v aplikaci vytvořené velké množství a my si můžeme pro řízení nastavit libovolný pomocí speciálních příkazů vačkového automatu.



Obr. 6.22 – Objekt vačkových profilů

6.2.8.1. Popis vytvořených vačkových profilů

V tab. 6.2 jsou popsaný funkce vytvořených vačkových profilů pro přehazování míčků z jedné strany modulu na druhou..

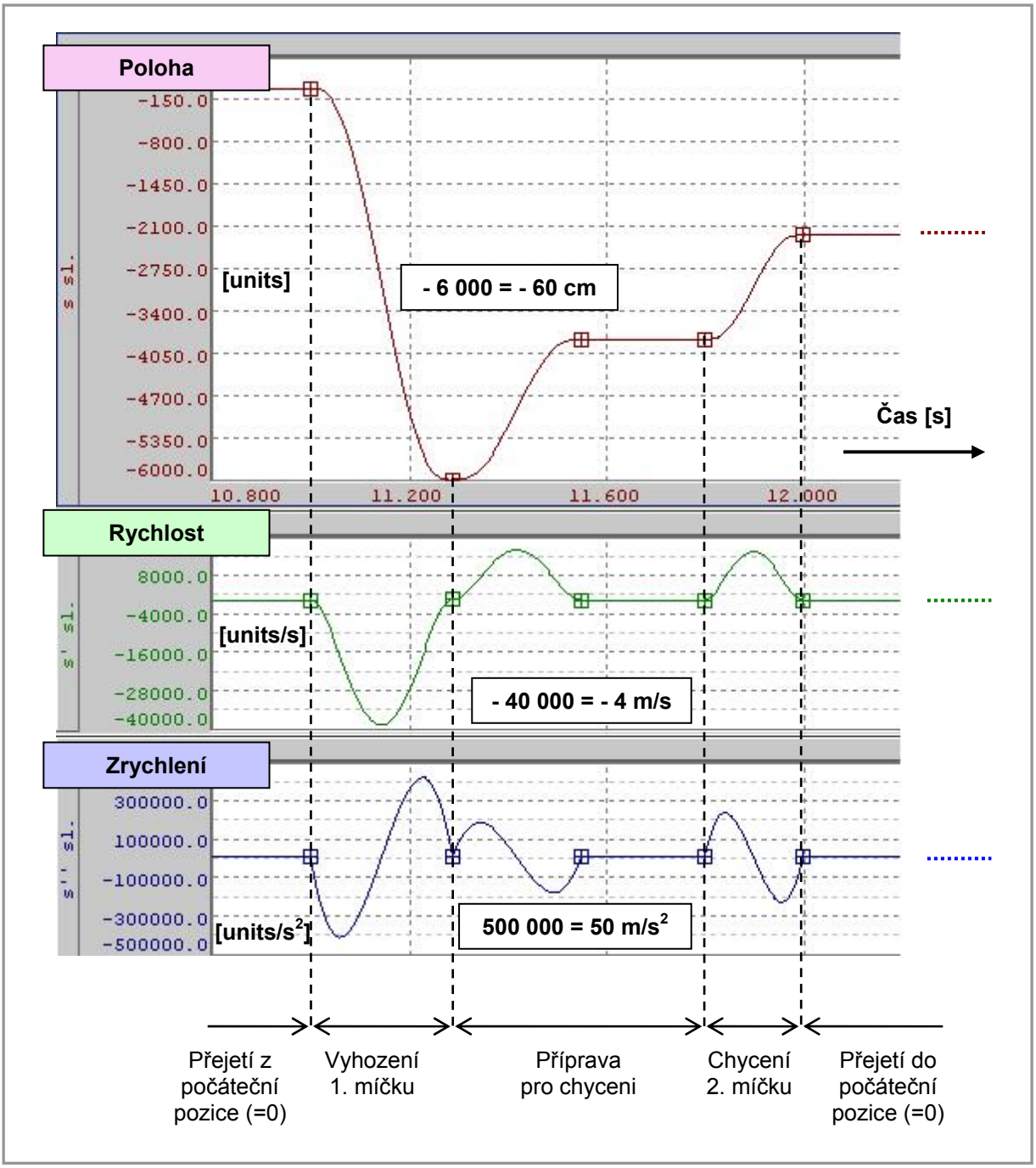
Vačkový profil – „vackaXY“ *X..adresa servozesilovače Y..typ přehození*	***Popis funkce profilu (trajektorie)***
00	Motor stojí
11, 21, 31 a 41	Realizace přehození ze strany B na stranu A 31, 41 - vyhození míčku 11, 21 - chycení míčku
12, 22, 32 a 42	Realizace přehození ze strany A na stranu B 12, 22 - vyhození míčku 32, 42 - chycení míčku
13, 23, 33 a 43	Přehazování jednoho míčku mezi moduly (složeno z profilů předchozích funkcí, které jsou spojeny za sebou)
14, 24, 34 a 44	Přehazování dvou míčku mezi moduly (trajektorie jsou realizovány na základě profilů předchozích funkcí, které jsou propojeny do sebe)

Tab. 6.2 – Popis funkce vytvořených vačkových profilů

6.2.8.2. Popis vybraného vačkového profilu

Všechny trajektorie polohy, rychlosti i zrychlení začínají na nulové hodnotě, do které se během cyklu vrací.

Vačkový profil (vacka44) realizující pohyb motoru pro vyhození jednoho míčku z jedné strany na druhou a současně chycení druhého míčku, je popsán na obr. 6.23. Doba jednoho cyklu je 20 s. Vačkový profil se po uplynutí této doby spustí od začátku.



Obr. 6.23 – Popis vybraného vačkového profilu

6.3. Výpočty parametrů pohybů

6.3.1. Výpočet rychlosti otáčení

Pro výpočet rychlosti jednotlivých motorů v m/s musíme znát průměr otáčení jednotlivých motorů a počet jednotek na jednu otočku (viz. tab. 6.3).

Parametr	***Směr***	***Hodnota***
Průměr otáčení	Vertikální	0,092 m
	Horizontální	0,30 m
Počet jednotek na otočku za 1s	Vertikální	2900 units / 1s
	Horizontální	3600 units / 1s
Směr	***[units /s] = [m/s]***	
Vertikální - pojezd	2900	0,290
	10000	1
Horizontální – držák míčku	3600	0,942

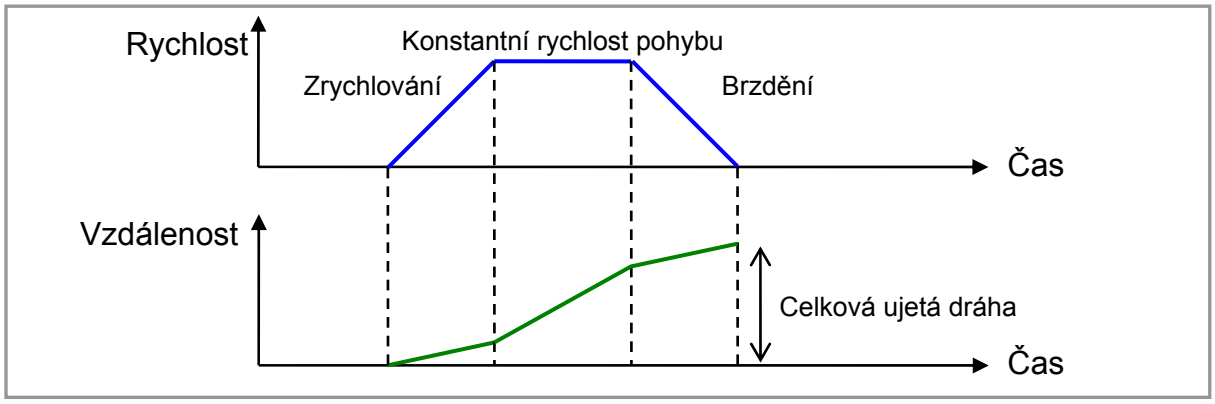
Tab. 6.3 – Výpočet rychlosti motorů v m/s

6.3.2. Výpočet parametrů funkce katapultu

Funkce katapult se skládá ze tří relativních pohybů: vyhození míčku, jeho chycení a chycení při jeho upuštění. Pro každý relativní pohyb je třeba zadat dva vstupní parametry : rychlost pohybu a dráhu, jakou má urazit. Relativní pohyb tedy spočívá v ujetí zvolené vzdálenosti danou rychlostí.

Výpočet těchto parametrů je počítán podle vzorců pro svislý vrh vzhůru a dolů.

Ideální průběh jednoho relativního pohyby se dá popsat dle obr. 6.24. Ve skutečnosti tento pohyb obsahuje překmity i zpoždění požadované hodnoty, které jsou závislé na nastaveném řídicím regulátoru servozesilovače a rychlosti pohybu.



Obr. 6.24 – Relativní pohyb

Počáteční požadovaná rychlost a dráha vyhození míčku se nemění a jsou pevně stanoveny na 5m/s resp. 0,6m. Skutečnou rychlost vyhození jsem získal po

změření parametrů pohyby příslušného motoru pomocí diagnostického nástroje „Trace“ (součást Automation Studia). Dále bylo zapotřebí změřit dobu a dráhu brzdění (viz. tab. 6.4) od místa maximální rychlosti do místa klidu, protože míček se v bodě maximální rychlosti odpoutá od držáku a než se motor zastaví, urazí míček nějakou vzdálenost.

Parametry pro „vyhození“	***Označení***	***Hodnota***
Celková dráha při vyhození (daná)	–	0,6 m
Skutečná rychlost vyhození (změřená)	v	4,95 m/s
Čas brzdění (změřená)	t_B	0,15 s
Dráha brzdění (změřená)	s_B	0,265 m

Tab. 6.4 – Parametry pro vyhození míčku (funkce katapult)

Výpočet parametrů pro chycení koule ve výšce výhozu (výsledky viz. tab. 6.5):

$$h = v^2/2g, \quad t_H = \sqrt{2h/g}, \quad t_{CH} = t_H - t_B, \quad s_{CH} = h - s_B, \quad v_{CH} = s_{CH}/t_{CH}$$

Skutečná rychlost vyhození se liší od vypočítané (z důvodu zrychlení a brzdění).

$$v_{CHSK} \approx v_{CH} * 1.4$$

Spočítané parametry pro „chycení“	***Označení***	***Hodnota***
Doba letu míčku	t_H	0,505 s
Doba pro chycení	t_{CH}	0,355 s
Výška vyhození míčku	h	1,24 m
Dráha pro chycení	s_{CH}	0,98 m
Rychlost chycení (potřebná)	v_{CHSK}	4,0 m/s

Tab. 6.5 – Spočítané parametry pro chycení míčku při jeho vyhození (funkce katapult)

Vypočtené parametry (viz. tab. 6.5) dráha pro chycení a rychlost chycení byly použity jako vstupní parametry relativního pohybu pro část chycení při vyhození.

Při výpočtu parametrů potřebných pro chycení míčku při jeho upuštění, známe výšku, ze které míček pouštíme a výšku chycení (viz. tab. 6.6).

Počáteční parametry pro „upuštění“	***Označení***	***Hodnota***
Výška upuštění	h	1,6 m
Výška chycení (zadávaná z OP)	h_U	0,6 m

Tab. 6.6 – Počáteční parametry pro upuštění míčku (funkce katapult)

Výpočet parametrů pro chycení míčku při jeho upuštění (výsledky viz. tab. 6.7):

$$s_U = h - h_U, \quad t_U = \sqrt{2s_U/g}, \quad v_U = s_U/t_U$$

Skutečná rychlost nutná pro chycení se liší od vypočtené (z důvodu zrychlení a brzdění). $v_{USK} \approx v_U * 1.1$

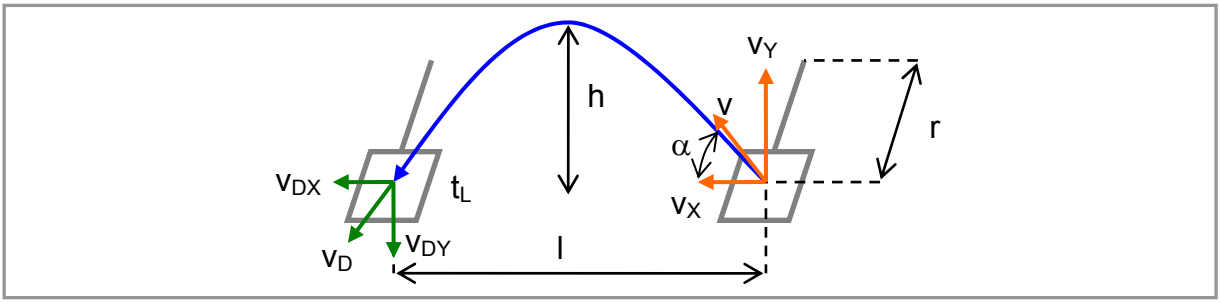
Spočítané parametry pro „chycení“	*Označení*	*Hodnota*
Doba pádu koule	t_U	0,452 s
Dráha pohybu pro chycení	s_U	1,0 m
Rychlost pro chycení při upuštění	v_{USK}	2,44 m/s

Tab. 6.7 – Spočítané parametry pro chycení míčku při jeho upuštění (funkce katapult)

Spočítané parametry rychlost chycení a dráha pro chycení byly opět použity jako vstupní parametry relativního pohybu pro část chycení při upuštění.

6.3.3. Výpočet parametrů funkce žonglér

Funkce žonglér je realizována pomocí vačkových profilů. Parametry (viz. obr. 6.25 a tab.6.8) potřebné pro vytvoření těchto profilů, jsou vypočteny podle vzorců pro šikmý vrh. Je třeba brát v úvahu, že ve skutečnosti se vačkové profily chovají jinak, než jak nadefinované. Dosahují větší rychlosti pohybu přibližně o 15% a pozdějšího ustálení požadované polohy.



Obr. 6.25 – Trajektorie přehození

Parametry	*Označení*	*Hodnota*	
Vzdálenost držáků	l	0,3 m	
Poloměr otáčení držáku míčku	r	0,15 m	
Rychlost vyhození – vertikální směr Skutečná / definovaná	v_Y	3,8 / 3,37 m/s	4,73 / 3,88 m/s
Rychlost vyhození – horizontální směr Skutečná / definovaná	v_X	0,39 / 0,35 m/s	0,31 / 0,28 m/s
Úhel výhozu	α	84°	86°
Výška výhozu	h	0,74 m	1,08 m
Doba letu	t_L	0,77 s	1,14 s
Skutečná rychlost v místě dopadu	v_D	3,82 m/s	4,75 m/s

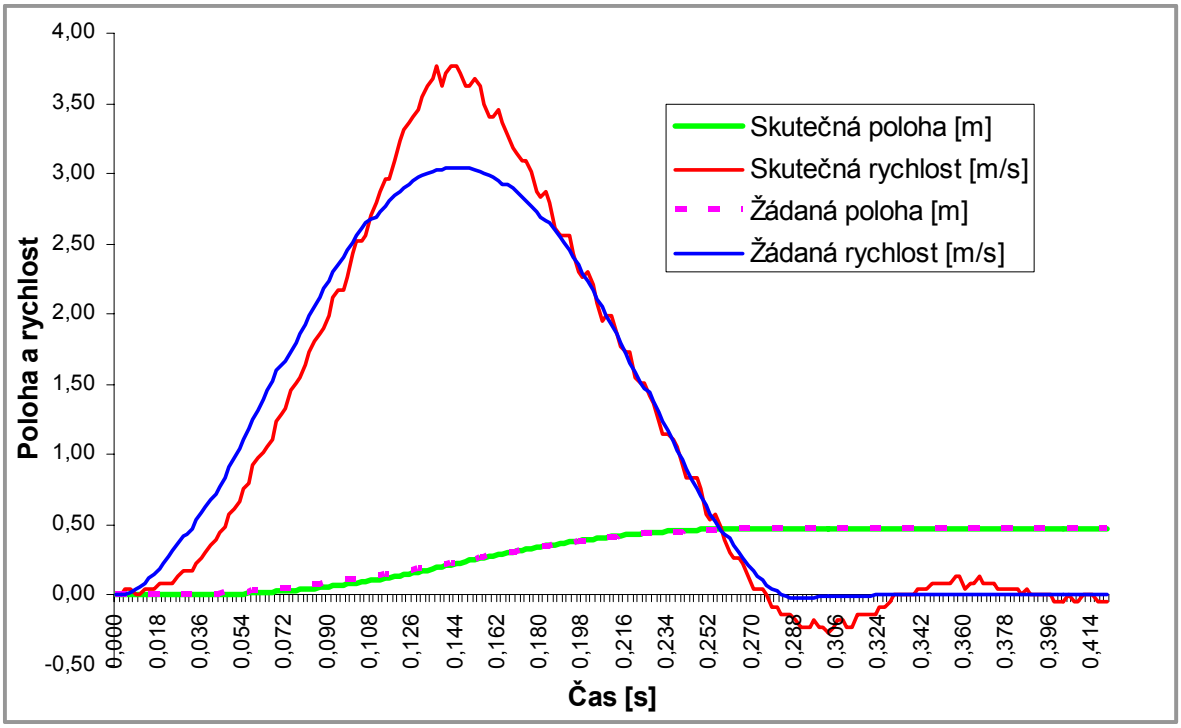
Tab. 6.8 – Parametry pro přehazování (funkce žonglér)

Vzorce pro spočítání parametrů:

$$t_L = 2v * \sin \alpha / g, \quad h = v^2 \sin^2 \alpha / 2g, \quad v_{DX} = v \cos \alpha, \quad v_{DY} = v \sin \alpha - gt$$

Na základě spočítaných parametrů byly vytvořeny trajektorie pohybu pro všechny motory.

