VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV AUTOMATIZACE A INFORMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

ŘÍZENÍ A SYNCHRONIZACE ELEKTRICKÝCH POHONŮ V NI LABVIEW

CONTROL AND SYNCHRONIZATION OF ELECTRIC DRIVES IN NI LABVIEW

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — SILVANO MARTINI
AUTHOR

VEDOUCÍ PRÁCE — Ing. PAVEL HOUŠKA, Ph.D.
SUPERVISOR

BRNO 2015

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav automatizace a informatiky
Akademický rok: 2014/2015

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

student(ka): Silvano Martini

který/která studuje v **bakalářském studijním programu**

obor: **Aplikovaná informatika a řízení (3902R001)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Řízení a synchronizace elektrických pohonů v NI LabVIEW

v anglickém jazyce:

Control and synchronization of electric drives in NI LabVIEW

Stručná charakteristika problematiky úkolu:

Cílem práce je seznámení s možnostmi řízení elektrických pohonů s řídicími jednotkami s rozhraním EtherCAT v prostředí NI LabVIEW. V rámci práce bude využit modul NI SoftMotion který obsahuje podporu pro komunikaci s řídicími jednotkami po EtherCAT a zároveň obsahuje podporu pro pohonů po zadaném profilu.

Cíle bakalářské práce:

1. Seznamte se vlastnostmi a použitím modulu SoftMotion.
2. Seznamte se s podporou rozhraní EtherCAT pro NI LabVIEW a SoftMotion.
3. Navrhněte řízení pohonu po zadaném profilu a řízení realizujte.
4. Seznamte se s možnostmi řízení více pohonů v SoftMotion a navrhněte řízení souřadnicového stroje.

Seznam odborné literatury:

[1] Pavelka, J. - Čeřovský, Z. - Javůrek, J.: Elektrické pohony, Nakladatelstvi ČVUT, Praha 2003
[2] SHELL, Richard L a Ernest L HALL. Handbook of industrial automation. New York: M. Dekker, c2000, xii, 900 p. ISBN 0-8247-0373-1.
[3] MARSHALL, Perry S a John S RINALDI. Industrial ethernet. 2nd ed. Research Triangle Park, NC: ISA, 2005, xi, 129 p. ISBN 15-561-7892-1.
[4] BITTER, Rick, Taqi MOHIUDDIN a Matt NAWROCKI. LabView advanced programming techniques. 2nd ed. Boca Raton: CRC Press, c2007, 499 s. ISBN 08-493-3325-3.

Vedoucí bakalářské práce: Ing. Pavel Houška, Ph.D.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2014/2015.

V Brně, dne 25.11.2014

L.S.

Ing. Jan Roupec, Ph.D. Ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. Děkan fakulty

ABSTRAKT

Tato práce se v počátku věnuje základům řízení motorů a principu funkce sběrnice EtherCAT. Většina práce je pak věnována rozšiřujícímu modulu SoftMotion pro vývojové prostředí NI LabVIEW. Je zde popsán základní koncept modulu, způsoby použití funkcí, různé přístupy programování a názorné příklady. Závěrem je zmíněna realizace řízení hardwaru.

KLÍČOVÁ SLOVA

Řízení pohonu, EtherCat, NI LabVIEW, SoftMotion

ABSTRACT

This thesis in his beginning describes basics of motor controlling and function principles of EtherCAT bus. Majority of the thesis is devoted to extending module SoftMotion for development environment NI LabVIEW. There is described the basic concept of the module, ways of functions using, various programming approaches and examples. Realization of hardware control is mentioned at the end.

KEYWORDS

Drive control, EtherCat, NI LabVIEW, SoftMotion

PROHLÁŠENÍ O ORIGINALITĚ

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a s použitím uvedené odborné literatury.

V Brně, dne 29.5.2015

BIBLIOGRAFICKÁ CITACE

MARTINI, S. *Řízení a synchronizace pohonů v NI LabVIEW.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2015. 41 s. Vedoucí bakalářské práce Ing. Pavel Houška Ph.D.

PODĚKOVÁNÍ

Tímto bych rád velmi poděkoval svému vedoucímu práce Ing. Pavlu Houškovi, Ph.D. za vedení, cenné rady při řešení problému a věnovaný čas v průběhu vytváření této práce. Poděkování patří také rodině, která mi vytvořila příhodné podmínky pro práci.

OBSAH

1. Úvod

Automatizace se zabývá řízením strojů, nebo celých výrobních procesů. Umožňuje nahradit lidskou práci, čímž je dosahováno vyšší přesnosti, rychlosti, snižování nákladů a chybovosti. V neposlední řadě je umožněno nasazení v podmínkách, kde by člověk pracovat nemohl.

V dnešní době zažívá automatizace rychlý vývoj hlavně díky rozvoji informačních technologií, které se stávají dostupnější. Realizace pak probíhá především elektronicky a to pomocí regulátorů, programovatelných logických automatů (PLC), mikro kontrolérů, průmyslových PC a programovatelných řídicích kontrolérů (PAC). Každý ze systémů je potřeba naprogramovat, k čemuž nám mohou posloužit buď klasické programovací jazyky (C, C++ atd.), nebo speciální software, díky nimž je programování ulehčeno (LOGO! Soft Comfort, LabVIEW atd.).

Všechny řídicí systémy pak využívají sběrnice, např. EtherCat, ProfiNet, CAN, USB a mnoho dalších, díky kterým jsou propojeny s ostatními zařízeními, senzory či řízenými prvky a mohou si vyměňovat data.

Tato práce se okrajově věnuje základům řízení motorů a popisuje princip funkce sběrnice EtherCAT.

Podrobněji je poté popsán rozšiřující modul pro NI LabVIEW s názvem SoftMotion, kterému se věnuje téměř polovina práce. V práci je tedy uveden jak základní koncept práce s modulem, tak i několik návrhů řízení u kterých je vysvětleno v čem spočívají, jaké mají vlastnosti a případně čemu je potřeba se vyvarovat.

Při návrzích řízení jsou použity simulované osy, v čemž spočívá výhoda modulu SoftMotion, že není pro návrh nutné znát konkrétní hardware, který je možné přiřadit k dané ose až dodatečně.

Použití konkrétního hardwaru se tato práce věnuje pouze minimálně, jelikož na něm není programování pomocí modulu SoftMotion přímo závislé.

2. Základy řízení elektrických pohonů

Elektrický pohon tvoří elektrický motor, snímače polohy anebo rychlosti a řídicí jednotka. Podle typu použitého motoru a požadavků kladených na pohon jsou používány různé typy snímačů a řídicí jednotky.

2.1. Elektrické motory

2.1.1. DC motory

Nejjednodušší motory z pohledu ovládání jsou stejnosměrné motory. Tyto motory pro změnu toku proudu do jednotlivých cívek rotoru využívají komutátory, což jsou mechanické rotační přepínače. Jelikož je změna toku proudu už řešena komutátorem, zbývá již jen řídit směr otáčení a rychlost, respektive výkon. Změna směru otáčení se řeší změnou polarity přiváděného napětí a výkon pomocí změny velikosti napětí.

2.1.2. Synchronní a asynchronní motory

Tyto motory fungují na principu točivého magnetického pole, jenž je vytvářeno střídavým proudem. Jelikož v elektrické síti je frekvence napětí 50 Hz (Evropa), znamenalo by to konstantní otáčky magnetického točivého pole 3000min^{-1}. Jelikož tento fakt by znemožňoval rozběhnutí synchronního motoru bez pomocného vinutí nebo motoru a u asynchronního motoru neefektivní provoz, je potřeba tuto frekvenci měnit a tím řídit motor. Ke změně frekvence elektrického proudu se používají frekvenční měniče.

2.1.3. Krokové motory

Krokové motory jsou speciálním typem synchronních motorů, které jsou výrazně jednodušší na řízení a jsou použitelné pro polohové řízení bez zpětné vazby. Ovládání krokového motoru je založeno na postupném pulsním napájení pólů statoru, které je řízeno elektronicky. Při změně napájené pólové dvojice dojde k pohybu o jeden krok. Velikost tohoto kroku je dána počtem pólů, či zubů rotoru. Výhodou těchto motorů je jejich předvídatelná poloha (pokud nedojde k přetížení nebo překročení mezní rychlosti) na základě počtu odeslaných pulsů a tedy provedených kroků.

2.2. Snímače

Při řízení elektrických motorů, kdy je požadováno dosažení přesné polohy natočení či rychlosti (servo motory), je nezbytné použít zpětnovazební prvky.

