

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 RAMENE MOTORU A URČENÍ HMOTNOSTI PŘEDMĚTU Z DYNAMIKY POHYBU

MOTOR ARM CONTROL AND OBJECT WEIGHT DETERMINATION FROM DYNAMIC MOTION

DIPLOMOVÁ PRÁCE
DIPLOMA THESIS

AUTOR PRÁCE
AUTHOR

RADEK POUR

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. ZDENĚK NĚMEC, CSc.

BRNO 2008

Zadání závěrečné práce

Název práce: Řízení ramene motoru a určení hmotnosti předmětu z dynamiky pohybu

Název práce anglicky: Motor arm control and object weight determination from dynamic motion

Vedoucí: Ing. Zdeněk Němec, CSc.

Ústav: Ústav automatizace a informatiky

Typ práce: diplomová práce

Cíle, kterých má být dosaženo:

1. Podrobně zpracovat metodu vyhodnocení hmotnosti ze zaznamenaných časových průběhů proudu motoru a zrychlení otáčivého pohybu.
2. Vyhodnocování hmotnosti realizovat na daném vyvíjeném zařízení pro vážení a třídění.
3. Pro řízení pohybu ramene motoru navrhnout a ověřit programové vybavení, určené k programovatelnému automatu fy Phoenix Contact.
4. Návrh řízení pohybu i vyhodnocování hmotnosti koncipovat tak, aby zařízení kooperovalo s navazující částí zařízení pro třídění zboží.

Charakteristika problematiky úkolu:

Úkol řeší polohové řízení pohybu ramene na ose stejnosměrného motoru při přemisťování zboží. Dynamika pohybu je následně vyhodnocena a vypočtena hmotnost přemisťovaného předmětu.

Základní literární prameny:

Šmejkal, L.-Martinásková, M. PLC a automatizace. Praha: BEN, 1999.

Firemní materiály o programovatelných automatech fy Phoenix Contact.

Němec, Z.: Vyhodnocování hmotnosti předmětu z dynamiky pohybu otáčivého ramene.

Návrh metody a výsledky předběžných zkoušek. Pracovní materiál FSI VUT v Brně ze dne 3.3.2007.

Abstrakt

Tématem projektu, jehož součástí je tato diplomová práce, je inteligentní vážení a třídění zboží. V první části je podrobně rozebrán princip zjišťování hmotnosti z dynamiky pohybu s odvozením výpočtových vztahů a popis použitých prostředků. Druhá část řeší návrh tří zpětnovazebních regulačních obvodů pro řízení ramene stejnosměrného motoru, principy komunikace mezi programovatelnými automaty a navržený kód programu.

Diplomová práce vznikla při řešení výzkumného záměru Inteligentní systémy v automatizaci podporovaného MŠMT ČR pod registračním číslem MSM 0021630529.

Abstract

The diploma thesis is a part of the project Intelligent weighting and sorting of goods. The principle of measuring of weight from movement dynamics with derivation calculation relation is detailed described in the first part. The description of used automation means is also in the first part of the diploma thesis. The second part includes design of three adaptive control systems for controlling DC electromotor's arm, communication principles between programmable logic controllers and designed code of program.

This thesis has been supported by the Czech Ministry of Education in the frame of MSM 0021630529 Research Intention Intelligent Systems in Automation.

Klíčová slova

programovatelný automat, PLC, inteligentní vážení, regulace, regulátor, Phoenix Contact

Keywords

programmable logic controller, PLC, intelligent weighting, regulation, controller, Phoenix Contact

Prohlášení

Prohlašuji, že jsem diplomovou práci *Řízení ramene motoru a určení hmotnosti předmětu z dynamiky pohybu* vypracoval samostatně pod vedením Ing. Zdeňka Němce, CSc. s využitím materiálů uvedených v seznamu použité literatury.

Dne 18.5.2008

Radek Pour

Poděkování

Tímto bych chtěl poděkovat vedoucímu mé diplomové práce Ing. Zdeňku Němcovi, CSc. za odborné vedení, poskytnutí potřebných materiálů, prostředků a především cenných rad a zkušeností, což výrazně pomohlo ke zdárnému vyřešení diplomové práce.

Radek Pour

Obsah

Zadání závěrečné práce.....	3
Abstrakt.....	5
Prohlášení.....	7
1 Úvod.....	11
2 Rozbor problému.....	13
2.1 Princip vyhodnocování hmotnosti	15
2.2 Odvození výpočtu.....	16
2.3 Výsledné vztahy	17
3 Použité automatizační prostředky	19
3.1 Programovatelný automat ILC 350ETH.....	19
3.1.1 Struktura ILC 350ETH.....	20
3.1.2 Význam signalizačních LED	21
3.1.3 Pracovní módy ILC 350ETH	22
3.1.4 Vnitřní schéma ILC 350ETH	22
3.1.5 Napájení ILC 350ETH	22
3.1.6 Digitální vstupy/výstupy	23
3.1.7 Základní parametry automatu ILC 350ETH	24
3.2 Servozesilovač IB IL DC AR 48/10A	25
3.2.1 Význam signalizačních LED	26
3.2.2 Připojení servozesilovače.....	27
3.2.3 Struktura servozesilovače	27
3.2.4 Základní parametry	28
3.3 Elektromotor.....	29
3.4 Absolutní snímač polohy CE-58-S P	30
3.4.1 Technické parametry	30
3.4.2 Označení vstupů/výstupů	30
4 Použité programové prostředky	31
4.1 PC WorX	31
4.1.1 Základní prvky PC WorX	31
4.1.2 Druhy kódů	33
4.1.3 Druhy spouštěných úloh.....	34
4.1.4 Datové typy	34
5 Navržené řešení	35
5.1 Struktura programu – vývojové diagramy	35
5.1.1 Projekt vážení a třídění zboží.....	35
5.1.2 Hlavní program	36
5.1.3 Inicializace konstant.....	37
5.1.4 Výpočet hmotnosti	37
5.1.5 Regulátor polohy	38
5.1.6 Regulátor rychlosti	39
5.1.7 Vážení	40
5.2 Návrh regulačního obvodu	40
5.2.1 Regulace a regulační obvod	40
5.2.2 Regulační obvod pro řízení DC motoru	40
5.2.3 Návrh regulátoru proudu.....	41
5.2.4 Návrh regulátoru rychlosti	43
5.2.4.1 Výpočet přepínacích úrovní regulátoru rychlosti	45

5.2.5	Návrh regulátoru polohy.....	46
5.2.5.1	Algoritmus výpočtu akční veličiny regulátoru polohy.....	48
5.3	Důležité úseky zdrojového kódu.....	49
5.3.1	PCP komunikace	49
5.3.1.1	PCP_CONNECT.....	49
5.3.1.2	PCP_WRITE	49
5.3.1.3	PCP_READ.....	50
5.3.2	Spuštění servozesilovače	50
5.3.3	Ethernetová komunikace	51
5.3.3.1	IP_CONNECT	52
5.3.3.2	IP_URCV	52
5.3.3.3	IP_USEND	52
5.3.4	Regulátor polohy	52
5.3.5	Regulátor rychlosti	53
5.3.6	Výpočet hmotnosti.....	55
5.4	Poznatky z řešení	56
6	Závěr	59
	Seznam použité literatury.....	61
	Seznam příloh	63

1 Úvod

Automatizace je označení pro používání řídicích prostředků, jako jsou programovatelné automaty a regulátory pro řízení průmyslových zařízení a procesů. Nasazování těchto systémů je dalším krokem, následujícím po mechanizaci, která lidem poskytla nástroje usnadňující jim práci. Automatizace snižuje potřebu účasti člověka na řízení a hlídání vykonávané činnosti.

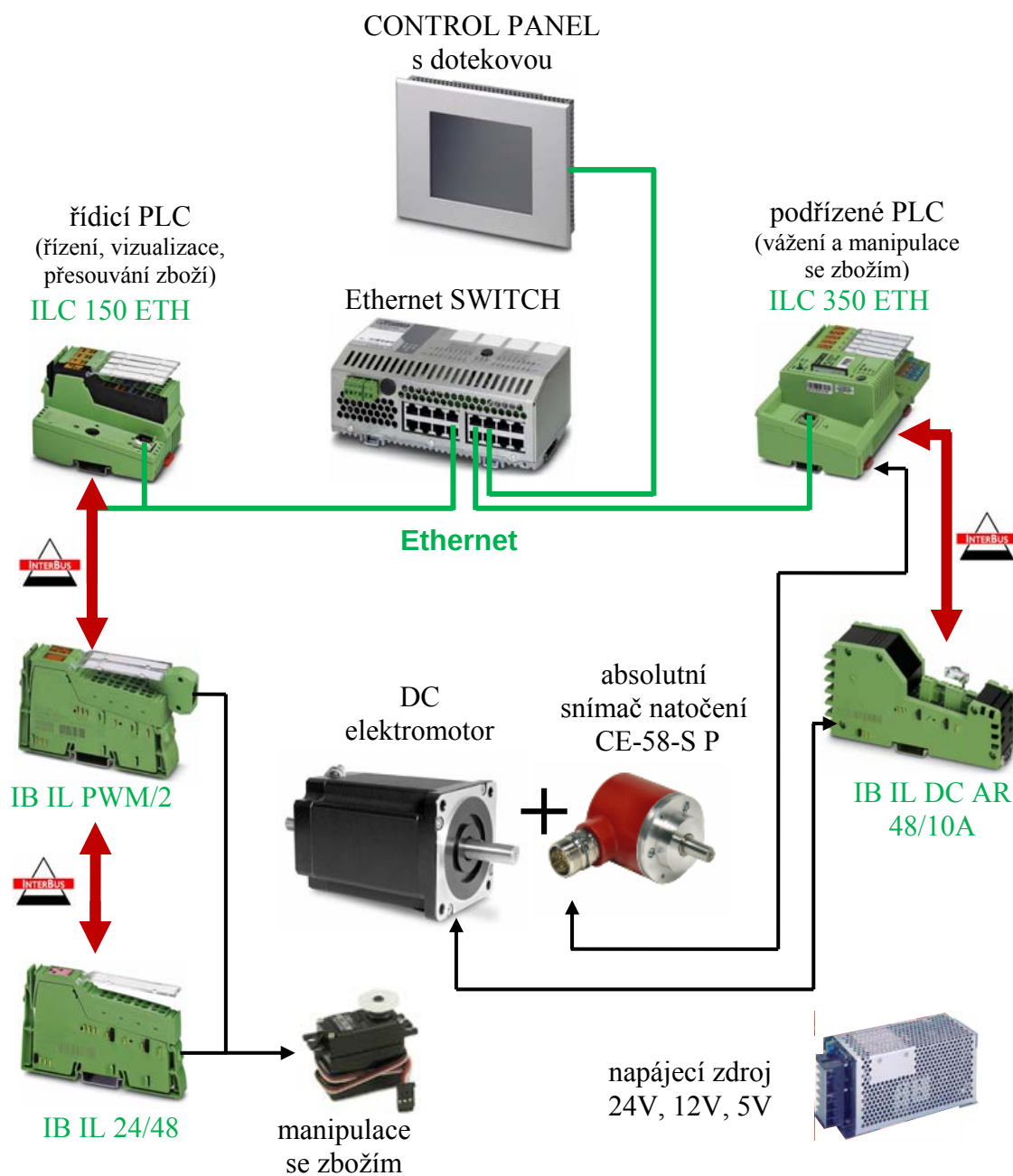
Před pár desítkami let jsme v provozech továren našli automaticky pracující linky jen velmi zřídka, veškerá rozhodovací a ovládací činnost náležela obsluze. V dnešní době asi stěží budeme hledat továrnu, kde není využito automatického řízení.

Dnes je automatizace nedílnou součástí života lidí. Bez ní se neobejde žádný výrobní ani technologický proces, najdeme ji v obchodních i komerčních centrech a stále více se uplatňuje při řízení moderních objektů a rodinných domů.

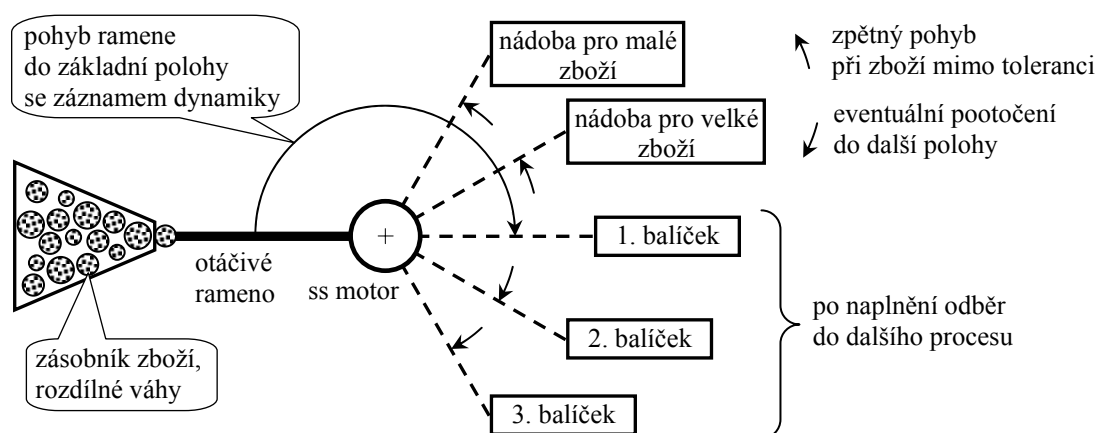
Problematika zjišťování hmotnosti se vyskytuje v mnoha aplikacích, které hrají v lidském životě důležitou roli. Kolikrát si ani neuvědomujeme, kde všude dochází k vážení předmětů. Příkladem zjišťování hmotnosti může být kontrola přetížení výtahu či ramene jeřábu, zatížení nosníků výrobních linek, silniční váhy pro nákladní automobily. Ani ve zdravotnictví se bez vážení neobejdeme, zde magistry v lékárnách odvažují na laboratorních vahách složky léků, dokonce i pro zjišťování obsahů některých složek krve se využívá principu vážení. Na konec nelze zapomenout na navažování zboží v průmyslu, a to nejen potravinářském, ale i strojírenském a dalších. Nedochozí vždy k vážení jen finálních produktů, ale například před začátkem mísení je mnohdy třeba znát hmotnosti jednotlivých složek, aby byly dodrženy technologické postupy a daná složení vyráběných směsí.

Je vidět, že bez toho, abychom znali hmotnost věcí, které se kolem nás vyskytují, se zřejmě v dnešní době neobejdeme, stejně jako se neobejdeme bez automaticky pracujících strojů a systémů. Vždyť dnes se v téměř každém elektronickém zařízení objevuje mikroprocesor, který nám usnadňuje ovládání a hlídání důležitých veličin.

Tato diplomová práce je součástí projektu, který řeší problematiku inteligentního vážení a následného třídění zboží za pomoci stejnosměrného elektromotoru. Projekt je rozdělen do dvou na sebe navazujících částí, přičemž každá je řešena jedním diplomantem. Cílem této diplomové práce je návrh a realizace vyhodnocení hmotnosti a řízení pohybu stejnosměrného motoru pomocí programového a hardwarového vybavení od firmy Phoenix Contact. Projekt se v roce 2008 účastnil mezinárodní soutěže pořádané společností Phoenix Contact, kde reprezentoval Ústav automatizace a informatiky, FSI VUT v Brně.



Obr. 2-2 Blokové schéma projektu [13]

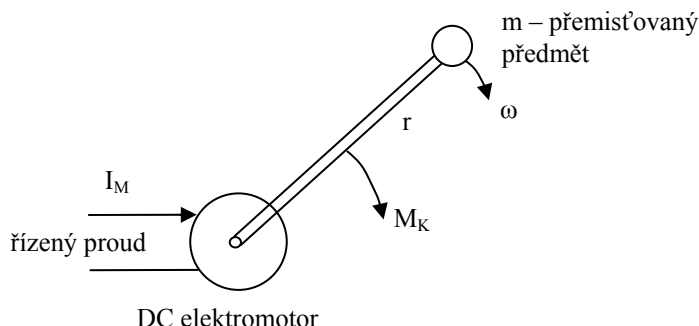


Obr. 2-3 Polohy otáčivého ramene při třídění (pohled shora)

2.1 Princip vyhodnocování hmotnosti [1]

K určení hmotnosti m přemisťovaného předmětu vyhodnotíme závislost úhlové rychlosti otáčení ω na proudu motoru I_M z časového úseku rozběhu nebo zastavování pohybu.