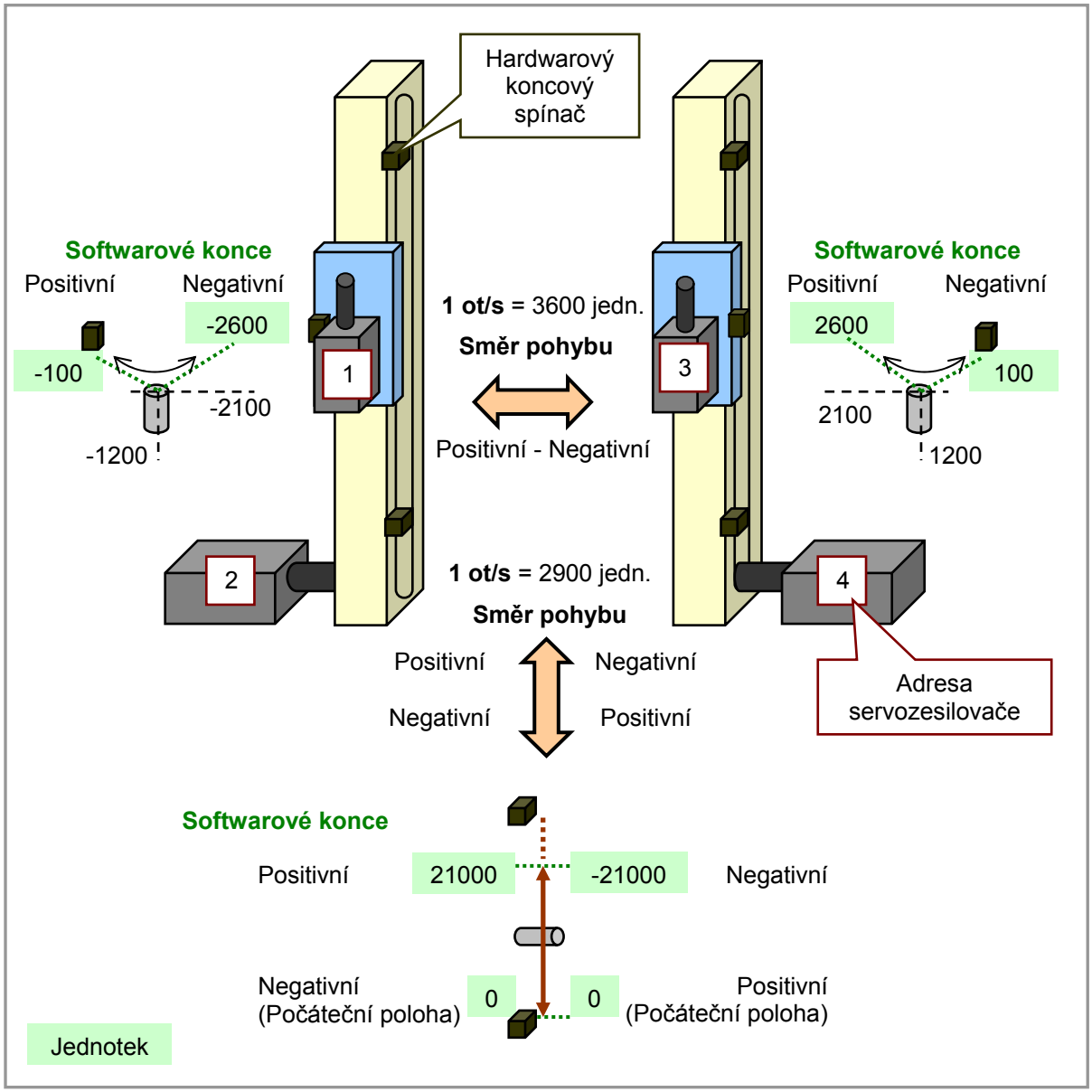
Porovnání parametrů rychlosti a polohy mezi definovaného vačkového profilu a skutečnými změřenými hodnotami je znázorněno na obr. 6.26.



Obr. 6.26 – Porovnání skutečných a žádaných parametrů vačkových profilů

6.4. Popis důležitých parametrů a dat modelu

Při ovládání modelu a realizaci řídicí aplikace je třeba brát v úvahu směr otáčení motorů a z toho vyplývající typy hardwarovýh spínačů, hodnoty softwarových konců, typy pohybů a data potřebná k jejich realizaci a další. Nejdůležitější informace jsou popsány na obr. 6.27.



Obr. 6.27 – Popis důležitých parametrů a dat modelu

Příslušné hardwarové koncové spínače plní kromě funkce pro hlídání přejetí (zastavení motoru) i funkci pro nastavení motoru do počáteční polohy.

6.5. Ukázkové zadání úlohy pro studenty

Naprogramujte řídicí algoritmus realizující vyhození míčku skrz překážku a jeho chycení v definované výšce. Vyhození míčku proveďte pomocí motorů a servozesilovačů jedné strany modelu (č. 1 a 2) a chycení pomocí druhé (č. 3 a 4).

Řízení realizujte pomocí řídicích akcí (nc-actions) nebo více stavovými vačkovými automaty a vačkovými profily.

Doporučení:

1. Pomocí komponenty Test otestujte parametry řízené osy - motoru (parametry regulátoru servozesilovačů, limitní hodnoty, nastavení spínačů, parametry pohybů atd.). Je nutné mít vytvořenou hardwarovou konfiguraci, není potřeba mít napsaný řídicí program.
2. Pro řízení pohybu motorů použijte řídicí příkazy (nc-actions) nebo knihovnu polohovacích funkcí (libacp10cz), která obsahuje funkce pro řízení reální i virtuální osy pomocí základních pohybů, inicializaci parametrů, realizaci vačkového automatu (nahrání vačkových profilů, nastavení struktury atd.) a jiné.
3. Pomocí diagnostického nástroje Trace zjistěte skutečnou hodnotu rychlosti a polohy pohybu realizovaného pomocí akcí a vačkových profilu. Na základě změřených skutečných parametrů spočítejte potřebné parametry pro chycení míčků.

6.5.1. Postup pro řízení pomocí akcí (nc-actions)

Jednotlivé akce je vhodné psát do stavového automatu a v každém kroku provést jen jednu akci. Přechod do dalšího stavu povolit, jen pokud se daná akce vykonala úspěšně (inicializace, nastavení) nebo pokud se vykonává (pro pohyby).

Níže je uveden popis akcí, které musím provést abych vykonal určitou funkci, její zápis a doporučený stav (status) pro přechod do dalšího stavu automatu. Pro každou volanou funkci je potřeba provést nastavení parametrů, které se provede jednou pomocí globální inicializace nebo kdykoliv pomocí inicializační akce každé skupiny.. Akce jsou sepsány postupně tak jak se mají v programu volat.

Deklarace proměnných:

```
_GLOBAL   ACP10AXIS_typ    * p_ax_dat;   - datová struktura řízené osy
_GLOBAL   UDINT            * ax_obj;     - adresa vytvořené struktury osy
```

Vytvoření datové struktury (v inicializační části úlohy):

```
status=ncaccess1 (ncACP10MAN,"název osy v Deployment Table",&ax_obj);
p_ax_dat = (ACP10AXIS_typ*)ax_obj;
```

Globální inicializace všech parametrů (může se provést pokud není zapnutá řídicí jednotka servozesilovače):

```
status = ncaction(ax_obj,ncGLOBAL,ncINIT);
if (p_ax_dat->global.init == ncTRUE)
```

Zapnutí řídicí jednotky servozesilovače, jen pokud je připraven:

```
If (p_ax_dat->controller.ready == ncTRUE)
status = ncaction(ax_obj,ncCONTROLLER,ncSWITCH_ON);
if (p_ax_dat->controller.status == ncON)
```

Nastavení motoru do počáteční polohy:

```
status = ncaction(ax_obj,ncHOMING,ncSTART);
if (p_ax_dat->move.homing.status.ok == ncTRUE)
```

Teď už můžeme volat funkce jednotlivých pohybů. Pro každý pohyb se některé parametry nastavují automaticky a není potřeba provést inicializaci.

Inicializace všech pohybových parametrů se provádí:

```
p_ax_dat->move.basis.parameter.v_pos  = ...;
p_ax_dat->move.basis.parameter.v_neg  = ...;
p_ax_dat->move.basis.parameter.a1_pos = ...;
status = ncaction(ax_obj,ncBASIS_MOVE,ncINIT);
if (p_ax_dat->move.basis.init == ncTRUE)
```

Spuštění vybraného pohybu (positivní resp. negativní směr, absolutní, relativní):

```
p_ax_dat->move.basis.parameter.v_pos = ...;
status = ncaction(ax_obj,ncPOS_MOVE,ncSTART);
status = ncaction(ax_obj,ncNEG_MOVE,ncSTART);
if ( action_status == ncOK )
p_ax_dat->move.basis.parameter.s = ...;
status = ncaction(ax_obj,ncABS_MOVE,ncSTART);
status = ncaction(ax_obj,ncREL_MOVE,ncSTART);
if (p_ax_dat->move.basis.status.in_pos = ncTRUE)
```

Zastavení pohybu:

```
status = ncaction(ax_obj,ncMOVE,ncSTOP);
```

Čtení identifikačního čísla vzniklé poruchy podle kterého vyhledáme její přesný popis a její potvrzení:

```
Porucha = message.record.number
status = ncaction(ax_obj,ncMESSAGE,ncACKNOWLEDGE);
```

Změna jednotlivých parametrů lze provádět i pomocí inicializační akce, pro jednotlivé skupiny parametrů. Volání lze provádět průběžně

Nastavení parametrů regulátoru:

```
p_ax_dat->controller.position.kv = ...;
p_ax_dat->controller.position.p_max  = ...;
p_ax_dat->controller.speed.kv  = ...;
p_ax_dat->controller.speed.tn = ...;
status = ncaction(ax_obj,ncCONTROLLER,ncINIT);
```

Nastavení mezních parametrů pohybu:

```
p_ax_dat->limit.parameter.v_pos      = ...;
status = ncaction(ax_obj,ncLIMITS,ncINIT);
```

Poznámky:

Při inicializaci jednotlivých skupin parametrů nemusí být vypnuta řídicí jednotka servozesilovače (vypnutá jen při globální inicializaci).

Volání ostatních akcí se provádí podobně.

Všechny akce, kromě vytvoření datové struktury se volají v cyklické části programu.

Přesný popis zápisu akcí a význam nastavovaných parametrů je popsán v helpu Automation Studia v části: B&R Software world / Automation Studio / NC Software / ACP10 / NC Software

6.5.2. Řízení pomocí funkcí knihovny libacp10cz

Práce s funkcemi knihovny libacp10cz se provádí změnou hodnoty parametrů (např. změna rychlosti, povel pro pohyb, nahrání vačkových profilu, spuštění automatu a další) z datové struktury každé funkce. Není zapotřebí psát stavový automat jako pro řízení pomocí akcí, to za nás obstará daná funkce.

Funkce „*RealAxis“ (VirtualAxis)* automaticky zajistí pomocí akcí vytvoření datové struktury dané osy, inicializaci servozesilovače, nastavení všech parametrů pro motor i servozesilovač. Dále umožňuje ovládání osy pomocí polohových příkazů.

Funkce „*DownCAM“* realizuje nahrání vačkových profilů do servozesilovače a funkce „*CamAxis“* zajišťuje vytvoření struktury vačkového automatu, jeho parametrizaci a ovládání. Tyto funkce používají k realizaci příkazy vačkového automatu a řídicí akce.

Příklad ovládání pomocí funkcí knihovny libacp10cz:

Deklarace:

```
_GLOBAL   RealAxis_typ      osaServo;      - reálná osa
_LOCAL    DownCamProf_typ   downCamProf;   - vačkové profily
_LOCAL    CamAxis_typ       camServo;      - vačkový automat
```

Volání funkcí v cyklické části:

```
RealAxis(&osaServo); CamAxis(&camServo); DownCamProf(&downCamProf);
```

Zápis vybraných parametrů:

```
osaServo.param.speed =  500;      - změna rychlosti osy
osaServo.cmd.absMove = 1;         - spuštění absolutního pohybu
camServo.cmd.start = 1;           - spuštění vačkového automatu
downCamProf.cmd.download = 1;     - nahrání vačkového profilu
```

Čten vybraných parametrů:

```
osaServo.info.moveActive == 1;    - vyhodnocení aktuálního stavu pohybu
aktuální_poloha = osaServo.info.position;
```

Stejným způsobem se provádí zápis a čtení ostatních parametrů datové struktury každé funkce z knihovny.

6.5.2.1. Popis softwarové realizace vačkového automatu

Protože je softwarová realizace vačkového automatu složitá a časově náročná na nastudování je zde uveden popis části programu, který má na starost nahrání vačkového profilu do servozesilovače, vytvoření struktury vačkového automatu (viz. obr. 6.13) a její nahrání do servozesilovače.

Nahrání vačkového profilu:

```
index vačkového profilu (1-10)        - downCamProf.param.camIndex = 0x0001;
název příslušného vačkového profilu   - strcpy(downCamProf.param.camName,
povel pro nahrání vačkového profilu   -downCamProf.cmd.download = 1;
```

Definice struktury automatu a její nahrání:

Smazání starých parametrů -

```
memset(&camServo1.param.params, 0, sizeof(CamAutomatParameters_typ));
```

-

Definováni typu master osy (liší se pro servozesilovač č.1, ve kterém běží virtuální osa a ostatní servozesilovače) -

```
camServo1.param.params.MA_AXIS = ACP10PAR_S_SET_VAX1;
camServo2.param.params.MA_AXIS = ACP10PAR_MA1_CYCLIC_POS;
```

Parametry pro master osu:

Startovací poloha - camServo1.param.params.MA_S_START = 0;
Startovací perioda - camServo1.param.params.MA_IVSTART = 1;
Maximální rychlost - camServo1.param.params.MA_V_MAX = 10;

Definování základního stavu automatu:

#define STATE 0

Nastaveni typu události pro přechod do dalšího stavu (na konci stavu)

camServo1.param.params.state[STATE].event[0].EVENT_TYPE = ncST_END;
camServo1.param.params.state[STATE].event[0].EVENT_ATTR = ncST_END;

Index následujícího stavu -

camServo1.param.params.state[STATE].event[0].EVENT_ST_INDEX = 1;
#undef STATE

Definování prvního stavu automatu:

#define STATE 1

Index vačkového profilu vykonávaného v tomto stavu -

camServo1.param.params.state[STATE].ST_DATA_INDEX =0x0001;

Nastaveni typu události pro přechod do dalšího stavu (po vykonání profilu) -

camServo1.param.params.state[STATE].event[0].EVENT_TYPE =ncST_END;

Index následujícího stavu -

camServo1.param.params.state[STATE].event[0].EVENT_ST_INDEX = 1;
#undef STATE

Nahrání automatu do servozesilovače:

camServo1.cmd.parametrize = 1;

7. VIZUALIZACE

Nedílnou součástí každého řídicího systému je jeho vizualizace, tzn. sledování nebo ovládání systému pomocí uživatelského rozhraní. Ve většině případů je vizualizace umístěna v bezprostřední blízkosti systému (tatáž místnost, budova).

Tato kapitola je zaměřena na vytvoření vzdálené vizualizace a vizualizace pomocí operátorského panelu .

Vzdálenou vizualizací rozumíme vizualizaci, která k danému systému přistupuje přes počítačovou síť internet. Jelikož je tato problematika značně rozsáhlá, zaměříme se speciálně na vizualizaci pomocí webových stránek a pomocí VNC (Virtual Network Computing).

Použití webových stránek jako grafické uživatelské rozhraní (GUI) aplikací je trend poslední doby a přináší značné úspory při vývoji aplikací a hlavně při následné údržbě. Přístupnost webových stránek znamená, že stejné stránky je možné bez změny kódu prohlížet na různých typech zařízení od osobních počítačů, přes mobilní počítače typu PDA až po mobilní telefony s implementací www prohlížeče. Pro realizaci vizualizace pomocí webových stránek je zapotřebí nastavit webový server, který je součástí řídicí aplikace a obstarává přenos dat.

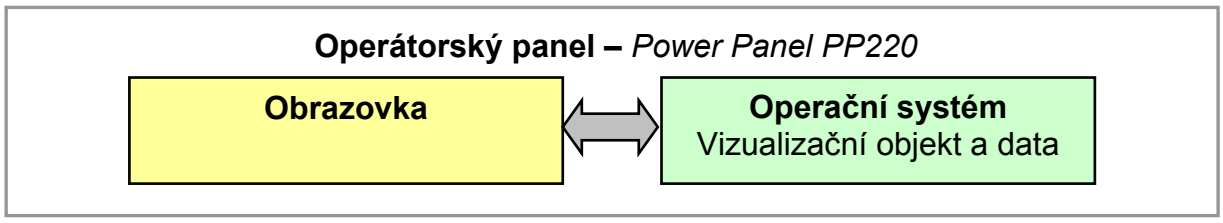
VNC (Virtual Network Computing) reprezentuje vzdálený řídicí software umožňující vizualizaci a ovládání jednoho zařízení (serveru), kterým může být počítač nebo průmyslový operátorský panel, pomocí jednoduchého programu (viewer) na jiném počítači.