2.2.1. Snímaní rychlosti a polohy

Pro snímání polohy a rychlosti se používají enkodéry a resolvery, pro snímání rychlosti se používají také tachogenerátory.

Enkodér je snímač polohy vyznačující se digitálním výstupem. Jeho princip spočívá v detekci značek na kruhovém či přímém pravítku, které se oproti snímači pohybuje. Dle typu pravítka se může jednat o enkodér inkrementální nebo absolutní.

Snímač polohy resolver je složen z rotoru s jedním vinutím a statoru s dvěma vinutími pootočenými vzájemně o 90°. Princip funkce je podobný transformátoru, kde na vinutí statoru je přivedena nosná frekvence a na cívkách statoru je snímán výstupní signál, závislý na natočení rotoru. Výstupní signál z cívek statoru je potřeba dále zpracovat.

Tachogenerátor, nebo také tachometr je snímačem rychlosti. Má stejnou konstrukci jako stejnosměrný motor s tím rozdílem, že vinutí je použito vysoko-odporové pro zamezení brzdění. Princip funkce je také totožný, přičemž tachogenerátor je provozován v generátorickém režimu. Výstupem tachogenerátoru je napětí úměrné rychlosti.

2.2.2. Snímání proudu

Snímání proudu je nebytné z důvodu řízení krouticího momentu a ochrany motoru před přetížením. Pro tyto účely se používají Hallovy senzory nebo bočníky.

Hallův senzor je polovodičová součástka snímající magnetické pole, vytvořené průtokem proudu cívkami. Výstupem Hallova senzoru je napětí, které je potřeba ve většině případů zesílit.

Bočník je rezistor zapojený sériově s cívkou motoru. Na tomto rezistoru je měřen úbytek napětí a na jeho základě vypočten protékající elektrický proud.

2.3. Řízení elektrických pohonů

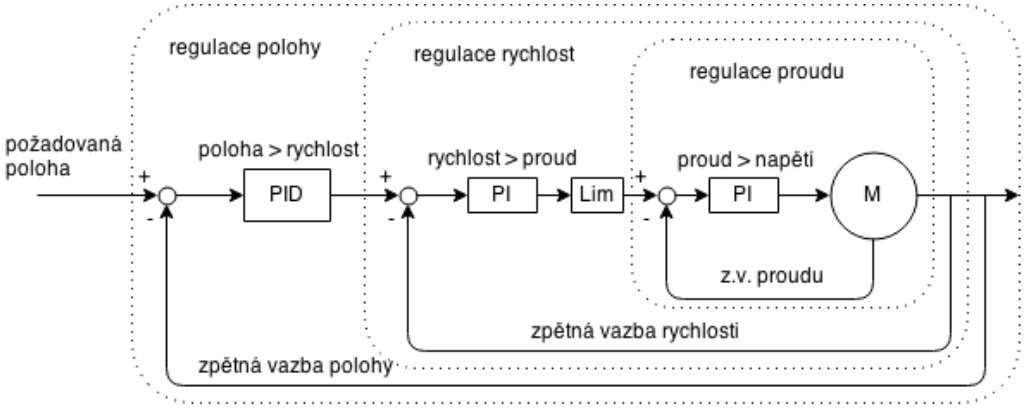
Řízení elektrických motorů zajištují řídicí jednotky, které jsou podle typu řízeného motoru nazývány také frekvenčním měniči nebo krokovacími jednotkami. Řízení pohonů je realizováno v různých pracovních režimech, mezi základní režimy patří:

1. momentové řízení (Torque Profile Mode)
2. rychlostní řízení (Velocity Mode, Profile Velocity Mode)
3. polohové řízení (Profile Position Mode, Interpolated Position Mode)

Podle podporovaných pracovních režimů a typu motoru je integruje řídicí jednotka tyto části:

- vstupy pro připojení nutných snímačů a zpracování signálů,
- moduly pro výpočty nutné kinematiky a dynamiky,
- regulátory,
- modulátory a výkonovou výstupní část,
- komunikační rozhraní.

Struktura regulačního obvodu závisí na používaném pracovním módu. Při velmi zjednodušeném pohledu je možné na regulační obvody nahlížet takto (obr. 1): V případě momentového řízení je regulační obvod tvořen pouze proudovou regulací. Při použití rychlostního řízení tvoří regulační obvod kaskáda regulátorů, kde je výstup rychlostního regulátoru přiveden na vstup proudového regulátoru. Pro polohové řízení je v kaskádě předřazen regulátor polohy, jehož výstup je přiveden na vstup regulátoru rychlosti.



Obr. 1 - Schéma řízení elektrického motoru.

3. EtherCAT

Jedná se o sběrnici vyvinutou firmou Beckhoff Automation, která ji představila v roce 2003, přičemž při jejím vývoji měla za cíl rychlost, přesnou synchronizaci a nízké ceny potřebného hardwaru. EtherCAT je postaven na základech technologie Ethernet. [1]

Jako sběrnice se v počítačové terminologii označuje systém komunikace pro přenos dat mezi elektronickými zařízeními určitého systému. Pojmem sběrnice se chápe jak fyzická část, tedy vodiče, tak i protokol, kterým jsou definována pravidla pro komunikaci mezi zařízeními.

3.1. Základ funkce

Princip funkce EtherCATu spočívá v odeslání zprávy (rámce) prostřednictvím hlavního kontroléru (master), která pak prochází všemi připojenými podřazenými zařízeními (slaves). Každé připojené slave zařízení čte z rámce data pro něj určená tzv. „za letu“ (on the fly) a vkládá svá data do rámce dál ve směru, jak prochází zařízením. Schopnost čtení a zápisu „on the fly“ je zařízením dána díky ESC (EtherCAT slave controller), realizovaným jako specifický integrovaný obvod (ASIC) nebo programovatelné pole FPGA. Rámec je tedy zpožděn pouze zanedbatelně, jelikož odpadá prodleva na zpracování, kde by musel být celý rámec přijat, zpracován a až poté odeslán dál. Takto prochází rámec všemi připojenými zařízeními do doby, než se dostane k poslednímu zařízení v segmentu topologie, který detekuje otevřený výstupní port a pošle rámec zpět stejnou cestou, což je umožněno díky plnému duplexu technologie Ethernet.

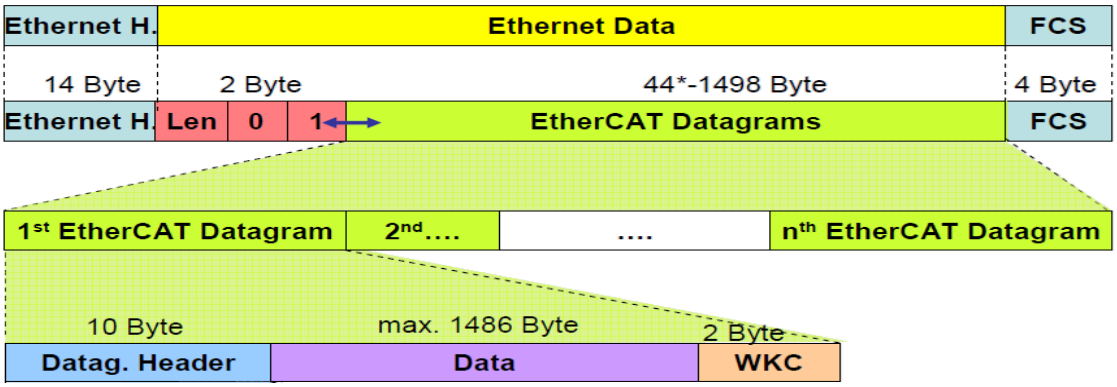
Master zařízení je jediné zařízení v celé síti, které má umožněno vytvořit a vyslat rámec dat, ostatní jej mohou pouze upravovat. Tímto konceptem je dosaženo prevence nepředvídatelných prodlev a garantována schopnost práce v reálném čase.

Komunikace mezi slave zařízeními je realizovatelná pouze dvěma způsoby. Slave zařízení může poslat data přímo jinému zařízení po směru toku dat, tato varianta je však závislá na fyzickém uspořádání. V druhé variantě, kdy by bylo potřeba poslat data proti směru komunikace, dochází ke komunikaci přes master zařízení a data jsou přenesena ve dvou cyklech. [1]

3.2. Protokol

Rámec

Rámec EtherCATu a jeho datové prvky, neboli datagramy jsou zapouzdřeny do datové částí standardního Ethernetového rámce, kde je rozpoznáno, že jde o EtherCAT komunikaci, podle identifikátoru 0x88a4 v poli EtherType. Struktura celého rámce tedy obsahuje hlavičku Ethernetu (MAC adresy a EtherType), údaj o datech (délka, reservováno, typ), EtherCAT datagramy a závěrečnou kontrolní sekvenci rámce. Samotný datagram je pak složen z hlavičky datagramu, dat a čítače přístupu k datům. Názorněji je struktura vyobrazena na Obr. 2. [1]



Obr. 2 - Struktura EtherCAT rámce [1].