Při výpočtu vycházíme z rovnosti impulsu síly a hybnosti hmoty. Pro rotační pohyb tedy platí $M_K \cdot \Delta t = J \cdot \Delta \omega$.



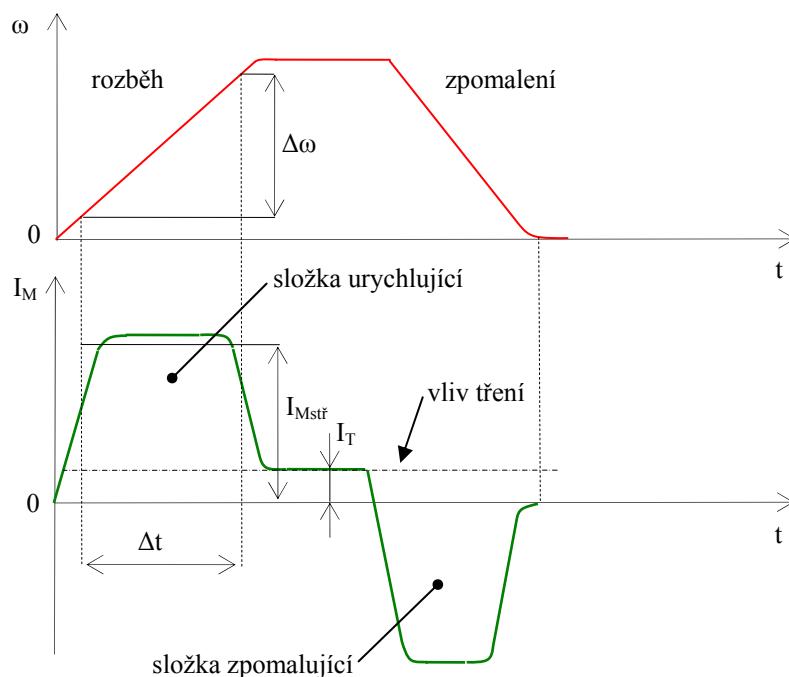
Obr. 2-4 Náčres ramene

Časový interval Δt k vyhodnocování volíme z takového úseku rozběhu či doběhu, ve kterém je co největší změna úhlové rychlosti $\Delta \omega$ a rychlost se mění monotónně. V takto zvoleném úseku pak z měřeného proudu $I_M(t)$ určíme jeho střední hodnotu $I_{Mstř}$.

M_K – mechanický krouticí moment, urychlující (zpomalující) rotační pohyb ramene

M_E – krouticí moment elektrický uvnitř motoru; pro daný účel je nejvhodnější stejnosměrný elektromotor, u kterého je krouticí moment přímo úměrný proudu $M_E = K_M \cdot I_M$

M_T – krouticí moment tření, související se ztrátami (tření v ložiskách, v komutátoru, v převodovce); předpokládáme nezávislost na rychlosti pohybu



Obr. 2-5 Průběh proudu a rychlosti

2.2 Odvození výpočtu

Výchozí rovnice $M_K \cdot \Delta t = J \cdot \Delta \omega,$ (2.1)

kde moment setrvačnosti $J = m \cdot r^2.$ (2.2)

Po dosazení (2.2) do (2.1) dostáváme $M_K \cdot \Delta t = m \cdot r^2 \cdot \Delta \omega.$ (2.3)

Urychlovací moment $M_K = M_E - M_T,$ (2.4)

kde $M_E = K_m \cdot I_M,$ (2.5)

$M_T = K_T \cdot \text{sign}(\omega).$ (2.6)

Po dosazení rovnic (2.4), (2.5) a (2.6) do rovnice (2.3) dostáváme vztah

$$(K_m \cdot I_{Mstř} - K_m \cdot I_T) \cdot \Delta t = m \cdot r^2 \cdot \Delta \omega, \quad (2.7)$$

ze kterého plyne $m = K \cdot (I_{Mstř} - I_T) \cdot \frac{\Delta t}{\Delta \omega},$ (2.8)

kde $K = \frac{K_m}{r^2},$ (2.9)

$$I_{Mstř} = \frac{\int_0^{\Delta t} I_M(t) dt}{\Delta t}. \quad (2.10)$$

2.3 Výsledné vztahy

– pro vyhodnocování z rozběhu je použito vztahu

$$m_R = K \cdot (I_{Mstř} - I_T) \cdot \frac{\Delta t}{\Delta \omega} - m_0 \quad (2.11)$$

– pro vyhodnocování z doběhu je použito vztahu

$$m_D = K \cdot (|I_{Mstř}| + |I_T|) \cdot \frac{\Delta t}{\Delta \omega} - m_0 \quad (2.12)$$

– výsledná hmotnost

$$m = \frac{m_R + m_D}{2} \quad (2.13)$$

- Konstantu K lze stanovit výše uvedeným výpočtem, ale přesnější je stanovit ji cejchováním.
- Složku proudu I_T pro překonání tření je možné stanovit třemi způsoby:
 - a) Jednorázovým změřením při rovnoměrném pohybu ramene. Pak I_T považujeme za konstantní.
 - b) Opakovaným vyhodnocováním v cyklech rozběh-doběh z úseků konstantní úhlové rychlosti.
 - c) Opakovaným vyhodnocováním celého cyklu pomocí modifikovaného výpočtu (srovnání ploch).
- Zbytková hmotnost m_0 shrnuje vliv setrvačnosti ramene a rotoru motoru, přepočítaný na konec ramene. Stanoví se zkouškou, kdy na rameni není umístěn předmět, tedy $m = 0$ g.

3 Použité automatizační prostředky

Programovatelný automat, nebo také PLC či PA, je zařízení určené pro automatické řízení zejména průmyslových aplikací. Jeho základem je mikropočítač, který vykonává logické a aritmetické operace. Celý automat je navržen tak, aby odolával ztíženým pracovním podmínkám, jako jsou vibrace, vyšší teploty, mechanické namáhání. Zároveň musí být jeho spolehlivost na vysoké úrovni, protože často zajišťuje bezpečnostní funkce. Dřívější PLC byly určeny jen pro řešení logických úloh, ale dnes je možné zpracovávat analogové i číslicové údaje.

Abychom automat mohli uvést do provozu, je třeba jej naprogramovat, tj. napsat program, podle kterého bude PLC vykonávat svou činnost. K tomuto účelu slouží programovací přístroj, v dnešní době řešen převážně jako software pro PC.

Existují dvě skupiny PLC, které se liší variabilitou. První skupinu tvoří tzv. kompaktní PLC, jejichž počet vstupů a výstupů je daný výrobcem. Jejich rozšíření je možné, avšak není příliš vhodné. Do další skupiny spadají tzv. stavebnicové PLC, u kterých si uživatel volí počty jednotlivých vstupů, výstupu a dalších jednotek, které lze snadno připojit.

Veškeré technické informace a obrázky uvedené v této kapitole jsou převzaty z literatury [4], [5], [10], [11], [12], [13].

3.1 Programovatelný automat ILC 350ETH

Programovatelný automat ILC 350ETH firmy Phoenix Contact je určen pro široké použití, a to především pro středně rozsáhlé aplikace. Automaty této řady mají výkonný procesor a integrované ethernetové rozhraní, které umožňuje propojení s dalšími automaty a systémy řízení. Pomocí bloků TCP/IP komunikace mohou také komunikovat se zařízeními, která jsou s TCP/IP kompatibilní.

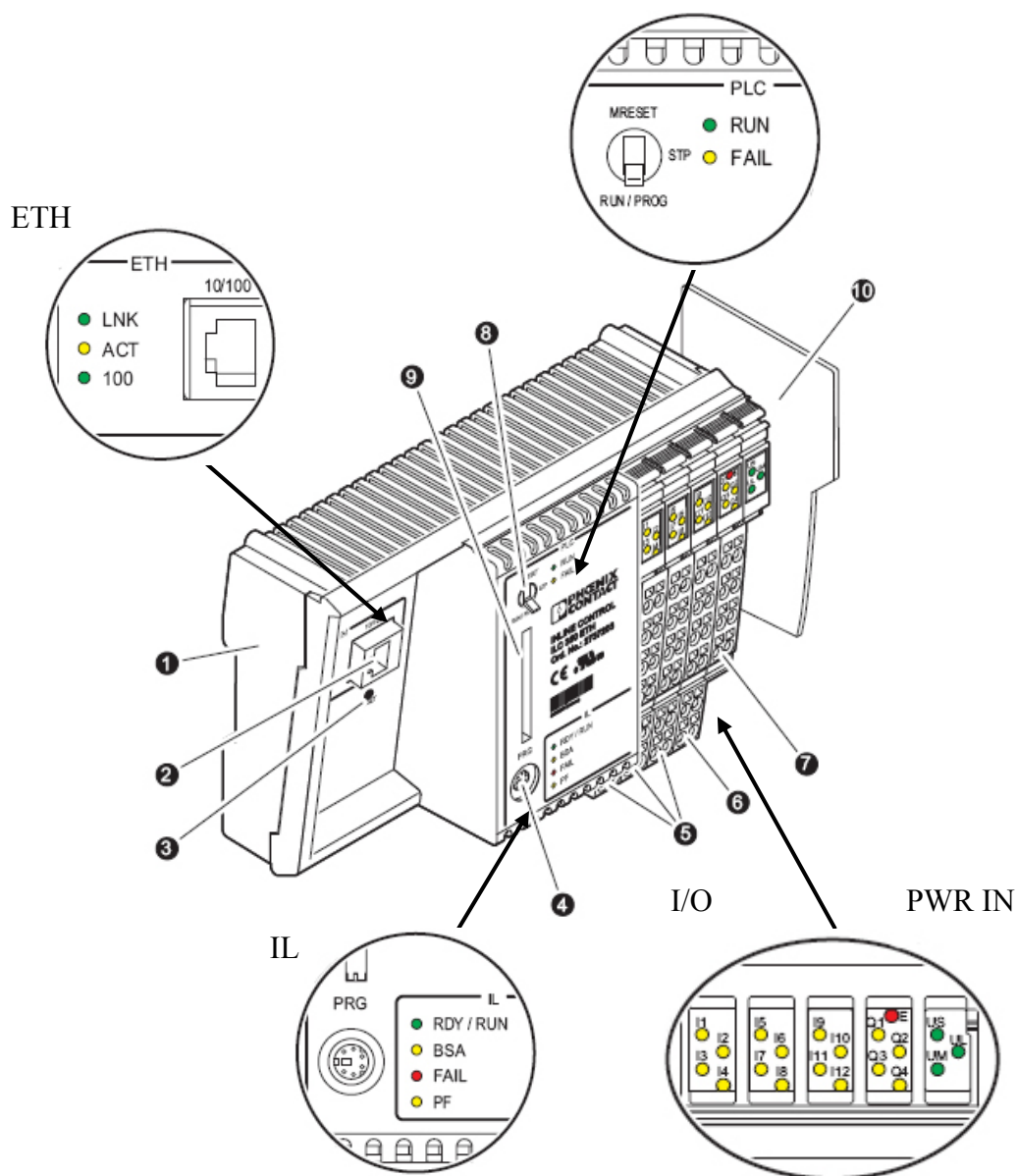


Obr. 3-1 Programovatelný automat ILC 350ETH

ILC 350ETH je řešen jako stavebnicový systém, tudíž je možné k samotnému automatu připojit další rozšiřující bloky podle řešené úlohy. Uvnitř automatu je zřízena sběrnice interbus, na kterou se připojují rozšiřující moduly.

Pro programování automatu se využívá software PC WorX, který bude popsán níže. Pro přenos programu z PC WorX do PLC a taktéž k odladění lze využít sériové linky RS-232, nebo rychlejší Ethernetové komunikace.

3.1.1 Struktura ILC 350ETH



Hlavní části ILC 350ETH:

- 1 základna elektroniky
- 2 konektor pro ethernetové připojení
- 3 tlačítko reset
- 4 konektor pro připojení linky RS-232
- 5 konektory 1 až 3: binární vstupy
- 6 konektor 4: binární výstupy
- 7 konektor 5: svorky napájení
- 8 přepínač módů
- 9 konektor paměťové karty
- 10 ukončovací destička

3.1.2 Význam signalizačních LED

Význam jednotlivých LED na čelním panelu automatu ILC 350ETH (viz obr. 3-2).

Označení	Barva	Význam
ETH : stavy ethernetového rozhraní		
LNK	zelená	spojení navázáno
	ON	PLC schopno navázat spojení
ACT	žlutá	přenos dat
	ON	přenos dat přes ethernetové rozhraní
100	zelená	rychlost přenosu
	ON	100 Mbit/s
	OFF	10 Mbit/s
PLC : stavy programovatelného automatu		
RUN	zelená	automat spuštěn
	OFF	automat není připraven k činnosti
	bliká	úspěšná inicializace, režim READY/STOP, program neprobíhá
	ON	úspěšná inicializace, program probíhá, režim RUN
FAIL	žlutá	chyba
	ON	nastala chyba při běhu programu
	OFF	není chyba
IL : diagnostika sběrnice INTERBUS		
RDY/RUN	zelená	řadič sběrnice INTERBUS připraven
	bliká	řadič sběrnice INTERBUS ve stavu READY nebo ACTIVE
	ON	řadič sběrnice INTERBUS v režimu RUN
BSA	žlutá	segment sběrnice ukončil spojení
	ON	vypnut jeden nebo více segmentů sběrnice
FAIL	červená	chyba na sběrnici
	ON	chyba na připojené sběrnici, nebo chyba automatu
PF	žlutá	chyba periferie
	ON	chyba periferie připojené na sběrnici

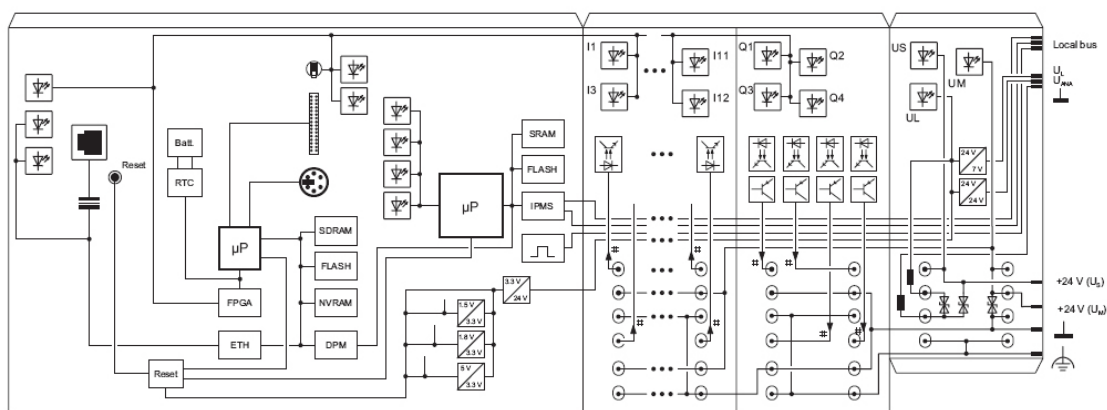
I/O : digitální vstupy a výstupy		
I1 až I12	žlutá	vstup 1 až 12
	ON	logická „1“ na vstupu
E	žlutá	Error
	ON	přetížení na jednom z výstupů 1 až 4
Q1 až Q4	žlutá	výstupy 1 až 4
	ON	logická „1“ na výstupu
PWR IN : napájecí napětí		
US	zelená	24 V pro obvod segmentu
	ON	napětí je přítomno
UM	zelená	24 V hlavní obvod
	ON	napětí je přítomno
UL	zelená	24 V pro generování napětí U_L a U_{ANA}
	ON	napětí je přítomno

3.1.3 Pracovní módy ILC 350ETH

Pracovní módy se přepínají ručně na čelním panelu automatu pomocí třípolohového přepínače (viz obr. 3-2).