Vizualizace pomocí operátorského panelu rozumíme vizualizaci, která bude prováděna z tohoto panelu umístěného v blízkosti řízeného modelu.

7.1. Vizualizace pomocí operátorského panelu

Pro místní vizualizaci pomocí operátorského panelu byla použita řídicí jednotka Power Panel PP200 s barevnou dotykovou LCD – QVGA obrazovkou.

Vizualizace se vytváří pomocí programu Automation Studio a její vizualizační komponenty [10]. Abychom mohli vytvořit vizualizaci pomocí panelu musí naše řídicí aplikace obsahovat tzv. vizualizační objekt (viz. obr. 7.1). Tento objekt obsahuje veškeré informace a parametry vizualizačních obrazovek. Data pro vizualizaci jsou potom definována přímo v aplikaci pomocí symbolických jmen.



Obr. 7.1 – Princip vizualizace pomocí operátorského panelu

7.1.1. Objekt pro vizualizaci

Pro možnost tvorby vizualizace musí řídicí aplikace obsahovat speciální vizualizační objekt typu „Visual components SG3“, který obsahuje veškeré informace o vizualizačních obrazovkách panelu (jednotlivé prvky, možnosti tlačítek a další).

Tento objekt se do programu vloží automaticky při jeho vytvoření, pokud jako řídicí systém definujeme modul obsahující vizualizační obrazovku. Nestane–li se tak, je tento objekt potřeba vložit do aplikace.

Po otevření tohoto objektu se spustí vizualizační komponenta prostředí Automation Studio pro tvorbu jednotlivých obrazovek.

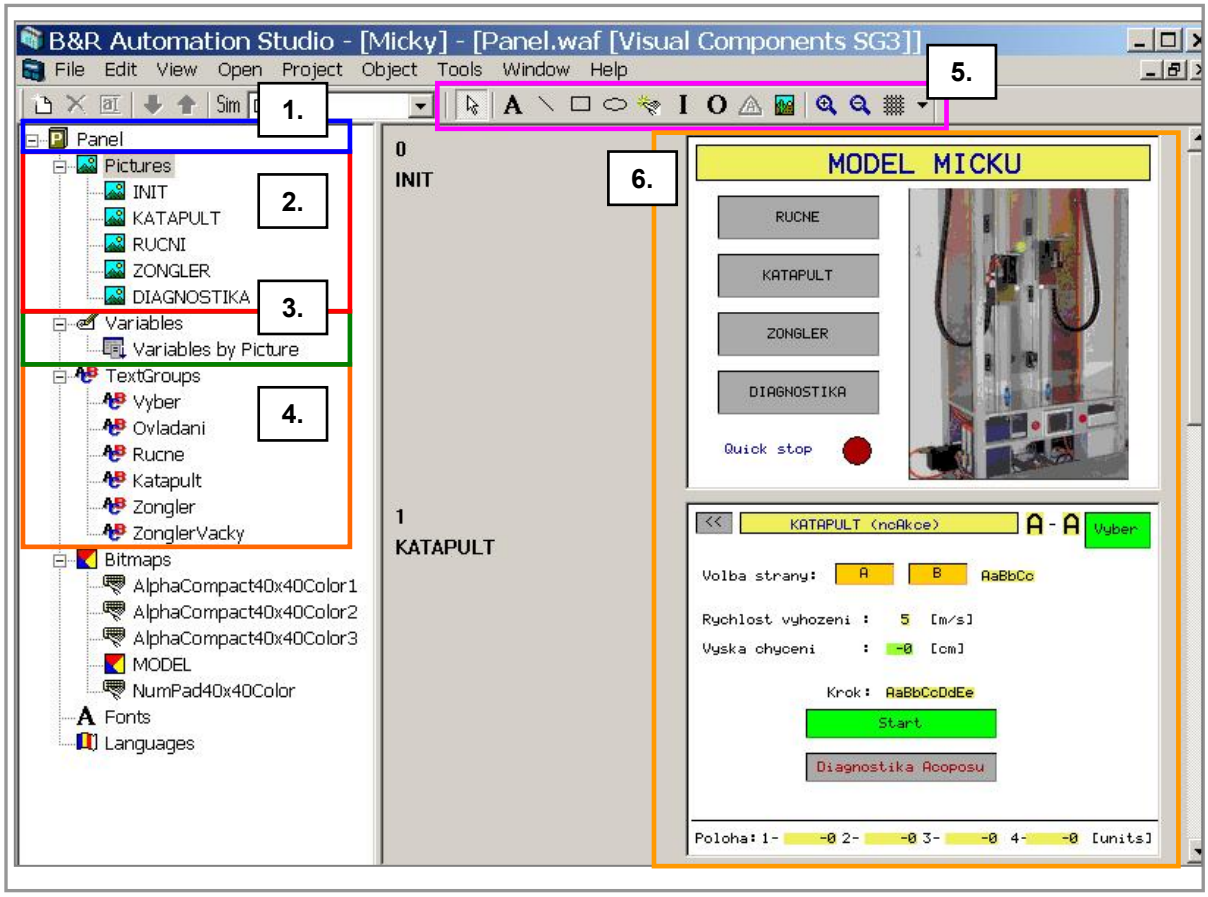
7.1.2. Prostředí pro tvorbu vizualizace

Prostředí pro tvorbu vizualizace se nám zobrazí po otevření vizualizačního objektu v naší řídicí aplikaci (programu). Tato vizualizační komponenta se používá ke konfiguraci textových a grafických vizualizačních projektů.

Vytvořené obrazovky mohou obsahovat vstupní a výstupní pole, statické nebo dynamické bitové mapy a texty.

Vývojové prostředí lze rozdělit do několika částí (viz. obr. 7.2):

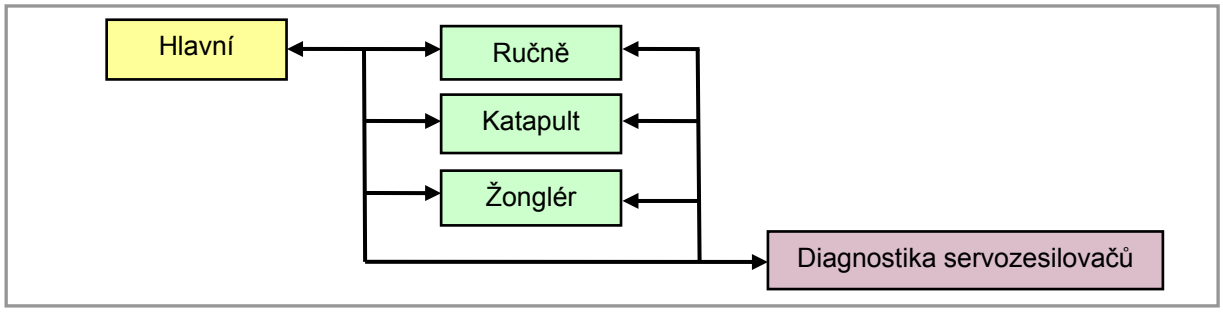
1. Panel – po kliknutí na něj se nám zobrazí obrazovka, kde se nastavují vlastnosti a parametry vizualizační obrazovky
2. Seznam vizualizačních obrazovek – po kliknutí na vybranou obrazovku se nám tato obrazovka objeví v pravé části
3. Seznam použitých proměnných, jejich umístění (obrazovka) a typ (čtení nebo zápis)
4. Seznam textů tlačítek, která lze rozdělit do jednotlivých skupin
5. Funkční klávesy, pomocí kterých se zadávají komponenty do obrazovky (kurzor, text, čára, elipsa, čtyřúhelník, tlačítko, vstupní pole, výstupní pole, bitmapa, zoom a vlastnosti rastru)
6. Okno pro tvorbu obrazovek



Obr. 7.2 – Prostředí pro tvorbu vizualizace operátorského panelu

7.1.3. Popis obrazovek operátorského panelu

Vizualizace pomocí operátorského panelu se skládá z pěti obrazovek, kde každá slouží k určité funkci. Na obr. 7.3 je znázorněn diagram přechodu mezi obrazovkami.

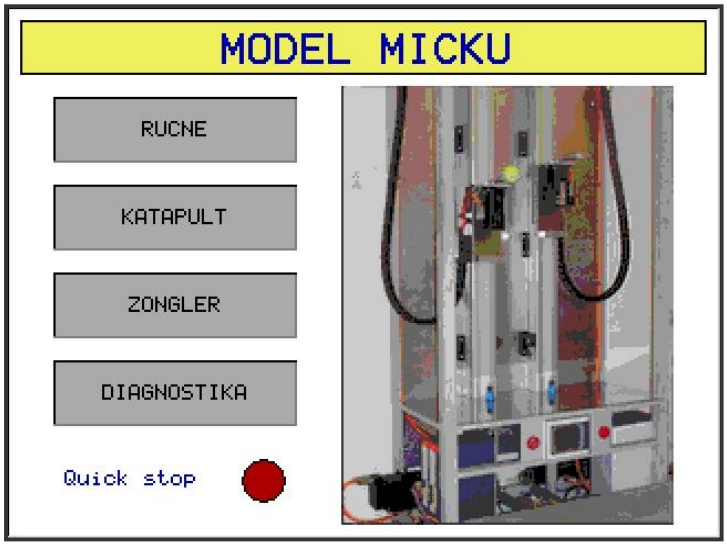


Obr. 7.3 – Struktura přechodu mezi obrazovkami panelu

Všechny obrazovky obsahují vybrané ovládací a informační prvky, kterými jsou: tlačítka, text, informační a zadávací pole.

7.1.3.1. Hlavní

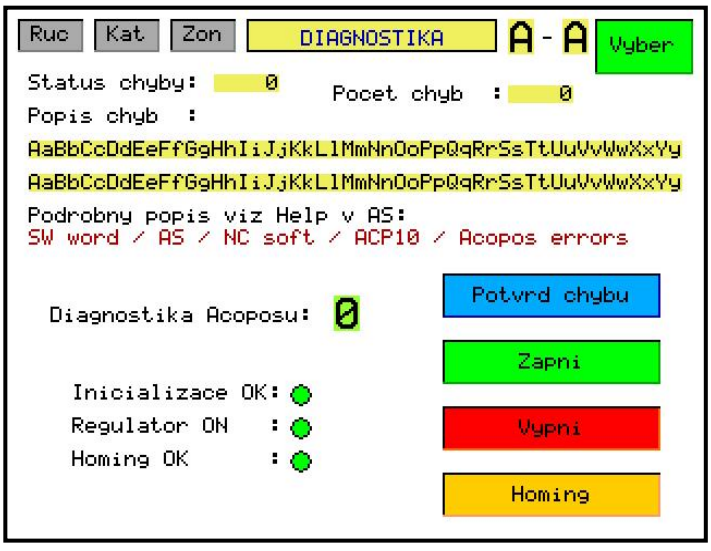
Hlavní obrazovka (viz. obr. 7.4) se zobrazí po spuštění řídicího systému (operátorského panelu). Slouží k výběru obrazovek jednotlivých funkcí pomocí tlačítek „Ručně, Katapult, Žongler a Diagnostika“. Dále obsahuje signalizaci stavu tlačítka okamžitého zastavení (Quickstop) motorů (červená ledka znamená, že model není možno ovládat – tlačítko je sepnuto).



Obr. 7.4 – Hlavní obrazovka panelu

7.1.3.2. Diagnostika servozesilovačů

Obrazovka diagnostiky (viz. obr. 7.5) slouží k získávání informací o aktuálním stavu vybraného servozesilovače a k jeho ovládání. Popis prvků obrazovky diagnostiky je v tab. 7.1.



Obr. 7.5 – Obrazovka pro funkci diagnostika servozesilovače

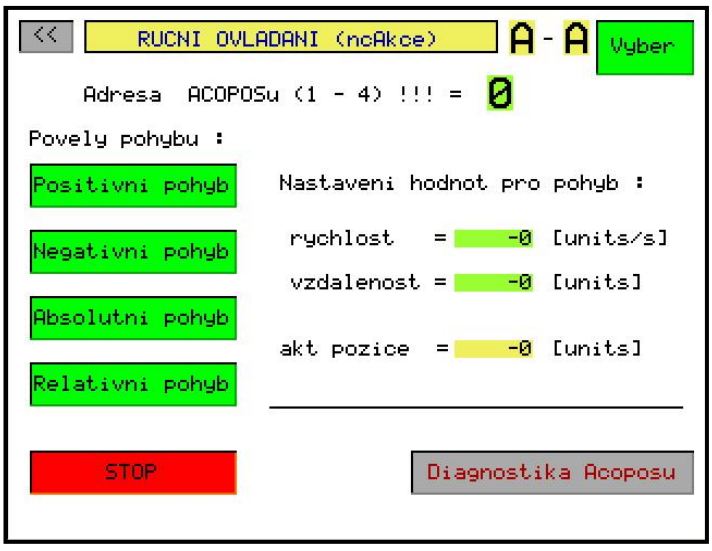
Popis prvků obrazovky diagnostika:

Typ	Název	Funkce
Informační pole	Místo ovládání	Informuje o místu ovládání P=operátorský panel a W=webové stránky
	Funkce modelu	Vybraná funkce – R=ručně, K=katapult, Z=žonglér a D=diagnostika
	Status chyby	Popisné ID číslo chyby
	Počet chyb	Počet poruchových stavů servozesilovače (napájení)
	Popis chyby	Podrobný popis chyby servozesilovače
Zadávací pole	Diagnostika servozesilovače	Adresa diagnostikovaného servozesilovače (1–4 pro servozesilovač, 5 pro virtuální osu)
Tlačítko	<<	Navrácení na hlavní obrazovku
	Ruc, Kat, Zon	Přechod na obrazovku vybrané funkce
	Vyber	Potvrzení funkce diagnostika
	Potvrď chybu	Potvrdí existující chybu
	Zapni	Zapnutí řídicí jednotky servozesilovače (pokud není žádná chyba)
	Vypni	Vypnutí řídicí jednotky servozesilovače
	Homing	Nastavení motoru do počáteční polohy
Signalizace (zelené LED)	Inicializace Ok	Provedla se inicializace motoru
	Regulátor On	Regulátor se zapnul
	Homing Ok	Provedlo se nastavení do počáteční polohy

Tab. 7.1 – Prvky obrazovky funkce diagnostika servozesilovače

7.1.3.3. Ruční ovládání

Obrazovka ručního ovládání (viz. obr. 7.6) slouží k ovládání jednoho zvoleného motoru pomocí základních pohybů, popis prvků obrazovky je v tab. 7.2.



Obr. 7.6 – Obrazovka funkce ruční ovládání

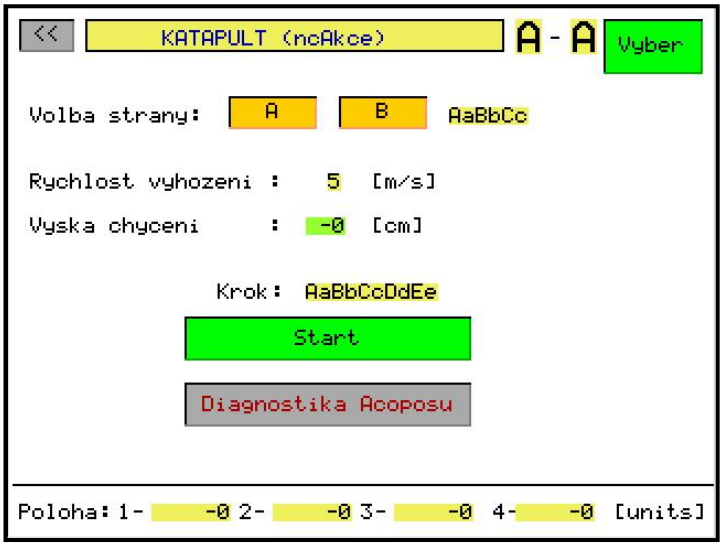
Popis prvků obrazovky ručně:

Typ	Název	Funkce
Informační pole	Místo ovládání	Informuje o místu ovládání P=operátorský panel a W=webové stránky
	Funkce	Vybraná funkce – R=ručně, K=katapult, Z=žonglér a D=diagnostika
	Akt. Pozice	Informuje o aktuální pozici řízeného motoru [units]
Zadávací pole	Adresa	Adresa řízeného motoru (1 – 4)
	Rychlost	Rychlost, jakou se bude motor otáčet [units/s]
	Vzdálenost	Vzdálenost otáčení na jakou pozici nebo o kolik [units]
Tlačítko	<<	Navrácení na hlavní obrazovku
	Vyber	Potvrzení funkce ručního ovládání
	Positivní pohyb	Zapnutí daného pohybu, předem nutno zadat rychlost otáčení a vzdálenost
	Negativní pohyb	
	Absolutní pohyb	
	Relativní pohyb	
	Stop	Zastavení daného pohybu
	Diagnostika ..	Přechod na obrazovku diagnostiky

Tab. 7.2 – Prvky obrazovky funkce ruční ovládání

7.1.3.4. Katapult

Obrazovka katapult (viz. obr. 7.7) slouží k ovládání modelu pomocí funkce katapult. Popis prvků obrazovky je v tab. 7.3.