Popis obrázku:

Header – hlavička

Len – length – délka

WKC – working counter – čítač operací s daty

FCS – frame control sequence – kontrolní sekvence rámce

Adresace

Adresa zařízení, pro které je datagram určen, je uchovávaná v hlavičce EtherCAT datagramu, je pro ní vyhrazeno 32 bitů a může mít tři podoby.

Hlavní využívaná adresace je logická, kde nejsou slave zařízení adresována individuálně, ale pomocí sekcí, které mohou zahrnovat více zařízení. Přiřazení fyzického zařízení k logické adrese je konfigurováno master zařízením při inicializaci. Pokud je v jedné sekci přiřazeno více zařízení, všechny mohou být adresovány jedním datagramem, z nějž dále čtou, nebo do něj zapisují svá data. Velikost rámce se všemi datagramy může nabývat velikosti až 4GB. Data v tomto rámci mohou být fragmentovaná a jejich pořadí je nezávislé na fyzickém uspořádání síťových prvků.

Další dvě možnosti představuje 16 bitů dlouhá informace o pozici a stejně dlouhá informace o adresování lokální paměti slave zařízení (celkem opět 32bitů). V prvním případě je hodnota pozice inkrementována každým slave zařízením, což je vhodné pro skenování hardwarové konfigurace a využívá se při inicializaci sítě. Po kontrole hardwarové konfigurace může být přidělena kterémukoli zařízení fixní adresa, což je druhá možnost, která již bude nezávislá na topologii a tím vznikne možnost přímé adresace. Fixní adresa po odpojení elektrické energie není zachována. [1]

Mailbox služby

Jednou z významných charakteristik EtherCATu je i podpora více protokolové komunikace, která je realizována pomocí standardizovaného síťového tunelování, tzv. mailboxu. Síťové tunelování spočívá v zapouzdření datových prvků jednoho protokolu, do datových prvků protokolu jiného. Jinými slovy stejně jak je zapouzdřen rámec EtherCATu do rámce Ethernetu, je možné dále zapouzdřit do datové části EtherCATu ještě další protokol. [2]

Mailbox služeb se využívá zejména pro následující komunikaci:

- Ethernet over EtherCAT (EoE) – klasická Ethernet komunikace při zachování real-time vlastností
- CANopen over EtherCAT (CoE) – umožňuje přístup ke CANopen zařízením, jejich objektům a komunikaci pomocí SDO a PDO
- File access over EtherCAT (FoE) – přenos souborů
- Servo drive over EtherCAT (SoE) – využití komunikačního prostředí SERCOS™

3.3. Topologie

Sběrnice EtherCAT umožňuje kombinaci více klasických topologii, jako jsou hvězda, strom, sériové zapojení a jiné. Kombinované topologie jsou umožněny díky zařízením se dvěma a více porty, které zprostředkovávají komunikaci v uzlech.

Ačkoli je možné sestavit jakoukoli topologii, rámec vždy probíhá sítí jako by se jednalo o kruh. Tedy pokud dojde na konec jedné větve a dostane se na uzavřený port, vrací se zpět do uzlu, kde je poslán do další větve.

V aplikacích, kde je potřeba připojovat a odpojovat zařízení za provozu se využívá automatického uzavření (otevření), odpojeného (připojeného) výstupního portu. Tím pádem je rámec poslán dál do sítě a není omezena její funkčnost.

Pro propojení zařízení se využívá klasických kabelů pro Ethernet a z toho plynou i limity. Je tedy například možné propojit zařízení až na vzdálenost 100m bez použití opakovače v případě varianty 100BASE-TX.

Do sítě může být teoreticky připojeno až 65536 zařízení. [1]

3.4. Synchronizace

V aplikacích vyžadujících simultánní akce, například řízení pohybu, je nezbytná přesná synchronizace. K tomu jsou v EtherCATu použity tzv. distribuované hodiny.

Jelikož každé slave zařízení používá k synchronizaci interní hodiny, je potřeba tyto hodiny kalibrovat. Kalibrace hodin v jednotlivých uzlech je prováděna ve třech fázích.

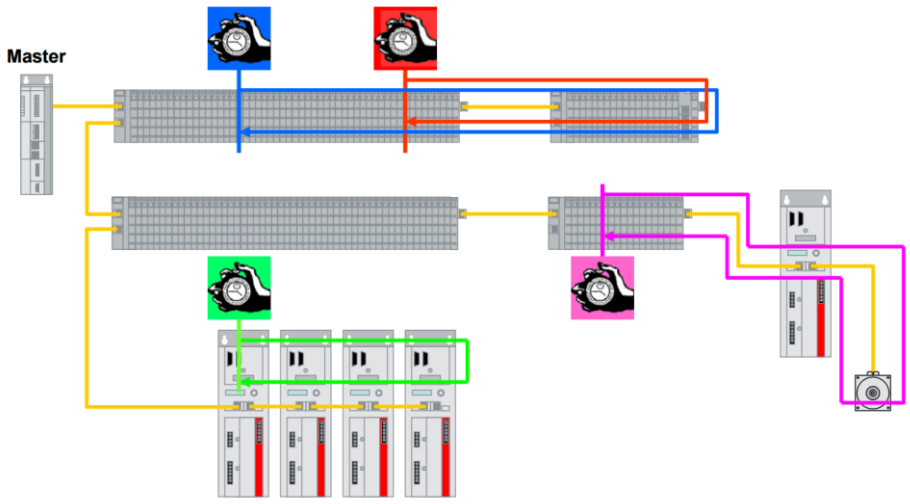
Prvním krokem k synchronizaci hodin je měření prodlevy procházejícího rámce topologií. Master pošle rámec a každé slave zařízení si zapíše časový rozdíl mezi průchodem paketu směrem „po proudu“ a jeho vracením se zpět.

Následuje kompenzace posunu. Master vypočítá z kopií interních hodin zaslaných slave zařízeními časové posuny a následně pošle rámec, kterým je synchronizuje.

Posledním krokem je kompenzace driftu, která je prováděna časovou regulační smyčkou.

Proces kalibrace se vykonává při startu sítě, nebo pokud je požadováno, může být spouštěn i při provozu.

Pokud mají všechna zařízení stejnou časovou informaci, mohou následně nastavit výstupy současně a taktéž vstupním informacím přiřadit velmi přesný čas pořízení. [2]



Obr. 3 - Ilustrace měření prodlevy na slave zařízeních [1].

3.5. Detekce chyb

EhterCAT slave kontrolér v každým uzlu kontroluje správnost rámce pomocí kontrolní sekvence, pokud nastane chyba, je inkrementován čítač chyb a všechna následující zařízení daný rámec ignorují, včetně master zařízení, který jej vyřadí.

Kontrola funkce samotných zařízení se pak provádí přes inkrementaci tzv. pracovního čítače (WKC), který je inkrementován podle způsobu přístupu k datům všemi adresovanými zařízeními daného datagramu. Pokud je hodnota čítače jiná než předpokládaná, master daný rámec opět vyřadí. [1]

4. NI LabVIEW a SoftMotion

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) je grafické vývojové prostředí využívající vizuální programovací jazyk G od společnosti National Instruments.

Oproti textově založeným programovacím jazykům, které využívají textových instrukcí k určení průběhu vykonávaní programu, LabVIEW využívá „dataflow“ programování. Při využívání „dataflow“ programování je určena posloupnost vykonávání instrukcí a funkcí podle toku dat napříč blokovým diagramem. Tento tok dat je určen pomocí propojovacích linek.

Z důvodu, že programy a podprogramy imitují přítomnost a funkčnost reálných fyzických instrumentů, jako jsou osciloskopy nebo multimetry, říká se jim virtuální instrumenty, neboli zkráceně VI. [3]

4.1. Podpora protokolu EtherCAT v LabVIEW

Podpora protokolu je v LabVIEW realizována prostřednictvím ovladače „NI-Industrial Communications for EtherCAT®“ na real-time zařízeních.

Práce s EtherCAT zařízeními v LabVIEW spočívá v jejich přidání do projektu a dále již jen využívání jejich vstupů / výstupů.