Mód	Význam
RUN/PROG	program běží, PC WorX může komunikovat s automatem
STP	program se neprovádí
MRESET	smazání uložených dat a programu

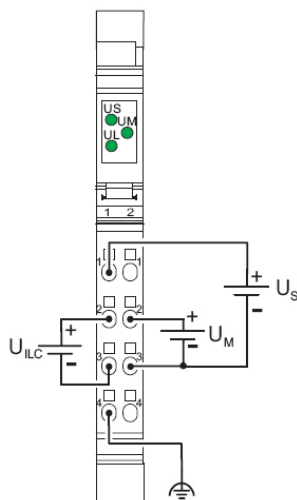
3.1.4 Vnitřní schéma ILC 350ETH



Obr. 3-3 Vnitřní schéma ILC 350ETH

3.1.5 Napájení ILC 350ETH

Automat ILC 350ETH je napájen externím zdrojem o hodnotě výstupního napětí 24 V DC. Připojení napájecího napětí je zřejmé z obr. 3-4.

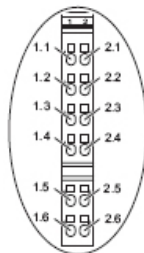


Obr. 3-4 Připojení napájecího napětí

Označení	Velikost napětí	Účel
U_S	24 V DC	napětí segmentu
U_{ILC}	24 V DC	pro generování napětí pro komunikaci U_L a pro zařízení připojená na lokální sběrnici; pro generování napětí pro analogové operace U_{ANA}
U_M	24 V DC	hlavní napájecí napětí

3.1.6 Digitální vstupy/výstupy

Automat ILC 350ETH je vybaven 12 digitálními vstupy a 4 digitálními výstupy. O další vstupy/výstupy lze PLC rozšířit připojením rozšiřujících bloků.



Obr. 3-5 Digitální vstupy/výstupy

Svorka	Určení	Poznámka
Konektor vstupů – konektory 1 až 3		
1.1	I1	vstup 1
2.1	I2	vstup 2
1.2, 2.2	24 V	napájecí napětí U_M pro 2 a 3 žilové připojení
1.3, 2.3	GND	zem pro 3 žilové připojení
1.4	I3	vstup 3
2.4	I4	vstup 4
1.5, 2.5	24 V	napájecí napětí U_M pro 2 a 3 žilové připojení
1.6, 2.6	GND	zem pro 3 žilové připojení

Konektor výstupů – konektor 4		
1.1	O1	výstup 1
2.1	O2	výstup 2
1.2, 2.2	GND	zemní svorka pro 2 a 3 žilové připojení
1.3, 2.3	FE	pracovní zem pro 3 žilové připojení
1.4	O3	výstup 3
2.4	O4	výstup 4
1.5, 2.5	GND	zemní svorka pro 2 a 3 žilové připojení
1.6, 2.6	FE	pracovní zem pro 3 žilové připojení

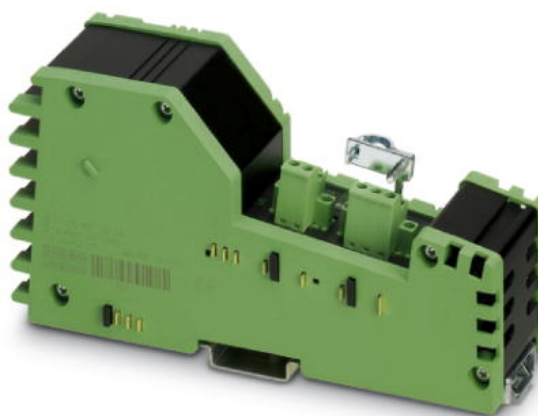
3.1.7 Základní parametry automatu ILC 350ETH

Mechanické parametry	
výška	140,5 mm
šířka	182 mm
hmotnost	440 g
stupeň ochrany	IP20
Komunikační rozhraní	
INTERBUS	
– typ spojení	Inline data jumper
Ethernet 10Base-T/100Base-TX	
– typ spojení	RJ45 female
– rychlost přenosu dat	10/100 Mbit/s
– využití	parametrizace, diagnostika, řízení
RS-232C	
– typ spojení	MINI-DIN female (PS2)
– využití	parametrizace, diagnostika, řízení
Parametry napájení	
proud	250 mA
napájecí napětí	24 V DC
rozsah napájecího napětí	20,4 V DC až 30 V DC
zbytkové kolísání	± 5%
INTERBUS	
zařízení s parametrizačním kanálem	max. 63
připojených zařízení	max. 512
počet I/O	max. 8192
baterie	integrovaná
počet tasků	16
paměť	2 Mbyte
Digitální vstupy	
počet	12
logické úrovně	
– logická „0“	max. 5 V DC
– logická „1“	min. 15 V DC
nominální vstupní napětí	24 V DC
dovolený rozsah	$-0,5 \text{ V} < U_{\text{IN}} < +30 \text{ V DC}$

nominální vstupní proud	7 mA běžně, max. 15 mA
časové filtry	
1. vstupy 1 až 8	
– změna 0 → 1	5 μs
– změna 1 → 0	2 μs
2. vstupy 9 až 12	
– změna 0 → 1	500 μs
– změna 1 → 0	700 μs
dovolená délka přívodu	30 m
Digitální výstupy	
počet	4
nominální výstupní napětí	24 V DC
nominální výstupní proud	500 mA

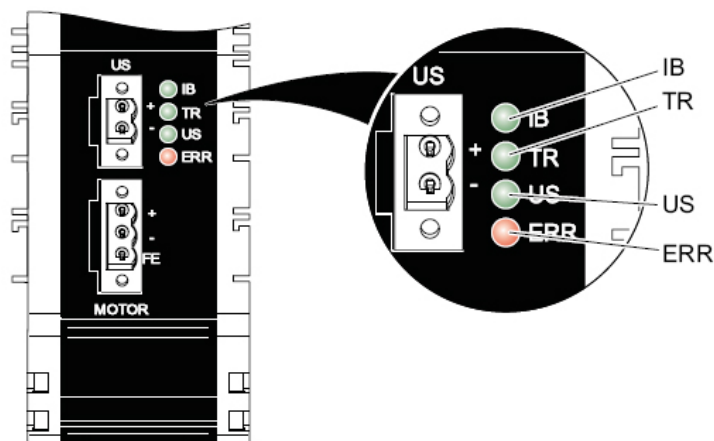
3.2 Servozesilovač IB IL DC AR 48/10A

Servozesilovač IB IL DC AR 48/10A je rozšiřujícím modulem pro stavebnicové automaty společnosti Phoenix Contact, který slouží pro řízení stejnosměrných motorů s napájecím napětím od 12 V DC do 48 V DC a výkonem do 450 W. Maximální proud, který může servozesilovač dodat do motoru, je hardwarově omezen na 10 A.



Obr. 3-6 Servozesilovač IB IL DC AR 48/10A

3.2.1 Význam signalizačních LED

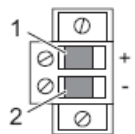


Obr. 3-7 Čelní panel servozesilovače IB IL DC AR 48/10A

Označení	Barva	Význam
IB	zelená	diagnostika
	ON	sběrnice aktivní
	bliká	
	0,5 Hz	komunikační napětí přítomno, neaktivní sběrnice
	2 Hz	komunikační napětí přítomno, sběrnice aktivní, I/O error
	4 Hz	komunikační napětí přítomno, chyba na sběrnici
	OFF	komunikační napětí nepřítomno, neaktivní sběrnice
TR	zelená	PCP aktivní
	ON	PCP zpráva odeslána do servozesilovače
	OFF	nepřenášejí se PCP data
US	zelená	napájecí napětí silové části
	ON	napájecí napětí je větší než 75% nominálního napětí
	OFF	napájecí napětí je menší než 75% nominálního napětí
ERR	červená	Error
	ON	nastala chyba, důvod lze přečíst v „ErrorCode“
	OFF	nenastala chyba

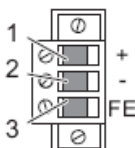
3.2.2 Připojení servozesilovače

d) Připojení napájení silové části



Svorka	Určení
1	$U_S +$
2	$U_S -$

e) Připojení motoru



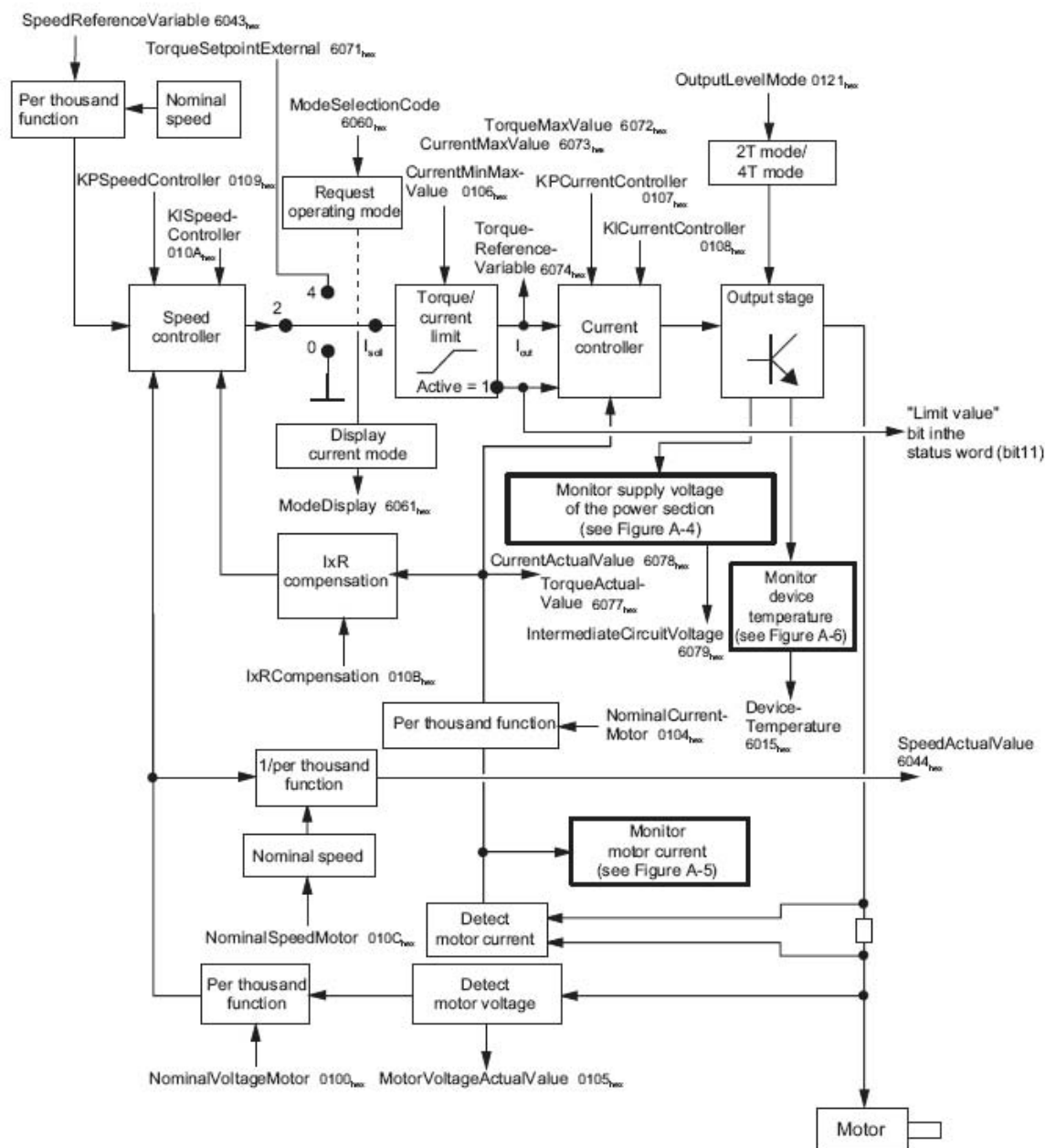
Svorka	Určení
1	Motor +
2	Motor -
3	Pracovní nula (FE)

3.2.3 Struktura servozesilovače

Servozesilovač obsahuje dosti složitou strukturu, kterou je možno jednoduše měnit pomocí PCP komunikace. Jak je patrné z *obr. 3-8*, v servozesilovači jsou realizovány dva zpětnovazební regulační obvody. První je regulátor proudu, druhý je regulátor rychlosti.

Pomocí PCP komunikace lze zvolit režim podle potřeb řešené aplikace. Samostatně může pracovat regulátor proudu, ale nikoliv regulátor rychlosti. PCP komunikace umožňuje nastavit veškeré parametry, jako jsou maximální, nominální hodnoty, velikosti konstant regulátorů, změnu módů výstupní části, číst některé hodnoty jako teplotu, nastavený mód a mnoho dalších.

Důležitou funkcí je, že lze zvolit, kterými parametry se bude motor ovládat a také které hodnoty budou ze servozesilovače čteny pomocí procesní komunikace. Máme možnost zvolit například zadávání požadovaného proudu nebo rychlosti, čtení aktuální hodnoty proudu, rychlosti, napětí nebo dalších. Ale vždy je možno číst a zadávat jen jeden parametr.



Obr. 3-8 Zjednodušená vnitřní struktura servozesilovače

3.2.4 Základní parametry

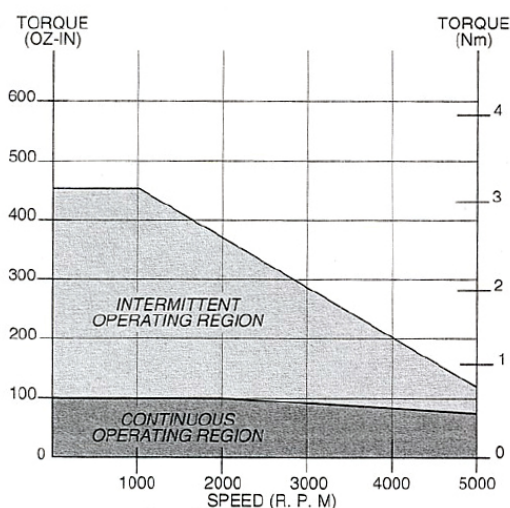
Mechanické parametry	
výška	85 mm
šířka	48 mm
hmotnost	460 g
Přenosová rychlost	
IB IL DC AR 48/10A	500 kbit/s
Napájení silové části	
napájecí napětí	12 V DC až 8 V DC $\pm$ 15%

napájecí proud	0 A až 10 A
vypnutí při překročení napětí	$U_S > 60 \text{ V DC}$
Výstup	
počet	1
zátěž	1 DC elektromotor s permanentními magnety
výstupní napětí	92% vstupního napětí
výstupní proud	max. 10 A
limitovaný proud motoru	0 A až 10 A, nastavení přes sběrnici
brzdění	energie je předána zpět do zdroje (třeba zajistit maření energie)
Časové intervaly spouštění regulátorů	
regulátor rychlosti	1 ms
regulátor proudu/momentu	250 μs

3.3 Elektromotor

Pro řešení je použit stejnosměrný elektromotor s komutátorem a permanentními magnety. Motor je starší výroby, a tak nebylo možné nalézt jeho technické parametry, proto se vychází z parametrů dosti podobného elektromotoru stejného výrobce.

krouticí moment	0,6 Nm
max. napětí	90 V
max. otáčky	5300 min^{-1}
hmotnost	2,7 kg
max. špičkový proud	27 A
max. proud	5,8 A



Obr. 3-9 Závislost krouticího momentu na rychlosti otáčení

3.4 Absolutní snímač polohy CE-58-S P

Absolutní snímač polohy slouží ke zjišťování aktuálního úhlu natočení hřídele elektromotoru vzhledem k předem definované nulové poloze.