Obr. 7.7 – Obrazovka ovládání funkce katapult

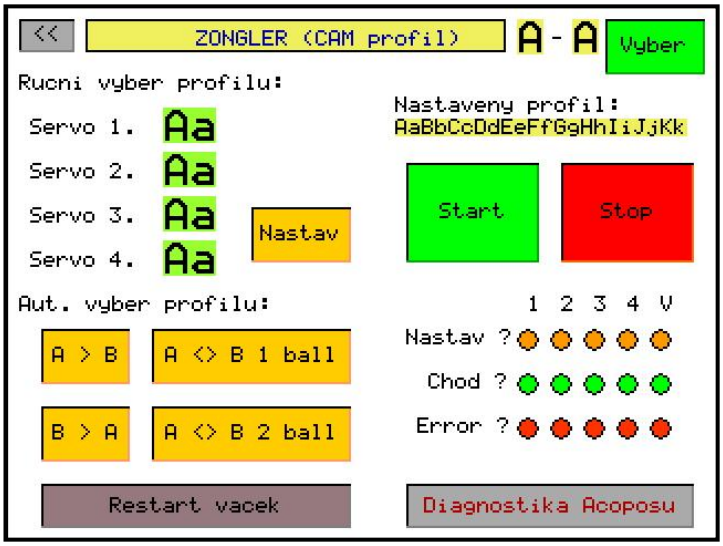
Popis prvků obrazovky ručně:

Typ	Název	Funkce
Informační pole	Místo ovládání	Informuje o místu ovládání P=operátorský panel a W=webové stránky
	Funkce	Vybraná funkce – R=ručně, K=katapult, Z=žonglér a D=diagnostika
	Strana	Informace o navolené straně
	Krok	Stav funkce v jaké se nachází (inicializace, vyhození, chycení atd.)
	Rychlost	Rychlost vyhození koule
	Poloha	Aktuální poloha všech motorů [units]
Zadávací pole	Výška chycení	Výška, v jaké se míček chytne, při jeho upuštění (1–100cm, 0 znamená, že se neprovede upuštění a míček se chytne nahoře a dolů se jenom přejede)
Tlačítko	<<	Navrácení na hlavní obrazovku
	Vyber	Potvrzení funkce ručního ovládání
	A, B	Vybrání str. modelu, která bude funkci vykonávat
	Start	Odstartování funkce katapult
	Diagnostika ..	Přechod na obrazovku diagnostiky

Tab. 7.3 – Prvky obrazovky funkce katapult

7.1.3.5. Žonglér

Obrazovka žonglér (viz. obr. 7.8) slouží k ovládání modelu pomocí vačkových profilů a funkce žonglér. Popis prvků obrazovky je v tab. 7.4.



Obr. 7.8 – Obrazovka ovládání funkce žonglér

Popis prvků obrazovky ručně:

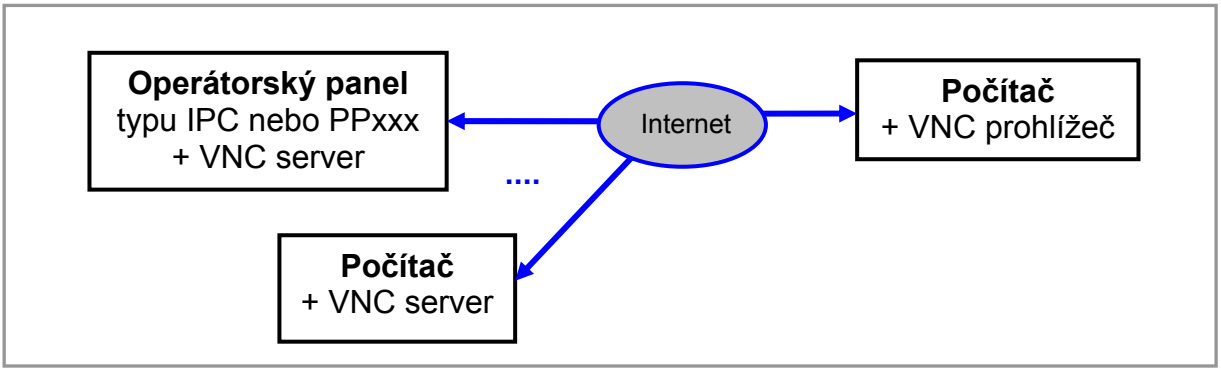
Typ	*Název*	*Funkce*
Informační pole	Místo ovládání	Informuje o místu ovládání P=operátorský panel a W=webové stránky
	Funkce	Vybraná funkce – R=ručně, K=katapult, Z=žonglér a D=diagnostika
	Nastavený profil	Informace o tom, jak byl profil nastaven
Zadávací pole	Servo 1,2,3 a 4	Ruční zadání vačkového profilu – např. 51, nutno znát co profil dělá
Tlačítko	<<	Navrácení na hlavní obrazovku
	Vyber	Potvrzení funkce vačkových profilů
	Nastav	Nastaví ručně navolené profily
	A>B B>A A<>B 1 ball A<>B 2 ball	Automaticky přiřadí profil pro všechny motory – přehození ze str. A na str. B – přehození ze str. B na str. A – přehazování tam a zpět jednoho míčku – přehazování tam a zpět dvou míčků současně
	Restart vaček	Při špatném ručním nastavení profilů, nutno potom zadat správné profily
	Start / stop	Spuštění / zastavení vykonávání funkce
	Diagnostika ..	Přechod na obrazovku diagnostiky
Signalizace LED	Nastav ?	Správně nastavené vačkové profily (oranžová)
	Chod ?	Servozesilovače i virtuální osa v chodu (zelená)
	Error?	Existuje chyba servozesilovače (přehřátí a další) nebo virtuální osy (výpadek komunikace Ethernet Powerlink) (červená)

Tab. 7.4 – Prvky obrazovky funkce žonglér

7.2. Vizualizace pomocí „Virtual Network Computing”

VNC znamená „Virtual Network Computing“ [17]. Jedná se o vzdálený řídicí software (viz. obr. 7.9) umožňující vizualizaci a ovládání jednoho zařízení (server), kterým může být počítač nebo průmyslový operátorský panel (v případě firmy B+R se jedná o operátorské panely typu IPC a PPxxx), pomocí jednoduchého programu (VNC prohlížeče) na jiném počítači. Komunikace probíhá prostřednictvím internetu. Komunikující zařízení dokonce nemusí být stejného typu. VNC je volně dostupné a je využívané, jak v průmyslu, tak v administrativě a v soukromém sektoru.

Po připojeni na VNC server operátorského panelu se pouze zobrazí aktuální obrazovka operátorského panelu. U novější verze operačního systému panelu (od 2.8), který bude podporován novou verzí Automation Studia půjde i aktivní prvky (tlačítka, zadávací prvky) ovládat, vizualizace bude obousměrná.



Obr. 7.9 – Vizualizace pomocí VNC (Virtual Network Computing)

Podmínkou pro vizualizaci a ovládání z počítače je mít na cílovém zařízení spuštěn VNC server. V případě operátorského panelu firmy B+R se server spustí pomocí funkce „VNC_Start()“ (viz. tab. 7.5) v aplikační úloze, která musí obsahovat příslušnou knihovnu (libvncserv.a). Systémový objekt serveru se do programu vloží automaticky při jeho překladu. Server je možné zastavit pomocí funkce „VNC_Stop()“.

U novější verze Automation Studia se server pouze nadefinuje v konfiguraci komunikaci Ethernet, nebude potřeba psát žádný kód.

Volání funkce		
Status=VNC_Start(rozlišení, port, obn_cas, jméno, typ_hw)		
Parametr	**Typ**	**Popis**
Rozlišení	USINT	Zobrazovaná rozlišení 1 - 640 x 480 (RES640x480) 2 - 320 x 240 (RES320x240) 3 - 800 x 600 (RES800x600) 4 - 1024 x 786 (RES1024x786)
Port	UINT	Číslo portu (defaultní pro VNC prohlížeč je 5900)
Obn_cas	USINT	Obnovovací čas 1 – Krátký (LOW) 2 – Střední (IMMEDIATE) 3 – Vysoký (HIGH) 4 – Velmi vysoký (VERYHIGH)
Jméno	STRING [80]	Jméno vizualizačního objektu (např. „Panel“)
Typ_HW	USINT	Typ hardwaru 0 - IPC (I386) 1 - PP100 (PPXXX)
Status	UINT	Status je 0 pokud neexistuje chyba

Tab. 7.5 – Funkce spuštění VNC serveru

Příklad volání:

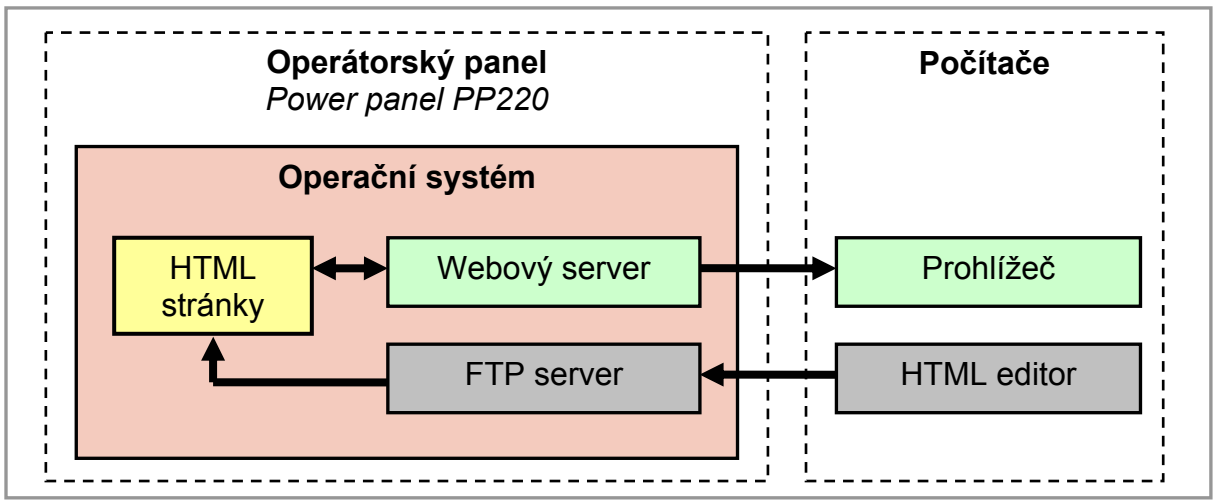
```
Status = VNC_Start(RES1024x768,5900,HIGH,"Panel",PPXXX);
Status = VNC_Stop();
```

7.3. Vizualizace pomocí webových stránek

K přístupu na vizualizační stránky modelu stačí do internetového prohlížeče zadat IP adresu řídicího systému (PLC). Jako první se nám zobrazí úvodní stránka s odkazovým menu na další stránky.

Stránky lze vytvořit pomocí libovolného editoru a do operátorského panelu se nahrají přes FTP.

Samotné webové stránky komunikují s PLC (operátorský panel) pomocí systémového objektu zvaného webový server. Webový server musí být vložen do řídicí aplikace a musí být správně nakonfigurován. Princip vizualizace pomocí webových stránek je znázorněn na obr. 7.10.



Obr. 7.10 – Princip vizualizace pomocí webových stránek

7.3.1. Webový server

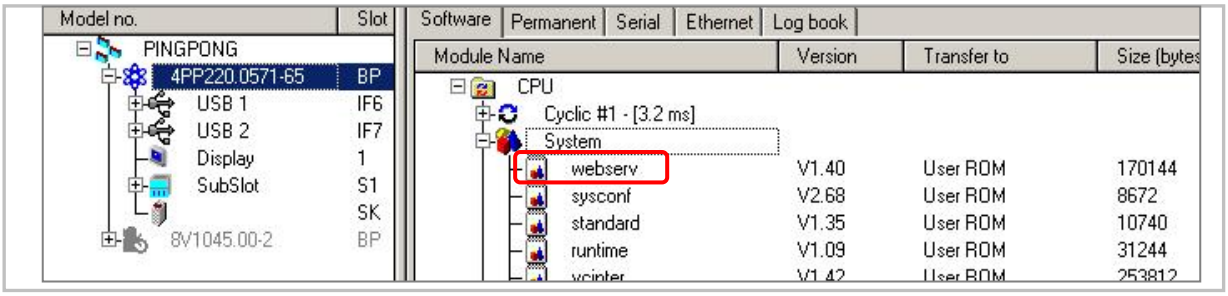
Webový server umožňuje pomocí standardního internetového prohlížeče zobrazovat a nastavovat hodnoty proměnných z/do operačního systému řídicí jednotky.

Webový server je v provozu, pokud operační systém (automation runtime) řídicí jednotky a projekt vytvořený v Automation Studio splňují následující požadavky:

1. Operační systém řídicí jednotky umí pracovat s webovým serverem
2. Řídicí aplikace obsahuje systémový objekt webového serveru
3. Existuje TCP/IP spojení mezi řídicí jednotkou a okolním prostředím a řídicí jednotka je správně nastavena
4. Správně vytvořen inicializační soubor webového serveru
5. Správně vytvořen konfigurační soubor webového serveru
6. Vytvořena úvodní stránka
7. Webové stránky a potřebné soubory byly nahrány na disk řídicí jednotky

7.3.1.1. Systémový objekt

Samotná instalace webového serveru se provede přidáním systémového objektu „webserv.br“ do vytvářeného projektu (viz. obr 7.11). Objekt je k nalezení v adresáři příslušného operačního systému (....\BrAutomation\BR_AS_240_L001\AS\ System\Verze operačního systému (V0268) \i386*.*).



Obr. 7.11 – Vložení systémového objektu webového Serveru

7.3.1.2. Inicializační soubor

Po spuštění řídicí jednotky hledá webový server inicializační soubor „C:\BrWebSvr.ini“ na disku řídicí jednotky. Jméno tohoto inicializačního souboru a jeho umístění na disku „C:\” nesmí být změněny. Inicializační soubor obsahuje klíčové slovo „CFG = cesta ke konfiguračnímu souboru a jeho jméno“ .

Jelikož je umístění konfiguračního souboru a html stránek pevně stanovené, bude potom v inicializačním souboru „CFG = C:\web\BrWebSvr.cfg“.

Tento soubor se zkopíruje na disk řídicí jednotky pomocí FTP.

7.3.1.3. Konfigurační soubor

Vytvořený konfigurační soubor „BrWebSvr.cfg „ musí být pomocí FTP nahrán do adresáře, který je nastaven v inicializačním souboru. Dále musí obsahovat určité parametry. Jejich popis a konfigurace je ukázána v obr. 7.12.

```
;Klíčové slovo= ..          ;Komentář
Port          = 80          ;Komunikační port
Retries       = 5           ;Opakování
Priority      = 78          ;Priorita přihlášení
BaseDir       = C:          ;Kořenový adresář
WebDir        = web         ;Adresář webových stránek
StartPage     = model.asp   ;Startovací stránka

;Seznam uživatelů
UserCount      = 30         ;Počet uživatelů
NAME[1]       = user1       ;Přihlašovací jméno
PASSWORD[1]= password       ;Heslo
RIGHTS[1]     = 100;  ...   ;Úroveň
```

Obr. 7.12 – Konfigurační soubor

7.3.2. Tvorba webových stránek (ASP funkce)

Webové stránky mohou být vytvořeny v libovolném editoru. Potom musí být nahrány do adresáře řídicí jednotky, který je definován v konfiguračním souboru (C:\web). Pokud webové stránky obsahují procesní proměnné čtené z operačního systému (Automation runtime) musí být vytvořeny jako „Active Server Page files“ (*.asp).

Zobrazení aktuálních hodnot proměnných, jejich zápis do PLC, přihlašování a odhlašování je realizováno pomocí ASP funkcí. Proměnné musí být v projektu definované jako globální (_GLOBAL typ proměnná).

7.3.2.1. Čtení proměnné

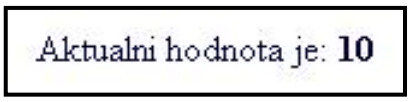
Funkce čtení proměnné slouží k získání aktuální hodnoty z PLC a k jejímu zobrazení na webových stránkách. Ukázka zápisu kódu a vzhled stránky je na obr. 7.13.

Kód v PLC:

```
_GLOBAL int Cislo;      Cislo = 10;
```

Kód webové stránky:

```
<body>
Aktuální hodnota je: <% ReadPLC ("Cislo"); %>
</body>
```



Obr. 7.13 – Čtení proměnné pomocí webové stránky

7.3.2.2. Zápis proměnné

Funkce zápis proměnné slouží k zapsání hodnoty z webových stánek do PLC. Hodnotu můžeme zadávat nebo ji předdefinovat. Ukázka kódu a vzhled stránky je na obr. 7.14. Parametry funkce jsou popsány v tab. 7.6.

Parametr	*Popis*
/goform/ReadWrite	Funkce pro čtení/zápis proměnné
redirect	Jméno stránky, na které je poslán výsledek funkce
variable	Jméno proměnné
value	Hodnota
write	Zápis hodnoty do PLC

Tab. 7.6 – Parametry ASP funkce zápis hodnoty do PLC z webových stránek

Po stisknutí příslušného tlačítka dojde k zápisu proměnné do PLC.