Pro přidání zařízení do projektu je potřeba nejdřív načtení konfiguračního .xml souboru (dodávaného výrobcem hardwaru) na EtherCAT master zařízení, z něhož získá master informace o vstupech a výstupech daného hardwaru a bude schopno jej v síti identifikovat. Načtení konfiguračního souboru se provede pomocí kontextové nabídky master zařízení Utilites > Import Device Prifiles… Po načtení souboru je již možné přidat zařízení do projektu pomocí kontextové nabídky master zařízení, New > Target and Devices…, kde se dané zařízení, po připojení, již zobrazí.

Jelikož je EtherCAT real-time sběrnice a vývoj aplikace v LabVIEW je prováděn většinou na zařízeních bez této vlastnosti, není v těchto případech možná přímá komunikace. Pro EtherCATovou komunikaci je tedy většinou nezbytné použít real-time EtherCAT master zařízení, které s počítačem komunikuje přes Ethernet, RS232 či PCI a zprostředkovává přístup do sítě EtherCAT.

4.2. SoftMotion

Jedná se o zásuvný modul pro vývojové prostředí NI LabVIEW, zprostředkovávající práci s pohony - jejich konfiguraci, řízení či simulaci.

4.2.1. Přístup

Základním konceptem práce s modulem SoftMotion je vytvoření tzv. motion resources, neboli pohybových prostředků, které reprezentují reálný pohon, nebo jeho součást. Těmito prostředky jsou osy, souřadnice a tabulky. Po vytvoření je v blokovém diagramu s prostředky zacházeno pouze pomocí referencí.

Pro realizaci řízení v blokovém diagramu jsou k dispozici funkční bloky obstarávající vykonání různých druhů pohybu, či jejich snímání.

4.2.2. Motion resources - Pohybové prostředky

Axis - Osa

Osa se konfiguruje pro řízení jednoho pohonu a to buď krokového, nebo servo motoru. Přidat do projektu můžeme i osu simulovanou.

V případě řízení krokového motoru se rozlišuje, zda se jedná o režim se zavřenou nebo otevřenou smyčkou, tedy jestli bude docházet ke korekci polohy a rychlosti na základě zpětné vazby, nebo bude poloha a rychlost vypočtena pouze na základě řídicích pulzů.

Servo motory pracují vždy v režimu uzavřené smyčky se zpětnou vazbou a pro dosažení přesného řízení využívají PID kontrolní smyčku. Parametry PID regulace je možné upravovat v konfiguraci osy. Tato regulace může pracovat ve dvou režimech, založených na kontrole pozice, nebo krouticího momentu. [4]

Coordinate - Souřadnice

Tzv. souřadnice slouží pro seskupení více os do jednoho prostředku. Nesmírnou výhodou a i účelem využívání souřadnic je automatické přepočítání průběhu pohybu tak, aby zahrnuté osy nejen započali, ale i dokončili pohyb synchronizovaně.

Table - Tabulka

Tabulky rozlišujeme dvojího druhu – contour table a camming table.

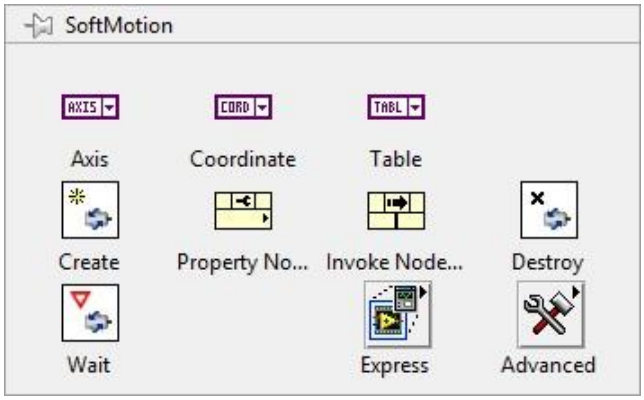
Contour table, je tabulka obsahující poziční body pro jednu, či více os, které tvoří konturu, kterou bude pohon následovat. K uchovávání hodnot lze použít textový soubor, nebo zásobník.

Druhý typ tabulky slouží k zápisu hodnot pro elektronické vačky a hodnoty jsou uchovávány pouze v textovém souboru.

4.2.3. Funkce

Po instalaci modulu přibyde v paletě nástrojů několik nových položek, které jsou k nalezení pod položkou Vision and Motion > SoftMotion (Obr. 4)

Hlavními položkami jsou již zmiňované reference na motion resource, funkce pro vytvoření SoftMotion objektu pohybu, jeho odstranění a čekání na dokončení. Dalšími nezbytnými prvky pro práci s objekty jsou Property a Invoke Node.



Obr. 4 - Hlavní nabídka funkcí modulu SoftMotion.

Create – vytvoření SoftMotion objektu pohybu

Jedná se o polymorfní VI, vytvářející objekt pro určitý druh pohybu. Pomocí selektoru se vybere, o jaký konkrétní objekt se má jednat. Možnosti jsou: pohyb přímočarý, kruhový a po kontuře, hledání referenčních bodů, snímání a porovnávání polohy, ukončení pohybu, převodování, vačkový vztah, tabulku kontury, nebo profil.

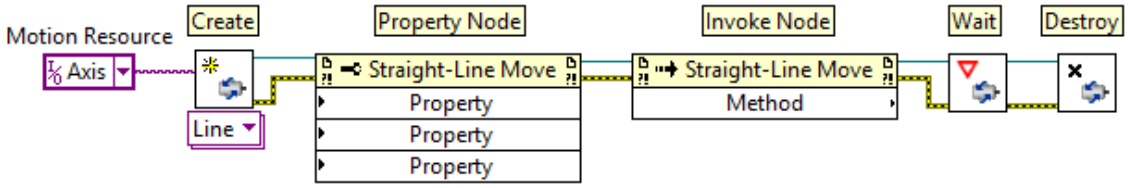
Property node – vlastnosti objektu

Pomocí property node se přistupuje k parametrům daného objektu. Po připojení reference objektu pohybu je možné vybírat z konkrétních vlastností daného objektu a nastavit, zda se bude hodnota číst či zapisovat.

Invoke node – metody objektu

Pro vyvolání metody daného objektu slouží právě invoke node. Po zvolení metody se mohou objevit ještě upřesňující parametry.

Základní sekvence funkcí (Obr. 5) pro vykonání určitého úkonu pomocí SoftMotion objektů je následující. Z konstanty, či ovládacího prvku je přiveden motion resource na funkci create, z níž povede reference na objekt, s jehož vlastnostmi a metodami budeme pracovat pomocí property a invoke node. Pokud je třeba, zařadí se prvek wait, který čeká na dokončení úkonu, případně provedeme odstranění. Odstranění proběhne samo při ukončení VI i bez použití funkce odstranění, nicméně je vhodné jej používat u složitějších, dlouho běžících aplikací, aby docházelo k uvolňování prostředků.

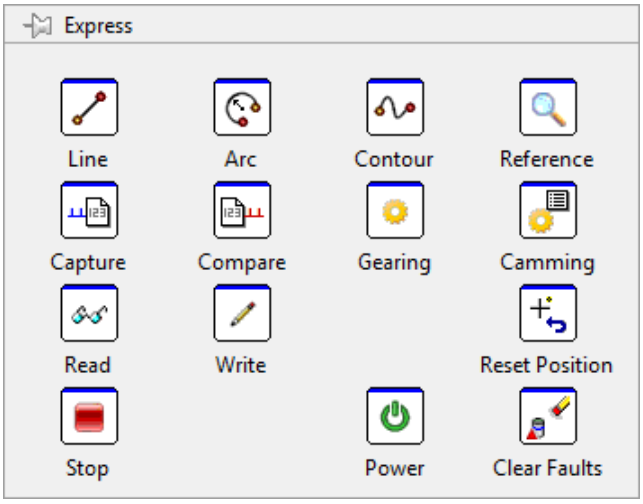


Obr. 5 - Základní sekvence práce s objekty pohybu.

Express

V nabídce Express (obr. 6) nalezneme již předpřipravené tzv. Express VI, které v sobě zahrnují vytvoření objektu i s nastavením parametrů a vyvoláním metod. Vlastnosti daného VI lze nastavit buď pomocí vstupů, nebo v dialogovém okně vlastností, vyvolaném kliknutím pravým tlačítkem myši a zvolením položky vlastnosti (properties). Výhodou dialogového okna vlastností může být např. zobrazení profilu polohy, rychlosti a zrychlení v grafech.

Ve výchozím stavu je Express VI nastaveno jako synchronní a tudíž dojde k jeho vykonání ihned, když je voláno a následující kód bude pokračovat až po jeho dokončení. Synchronicitu lze změnit ve vlastnostech VI.