Obr. 3-10 Absolutní snímač polohy CE-58-S P

3.4.1 Technické parametry

počet bitů	10
počet impulzů/otáčka	1024
počet otáček	1
napájecí napětí	11 V až 27 V DC
kód výstupu	binární
logické úrovně	
– logická „0“	< 2 V DC
– logická „1“	> 8 V DC
maximální rychlost otáčení	12 000 min ⁻¹
hmotnost	0,3 kg

3.4.2 Označení vstupů/výstupů

Pin	Označení	Účel	Úroveň	Barva
1	O_D0	Data		bílá
2	O_D1	Data		hnědá
3	O_D2	Data		zelená
4	O_D3	Data		žlutá
5	O_D4	Data		šedá
6	O_D5	Data		růžová
7	O_D6	Data		modrá
8	O_D7	Data		červená
9	O_D8	Data		černá
10	O_D9	Data		fialová
11	Direction IN	Směr	Supply V	šedá/růžová
12	I_Latch	Pozdržení	Supply V	červená/modrá
13	Preset1_IN	Přednastavení	Supply V	bílá/zelená
14	Ser.Program+ IN/OUT	Programování	RS 485	hnědá/zelená
15	Ser.Program- IN/OUT	Programování	RS 485	bílá/žlutá
16	Supply Voltage IN	Napájení	11 V až 27 V	žlutá/hnědá
17	Ground IN	Zem	0 V	bílá/šedá

4 Použité programové prostředky

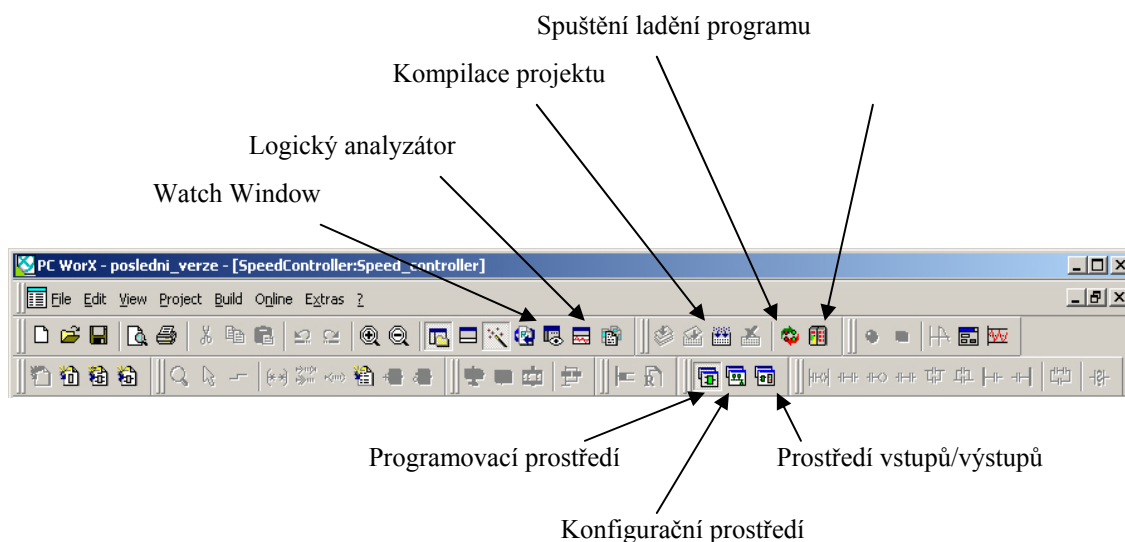
4.1 PC WorX

PC WorX je programovací přístroj vyvíjený společností Phoenix Contact a je určen pro programování PLC tohoto výrobce. Mimo samotné tvorby programů umožňuje i diagnostikovat automat a také testovat správnou funkčnost celého zařízení.

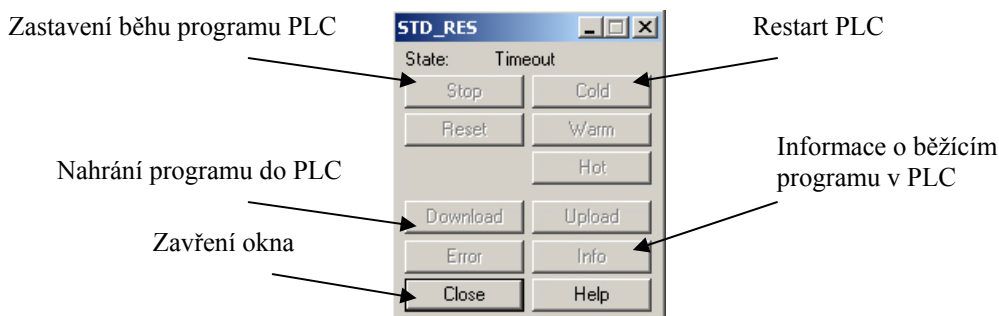
Program je určen pro prostředí Windows a je řešen jako uživatelsky příjemný software s grafickým prostředím. Nabízí širokou škálu možností nastavení, a to jak samotného programovacího prostředí, tak i připojeného automatu, se kterým umožňuje komunikovat přes Ethernet nebo sériovou linku RS-232C.

Programovací přístroj PC WorX umožňuje mimo jiné použití programovacích jazyků, které jsou zahrnuty v normě IEC – 61 131. Mezi grafické patří jazyk kontaktních schémát (LD – Ladder Diagram), jazyk funkčních bloků (FBD – Function Block Diagram), mezi textové jazyk mnemokódů (IL – Instruction List), strukturovaný textový jazyk (ST – Structured Text).

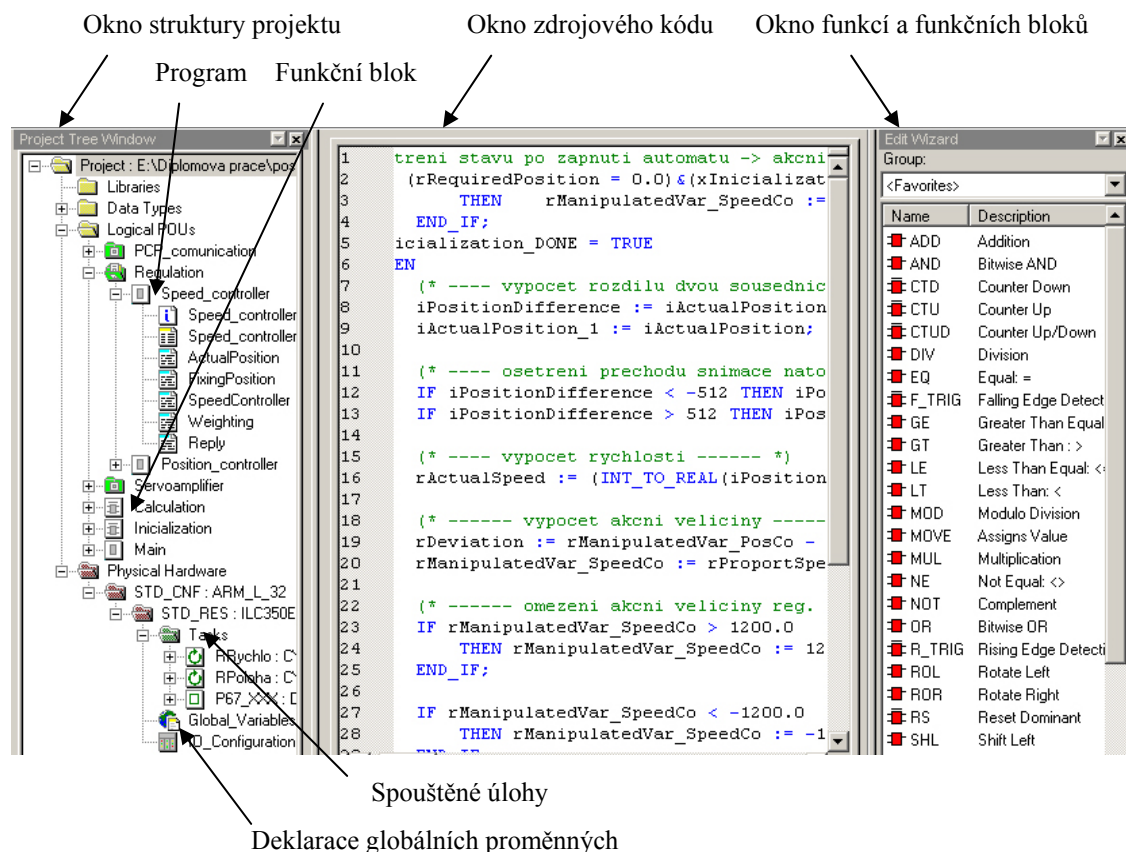
4.1.1 Základní prvky PC WorX



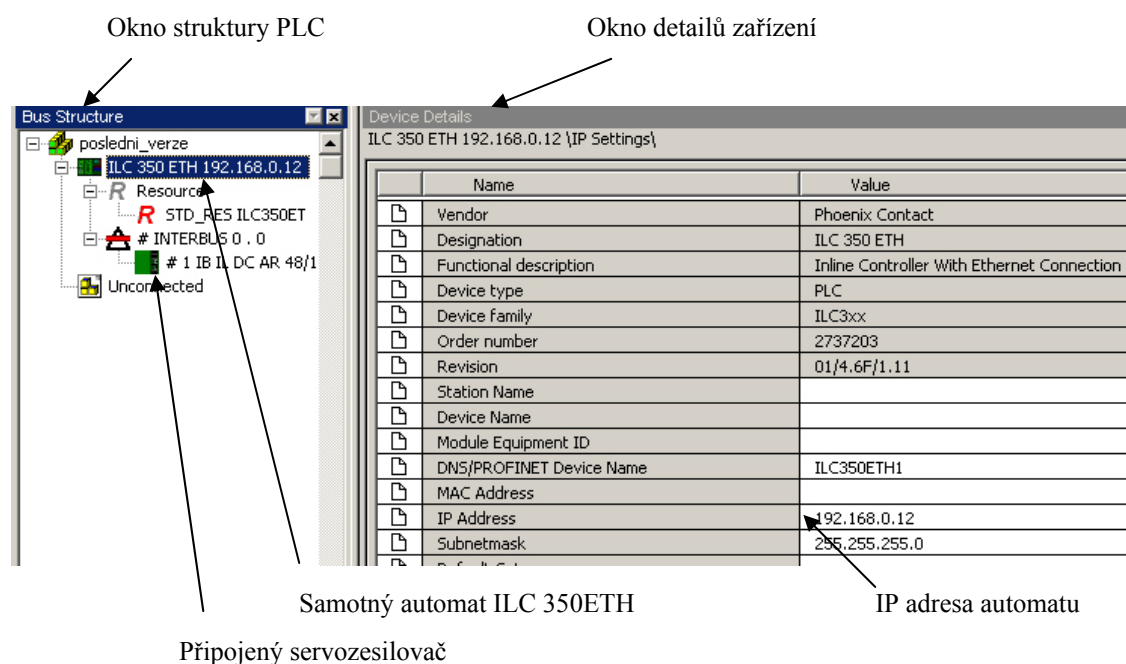
Obr. 4-1 Hlavní panel programu PC WorX



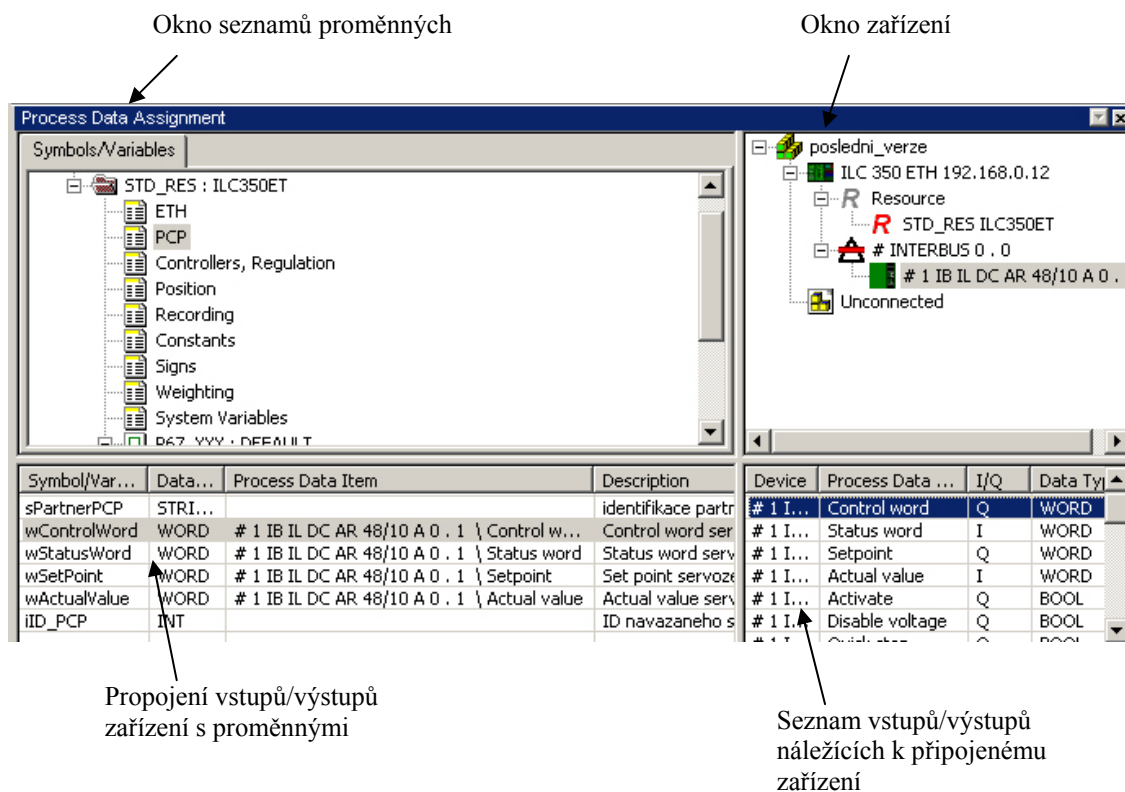
Obr. 4-2 Prostředí pro ovládání PLC



Obr. 4-3 Programovací prostředí



Obr. 4-4 Konfigurační prostředí



Obr. 4-5 Prostředí vstupů/výstupů

4.1.2 Druhy kódů

Ve struktuře projektu se mohou vyskytnout tři různé druhy kódů, přičemž každý má své specifické určení.

- Program – každý program musí být zařazen ve spouštěných úlohách (Tasks), jinak nebude vykonáván. Z programu je možné přistupovat ke vstupům a výstupům automatu a dalších zařízení, která jsou připojena na některou ze sběrnic.
- Funkční blok – program, který nelze umístit do spouštěných úloh, a je ho tedy třeba volat z programu nebo jiného funkčního bloku, přičemž se při volání mohou zadat vstupní hodnoty. Je schopen vracet výstupní hodnoty. Funkční blok je obdobou podprogramu u jiných programovacích prostředí. Pamatuje si poslední hodnoty proměnných. Je možné z něj přistupovat ke vstupům a výstupům automatu a dalších zařízení, která jsou připojena na některou ze sběrnic.
- Funkce – volána z programu nebo podprogramu, je ji možné přiřadit vstupní hodnotu. Po vykonání navrací hodnotu výsledku. Není možné z ní přistupovat ke vstupům a výstupům automatu a dalších zařízení, která jsou připojena na některou ze sběrnic.

4.1.3 Druhy spouštěných úloh

Spouštěné úlohy (Tasks) jsou skupiny programů, které jsou spouštěny různými způsoby. Počet Tasků je omezen na 16, každému je možné nastavit prioritu, podle které bude spouštěn.

- a) Základní (Default) – úloha, ve které je spouštěn hlavní program, má nejnižší prioritu a může být jen jedna.
- b) Časově spouštěné – programy jsou spouštěné s určitou periodou.
- c) Spouštěné událostí – programy jsou spouštěné, pokud nastane předem definovaná situace, např. hodnota na vstupu.
- d) Spouštěné systémovou událostí - programy jsou spouštěné, pokud se systém dostane do konkrétního stavu, např. chyba na sběrnici, restart automatu atd.