Kód webové stránky:

```
<body>
Nastavení hodnoty = konstanta
<form method="POST" action="/goform/ReadWrite"  >
<input type=hidden name=redirect size=50 value="stranka.asp" >
<input type="hidden" name="variable" value="Hodnota_Konstanta">
<input type="hidden" name="value" value="1">
<input type="submit" value="Nastav" name="write">
</form>

Nastavení hodnoty = libovolné číslo
<form method="POST" action="/goform/ReadWrite" >
<input type=hidden name=redirect size=50 value=" stranka.asp" >
<input type="hidden" name="variable" value=" Hodnota_Cislo">
<input type="text" name="value" value="" size="4">
<input type="submit" value=" Nastav " name="write">
</form>
</body>
```

Obr. 7.14 – Zápis hodnoty proměnné z webové stránky

7.3.2.3. Přihlášení a odhlášení uživatele

Funkce přihlášení a odhlášení uživatele slouží k zabezpečení přístupu na webové stránky. Seznam přihlašovacích jmen a hesel je definován v konfiguračním souboru. Ukázka kódu a vzhled stránky je na obr. 7.15. Parametry přihlašovací a odhlašovací funkce jsou popsány v tab. 7.7.

Parametr	*Popis*
/goform/UserLogin	Použití přihlašovací funkce
username	Přihlašovací jméno
password	Heslo pro přihlášení
redirect	Stránka po přihlášení / odhlášení
error_redirect	Stránka při špatném přihlášení

Tab. 7.7 – Parametry ASP funkce přihlašování a odhlašování

Při správném zadání jména a hesla (user1 a password) přejdeme na požadovanou stránku (dobře.asp), to platí i pří špatném přihlášení (špatně.asp) a odhlášení uživatele (odhlášení.asp).

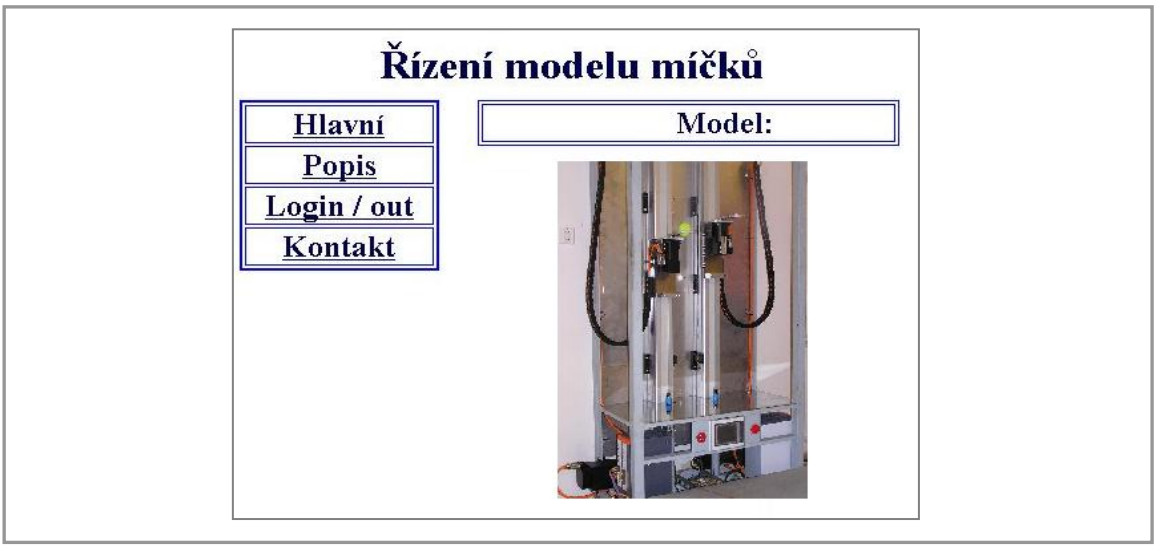
Kód webové stránky:

```
<body>
Přihlášení uživatele:
<form method="POST" action="/goform/UserLogin">
<input type=hidden name=redirect value="dobre.asp">
<input type=hidden name=error_redirect value="spatne.asp">
Jméno
<input type="text" name=username>
Heslo
<input type="password" name=password>
<input type="submit" value="Přihlaš" >
<input type="reset" value="Vymaž">
</form>
Odhlášení uživatele:
<form method="POST" action="/goform/UserLogout">
<input type=hidden name=redirect value="odhlaseni.asp">
<input type="submit" value="Odhlaš" >
</form>
</body>
```

Obr. 7.15 – Přihlášení a odhlášení uživatele pro ovládání z webových stránek

7.3.3. Popis webových stránek

Po přístupu na webové stránky, pro řízení modelu míčků, zadáním IP adresy řídicího systému do internetového prohlížeče, se jako první zobrazí úvodní – „Hlavní“ stránka (viz. obr. 7.16) s odkazovým menu na další stránky (popis, přihlášení a odhlášení a kontakt).

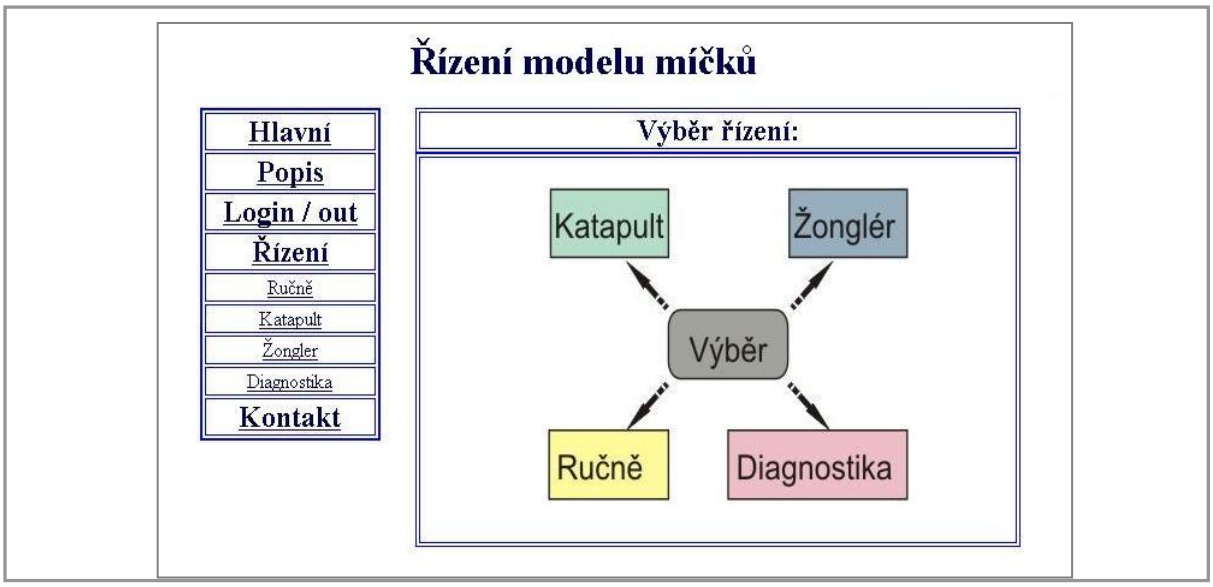


Obr. 7.16 – Úvodní webová stránka pro řízení modelu

Webová stránka popis obsahuje všechny informace potřebné k řízení modelu, od popisu funkce modelu až po popis a postup ovládání.

Stránka login/out slouží k přihlášení a odhlášení uživatele. Po správném přihlášení se zobrazí stránka „Řízení“, obsahující výběrové menu pro zvolení funkce

řízení modelu (viz. obr. 7.17). Odtud se potom dostaneme na stránky ovládání příslušné funkce – ručně, katapult žonglér a diagnostika. Na obr. 7.18 je ukázána vizualizace funkce katapult, vizualizace ostatních funkcí je v příloze 17.



Obr. 7.17 – Webová stránka pro výběr řízení modelu

Obr. 7.18 – Webové stránky pro ovládání funkce katapult

Řídicí obrazovky obsahují informační popisy (např. Ovládání z), vstupní pole zobrazení aktuálních hodnot z PLC (např. Web server a katapult), výstupní pole pro zadávání hodnot do PLC (hodnota pro výšku) a tlačítka, která potvrdí zadání hodnot předdefinovaných (např. str. A, Start) nebo definovaných ve výstupním poli (např. Výška).

7.4. Popis ovládání modelu z operátorského panelu a webových stránek

Princip a postup ovládání modelu je společný pro ovládání z operátorského panelu i z webových stránek.

Pozn.:

Zadávané hodnoty z webových stránek se musí potvrdit příslušným tlačítkem.

7.4.1. Diagnostika servozesilovače

Pro možnost diagnostiky servozesilovače musíme nejprve na příslušné obrazovce potvrdit tuto funkci tlačítkem „vyber“. Potom musíme vybrat adresu diagnostikovaného servozesilovače (1-4 pro osu reálnou a 5 pro osu virtuální) v zadávacím poli „diagnostika acoposu“. Dále musíme potvrdit existující chybu tlačítkem „Potvrď chybu“. Neexistuje–li žádná chyba můžeme servozesilovač zapnout tlačítkem „Zapni“, servozesilovač můžeme kdykoliv vypnout tlačítkem „Vypni“. Potom nastavíme motor do počáteční polohy tlačítkem „Homing“. A nakonec se příslušným tlačítkem přepneme na obrazovku funkce, kterou chceme aby model vykonal.

7.4.2. Ruční ovládání

Nejprve musíme spustit ovládaný servozesilovač. To se provádí z obrazovky diagnostika.

Pro možnost ručního ovládání musíme na příslušné obrazovce nejprve potvrdit tuto funkci tlačítkem „Vyber“. Potom musíme vybrat adresu řízeného motoru v zadávacím poli „Adresa“. Dále zadáme rychlost a vzdálenost pro pohyb v příslušném zadávacím poli. Nakonec spustíme pohyb pomocí tlačítka příslušného pohybu (Positivní, negativní, Absolutní a relativní), pohyb se ukončí sám po splnění zadaných podmínek, nebo je ho možno kdykoliv zastavit pomocí tlačítka „Stop“.

7.4.3. Ovládání funkce katapult

Nejprve musíme spustit ovládaný servozesilovač. To se provádí z obrazovky diagnostika.

Pro možnost ovládání funkce katapultu musíme na příslušné obrazovce nejprve potvrdit funkci tlačítkem „Vyber“. Potom musíme vybrat stranu modelu, která funkci vykoná, pomocí tlačítka „A“ nebo „B“. Dále zadáme výšku kliknutím na zadávací pole „Výška“. Nakonec spustíme pohyb pomocí tlačítka „Start“.

7.4.4. Ovládání funkce žonglér

Nejprve musíme spustit ovládaný servozesilovač. To se provádí z obrazovky diagnostika.

Pro možnost ovládání žonglér musíme na příslušné obrazovce nejprve potvrdit funkci tlačítkem „Vyber“. Potom musíme nastavit řídicí vačkové profily ručně pomocí zadávacího pole „servo X“ a tlačítka „Nastav“ nebo automaticky dalšími nastavovacími tlačítky (A>B, B>A, A<>B 1 Ball a A<>B 2 Ball). Nakonec spustíme pohyb pomocí tlačítka „Start“, který se dá ukončit jenom pomocí tlačítka „Stop“.

8. ZÁVĚR

Výsledkem této práce je realizovaný funkční model míčků, který splňuje všechny požadavky, které byly požadovány před jeho vytvoření. Vytvořený mechanický model demonstruje řízení a synchronizaci čtyř motorů. K řízení byly použity servozesilovače a průmyslový programovatelný logický automat. Komunikace je realizována pomocí moderní vysoce rychlostní komunikace reálného času Ethernet Powerlink. Model můžeme ovládat z operátorského panelu, ale i pomocí vzdáleného přístupu přes internet.

Navržený model bude v budoucnu využíván studenty na cvičení z předmětu Řídicí systémy. Při řízení modelu si studenti budou moci důkladně procvičit programování PLC, operátorského panelu a servozesilovačů firmy B+R automatizace v prostředí Automation Studio.

K realizaci pohybů modelu jsem použil synchronní motory malých rozměrů s vysokým výkonem a krouticím momentem, které jsou ovládány digitálními servozesilovači ACOPOS pomocí PWM napájecího napětí. Pro jednotlivé typy motorů jsou použity odlišné servozesilovače (pro větší motor typ 1090 a pro menší 1020), které se liší maximálním možným dodávaným výkonem a proudem. Jako zpětná vazba pro získání polohy motoru je použit rezolver, který přenáší data do modulu AC122 v servozesilovači. Řízení motorů servozesilovači je realizováno na základě programu běžicího v programovatelném logickém automatu typu Power Panel PP220, který zajišťuje komunikaci se servozesilovači, jejich řízení a vizualizaci.

Komunikace mezi řídicím automatem a servozesilovači je realizována pomocí moderní komunikace v reálném čase Ethernet Powerlink. Ethernet Powerlink kromě rychlé cyklické výměny dat zajišťuje i acyklickou komunikaci, která však cyklickou nesmí nijak ovlivnit. Pro rychlou cyklickou výměnu dat jsem nastavil dobu cyklu sběrnice na 400 µs. To je nejkratší možná doba, která navíc odpovídá periodě regulátoru polohy servozesilovače B+R. Adresa každého účastníka v síti musí být unikátní s tím, že řídicí stanici masteru (PLC) jsem musel nastavit adresu 0 a ostatním zařízením (servozesilovačům) adresu 1 - 4. Doba cyklu komunikace Ethernet Powerlink roste s počtem stanic v síti.

K realizaci komunikace musí PLC - Power Panel PP220 a servozesilovače obsahovat komunikační Ethernet Powerlinkové moduly, v našem případě IF786 (pro PLC) a AC112 (pro servozesilovače).

Model se skládá ze dvou lineárních pohybových pásových modulů obsahující pojezd. Jeden modul obsahuje osu vertikálního a horizontálního pohybu. Každý modul je ve vertikálním směr poháněn jedním synchronním motorem typu 8MSA5L,

který může jet rychlostí až 5 m/s se zrychlením 50 m/s^2. Dochází zde k přeměně rotačního pohybu na lineární. Ke každému pojezdu modulu je připevněn další, ale ne tak výkonný, synchronní motor 8MSA3L obstarávající rotační pohyb do strany (v horizontálním směru). K hřídeli tohoto motoru je upevněn držák míčků.

Pohyb vertikálního motoru po celé délce lineárního modulu (210 cm) se uskuteční pomocí osmi otočení a pohyb v horizontálním směru je realizován krátkým prudkým pohybem do strany (cca 45°).

Řídicí i vizualizační část programu jsem vytvořil ve vývojovém prostředí Automation Studio verze 2.4.0.9.

Vlastní řízená osa (servozesilovač a motor) zde vystupuje jako datová struktura, jejíž jednotlivé položky (parametry) jsem musel nastavit a otestovat. Tyto parametry přesně popisují vlastnosti osy – od vstupních a výstupních signálů, přes nastavení vnitřních regulátorů až po okamžité hodnoty veškerých známých veličin a další. Jediný rozdíl je v nastavení parametru pro spínač „Quickstop“ při použití funkcí PLCopen, pro který nelze nastavit funkci rychlého vypnutí (při nastavení se neprovede inicializace dat pro osu).

Vizualizace modelu je realizována pomocí dotykové obrazovky, která je součástí Power Panelu PP220. Dále je vizualizace realizována pomocí webových stránek, které běží na webovém serveru panelu a pomocí vzdáleného přístupu tzv. Virtual Network Computing (VNC). Pro vizualizaci modelu z webových stránek stačí do prohlížeče zadat IP adresu řídicího systému (Power Panelu). Webové stránky a konfigurační soubory webového serveru se do řídicího systému dají nahrát pomocí FTP. Pro vizualizaci pomocí VNC je zapotřebí nainstalovat VNC server do řídicího systému, na který se připojíme pomocí prohlížeče z počítače. Stávající verze operačního systému řídicí jednotky neumožňuje ovládání pomocí VNC, tento nedostatek bude odstraněn s novým operačním systémem, který bude podporován novou verzí Automation Studia (V2.5).

Pro řízení modelu jsem naprogramoval dvě ukázkové aplikace. První demonstruje ovládání motoru pomocí základních funkcí knihovny PLCopen, k této části nebyla vytvořena vizualizace. Druhá aplikace obsahuje vizualizaci modelu a realizuje ovládání modelu pomocí řídicích akcí (nc-actions) a vačkového automatu (CAM profile automat), který používá vačkové profily (CAM profile). Tato aplikace je naprogramována k vykonávání tří pohybových funkcí, kterými jsou ruční ovládání, funkce katapult a funkce žonglér (vačkové profily), a k diagnostice servozesilovačů. Řízení servozesilovače pomocí funkčních bloků knihovny PLCopen není možné používat ve stejném projektu jako řídicí akce (nc-actions), ale struktura programu (vložené objekty) je totožná pro obě možnosti řízení.

Při řízení pomocí vačkových automatů je synchronizace zajištěna virtuální master osou, která všechny podřízené reálné osy (motor a servozesilovač) synchronizuje pomocí času. Každá osa má pevně stanovené trajektorie pohybu v závislosti na čase, který se opakuje dokola.

Ruční ovládání řídí motory pomocí základních pohybů, kterými jsou positivní nebo negativní pohyb, který spočívá v otáčení motoru v daném směru danou rychlostí, a relativní nebo absolutní pohyb, který spočívá v ujetí definované vzdálenosti nebo najetí na požadovanou polohu definovanou rychlostí. Funkce katapult ovládá model automaticky a provede jeden cyklus, který se skládá ze dvou částí. První z nich spočívá ve vyhození míčku ve svislém směru skrz překážku a jeho chycení v maximální výšce. Druhá spočívá v upuštění míčku a v jeho opětovném chycení. Během obou částí se držák míčku vyhne překážce uhnutím do strany. Funkce vačkových profilů umožňuje ovládat všechny motory pomocí vybraného předem definovaného vačkového profilu, u kterého definujeme trajektorii polohy, rychlosti a zrychlení v závislosti na čase. Pomocí vačkových profilů je realizována i funkce žonglér, která spočívá v přehazování si dvou míčků navzájem z jedné strany modelu na druhou.