Obr. 6 - Nabídka Express a její položky.

Advanced – pokročilé

V podnabídce pokročilé je k dispozici VI pro změnu režimu motion resource (aktivní, konfigurační) a další dvě podnabídky - funkční bloky a synchronizace.

Nabídka funkčních bloků je totožná s nabídkou express VI. Funkční bloky zastávají stejné funkce jako Express VI, s tím rozdílem, že při použití funkčního bloku je využíván asynchronní režim bez možnosti změny na synchronní. Vykonání funkčních bloků je tedy řízeno pomocí vstupu execute (vykonat) a done (dokončeno). Při využívání asynchronních funkčních bloků je nutné dávat pozor, aby nedošlo k volání jednoho funkčního bloku dříve, než bude dokončen jiný, s přiřazenými stejnými motion resource.

V nabídce synchronizace jsou k dispozici nástroje pro zajištění přesnější synchronizace běžících smyček při práci s real-time a FPGA zařízeními.

4.2.4. Konfigurace osy

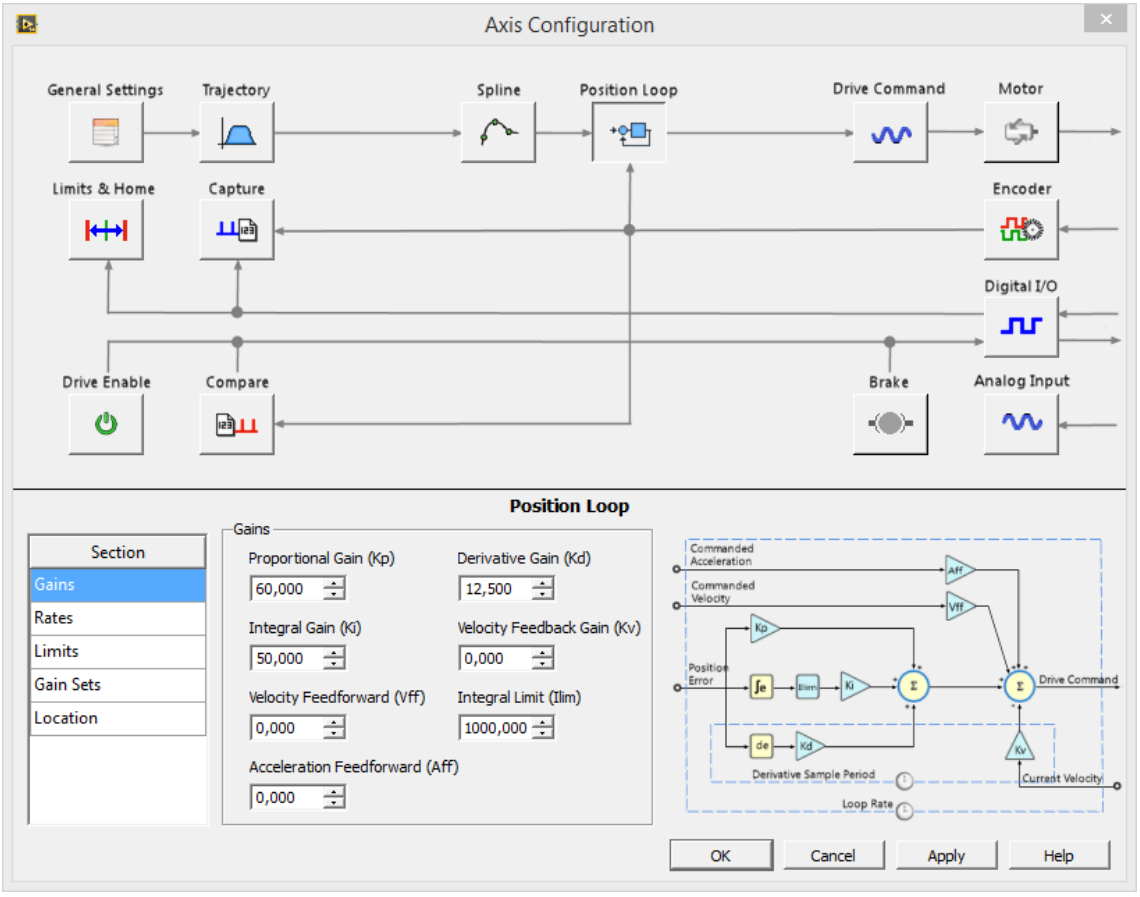
Před použitím SoftMotion osy je nutné provést její konfiguraci. K tomu slouží panel vlastností osy a dva nástroje – gain tuning panel a interactive test panel, které jsou dostupné v kontextové nabídce po kliknutí pravým tlačítkem myši na danou osu.

Panel vlastností

Pomocí tohoto panelu se konfigurují všechny nezbytné vlastnosti osy. Panel je rozdělen do dvou částí. Vrchní část zobrazuje schéma hlavních skupin parametrů, rozdělených dle jejich účelu (hlavní nastavení, trajektorie, koncové body, enkodér, digitální vstupy/výstupy atd.). Po zvolení určité skupiny ve vrchní části panelu se zobrazí jednotlivé parametry v části spodní, kde je možné je přenastavit. Zobrazené schéma a dostupné parametry se mění podle připojeného zařízení.

Před konfigurací by měli být připojeny všechny zařízení jako řídicí jednotky, motory, enkodéry, koncové snímače a ostatní související vstupy a výstupy.

Všechna nastavení je možné provést také programově přímo ve VI pomocí property node připojeného přímo na motion resource. Programové nastavení má nesmírnou výhodu, že předchází ztrátě nastavení v případě, že by došlo k restartování řídicího zařízení, nebo zavření projektu. V těchto případech se totiž nastavení vrací do výchozích hodnot a využitím programového nastavením dojde k opětovné konfiguraci při spuštění VI.



Obr. 7- Panel vlastností pro konfiguraci osy.

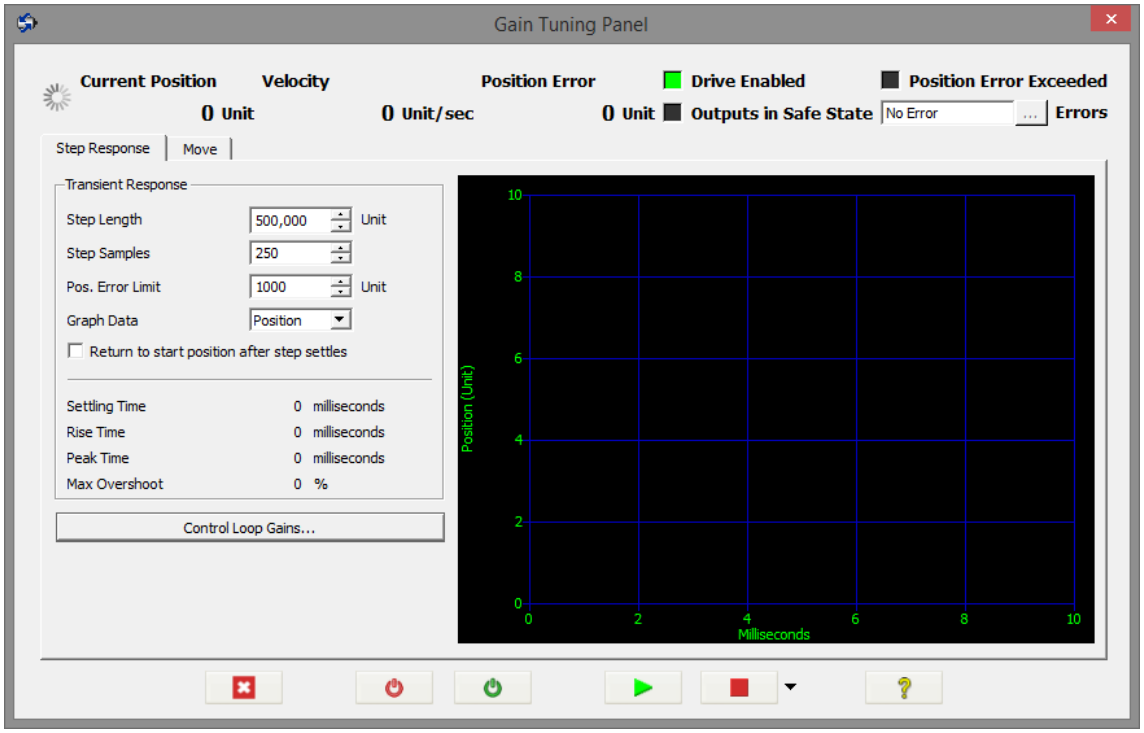
Gain Tuning Panel - Nastavení regulátoru polohy

Účelem tohoto panelu je ladění regulace polohy.