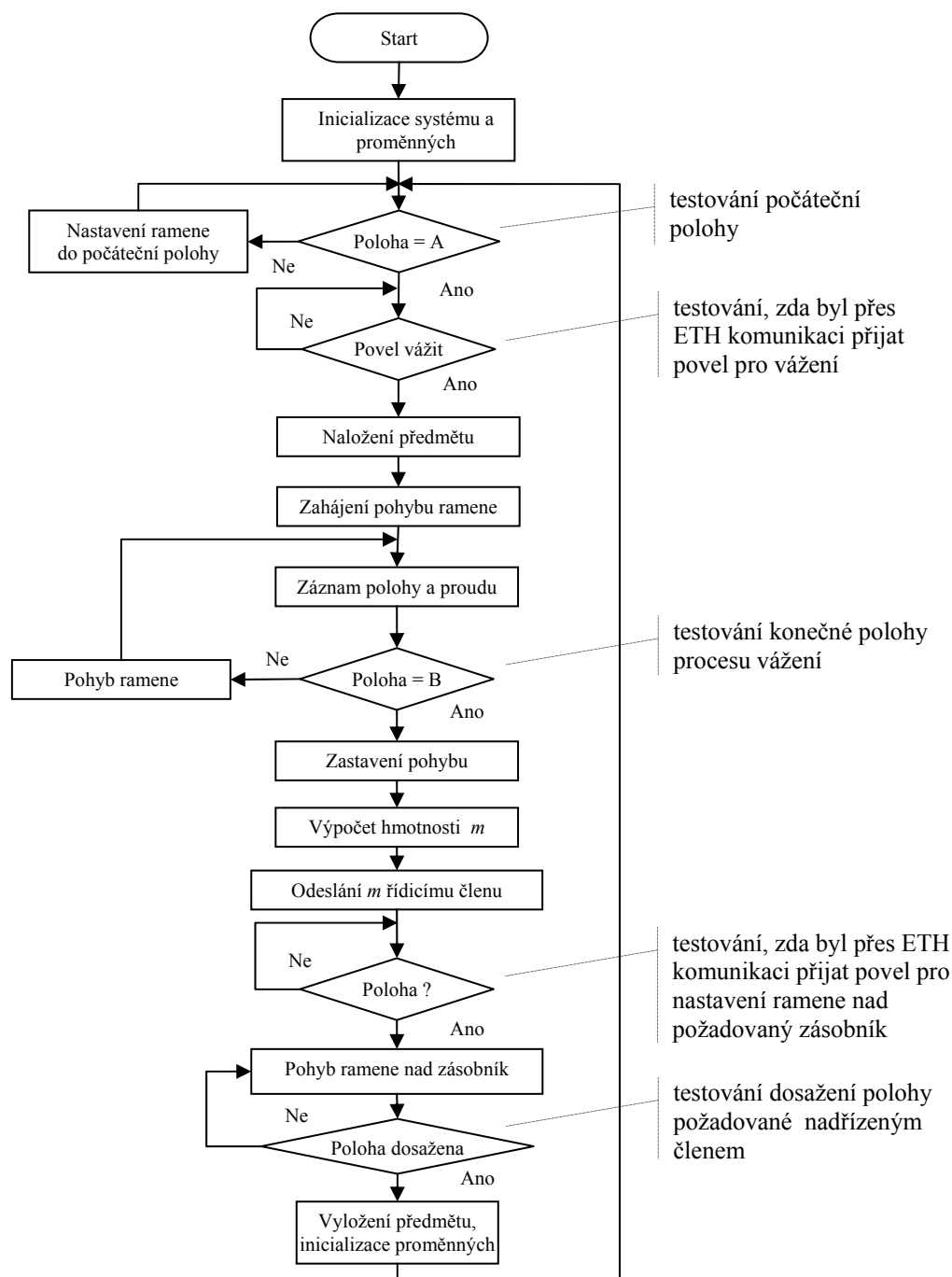
4.1.4 Datové typy

Datový typ	Název	Délka	Rozsah	Poč. hod.
BOOL	Boolean	1	0...1	0
SINT	Short integer	8	-128...127	0
INT	Integer	16	-32768...32767	0
DINT	Double integer	32	-2.147.483.648 do 2.147.483.647	0
USINT	Unsigned short integer	8	0 do 255	0
UINT	Unsigned integer	16	0 do 65535	0
UDINT	Unsigned double integer	32	0 do 4.294.967.295	0
REAL	Real numbers	32	-3.402823466 E+38 do -1.175494351 E-38 a +1.175494351 E-38 do +3.402823466 E+38	0.0
LREAL	Long real numbers	64	-1.7976931348623158 E+308 do -2.2250738585072014 E-308 a +2.2250738585072014 E-308 do +1.7976931348623158 E+308	0.0
TIME	Duration	32	0... 4.294.967.295 ms	t#0s
BYTE	Bit string of length 8	8	0...255 (16#00...16#FF)	0
WORD	Bit string of length 16	16	0...65.535 (16#00...16#FFFF)	0
DWORD	Bit string of length 32	32	0...4.294.967.295 (16#00....16#FFFFFFFF)	0

5 Navržené řešení

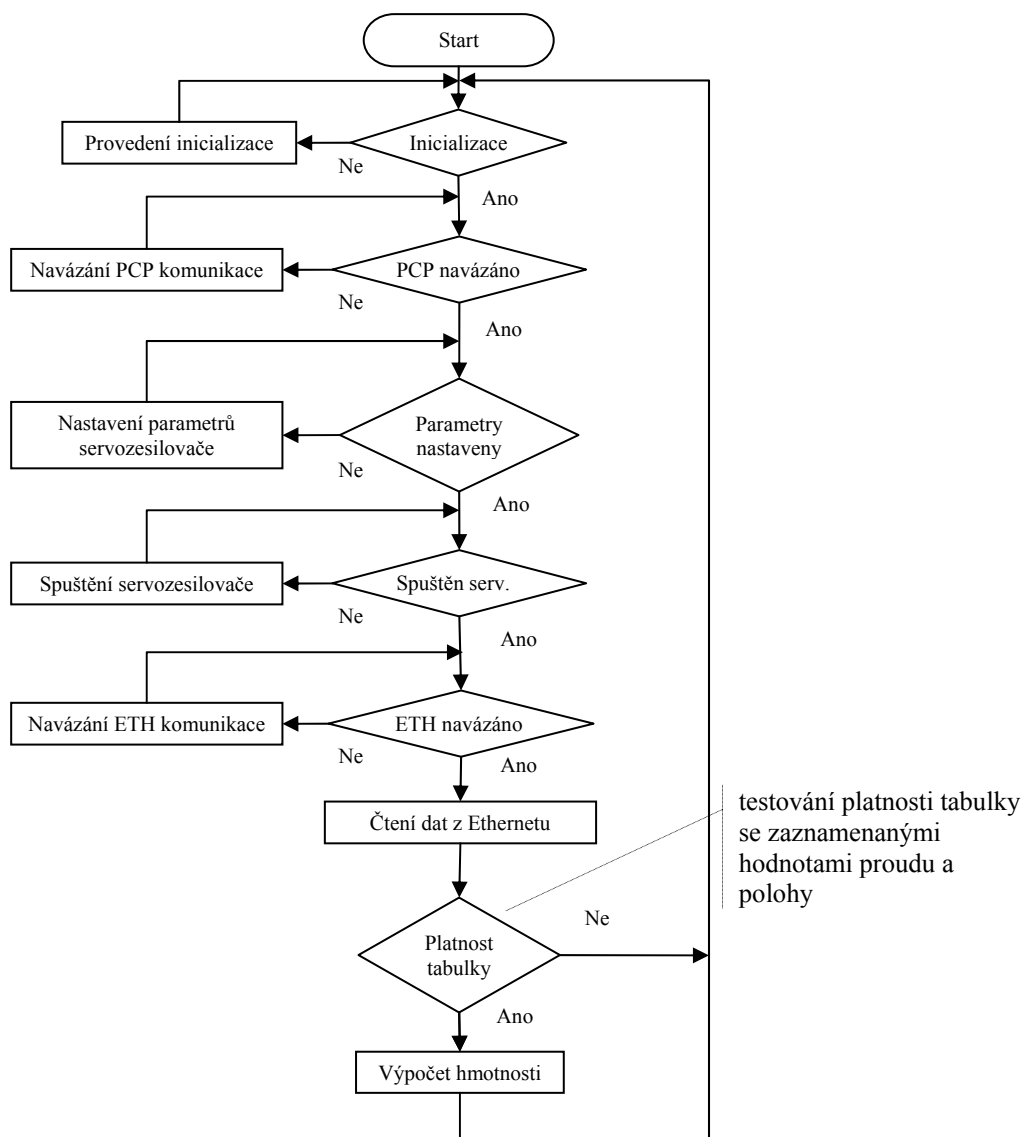
5.1 Struktura programu – vývojové diagramy

5.1.1 Projekt vážení a třídění zboží



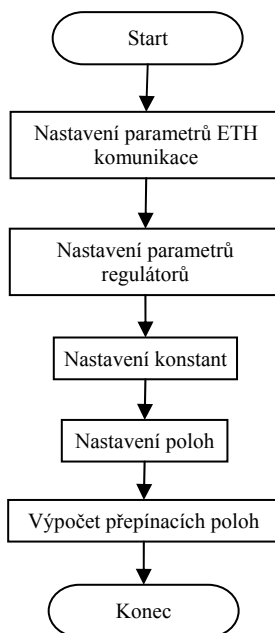
Obr. 5-1 Vývojový diagram projektu

5.1.2 Hlavní program



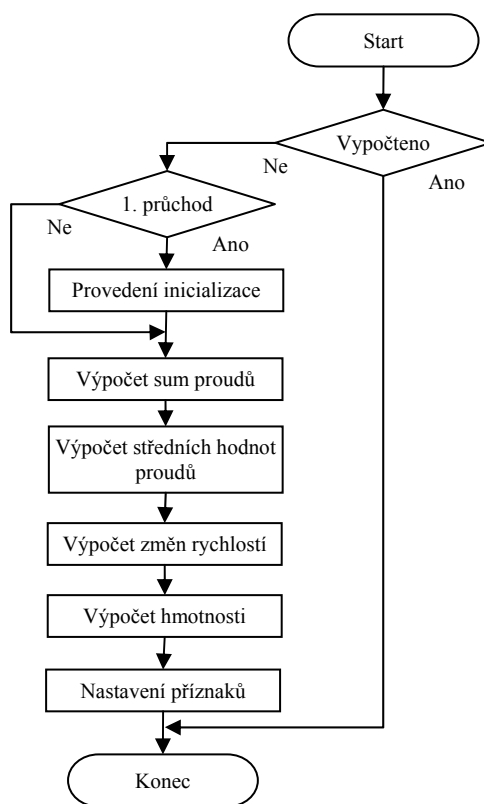
Obr. 5-2 Vývojový diagram hlavního programu

5.1.3 Inicializace konstant



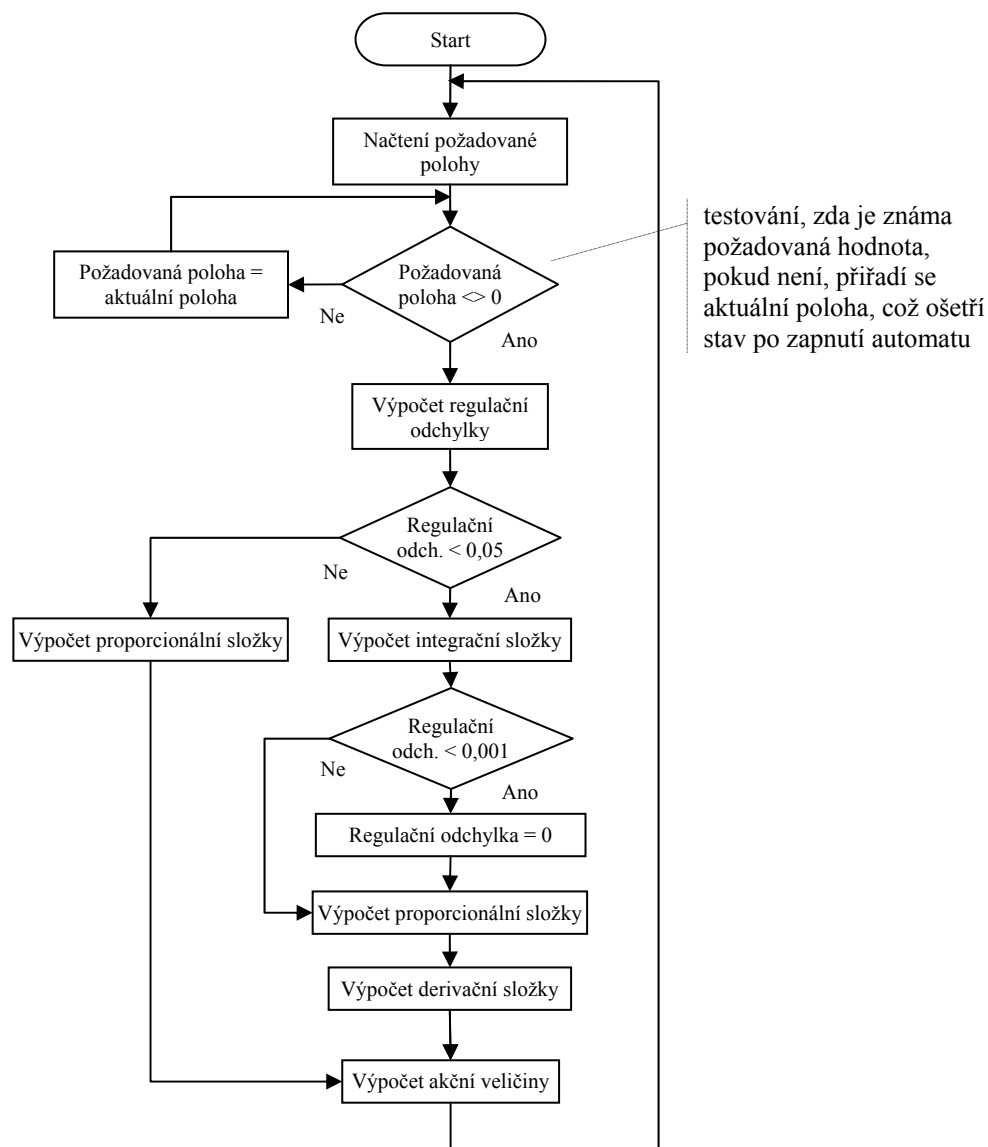
Obr. 5-3 Vývojový diagram inicializace

5.1.4 Výpočet hmotnosti



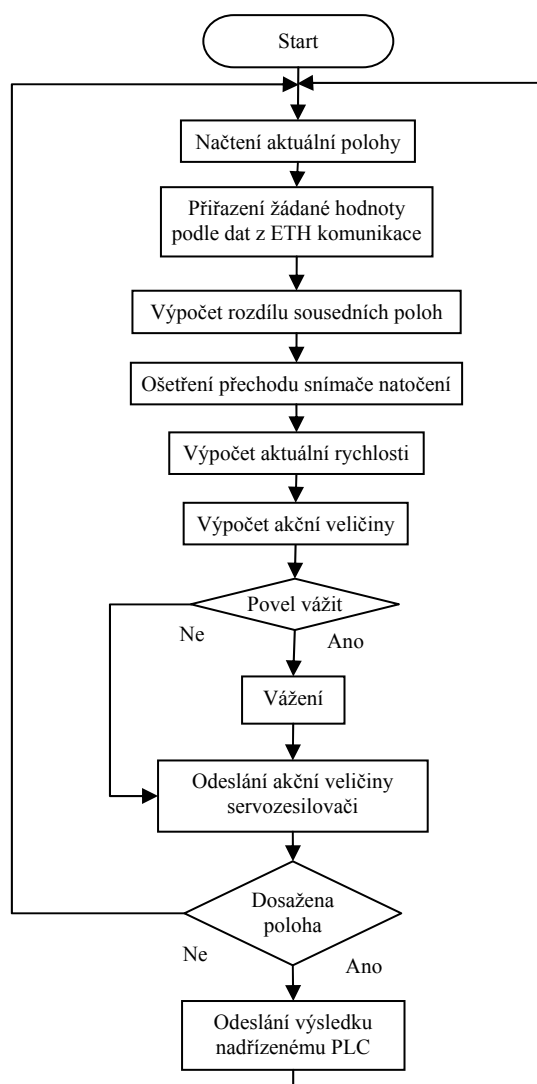
Obr. 5-4 Vývojový diagram výpočtu hmotnosti