Ukázkové řízení pomocí funkcí knihovny PLCopen umožňuje ovládat jeden motor pomocí pohybu v daném směru (positivním a negativním).

Při výpočtů parametrů pro vykonávání funkce katapult a žonglér jsem musel zohlednit odchylky mezi žádanou a skutečnou hodnotou rychlosti motorů, která rostla se zvětšující se rychlostí pohybu motorů a pohybovala se v rozmezí 5-15%. Odchylku nedokážeme odstranit z těchto důvodů: motor je využíván na maximální možný výkon, požadovanou rychlost a zrychlení potřebujeme dosáhnout během jedné až dvou otáček pro motor vertikálního pohybu nebo krátkým a prudkým pohybem do strany pro motor horizontálního pohybu (motor se dostatečně neroztočí).

Potřebný moment a síla $M=F*r=m*(g+a)*r$ k realizaci pohybu předmětu o hmotnosti 16 kg se zrychlením 50m/s^2 ve vertikálním směru je $M = 44\ Nm$ a $F = 960N$. Motor dokáže vyvinout krátkodobě maximální moment $M = 43\ Nm$ a sílu $F = M*2\pi/obvod = 934N$.

Odchylka motoru vertikálního pohybu by se dala částečně eliminovat použitím převodovky, která by se zapojila mezi hřídel motoru a lineárního modulu. Převodovka nebyla k modelu pořízena z finančních důvodů.

Zapojení napájecích zdrojů, kabelů, tlačítek, spínačů, servozesilovačů, řídicího systému bude v budoucnu realizováno tak, aby splňovalo podmínky řídicí kategorie 3 podle normy EN 954-1. V současně době toto není realizováno, z důvodu opoždění dodávky potřebných komponent.

Model bude v budoucnu také vybaven laserovými čidly pro odměřování pozice míčku na obou lineárních modulech. K modelu bude také přimontována kamera, která bude ze shora snímat scénu modelu a rozpoznávat pozici (trajektorii) míčků. Na základě dat získaných z čidel a z kamery bude možnost měnit řídicí požadavky.Tyto novinky budou realizovány v další diplomové práci, která bude navazovat na tuto.

LITERATURA

[1] *ACOPOS User's Manual.* B&R Automation, 2004.

[2] *Automation Software™ Overview Catalog.* B&R Automation, 2003.

[3] *Automation Studio™ System Introduction*. B&R Automation, 2001.

[4] *Automation Studio™ Programming*. B&R Automation, 2001.

[5] *CAM Profile Automat ACOPOS Firmware*. B&R Automation, 2004

[6] *Ethernet Powerlink User's Manual*. B&R Automation, 2003.

[7] *Positioning Introduction*. B&R Automation, 2001.

[8] *Power Panel 100/200 User's Manual.* B&R Automation, 2003.

[9] *Power Supply PS320*. B&R Automation, 2004.

[10] *Visual Components Introduction*. B&R Automation, 2002

[11] Valentine Richard. *Motor Control Electronics Handbook.* McGraw–Hill, 1998, ISBN 0-07-066810-8

[12] Werner Defren a Franz Kreutzkampf. *Machine safety in the European Community*. K. A. Schmersal GmbH, 2003.

[13] *B&R Automation* [online], 2005
http://www.br-automation.com/

[14] *Ethernet Powerlink Standardization Group* [online], 2005
http://www.ethernet-powerlink.com/

[15] Jankovský Dušan. *Jak psát web* [online], 2005
http://www.jakpsatweb.cz/

[16] *PLCopen for efficiency in automation* [online], 2005
http://www.plcopen.org/

[17] RealVNC. *Virtual Network Computing* [online], 2005
http://www.realvnc.com/

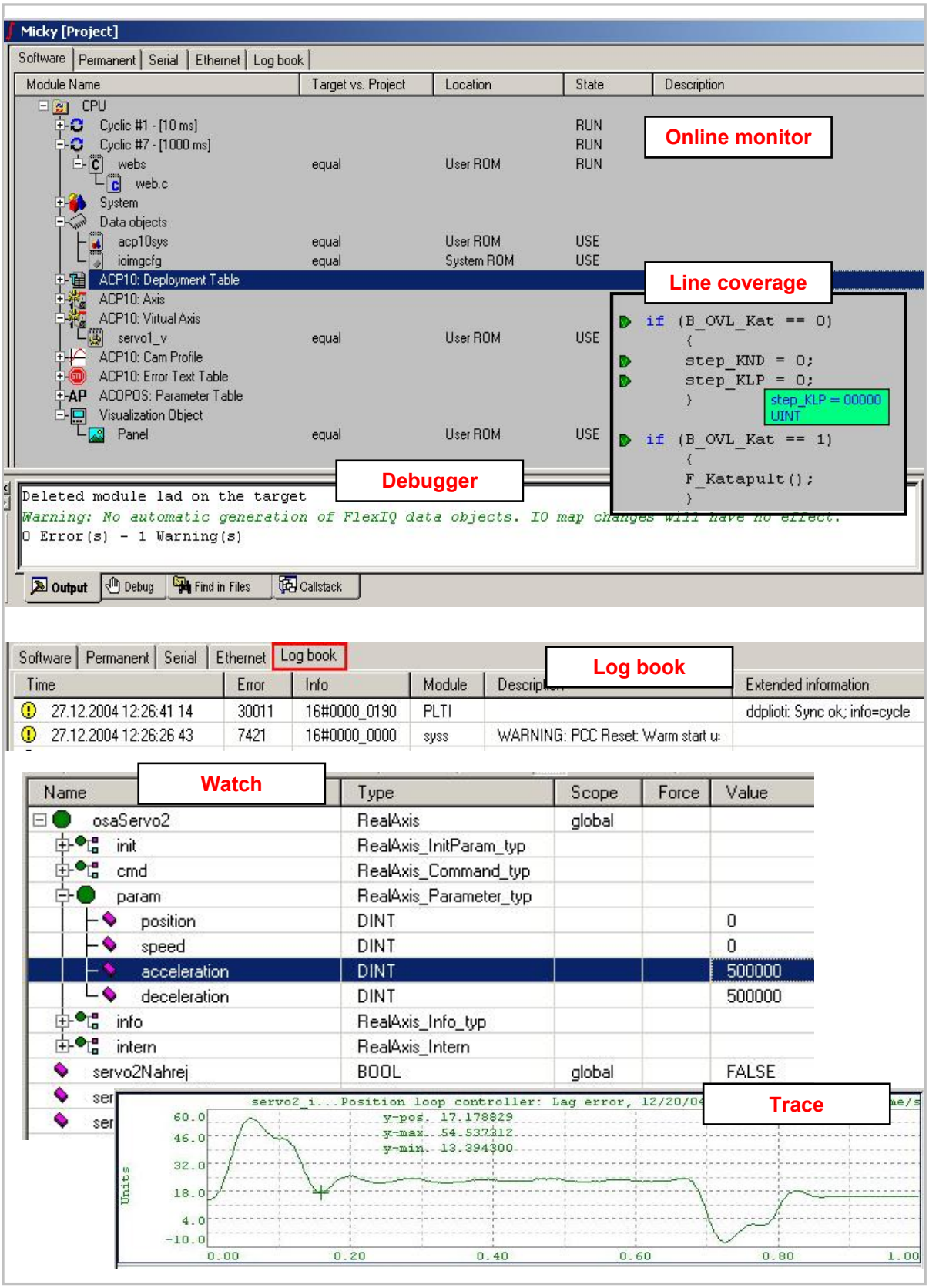
SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH

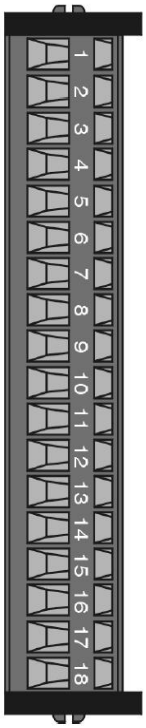
Příloha č. 1 – Řízený model míčků

Příloha č. 2 – Ladící, monitorovací a vyhodnocovací nástroje

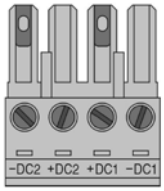


Příloha č. 3 – Konektory servozesilovače

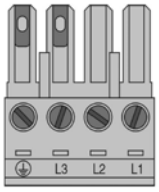
Popis konektorů servozesilovače:



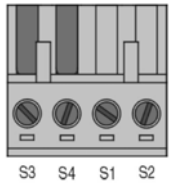
X1		
Pin	**Popis**	**Funkce**
1	Trigger 1	Vypínač
2	Trigger 2, Quick stop	Vypínač
3	COM (1, 2)	Napájení -24V
4	Shield	Stínění
5	End +	Positivní HW mez
6	End –	Negativní HW mez
7	Ref	Referenční spínač
8	Enable	Povolení
9	Enable	Povolení
10	COM (8, 9)	Napájení -24V
11	COM (8, 9)	
12	–––	–––
13	–––	–––
14	+24V	Napájení +24V
15	+24V	
16	COM (5–7,13–15)	Napájení -24V
17	COM (5–7,13–15)	
18	COM (5–7,13–15)	



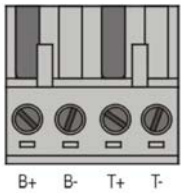
X2		
Pin	**Popis**	**Funkce**
1	– DC1	DC sběrnice 24 V
2	+ DC1	
3	– DC2	
4	+ DC2	



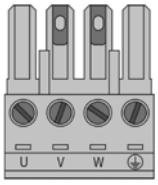
X3		
Pin	**Popis**	**Funkce**
1	L1	Výkonové hlavní napájení servozesilovače
2	L2	
3	L3	
4	PE	



X4a		
Pin	**Popis**	**Funkce**
1	S2	Konektor pro externí napájení brzdy motoru
2	S1	
3	S4	
4	S3	



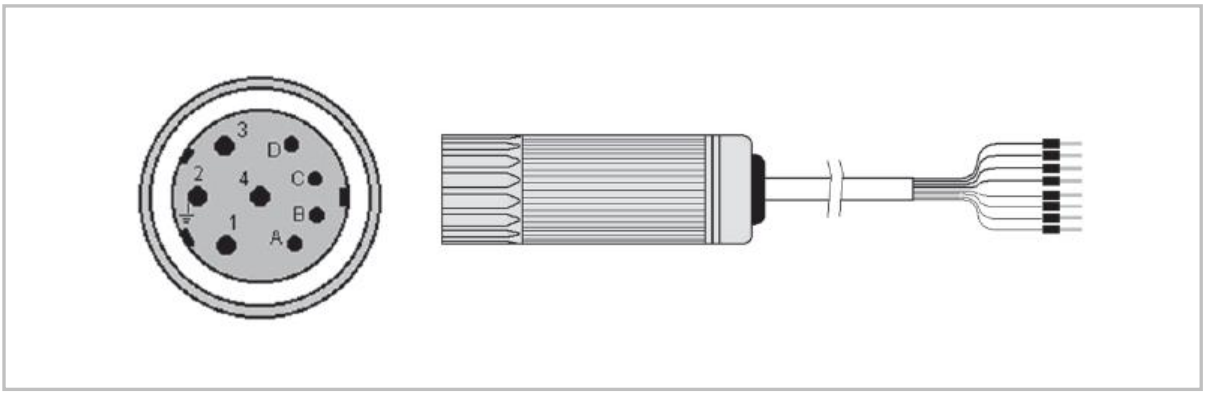
X4b		
Pin	**Popis**	**Funkce**
1	T–	Senzor teploty motoru
2	T+	
3	B–	Brzda motoru
4	B+	



X5		
Pin	**Popis**	**Funkce**
1	PE	Napájení motoru 380 V
2	W	
3	V	
4	U	

Příloha č. 4 – Kabel pro napájení a hlídání motoru

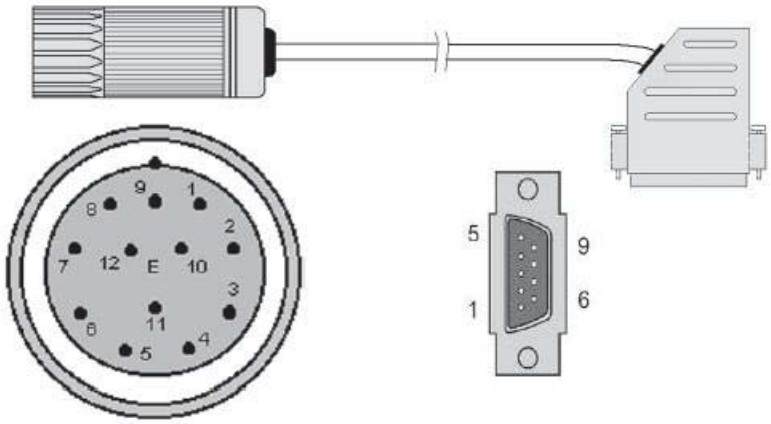
Kabel propojující napájecí a diagnostický konektor motoru a konektor servozesilovače X5 a X4b.



Kulatý konektor motoru	*Barva kabelu*	*Popis*	*Konektor servozesilovače / kontakt*
1	Modrý	Motor – U	X5 / 2
2	Žlutý / zelený	Motor – GND	X5 / 1
3	Černý	Motor – W	X5 / 3
4	Hnědý	Motor – V	X5 / 4
A	Bílý	Teplota +	X4 / 1
B	Bílý / červený	Teplota –	X4 / 2
C	Bílý / modrý	Brzda +	X4 / 3
D	Bílý / zelený	Brzda –	X4 / 4

Příloha č. 5 – Kabel rezolveru

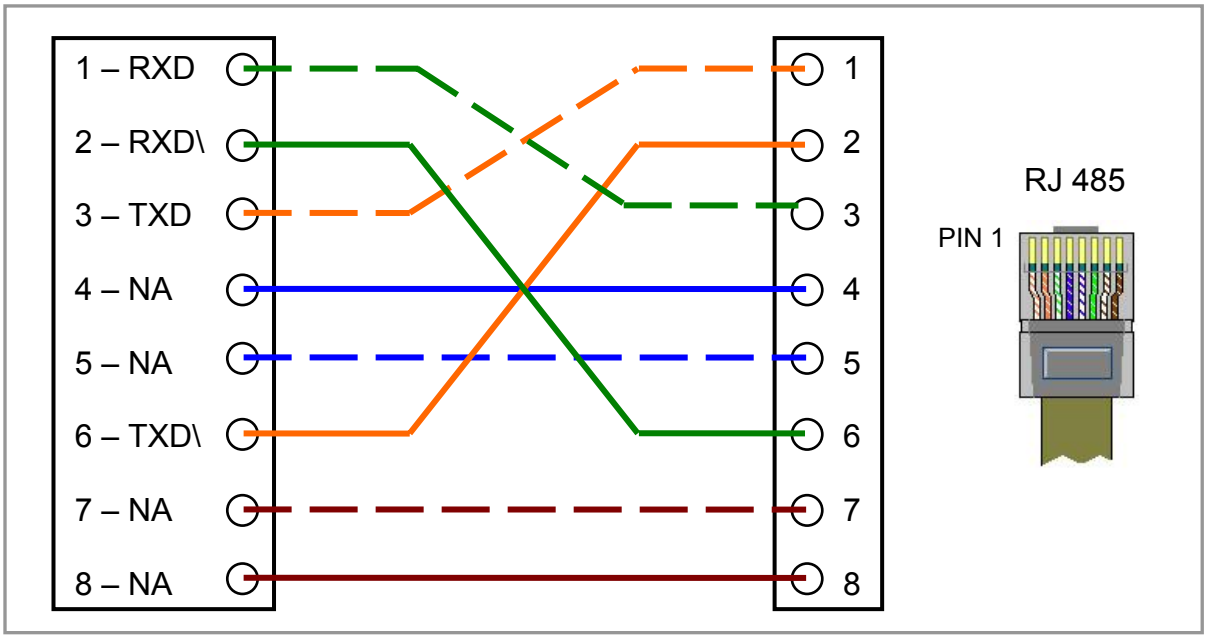
Kabel propojující řídicí konektor motoru s modulem rezolver AC122 ve slotu servozesilovače (přenos aktuální pozice).