Hlavní parametry tohoto panelu jsou parametry PID regulace, tedy hlavně konstanty proporcionální, integrační a derivační složky, případně ještě související doplňkové parametry. Tyto parametry jsou k nalezení pod tlačítkem Control Loop Gains.

V levé části panelu se také nastavují parametry testu, např. při měření odezvy na skokovou změnu se nastaví velikost skoku, počet vzorků a maximální chyba polohy, nebo v případě potřeby sledování odezvy na určitý pohyb, se přepne na kartu Move, kde se nakonfigurují požadované parametry.

Po spuštění tlačítkem start ve spodní liště dojde k vykonání testu a odezva motoru se zapíše do grafu. Na základě grafu vyhodnotíme odezvu a případně budeme upravovat PID parametry až do doby, než docílíme námi požadované odezvy.



Obr. 8 - Gain Tuning Panel pro ladění regulace.

Interactive Test Panel

Posledním z trojice panelů je Interactive Test Panel, sloužící k testování chování již nakonfigurované osy. Nastaví se zde parametry pohybu, tedy cílová poloha, rychlost a akcelerace a po spuštění se do grafu vykreslí poloha v závislosti na čase.

5. Návrh řízení pohonu v NI LabVIEW

Tato kapitola se věnuje pouze samotnému programování v LabVIEW za použití modulu SoftMotion, tedy bez připojeného hardwaru, jen s pomocí simulovaných os.

Jelikož je možné navrhnout řízení v mnoha variantách, jsou zde uvedené alespoň tři možné příklady za použití různých metod.

Před samotným programováním je vhodné vytvořit projekt a přidat simulované osy, na kterých se bude program testovat. Po založení projektu v úvodní nabídce programu se provede přidání os kliknutím pravým tlačítkem myši na zařízení, na kterém se bude program spouštět (např. My Computer) a zvolí se položka New > SoftMotion Axis. V nově otevřeném okně Axis Manager, kliknutím na tlačítko Add New Axis se přidá libovolný počet os, se kterými se bude dále pracovat. Obdobným způsobem je potřeba přidat i tabulku poloh. Nyní se již vytvoří nové VI, které se naprogramuje pro řízení jedné, či více os.

5.1. Program

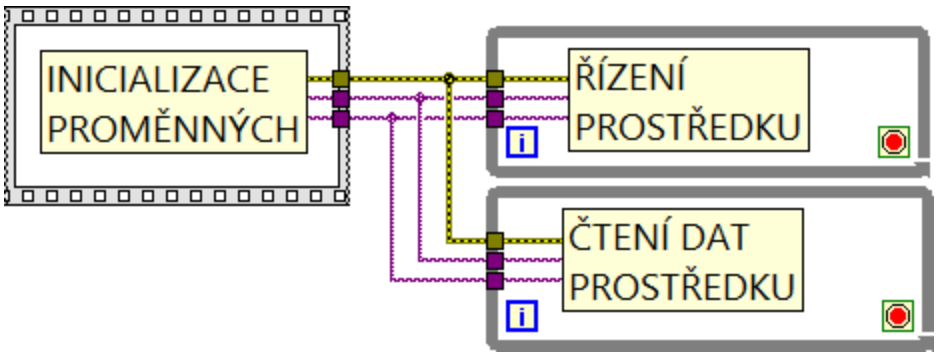
Hlavní struktura programu (Obr. 9) je rozdělena do tří částí, na inicializaci, řídicí smyčku a čtecí smyčku.

Pro inicializaci je zvolena sekvenční struktura, která proběhne pouze jednou při spuštění VI. Při inicializaci jsou přiřazeny Motion Resources, vymazána historie grafu a nastaveny výchozí hodnoty některých proměnných.

Z důvodu nezávislosti je řízení prostředku odděleno od čtení informací o daném prostředku pomocí dvou while smyček.

Čtecí smyčky všech variant jsou téměř stejné, jelikož při každé iteraci smyčky je spouštěno Express VI ať už synchronní, či asynchronní. Následně jsou přečtená data zapsána do grafu, či indikátoru. Ukončení smyčky je řízeno chybovým clusterem, nebo pomocí lokální proměnné ze smyčky řídicí. Z důvodu jednoduchosti již nebudou dále čtecí smyčky zmíněny.

Kód v jednotlivých řídicích smyčkách je popsán v následujících kapitolách, v závislosti na použitých funkcích.



Obr. 9 - Hlavní struktura programu.

5.1.1. Express VI – synchronní

Hlavní vlastností synchronních funkcí je, že při jejich vykonávání je blokováno vykonání následujícího kódu, dle konceptu dataflow, dokud nebude funkce dokončena. Omezeno je taktéž ovládání pomoci front panelu.

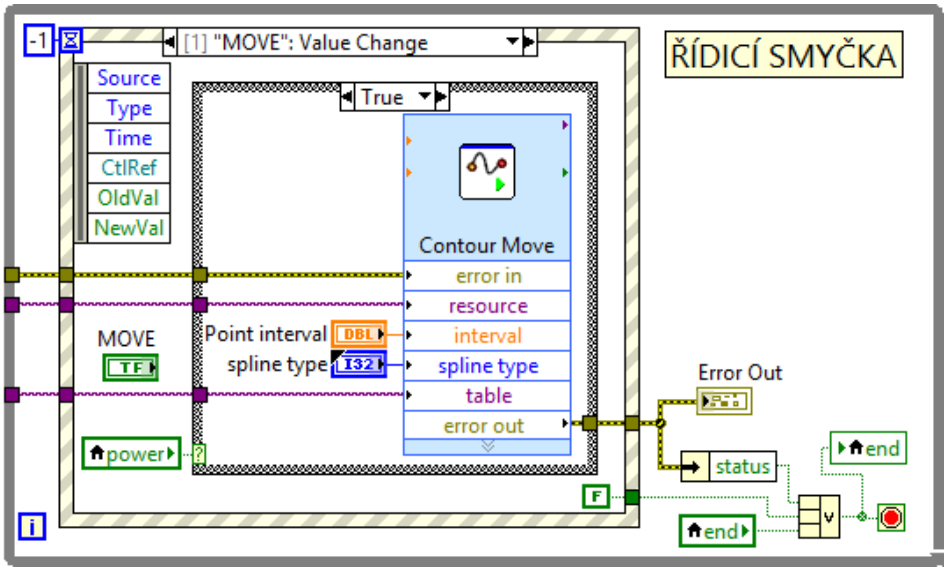
Jelikož synchronní Expres VI je vykonáno vždy, když na něj přijde řada dle dataflow, není možné jej použít volně ve while smyčce, ale je nutné jej spouštět podmíněně, například pomocí case, nebo event struktury.

Výhodou synchronního vykonávání funkcí je zamezení kolizí jednotlivých funkcí při práci se stejnými prostředky, jak tomu může nastat u vykonávání asynchronního.

Příklad

Pro ukázku práce se synchronními bloky je použita event struktura, kde se kód spouští na základě událostí. Použitá event struktura reaguje na stisknutí tlačítek Power, Move, Stop VI, Update a Position reset, přičemž v každém případě je spuštěno Express VI s odpovídající funkcí. Na ilustračním obrázku (Obr. 10) je zobrazen kód vykonávající se při stisknutí tlačítka Move.

Průběh programu je následující. Řídicí smyčka čeká na vykonání event struktury (není potřeba použít časování), tedy než dojde ke změně hodnoty jednoho z tlačítek. V případě stisknutí tlačítka Move dojde k načtení lokální proměnné Power, která indikuje, zda je povolená daná osa a řízení. Hodnota proměnné Power je přivedena na rozhodovací podmínku case struktury, kde se podle její hodnoty vykoná určitý kód obsažený uvnitř case struktury. Pokud je tedy povoleno řízení i osa, načtou se potřebné proměnné a spustí se Express VI pro vykonání pohybu po kontuře. Během vykonávání pohybu je běh ostatního kódu blokován a není tedy možné pohyb přerušit. Po dokončení pohybu je vyveden chybový cluster na výstup case a event struktury a v řídicí smyčce je zkontrolována bezchybovost průběhu. V případě chyby nebo vnějšího příkazu pomocí lokální proměnné se řídicí smyčka ukončí. Obdobným způsobem jsou řešeny všechny ostatní události event struktury a pro jejich podobnost zde nebudou všechny představeny.