5.1.5 Regulátor polohy



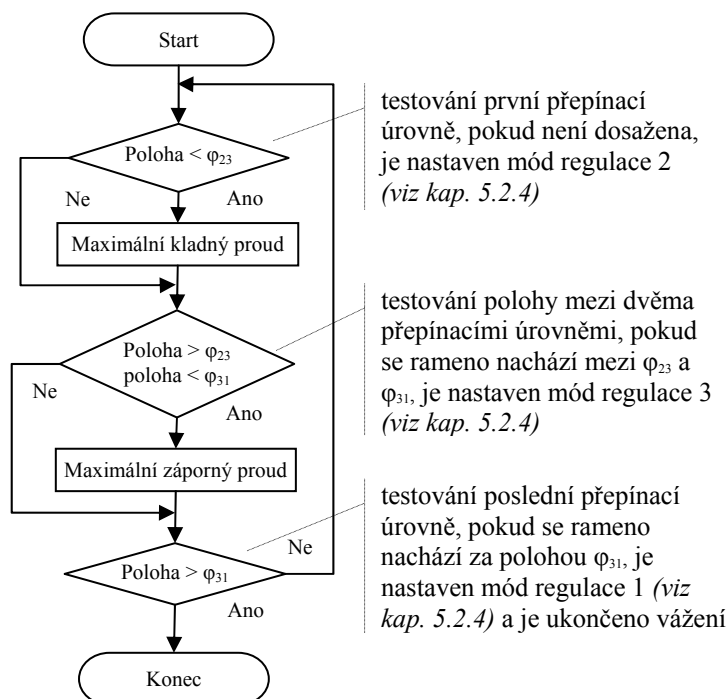
Obr. 5-5 Vývojový diagram regulátoru polohy

5.1.6 Regulátor rychlosti



Obr. 5-6 Vývojový diagram regulátoru rychlosti

5.1.7 Vážení

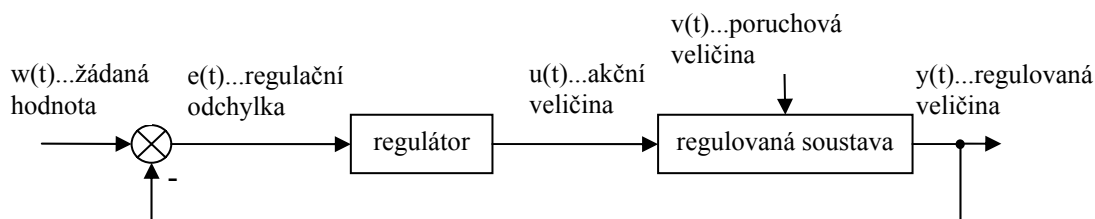


Obr. 5-7 Vývojový diagram procesu vážení

5.2 Návrh regulačního obvodu

5.2.1 Regulace a regulační obvod

Regulací nazýváme řízení, kde se využívá zpětné vazby. Regulace nám umožňuje udržovat regulovanou veličinu na požadované hodnotě při působení poruch, popřípadě ji měnit v závislosti na změně žádané hodnoty.



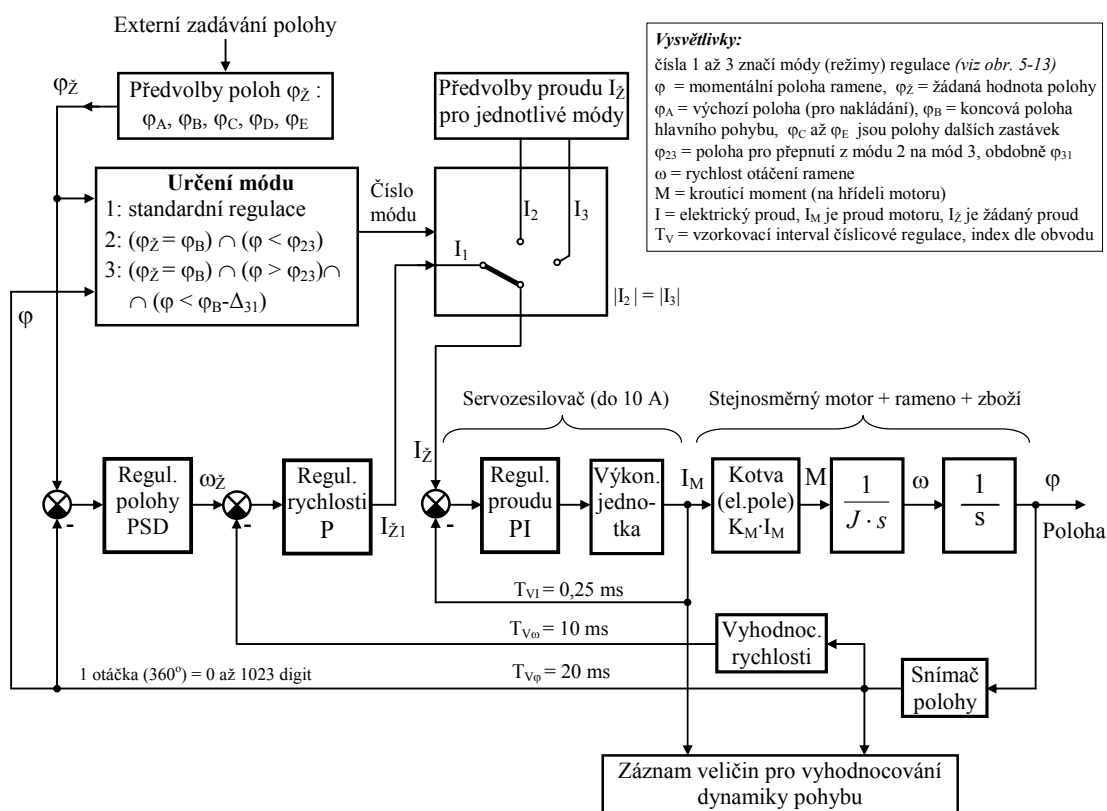
Obr. 5-8 Schéma regulačního obvodu

5.2.2 Regulační obvod pro řízení DC motoru

V regulačním obvodu je jako řízená soustava elektromotor, na jehož hřídeli je připevněno rameno. Na toto rameno se umísťuje předmět, jehož hmotnost se zjišťuje, a následně přemísťuje na danou polohu. Regulovaná soustava představuje setrvačnost

druhého řádu, která klade na regulaci vyšší nároky. Soustava je provozována převážně v nelineárním režimu, a proto použití obecných metod návrhu regulátorů je omezené.

Celý obvod je složen ze tří na sebe navazujících regulátorů. Nejtěsněji s motorem je spojen regulátor proudu (krouticího momentu), který je realizován v servozesilovači (viz kap. 3.2). Žádanou hodnotu proudu, která do tohoto regulátoru vstupuje, vypočítává regulátor rychlosti, který je řešen programem v PLC. Žádaná hodnota rychlosti je počítána v programově řešeném regulátoru polohy, jehož vstupem je poloha snímaná absolutním snímačem natočení (viz kap. 3.4). Veškeré veličiny, které jsou používány pro výpočty zásahů regulátorů, jsou normovány na rozmezí 0 až 1 jako bezrozměrné, což usnadňuje orientaci v řízení a také pochopení principů použitých v tomto řešení.



Obr. 5-9 Struktura regulace pro inteligentní vážení

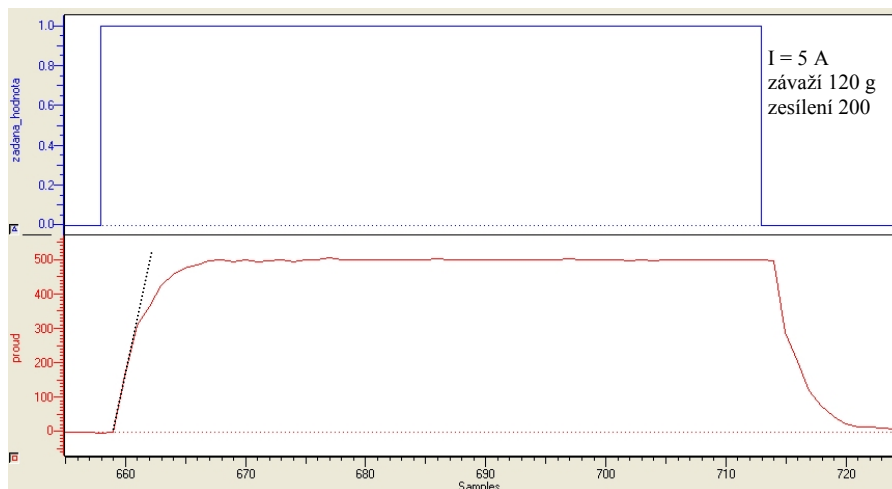
5.2.3 Návrh regulátoru proudu

Regulátor proudu (krouticího momentu) je realizován hardwarově uvnitř servozesilovače. Perioda opakování regulačního procesu je dána výrobcem, u tohoto typu servozesilovače je 250 μ s. Přednastavené parametry regulátoru jsou:

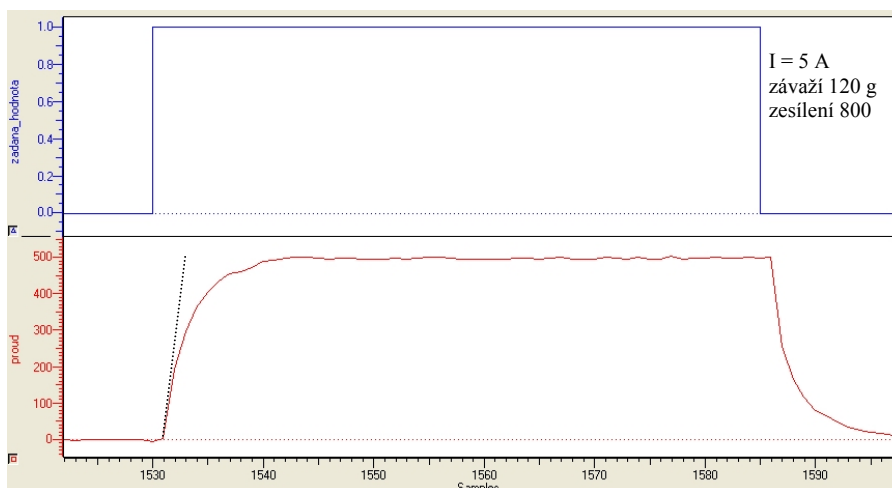
- proporcionální zesílení 200,
- integrační časová konstanta 30.

Pro dostatečnou přesnost zjišťování hmotnosti je třeba, aby náběh maximální hodnoty proudu byl co nejrychlejší. Z tohoto důvodu je volena vyšší hodnota proporcionálního zesílení. Jak je patrné z následujících obrázků, je vhodné zvolit

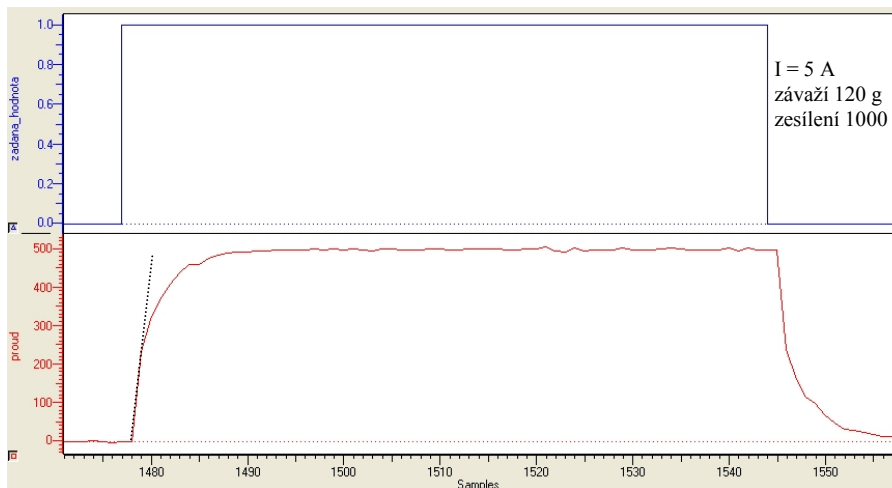
zesílení 800, protože vyšší hodnoty již nemají na přechodovou charakteristiku podstatný vliv.



Obr. 5-10 Přechodová charakteristika regulátoru proudu, zesílení 200



Obr. 5-11 Přechodová charakteristika regulátoru proudu, zesílení 800



Obr. 5-12 Přechodová charakteristika regulátoru proudu, zesílení 1000

5.2.4 Návrh regulátoru rychlosti

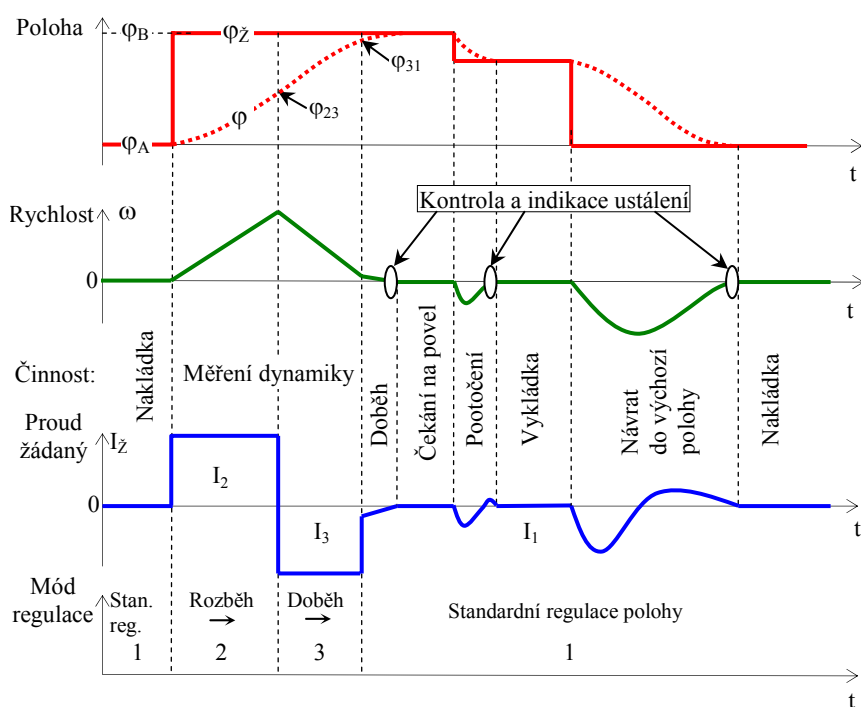
Pro potřeby zjišťování hmotnosti je třeba, aby regulátor rychlosti pracoval ve více režimech. Je zde uplatněno přepínání tří módů regulátoru. První mód je realizován jako proporcionální, pro určení akční veličiny se používá jednoduchý výpočet:

$$e = w - y \quad (5.1)$$

$$u = K_{p,v} \cdot e \quad (5.2)$$

Kde: e regulační odchylka regulátoru rychlosti
 w žádaná hodnota rychlosti
 y skutečná hodnota rychlosti
 u akční veličina regulátoru rychlosti
 $K_{p,v}$ proporcionální zesílení regulátoru rychlosti

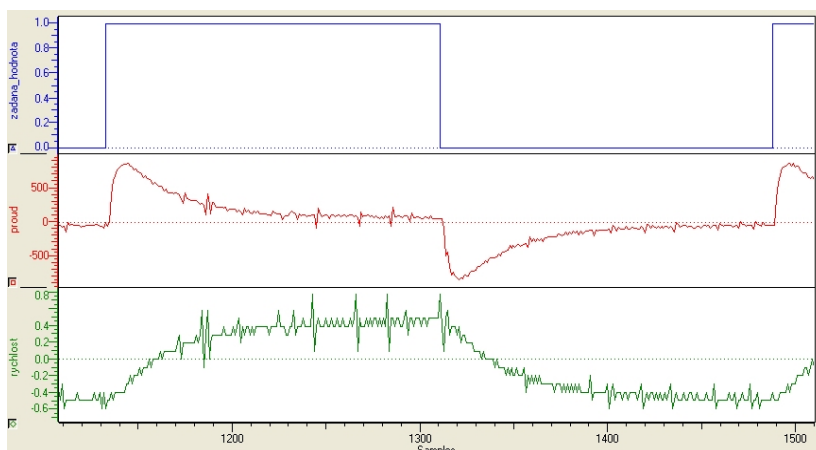
Druhý mód slouží ke zjišťování hmotnosti předmětu při rozběhu, a proto je do motoru pouštěn proud o maximální hodnotě 9 A. Poslední mód je pro výpočet hmotnosti při brzdění pohybu ramene a do motoru je dodáván maximální proud o hodnotě -9 A. Aby přechod mezi jednotlivými módy byl optimální a doby působení maximálních proudů byly co nejdelší, jsou přepínací polohy předem počítány podle speciálních vzorců, jejichž odvození je uvedeno v *kap. 5.2.4.1*.