Kulatý konektor motoru	*Popis*	*9 pin konektor (modul servozesilovače AC122)*
3	Cos +	3
4	Sin +	4
5	Ref +	5
7	Cos –	7
8	Sin –	8
9	Ref –	9

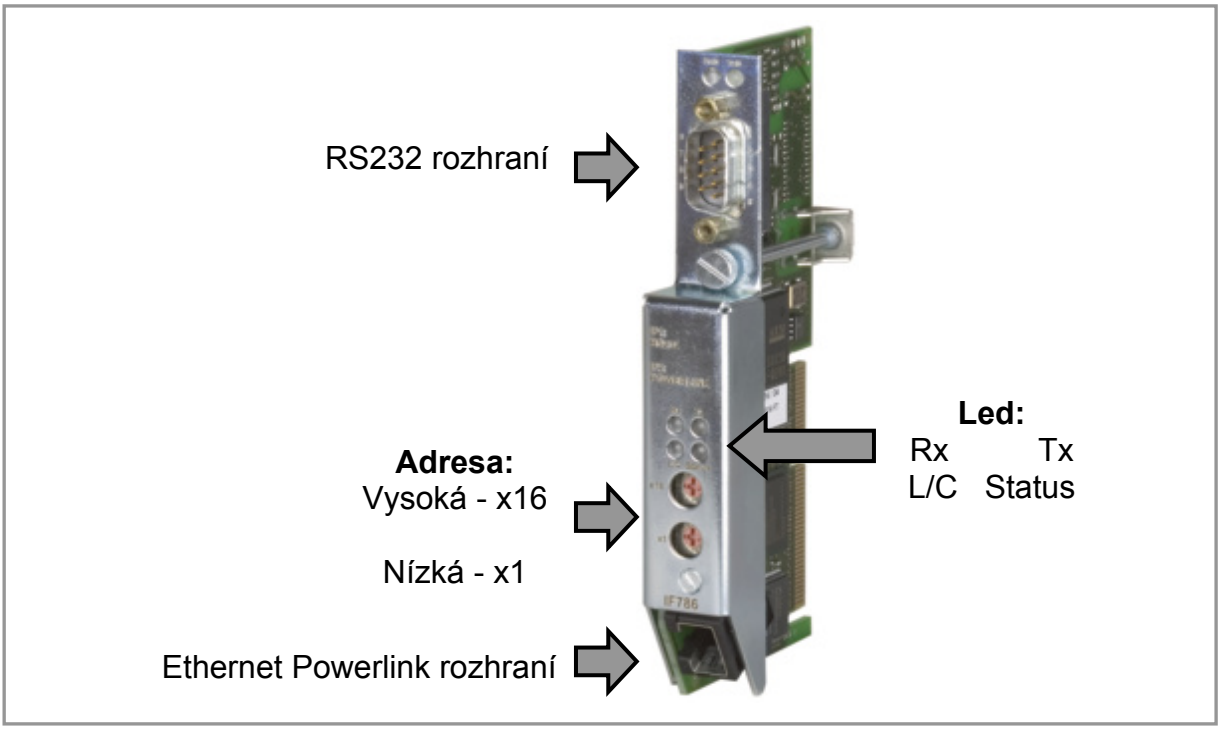
Příloha č. 6 – Kabel komunikace Ethernet Powerlink

Zapojení kříženého kabelu pro realizaci komunikace Ethernet Powerlink:



Příloha č. 7 – Komunikační karta Ethernet Powerlink IF786

Popis komunikační karty (součást PLC):



Indikace stavu karty:

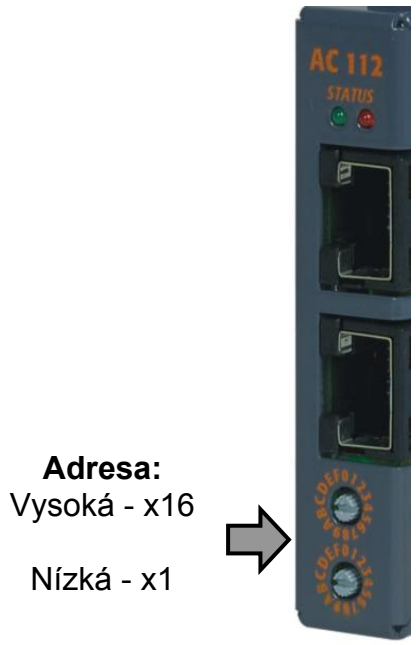
Označení	***Led***	***Význam***	
Status	Zelená / červená		Stanice v pořádku bez poruch
			Fatální systémová chyba [1)]
		B) B)	Špatná adresa, není v povoleném rozsahu
		B)	Systémové selhání (viz. systémové poruchy)
Tx	Oranžová		Stanice posílá data
Rx	Oranžová		Stanice přijímá data
L/C	Zelená / červená		Spojení v pořádku
			Existuje kolize

[1)] Tento problém nelze opravit, je potřeba restartovat modul, popřípadě řídicí systém, chyba je zobrazena v seznamu poruch v PLC (Log book)

[B)] Led bliká

Příloha č. 8 – Modul Ethernet Powerlink AC112

Popis modulu AC112 (součást servozesilovače):



Ethernet Powerlink rozhraní

Adresa:
Vysoká - x16

Nízká - x1

Povolené jsou adresy v rozmezí 01 – FD

Adresy 00, FE a FE jsou rezervovány pro systémové účely (bootovací, diagnostický režim)

Indikace stavu modulu:

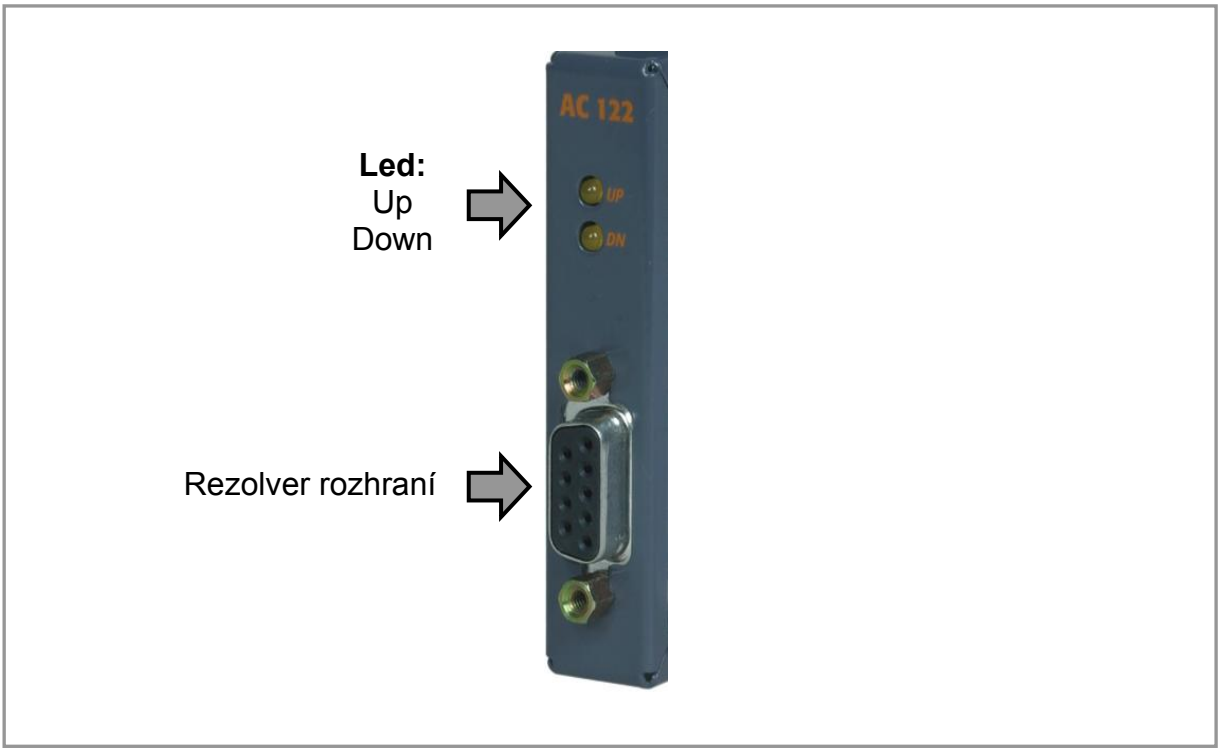
Popis	***Led***	***Zobrazení***
Komunikace v pořádku	Zelená	
	Červená	
Fatální systémová chyba [1)]	Zelená	
	Červená	
Výpadek master stanice	Zelená	
	Červená	
Systémové zastavení [2)]	Zelená	
	Červená	Viz. systémové poruchy

[1)] Tento problém nelze opravit, je potřeba restartovat modul, popřípadě řídicí systém

[2)] Blikání v různých intervalech označující kód poruchy

Příloha č. 9 – Modul rezolver AC122

Popis modulu AC122 (součást servozesilovače):

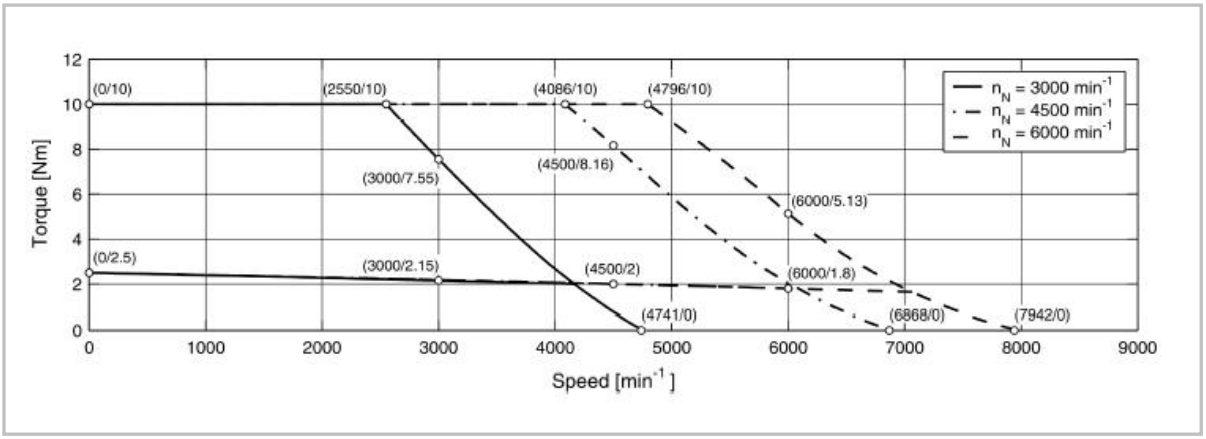


Indikace stavu modulu:

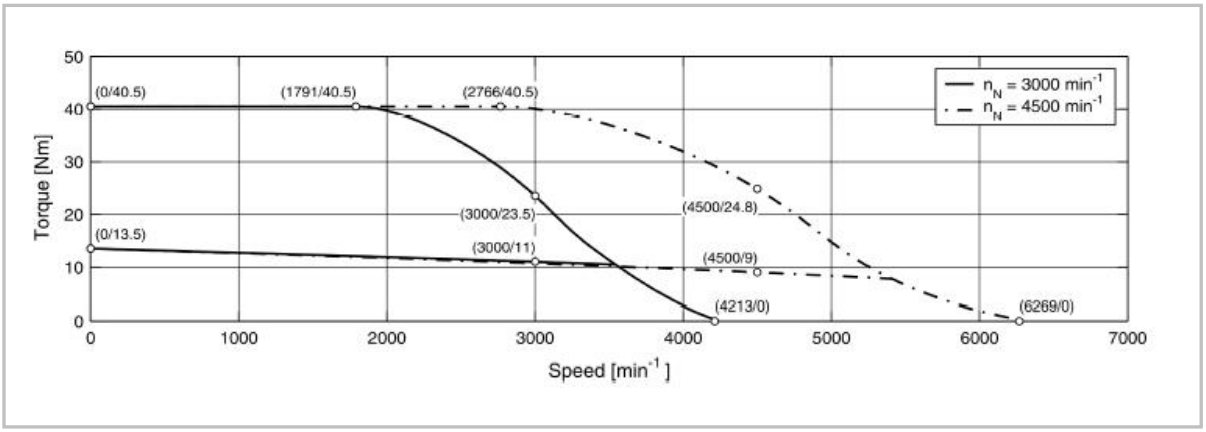
Popis	***Led***	***Význam***
Up	Oranžová	Pozice enkoderu se mění v pozitivním směru
Down	Oranžová	Pozice enkoderu se mění v negativním směru

Příloha č. 10 – Závislost momentu na rychlosti otáčení motorů

Motor typu 8MSA3L:

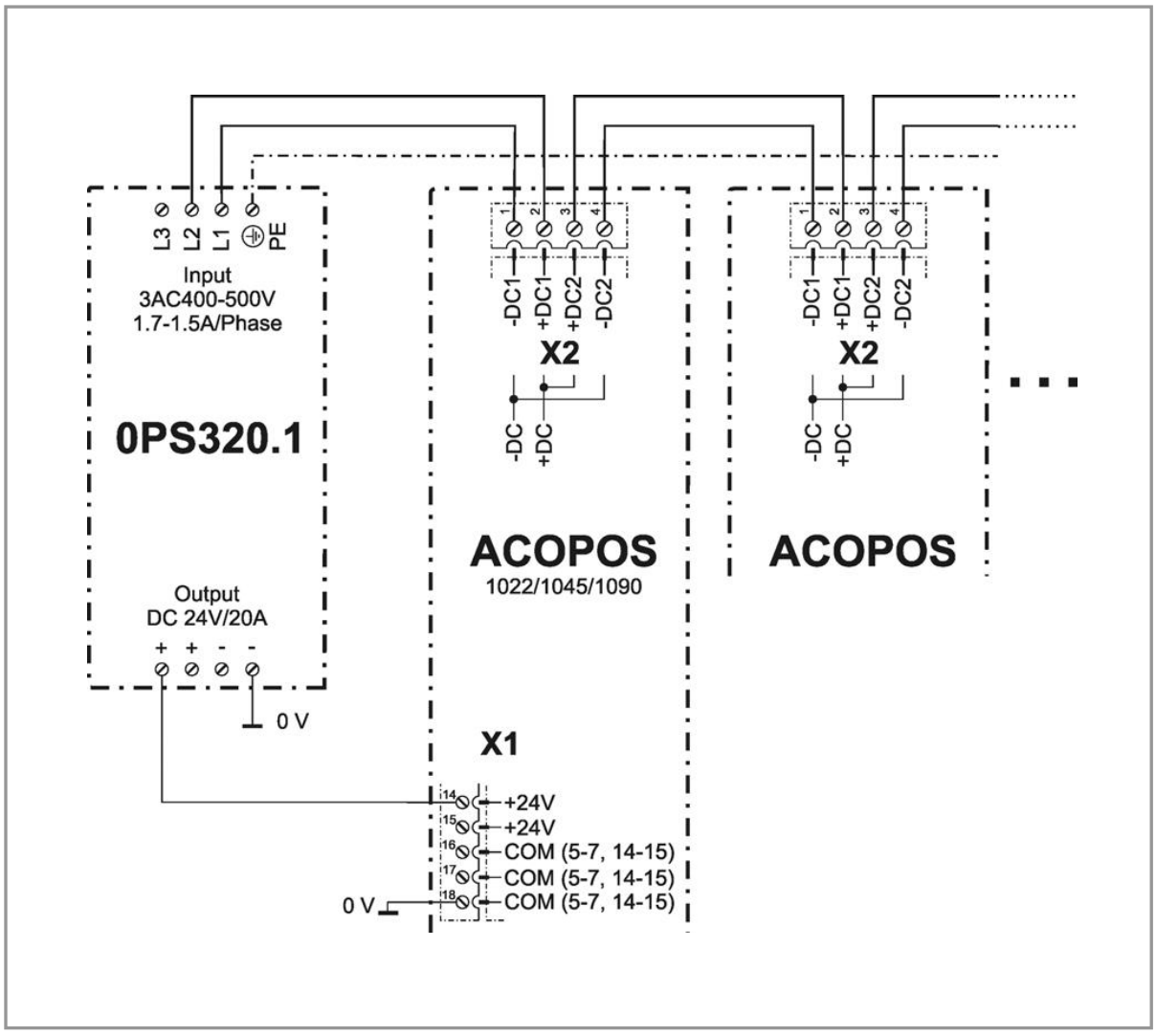


Motor typu 8MSA5L:



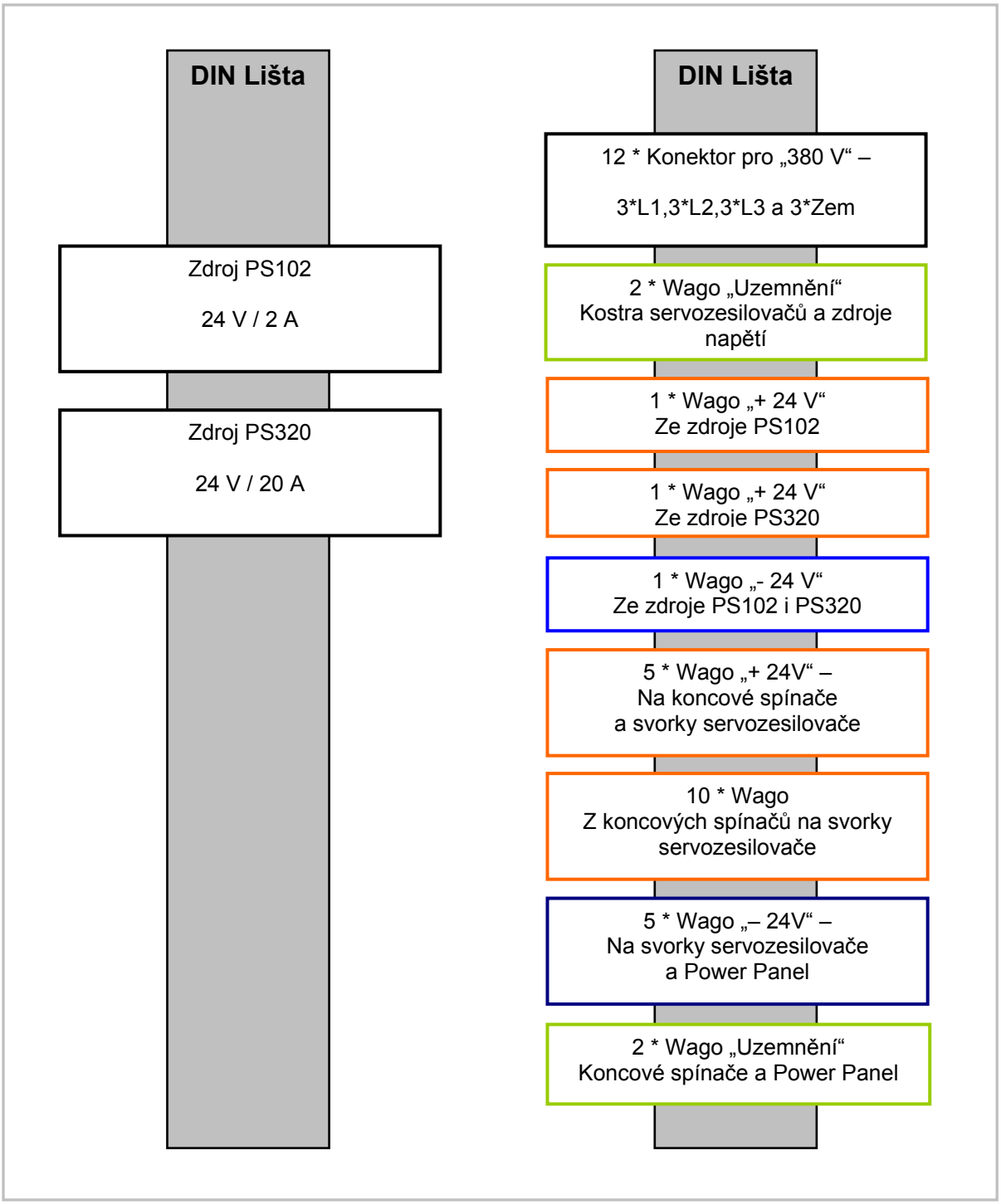
Příloha č. 11 – Propojení DC sběrnice servozesilovačů a zdroje

Realizace propojení DC sběrnice servozesilovačů a zdroje PS320, který z přivedeného napětí cca 500VDC vytvoří 24VDC s odběrem maximálně 20 A.