Obr. 10 - Ukázka řídicí smyčky s použitím synchronních Express VI.

5.1.2. Express VI / Funkční bloky – asynchronní

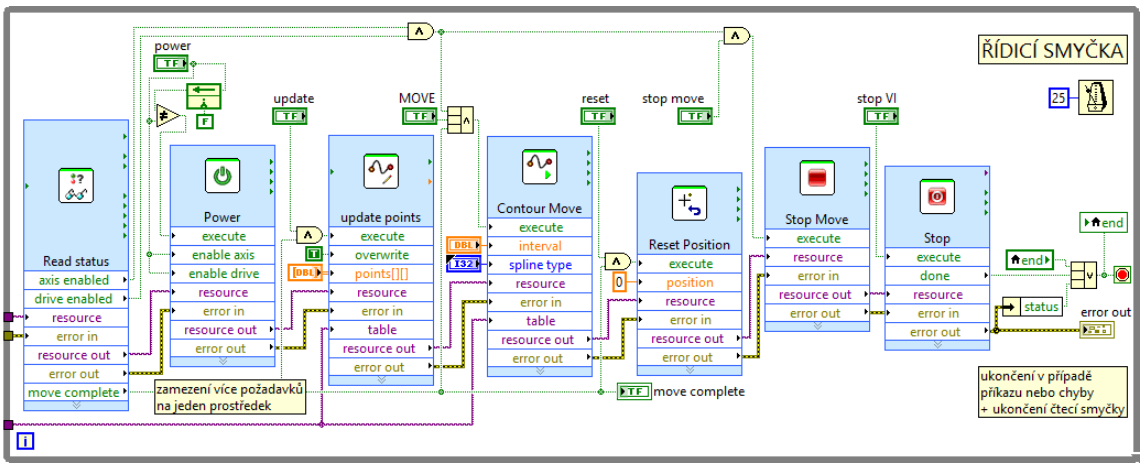
Jak funkční bloky, tak i Express VI vždy disponují vstupem execute (vykonat) a výstupy oznamující stav daného úkonu (aktivní, přerušeno, dokončeno, zaneprázdněn). Od těchto vstupů a výstupů a skutečnosti, že vykonání funkce daného bloku je po spuštění nezávislé na běhu volajícího programu, se odvíjí i způsob použití bloků.

Při používání asynchronních bloků je nutné kontrolovat stav vykonání dané funkce, abychom zamezili spuštění dvou funkcí pracujících se stejnými prostředky, což by vedlo ke kolizi. Z tohoto důvodu je nezbytné použít bloky ve smyčce while tak, aby byly stavy funkcí neustále aktualizovány.

Pokud je potřeba použít podmíněné struktury case, nebo event, je nezbytné brát v potaz, že pokud je podmínka splněna, funkce se sice spustí, ale její stav dokončení již nebude detekován. V takovém případě je potřeba zařadit blok čtení stavu daného prostředku mimo podmíněnou strukturu a zajistit, aby se neustále vykonával (například využitím časového vypršení, neboli timeoutu u event struktury). Výstupy z tohoto čtecího bloku je pak možné řídit (ne)spustitelnost bloků, které by způsobili kolizi.

Příklad

Na ukázce (Obr. 11) lze vidět sériové uspořádání všech asynchronních Express VI do jedné while smyčky. V tomto případě se při každé iteraci aktualizují všechny vstupy a výstupy. Pomocí logických funkcí je ošetřena (ne)spustitelnost bloků, které by způsobily kolizi, v závislosti na dokončení spuštěných funkcí.



Obr. 11 - Ukázka řídicí smyčky s použitím asynchronních Express VI.

5.1.3. Řízení více os

Pro návrh řízení více os je vhodné využít coordinate space prostředek, který sloučí více os do jednoho prostředku. To se provede pomocí volby New > SoftMotion Coordinate Space… V nově otevřeném okně se ze seznamu dostupných os vyberou ty, které budou zahrnuty do coordinate space. Tabulka pro uchování poloh jednotlivých os musí být nakonfigurována pro daný počet os.

Programování samotného řízení téměř stejné, jako bylo popsáno v přechozích kapitolách. Jelikož jsou všechny funkční bloky i Express VI polymorfní, dojde při připojení coordinate space k jejich přepnutí a zobrazí se adekvátní vstupy a výstupy, které jsou ve většině případů stejné.

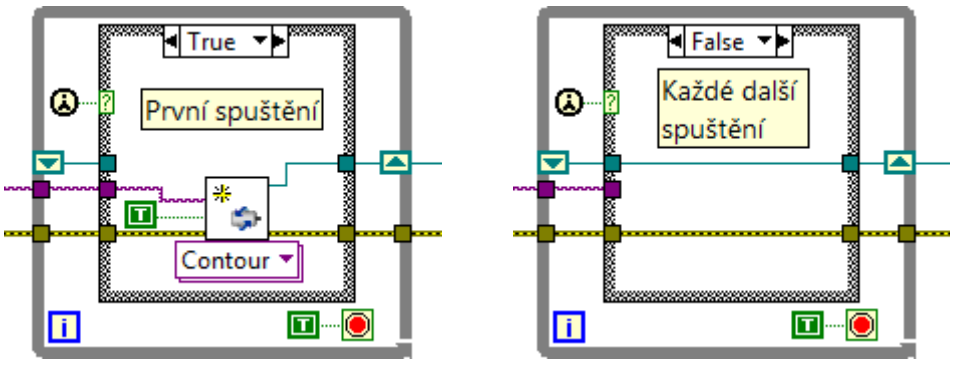
5.2. Programování pomocí SoftMotion objektů

Při využívání SoftMotion objektů jimi nahrazujeme funkci Express VI nebo funkčních bloků. Jedná se o nižší úroveň programování (blíž k low-level, ne jednodušší) a tudíž tato varianta umožňuje větší kontrolu nad vykonávaným programem.

Základní princip práce s objekty je již zmíněn a vyobrazen (Obr. 5) v kapitole 5.3, proto v této kapitole bude pouze ukázka možného použití.

Vytváření objektů

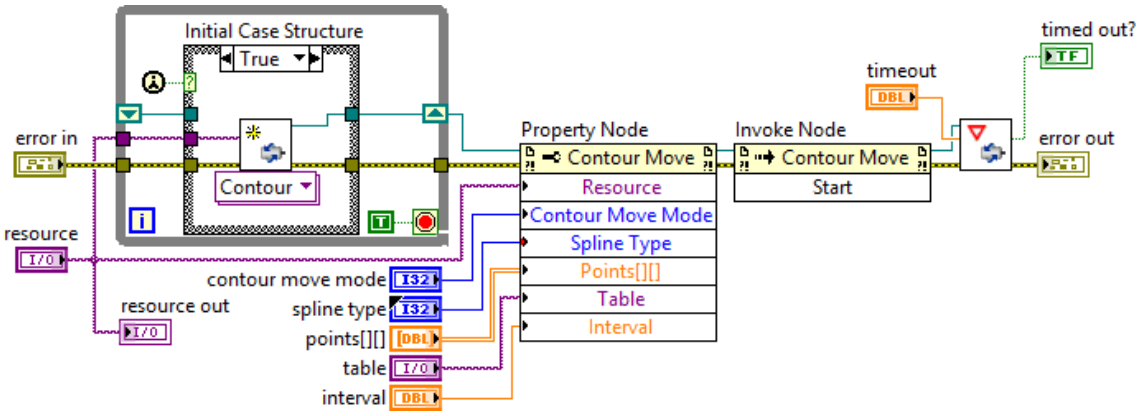
Při vytváření subVI vytvářejícího SoftMotion objekty (je použita funkce create) (Obr. 12), které se bude opakovaně spouštět, je vhodné použít konstrukci využívající shift registru pro uchování proměnné mezi iteracemi. Tato konstrukce je složená ze smyčky while, která proběhne vždy pouze jednou, a vnořené case struktury, která proběhne pouze při prvním spuštění. Při prvním spuštění se uvnitř case struktury vytvoří požadovaný objekt, který je následně přiveden na shift registr, který jej uchová pro každé další spuštění. Tento způsob je vhodnější, než při každém spuštění vytvářet nový objekt a následné jej smazat. Pro vytvoření asynchronního subVI je vytvoření této konstrukce nezbytné.



Obr. 12 - Konstrukce využívající shift registru pro uchování proměnné.

Synchronní subVI

Synchronní subVI (Obr. 13) lze vytvořit relativně snadno, použije se vytvoření objektu, pomocí property node se nastaví vlastnosti, spustí se požadovaná metoda a pomocí bloku wait se počká na její dokončení.