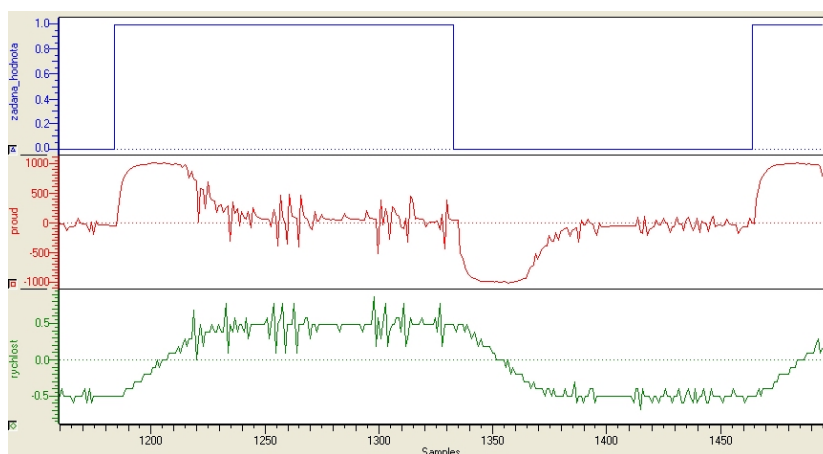
Obr. 5-13 Časové průběhy důležitých veličin a módů regulace

Jak je patrné z *obr. 5-11*, k podstatné změně asi 90% dojde během osmi cyklů programu. Program, kterým jsou charakteristiky zaznamenány, má periodu 10 ms, takže přechod z nulové hodnoty proudu na hodnotu 5 A trvá asi 80 ms. Je tedy vhodné zvolit vzorkovací periodu asi 1/10 z podstatné doby přechodové charakteristiky. Z toho vyplývá, že je optimální, pokud regulátor rychlosti pracuje s periodou 10 ms.

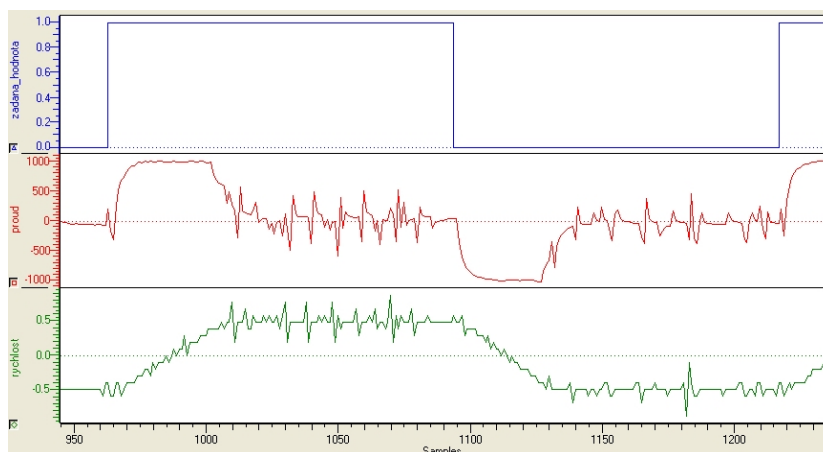
Pro návrh zesílení regulátoru je použita stejná metoda jako pro regulátor proudu. Tedy zesílení je voleno tak, aby přechodový děj odezněl v co nejkratším čase, avšak aby nedocházelo k překmitům. Bohužel kvůli podstatným zpožděním, která nastávají na sběrnici, nelze přechodový děj urychlit. Z tohoto důvodu není možné stanovit nejvhodnější proporcionalní zesílení.



Obr. 5-14 Přechodová charakteristika regulátoru rychlosti, zesílení 1000



Obr. 5-15 Přechodová charakteristika regulátoru rychlosti, zesílení 3000



Obr. 5-16 Přechodová charakteristika regulátoru rychlosti, zesílení 5000

Jak je patrné z předchozích obrázků, rozdíl v rychlosti přechodu při zesílení 3000 a 5000 není až tak patrný. Avšak rozdíl je dosti podstatný při porovnání přechodové charakteristiky se zesílením 1000 a 3000. Všechny charakteristiky byly měřeny pro skokové změny žádané hodnoty rychlosti mezi -0,5 a 0,5. Z výše uvedených skutečností vyplývá, že je vhodné stanovit zesílení o hodnotě 3000.

5.2.4.1 Výpočet přepínacích úrovní regulátoru rychlosti

Ve výpočtu je použito přírůstků

$$\Delta\varphi_{23} = \varphi_{23} - \varphi_A, \quad (5.3)$$

$$\Delta\varphi_{31} = \varphi_{31} - \varphi_{23}, \quad (5.4)$$

$$\Delta_{31} = \varphi_B - \varphi_{31} \quad \text{volitelná rezerva.} \quad (5.5)$$

Pro rozběh platí vztah

$$M_{KR} = K_m \cdot (I_{M \max} - I_T) \text{ po dobu } t_{23}. \quad (5.6)$$

Pro doběh platí vztah

$$M_{KD} = K_m \cdot (I_{M \max} + I_T) \text{ po dobu } t_{31}, \quad (5.7)$$

kde $I_{M \max} = |I_2| = |I_3|$, I_T je třecí složka proudu motoru.

Výchozí rovnice:

$$\text{a) pro polohy} \quad \varphi_B - \varphi_A = \Delta\varphi_{23} + \Delta\varphi_{31} + \Delta_{31}, \quad (5.8)$$

$$\text{b) rovnost impulzu síly} \quad M_{KR} \cdot t_{23} = M_{KD} \cdot t_{31}, \quad (5.9)$$

c) přírůstek pootočení

$$\Delta\varphi_{23} = \int_0^{t_{23}} \omega(t) dt = \int_0^{t_{23}} \left[\int_0^t \frac{M_{KR}}{J} dt \right] dt = \frac{M_{KR}}{J} \cdot t_{23}^2, \quad (5.10)$$

d) přírůstek pootočení

$$\Delta\varphi_{31} = \int_0^{t_{31}} \omega(t) dt = \int_0^{t_{31}} \left[\int_0^t \frac{M_{KD}}{J} dt \right] dt = \frac{M_{KD}}{J} \cdot t_{31}^2. \quad (5.11)$$

Řešení této soustavy 4 rovnic o 4 neznámých

Z rovnice (5.9) vyjádříme $t_{23} = \frac{M_{KD} \cdot t_{31}}{M_{KR}}$, dosadíme do rovnice (5.10) a dostáváme

$$\Delta\varphi_{23} = \frac{M_{KD}^2 \cdot t_{31}^2}{J \cdot M_{KR}}. \quad (5.12)$$

Z rovnice (5.11) vyjádříme $t_{31} = \sqrt{\frac{\Delta\varphi_{31} \cdot J}{M_{KD}}}$ a dosadíme do (5.12), dostáváme

$\Delta\varphi_{23} = \frac{M_{KD}}{M_{KR}} \cdot \Delta\varphi_{31}$, odtud $\Delta\varphi_{31} = \frac{\Delta\varphi_{23} \cdot M_{KR}}{M_{KD}}$. Po dosazení do (5.8) a úpravě obdržíme

$\Delta\varphi_{23} = \frac{M_{KD}}{M_{KR} + M_{KD}} \cdot (\varphi_B - \varphi_A - \Delta_{31})$, za momenty dosadíme rovnice (5.6) a (5.7),

$$\text{získáme } \Delta\varphi_{23} = \left[1 - \frac{I_T}{I_{M \max}}\right] \cdot \frac{(\varphi_B - \varphi_A - \Delta_{31})}{2}.$$

Zavedeme poměrnou hodnotu rezervy doběhu δ , pro niž platí: $\Delta_{31} = (\varphi_B - \varphi_A) \cdot \delta$, pak

$$\Delta\varphi_{23} = \frac{(\varphi_B - \varphi_A)}{2} \cdot \left[1 - \frac{I_T}{I_{M \max}}\right] \cdot \left[1 - \frac{\delta}{2}\right].$$

Výsledek řešení je:

$$\varphi_{23} = \varphi_A + \frac{(\varphi_B - \varphi_A)}{2} \cdot \left[1 - \frac{I_T}{I_{M \max}}\right] \cdot \left[1 - \frac{\delta}{2}\right] \quad (5.13)$$

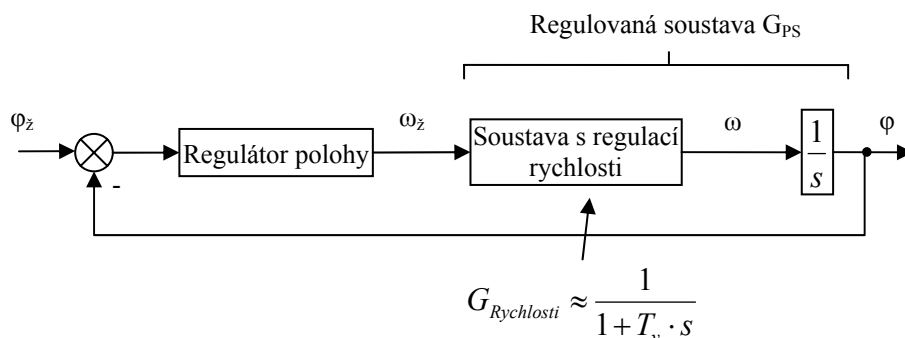
$$\varphi_{31} = \varphi_B - (\varphi_B - \varphi_A) \cdot \delta \quad (5.14)$$

5.2.5 Návrh regulátoru polohy

Regulátor polohy slouží k přesnému umístění ramene motoru nad požadovaný zásobník, popřípadě pod plnicí mechanismus. K vykládání předmětů je použito pákového mechanismu ovládaného servomotory. Síla působící od servomotoru na rameno ve směru tečny způsobuje jeho vychylování ze žádané polohy. Z tohoto důvodu jsou kladeny větší nároky na regulátor, který musí v případě vychýlení vyvolat krouticí moment motoru takový, aby síla působící na rameni vykompenzovala sílu od servomotoru.

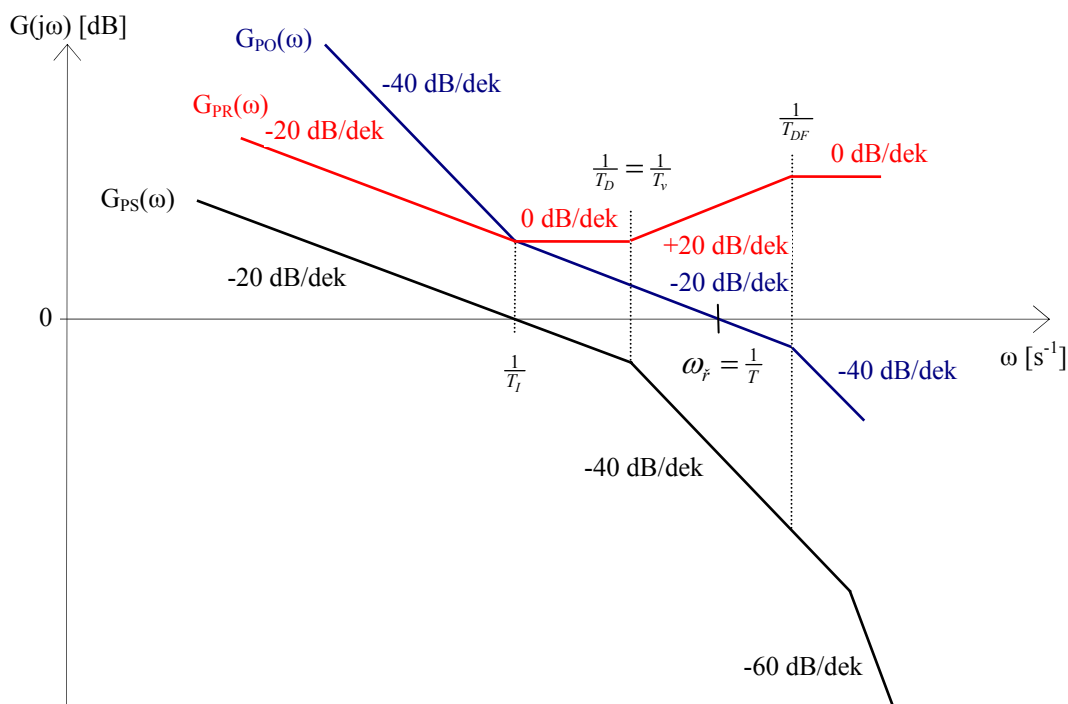
Z výše uvedených důvodů je použit PSD adaptivní regulátor, kde složka S zajišťuje přesnou polohu ramene a D zrychluje reakci na žádanou změnu. Adaptivita je řešena přepínáním módů, podle aktuální regulační odchylky (viz obr. 5-5).

První mód je určen pro velké změny polohy, kdy vzniká velká regulační odchylka. Zde je použito pouze proporcionálního zesílení regulátoru. Na druhý mód je přepnuto, pokud je regulační odchylka menší než 0,005, zde je využit plnohodnotný PSD regulátor. Třetí mód je použit, aby bylo dosaženo absolutního ustálení ramene motoru, což souvisí s přesností snímače polohy, jak bude vysvětleno později.



Obr. 5-17 Schéma regulátoru polohy

Je zřejmé, že perioda opakování výpočtu akční veličiny regulátoru polohy musí být větší než podřízených regulátorů. Je tedy volena na 20 ms. Pro určení parametrů regulátoru je použito frekvenční charakteristiky, která je pouze ilustrativní. Jelikož se řízení chová nelineárně, nelze pro určení parametrů použít běžných metod, jako je Ziegler-Nichols. Přesné hodnoty jednotlivých složek jsou zvoleny po mnoha zkouškách, kde bylo posuzováno, za jakou dobu odezní přechodový děj a také jak velký je překmit. Toto vše bylo zjišťováno pro poměrně malé změny polohy. Při pozdějším ožívání systému se ukázalo, že dané hodnoty nelze použít pro velké změny, a tak bylo přistoupeno k přepínání několika módů, jak je uvedeno výše.



Obr. 5-18 Frekvenční charakteristika regulace polohy

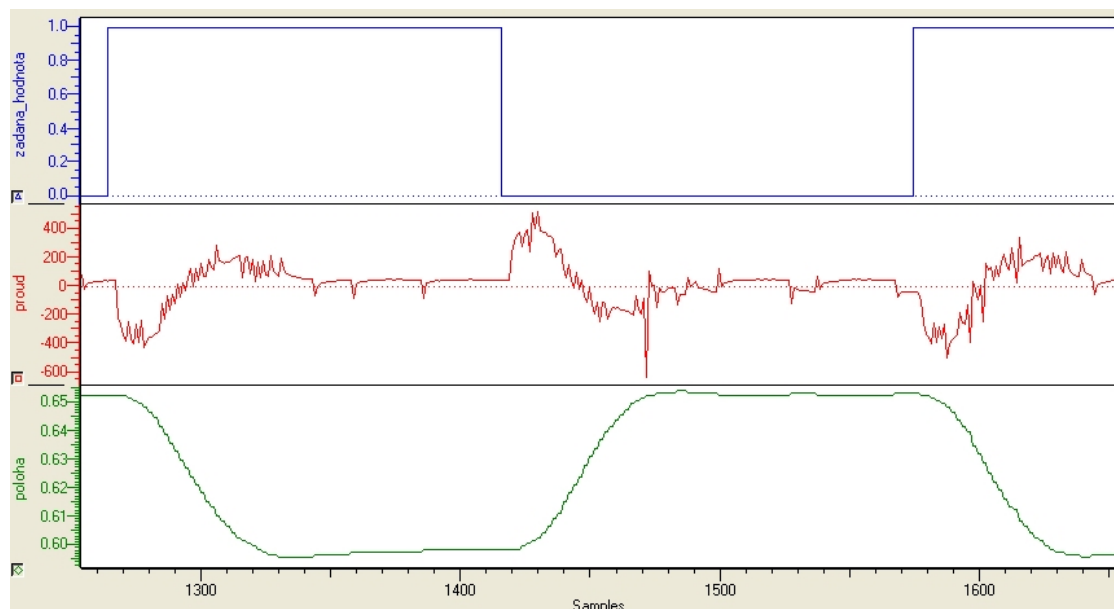
Na obr. 5-18 je znázorněna frekvenční charakteristika regulované soustavy $G_{PS}(\omega)$, navrženého regulátoru $G_{PR}(\omega)$ a výsledného regulačního obvodu $G_{PO}(\omega)$. Při návrhu parametrů regulátoru se vycházelo z přibližných hodnot, kde proporcionální zesílení bylo stanoveno experimentem na hodnotu 8. Po vykreslení přechodové charakteristiky byla změřena perioda kmitání, která je 0,88 s. Z této hodnoty je odvozena úhlová rychlost řezu $\omega_f = \frac{1}{0,88}$. Z přechodové charakteristiky regulátoru

rychlosti byla zjištěna hodnota $T_V = 0,25s$, která podle obr. 5-18 odpovídá derivační časové konstantě. Hodnota derivační filtrační časové konstanty je volena $T_{DF} = \frac{T_D}{4} = \frac{0,25}{4} = 0,06s$ a hodnota integrační časové konstanty jako $T_I = 10 \cdot T_{DF} = 0,6s$.