Příloha č. 12 – Rozmístění komponent na DIN liště

Ukázka rozmístění jednotlivých svorkovnic a zdrojů na DIN liště.



Příloha č. 13 – Zapojení kontaktů Wago svorek

Popis jednotlivých kontaktů wago svorek a jejich propojení (odkud nebo kam).

Uzemnění

1	2	S	3	4
Servozesilovač 1,2	R)		Kostra	Servozesilovač 3,4
Zdroj PS320	Zdroj PS102		R)	R)

Napájení +24 V

1	2	S	3	4
Tlačítko Quickstop	+24V, Zdroj PS102		R)	R)
Svorkovnice + 24 V	+24V, Zdroj PS320		R)	R)

Napájení -24 V

1	2	S	3	4
Svorkovnice -24 V	R)		-24 V, Zdroj PS102	-24 V, Zdroj PS320

Svorkovnice +24 V

1	2	S	3	4
+24V, Zdroj PS320	Tlačítko Quickstop	P)	R)	Power Panel
Acopos 1 / k 14	Acopos 2 / k 14	P)	Acopos 3 / k 14	Acopos 1 / k 14
Acopos 1 / k 9	Acopos 2 / k 9	P)	Acopos 3 / k 9	Acopos 1 / k 9
Koncový spínač A – Horní	Koncový spínač A – Dolní	P)	Koncový spínač B – Horní	Koncový spínač B – Dolní
Koncový spínač A – Levý	R)	P)	R)	Koncový spínač B – Pravý

Signály ze spínačů a tlačítka

	Tlačítko Quickstop	P)		
Acopos 1 / k 2	Acopos 2 / k 2	P)	Acopos 3 / k 2	Acopos 4 / k 2
Koncový spínač A – Horní	Acopos 2 / k 5		R)	R)
Koncový spínač A – Dolní	Acopos 2 / k 6		R)	R)
R)	R)		Koncový spínač B – Horní	Acopos 4 / k 6
R)	R)		Koncový spínač B – Dolní	Acopos 4 / k 5
Acopos 1 / k 5	Koncový spínač A – Levý		R)	R)
Acopos 1 / k 6	R)		R)	R)
R)	R)		Acopos 3 / k 5	R)
R)	R)		Acopos 3 / k 6	Koncový spínač B – Pravý

Svorkovnice -24 V

1	2	S	3	4
Napájení - 24V	R)	P)	R)	Power Panel
Acopos 1 / k 16	Acopos 2 / k 16	P)	Acopos 3 / k 16	Acopos 4 / k 16
Acopos 1 / k 10	Acopos 2 / k 10	P)	Acopos 3 / k 10	Acopos 4 / k 10
Acopos 1 / k 3	Acopos 2 / k 3	P)	Acopos 3 / k 3	Acopos 4 / k 3
R)	R)	P)	R)	R)

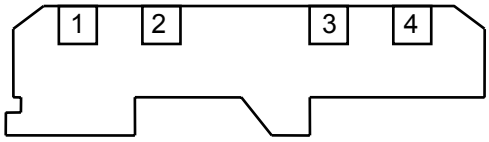
Uzemnění

1	2	S	3	4
Koncový spínač A – Horní	Koncový spínač A – Dolní		Koncový spínač B – Horní	Koncový spínač B – Dolní
R)	R)		R)	Power Panel

P) Propojení dvou svorkovnic
R) Rezerva

Vysvětlení:

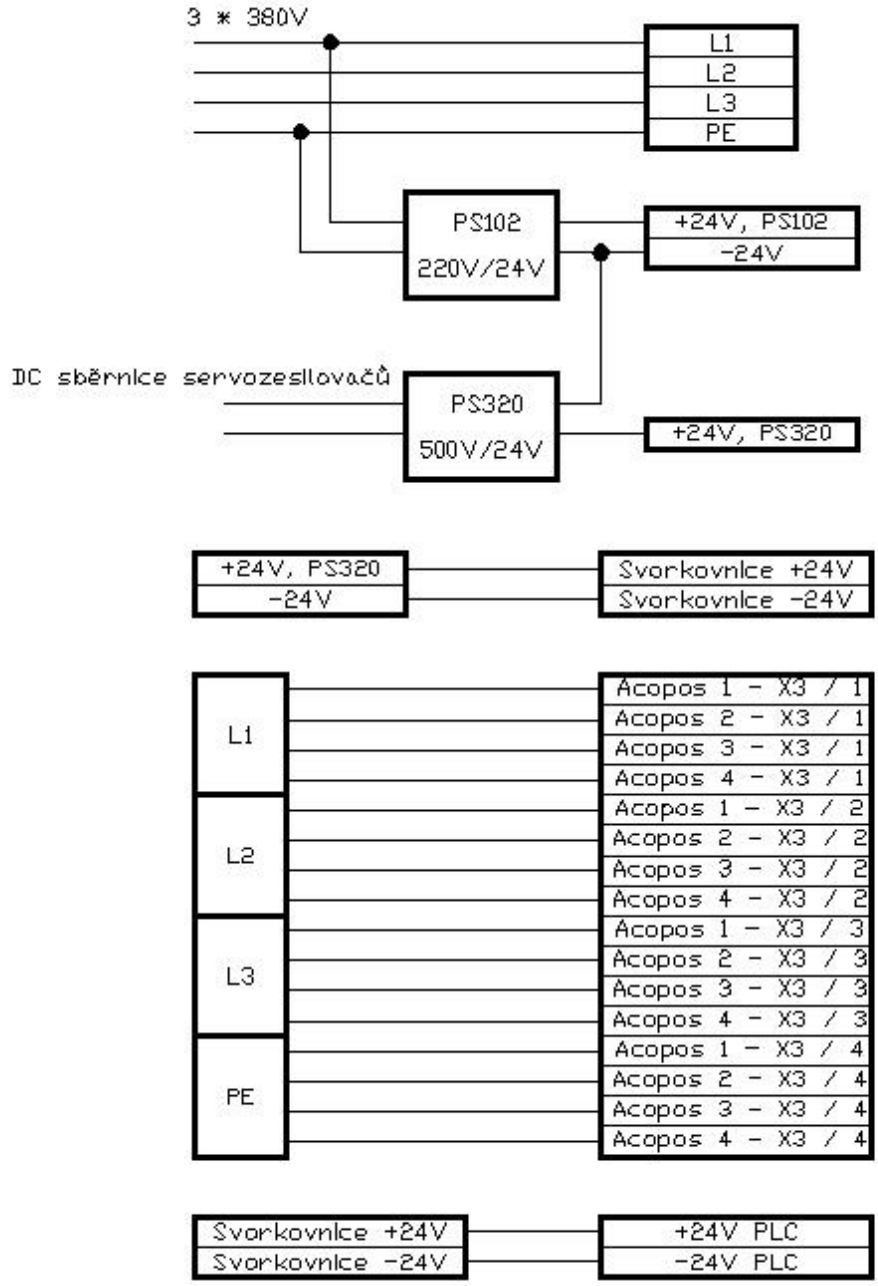
Popis označení kontaktů Wago svorkovnice



Acopos 2 / k 3 znamená: Servozesilovač (Acopos) č.2 / kontakt 3 na konektoru X1

Příloha č. 14 – Zapojení napájení modelu

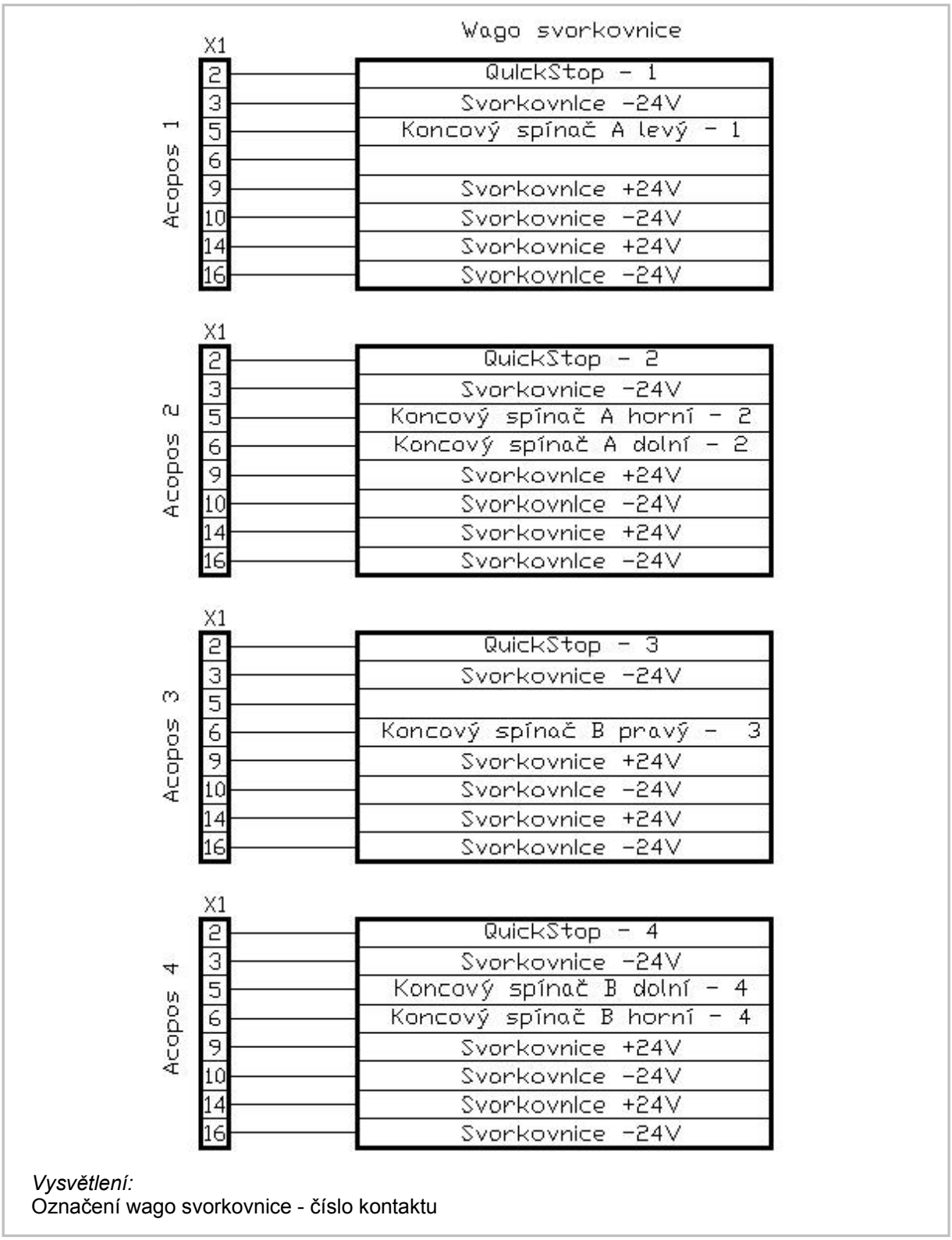
Realizace propojení napájení 380 V a 24 V.



Vysvětlení:
Acopos 1 – X3 / 1 znamená: Servozesilovač adresa č.1 konektor X3 kontakt 1

Příloha č. 15 – Propojení řídicích konektoru X1 servozesilovače

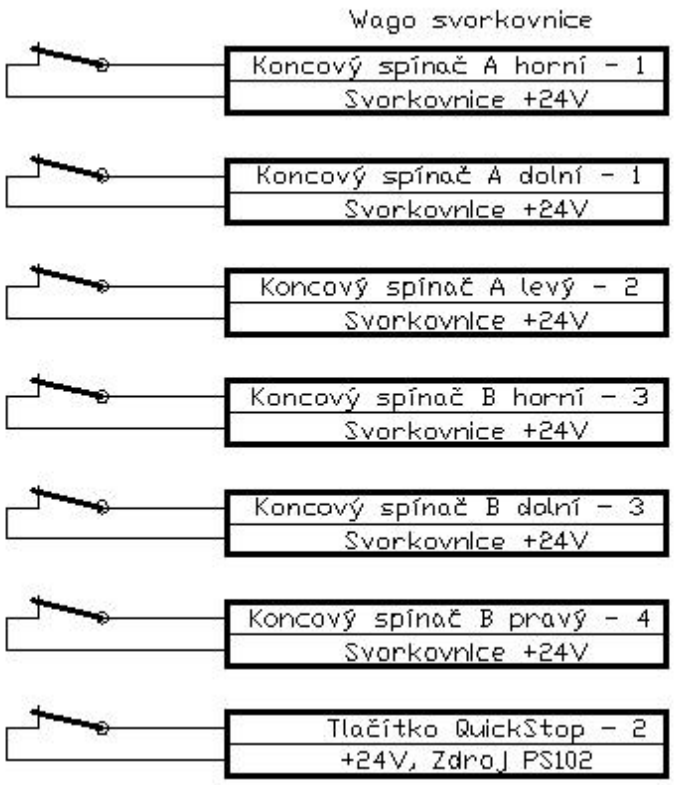
Propojení řídicího konektoru X1 jednotlivých servozesilovačů z kotakty wago svorkovnice.



Vysvětlení:
Označení wago svorkovnice - číslo kontaktu

Příloha č. 16 – Zapojení spínačů a tlačítek

Propojení koncových spínačů a tlačítka quickstop s kontakty wago svorkovnice.



Vysvětlení:
Označení wago svorkovnice - číslo kontaktu

Příloha č. 17 – Webové stránky pro ovládání modelu

Ukázka webových stánek pro ovládání jednotlivých funkcí modelu.

Žongler :

Ovládání z : Web Server

Funkce : Zongler

Potvrzení ovládání žongléu :

Žongler

Nastaveni profilu - ručně :

V1 V2 V3 V4 Nahrej

Nahrání profilu do servozesilovače - automaticky : (není třeba ručne zadávat)

1* Přehození A -> B | Přehozování A <-> B 1 ball

1* Přehození B -> A | Přehozování A <-> B 2 ball

Restart sekvence nastaveni vacek :

Restart

Ovládání modelu : pomocí profilu - Prehozeni z A na B

Start Stop

Stav : 1 = ano, 0 = ne

.	S 1	S 2	S 3	S 4	Virt
Nastaveno ?	.0	.0	.0	.0	.0
Chod ?	.0	.0	.0	.0	.0
Je porucha ?	.0	.0	.0	.0	.1

Katapult :

Ovládání z : Web Server

Funkce : Katapult

Potvrzení ovládání katapultu :

Katapult

Volba strany : Aktuální strana je = Str. A

str.A str.B

Rychlost vyhození : 5 m/s

Výška chycení míčku : Aktuální výška chycení je = 50.000000

Výška

Aktuální krok : Wait

Spuštění :

Start

Příloha č. 18 – Přiložené CD

Součástí diplomové práce je přiložené CD obsahující čtyři hlavní adresáře:

Manuály

Manuály - pro práci v prostředí Automation Studio, komunikace Ethernet Powerlink, motorů, servozesilovače, programovatelného logického automatu a interface modulů.

Řídicí SW

Řídicí program pro Power Panel, konfigurační soubory pro webový server a html stránky pro vizualizaci. Instalační soubor programu PVI Tree View a VNC.

Text

Tato práce v digitální podobě a podklady pro její přípravu, jako obrázky a fotografie.

Zapojení

Dokumentace k zapojení všech komponent modelu, od zdroje, přes tlačítka až po řídicí jednotky.