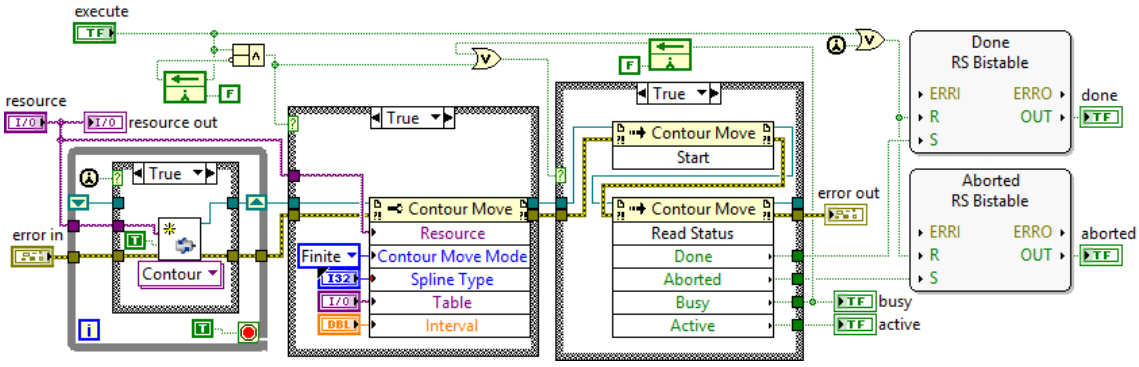
Obr. 13 - SubVI vytvářející synchronní SoftMotion objekt pro vykonání pohybu po kontuře s použitím shift registru.

Asynchronní subVI

U vytváření asynchronních subVI je možností více a odvíjí se od požadavku, zda je potřeba, aby dané subVI vracelo nadřazenému VI o dokončení procesu.

Pokud není kladen zmíněný požadavek, je možné vytvořit totožné subVI podobné synchronnímu, s tím rozdílem, že při vytváření objektu bude přivedena hodnota true na vstup „async?“ a na konci sekvence nebude zařazen blok pro čekání na dokončení operace. Stav dokončení operace je pak možné kdekoli v programu kontrolovat pouze pomocí invoke node (metoda read), kde bude na vstup připojen daný motion resource.

V případě požadované signalizace dokončení procesu přímo od subVI je nezbytné, aby kontrola stavu byla spouštěna uvnitř subVI i v případě false hodnoty na vstupu execute. Toho se docílí podmíněním čtení pomocí hodnot výstupů busy nebo active z předchozí iterace. Jelikož výstupy done a aborted nezůstávají na hodnotě true dlouhodobě, ale jen v jedné iteraci, je vhodné použít ještě bistabilní RS relé, které hodnotu uchová do příštího spuštění execute.



Obr. 14 - SubVI využívající asynchronní SoftMotion objekt, plně zastupující funkce Express VI.

6. Realizace řízení hardwaru

Tato kapitola pojednává o použití konkrétního hardwaru. Oddělení této kapitoly vyplívá z podstaty použití modulu SoftMotion, tedy že programové řízení je navrženo pro „motion resource“ osu, ke které se pomocí „Axis manager“ přiřadí hardware nebo zůstane simulovaná.

Jelikož konfigurace hardwaru není předmětem této práce, je zde uvedena jen obecně.

6.1. cRIO-9024

Pro funkcí master zařízení v síti EtherCat byl použit real-time kontrolér od společnosti National Instruments s označením cRIO-9024 (CompactRIO).

Kontrolér disponuje procesorem společnosti Freescale s frekvencí 800 MHz, 512 MB operační paměti DDR2 a 4 GB úložištěm. Konektivita kontroléru je řešena sériovým portem RS-232 pro připojení periferií, USB portem a dvěma porty pro Ethernet, které se mohou využít pro síťovou komunikaci.

Pro konfiguraci kontrolérů je nutné jej připojit pomocí Ethernet portu 1 přímo k počítači, nebo do stejné sítě. Konfigurace se pak provádí pomocí softwaru NI MAX a spočívá v nastavení IP adresy, pojmenování zařízení a instalaci softwaru.

6.2. Pohon

V řešení byl použit AC synchronní servo motor značky Kollmorgen, řady AKM. Z označení AKM31C-ANC2R-00 lze pomocí manuálu vyčíst kromě určitých rozměrů i typ vinutí, přítomnost 24V brzdy a použití zpětné vazby v podobě resolveru.

Pro řízení tohoto motoru byla použita řídicí jednotka AKD od společnosti Kollmorgen. Řídicí jednotka v tomto případě zastává funkci frekvenčního měniče, regulátoru a zároveň slave zařízení pro síť EtherCAT.

Jelikož řídicí jednotku je možné použít k řízení více typů motorů, je nutné ji nakonfigurovat pro použitý motor. Ke konfiguraci je nejvhodnější použít software Kollmorgen Workbench, který je pro tento účel vyvinut. Řídicí jednotku je potřeba připojit dle návodu k napájení a s počítačem propojit pomocí servisního konektoru. Následně po spuštění softwaru je již řídicí jednotka detekována a je možné provést konfiguraci, spočívající v nastavení parametrů motoru, zpětné vazby, limitů a parametrů regulace. Konfiguraci je možné částečně zjednodušit zadáním označení připojeného motoru, kdy se základní parametry nastaví automaticky.

6.3. Realizace řízení hardwaru v LabVIEW

Po připojení všech zařízení je prvním krokem přidání kontroléru CompactRIO do projektu LabVIEW. Přidání se provede pomocí kontextové nabídky New > Target and Devices…, kde se v seznamu vybere požadovaný kontrolér.

Následuje přidání řídicí jednotky AKD. Pro identifikaci v síti se načte konfigurační .xml soubor na EtherCAT master zařízení, tedy CompactRIO. V kontextové nabídce se zvolí Utilities > Import Device Prifiles…, kde se v nově otevřeném okně zvolí konfigurační soubor dodávaný výrobcem. Dalším krokem je přidání řídicí jednotky do projektu, provedené opět pomocí kontextové nabídky New > Target and Devices…

V posledním kroku se přiřadí AKD řídicí jednotka k SoftMotion ose pomocí Axis Manageru a volby Change Binding.

7. Závěr

Práce se zabývá možnostmi řízení elektrických pohonů s řídicími jednotkami s rozhraním EtherCAT v prostředí NI LabVIEW. Softwarové řízení, kterému je věnována tato práce, je realizováno prostřednictvím modulu SoftMotion, který rozšiřuje možnosti vývojového prostředí NI LabVIEW o nové funkce a nabídky. Před samotným návrhem řízení je uveden popis konceptu, nových funkcí a možností konfigurace. Zpracování návrhu řízení je rozděleno do tří částí, popisujících použití rozdílných funkcí a přístupů. Funkčnost vytvořených programů je demonstrována na simulovaných osách. Závěrem práce je uvedena implementace hardwaru, tedy nahrazení simulovaných os reálnými pohony.

Během práce se především podařilo porozumět principu využívání modulu SoftMotion, a zformulovat jeho popis. V souvislosti s tím byly navrženy dva funkční programy řídící jeden pohon a jeden program pro synchronizované řízení tří pohonů, použitelný například pro souřadnicový stroj. Všechny programy byli navrhnuty s pomocí simulovaných os, přičemž jeden z nich byl použit i pro řízení konkrétního hardwaru.

Další pokračování práce spatřuji v analýze nejnižší úrovně modulu SoftMotion, s ohledem na možnost realizace modulů pro podporu řídicích jednotek dalších výrobců, kteří nejsou zatím podporovány.

Použité informační zdroje

[1] **Beckhoff.** Information System. [Online] 2014. infosys.beckhoff.com.

[2] **Marshall, Perry S a Rinaldi, John S.** *Industrial ethernet: how to plan, install, and maintain TCP/IP ethernet networks : the basic reference guide for automation and process control engineers.* 2nd ed. Research Triangle Park, NC : ISA, 2005. stránky xi, 129 p. ISBN 15-561-7892-1.

[3] National Instruments. 2014. *LabVIEW 2014 Help* [online]. [cit. 2015-05-28]. Dostupné z: http://zone.ni.com/reference/en-XX/help/371361L-01/

[4] National Instruments. 2010. *NI SoftMotion Module Help* [online]. [cit. 2015-05-28]. Dostupné z: zone.ni.com/reference/en-XX/help/371093G-01/

Seznam obrázků

Seznam příloh

1. CD obsahující elektronickou verzi práce a vypracované programy.