Jak již bylo zmíněno, tyto hodnoty jsou jen orientační a dalšími zkouškami byly nalezeny přesnější parametry, které regulaci polohy zpřesnily a urychlily.

Zvolené parametry:

- zesílení $K_p = 4,5$,
- integrační časová konstanta $T_I = 3s$,
- derivační časová konstanta $T_D = 0,35s$,
- derivační filtrační časová konstanta $T_{DF} = 0,06s$.



Obr. 5-19 Přechodová charakteristika regulátoru polohy

5.2.5.1 Algoritmus výpočtu akční veličiny regulátoru polohy

Proporcionální složka

$$u_p(k) = K_p \cdot e(k), \quad (5.15)$$

kde $u_p(k)$ je akční zásah v k -tém kroku od proporcionální složky, K_p je proporcionální zesílení regulátoru, $e(k)$ je regulační odchylka v k -tém kroku.

Sumační složka

$$u_s(k) = u_s(k-1) + C_s \cdot e(k), \quad (5.16)$$

kde $u_s(k)$ je akční zásah v k -tém kroku od sumační složky, $u_s(k-1)$ je akční zásah v předešlém kroku od sumační složky, $e(k)$ je regulační odchylka v k -tém kroku, C_s je konstanta vypočtená podle vztahu $C_s = K_p \cdot \frac{T_V}{T_S}$, K_p je proporcionální zesílení regulátoru, T_V je vzorkovací perioda regulátoru polohy a T_S je sumační časová konstanta.

Sumační složka musí být ošetřena, aby při dlouhotrvajícím vychýlení ze žádané polohy nedošlo k nasčítání do nevhodných hodnot, což by po odeznění síly, která vychýlení způsobila, znamenalo značné překmitnutí ramene.

Diferenční složka

$$u_d(k) = C_{d1} \cdot u_d(k-1) + C_{d2} \cdot (e(k) - e(k-1)), \quad (5.17)$$

kde $u_D(k)$ je akční zásah v k -tém kroku od diferenční složky, $u_D(k-1)$ je akční zásah v předešlém kroku od diferenční složky, $e(k)$ je regulační odchylka v k -tém kroku, $e(k-1)$ je regulační odchylka v předchozím kroku, C_{d1} a C_{d2} jsou konstanty vypočtené podle vztahů $C_{d1} = \frac{T_{DF}}{T_V + T_{DF}}$ a $C_{d2} = K_P \cdot \frac{T_D}{T_V + T_{DF}}$, K_P je proporcionální zesílení regulátoru, T_V je vzorkovací perioda regulátoru polohy, T_D je diferenční časová konstanta a T_{DF} je diferenční filtrační časová konstanta.

5.3 Důležité úseky zdrojového kódu

5.3.1 PCP komunikace

PCP komunikace je určena pro acyklickou komunikaci mezi PLC a přídatnými moduly. Pro řešení zadaného problému je nutné pomocí PCP nastavit parametry servozesilovače. Aby bylo možné začít posílat data, je třeba navázat spojení mezi účastníky přenosu. Pro tento účel je výrobcem předpřipraven funkční blok s názvem *PCP_CONNECT*, pro odesílání dat *PCP_WRITE* a pro čtení údajů ze servozesilovače *PCP_READ*. Tvary zápisu těchto bloků v programu jsou uvedeny v kapitolách 5.3.1.1 až 5.3.1.3.

5.3.1.1 PCP_CONNECT

```
PCP_CONNECT_1(EN_C:=TRUE,ADD_ERROR:=awErrorPCP_connection,PARTNER:=sPartner);
xPCPConnect_valid := PCP_CONNECT_1.VALID;
awErrorPCP_connection := PCP_CONNECT_1.ADD_ERROR;
sPartner := PCP_CONNECT_1.PARTNER;
iID_PCP := PCP_CONNECT_1.ID;
```

Kde *EN_C* je parametr typu BOOL, který určuje povolení navázání spojení. *ADD_ERROR* je pole typu WORD, do kterého jsou vypisována případná chybová hlášení, *PARTNER* je typu STRING a určuje jméno připojovaného zařízení. Když je spojení úspěšně navázáno, do proměnné *PCP_CONNECT_1.VALID* je vložena hodnota TRUE a proměnné *PCP_CONNECT_1.ID* je přiřazeno pořadové číslo spoje, které se následně využívá při odesílání a přijímání dat.

5.3.1.2 PCP_WRITE

```
PCP_WRITE_1(REQ:=TRUE,ID:=iID,DATA_CNT:=iByteCount,ADD_ERROR:=awError,
VAR_1:=awIndexPCP,SD_1:=awSendData);
xSend_DONE := PCP_WRITE_1.DONE;
awError := PCP_WRITE_1.ADD_ERROR;
awIndexPCP := PCP_WRITE_1.VAR_1;
awSendData := PCP_WRITE_1.SD_1;
```

Kde *REQ* je typu BOOL a značí povolení odeslání dat, *ID* je typu INT a určuje které spojení má být použito, *DATA_CNT* typu INT, kde je uvedena délka přenášených dat v BYTE, *ADD_ERROR* je pole typu WORD, do kterého jsou vypisována případná chybová hlášení, *VAR_1* je pole typu WORD, kde je uvedena adresa, na kterou jsou data odesílána (základní lze vyčíst z obr. 3-8) a v poli typu *WORD SD_1* jsou odesílána

data. Pokud dojde k úspěšnému odeslání, proměnná *PCP_WRITE_1.DONE* bude obsahovat hodnotu TRUE.

5.3.1.3 PCP_READ

```
PCP_READ_1(REQ:=TRUE,ID:=iID,ADD_ERROR:=awError,VAR_1:=awIndexPCP,RD_1:=awReadData);
xRead_DONE := PCP_READ_1.NDR;
iByteCount := PCP_READ_1.DATA_CNT;
awError := PCP_READ_1.ADD_ERROR;
awIndexPCP := PCP_READ_1.VAR_1;
awReadData := PCP_READ_1.RD_1;
```

Většina parametrů je shodná jako u bloku *PCP_WRITE*. V proměnné *VAR_1* je uvedena adresa, ze které se mají data číst (základní lze vyčíst z *obr. 3-8*). Po úspěšném přečtení je příznak *PCP_READ_1.NDR* nastaven na hodnotu TRUE a do pole typu WORD *RD_1* jsou zapsána přenesená data.

5.3.2 Spuštění servozesilovače

Aby bylo možné začít ovládat motor, je třeba nejprve uvést v činnost samotný servozesilovač. To se provede postupným projitím jeho několika stavů, které jsou zřejmé z *obr. 5-20*. Po spuštění je automaticky navozen stav *Start inhibit*. Zapisováním hodnot do komunikačního registru *control word* se servozesilovač přepíná. Ve stavu *Operation enabled* lze již posílat žádanou hodnotu proudu servozesilovači, který požadavek vykoná.

Úsek programu vypadá následovně:

```
(* ---- osetreni error ---- *)
IF (wStatusWord.X0 = FALSE)&(wStatusWord.X1 = FALSE)&(wStatusWord.X2 = FALSE)&(wStatusWord.X3 = TRUE)
    THEN wControlWord.X7 := FALSE;
        wControlWord.X7 := TRUE;
END_IF;

(* ---- Start inhibit -> Ready to operate ---- *)
IF (wStatusWord.X0 = FALSE)&(wStatusWord.X1 = FALSE)&(wStatusWord.X2 = FALSE)&(wStatusWord.X3 = FALSE)&(wStatusWord.X6 = TRUE)
    THEN wControlWord.X0 := FALSE;
        wControlWord.X1 := TRUE;
        wControlWord.X2 := TRUE;
END_IF;

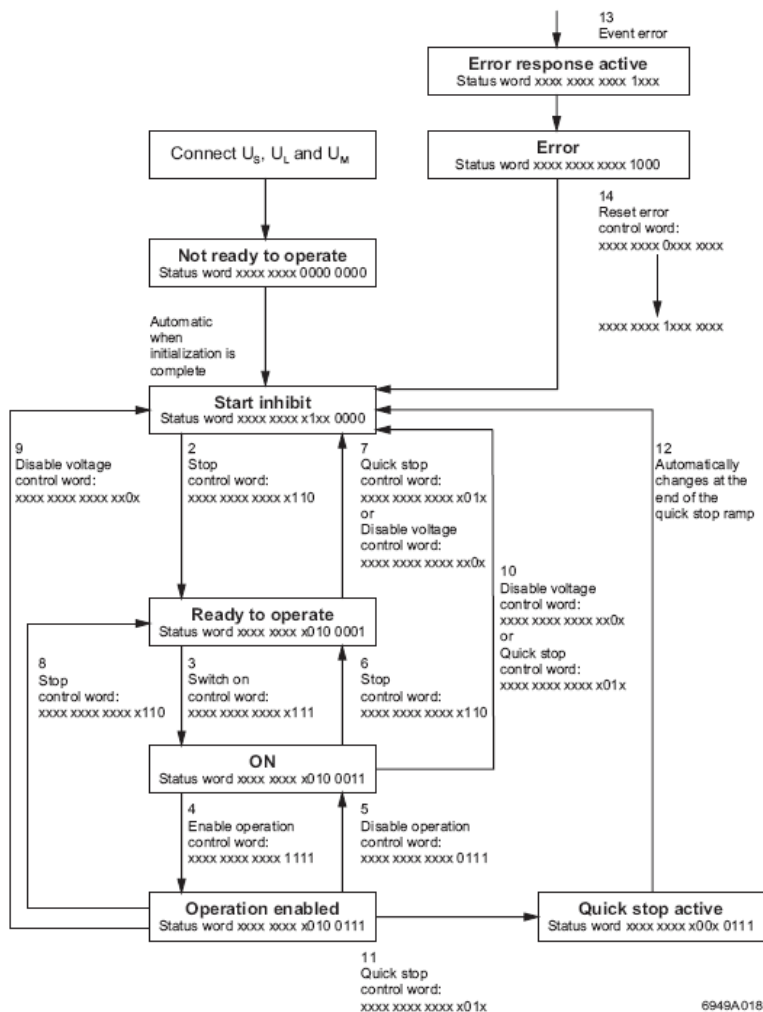
(* ---- Ready to operate -> ON ---- *)
IF (wStatusWord.X0 = TRUE)&(wStatusWord.X1 = FALSE)&(wStatusWord.X2 = FALSE)&(wStatusWord.X3 = FALSE)&(wStatusWord.X4 = FALSE)&(wStatusWord.X5 = TRUE)&(wStatusWord.X6 = FALSE)
    THEN wControlWord.X0 := TRUE;
        wControlWord.X1 := TRUE;
        wControlWord.X2 := TRUE;
END_IF;

(* ---- ON -> Operation enabled ---- *)
```

```

IF (wStatusWord.X0 = TRUE)&(wStatusWord.X1 = TRUE)&(wStatusWord.X2 =
FALSE)&(wStatusWord.X3 = FALSE)&(wStatusWord.X4 = FALSE)&(wStatusWord.X5
= TRUE)&(wStatusWord.X6 = FALSE)
    THEN  wControlWord.X0 := TRUE;
          wControlWord.X1 := TRUE;
          wControlWord.X2 := TRUE;
          wControlWord.X3 := TRUE;
END_IF;

```



Obr. 5-20 Pracovní stavy servozesilovače

5.3.3 Ethernetová komunikace

Aby bylo možné přenášet data mezi programovatelnými automaty, je nejprve třeba navázat spojení. Princip je velice podobný již výše zmíněné PCP komunikaci. Pro samotné propojení je použit funkční blok *IP_CONNECT*, pro příjem dat *IP_URCV* a pro odesílání *IP_USEND*. Tvary jejich zápisu v programu jsou uvedeny v kapitolách 5.3.3.1 až 5.3.3.3.

5.3.3.1 IP_CONNECT

```
IP_CONNECT_1(EN_C:=TRUE,PARTNER:=sPartnerETH);
xETHConnection_VALID := IP_CONNECT_1.VALID;
iETH_ID := IP_CONNECT_1.ID;
```

Kde *EN_C* je proměnná typu BOOL a značí povolení k navázání spojení, *PARTNER* je proměnná typu STRING, vkládá se do ní název zařízení, se kterým má být komunikace navázána. Pokud je spojení úspěšně utvořeno, do příznaku *IP_CONNECT_1.VALID* se zapíše TRUE a v proměnné *IP_CONNECT_1.ID* je uloženo číslo spojení, které se dále používá při komunikaci.

5.3.3.2 IP_URCV

```
IP_URCV_1(EN_R:=TRUE,ID:=iETH_ID,RD_1:=stReceivedData);
stReceivedData := IP_URCV_1.RD_1;
xETH_received := IP_URCV_1.NDR;
```

Kde *EN_R* je proměnná typu BOOL a značí povolení ke čtení z ethernetového spojení, *ID* je číslo spojení, ze kterého má být čtení provedeno. Pokud jsou data přečtena, nastaví se příznak *IP_URCV_1.NDR* na TRUE a data jsou vložena do strukturovaného textu (podobná struktura jako pole) *IP_URCV_1.RD_1*.

5.3.3.3 IP_USEND

```
IP_USEND_1(REQ:=xSendEnable,ID:=iETH_ID,SD_1:=stSendData);
xSend_DONE:=IP_USEND_1.DONE;
stSendData:=IP_USEND_1.SD_1;
```

Kde *REQ* je proměnná typu BOOL a značí povolení k odeslání přes ethernetové spojení, *ID* je číslo spojení, přes které má být odeslání provedeno, strukturovaný text *SD_1* obsahuje data, která se mají přenést. Pokud jsou data odeslána, nastaví se příznak *IP_USEND_1.DONE* na TRUE.

5.3.4 Regulátor polohy

Výpočet akční veličiny regulátoru polohy je dán algoritmem uvedeným v *kap. 5.2.5*. Níže uvedenému zdrojovému kódu předchází jen jednoduchý výpočet regulační odchylky, který spočívá v odečtení skutečné hodnoty od žádané.

```
IF (ABS(rDeviation) < 0.05)
THEN
  (* --- integracni slozka --- *)
  rUi_1 := rUi;
  IF iRegulationMod <> 1
  THEN rUi := 0.0;
  ELSE rUi := rUi_1 + rCi * rDeviation;
END_IF;

IF rUi > 0.3
THEN rUi := 0.3;
END_IF;

IF rUi < -0.3
THEN rUi := -0.3;
```

```

END_IF;

IF ABS(rDeviation) < 0.001
    THEN rDeviation := 0.0;
END_IF;

(* --- proporcionalni slozka --- *)
rUp := rProportPosition * rDeviation;

(* --- derivacni slozka --- *)
rUd_1 := rUd;
rUd := rCd1 * rUd_1 + rCd2 * (rDeviation - rDeviation_1);

(* --- akcni velicina --- *)
rManipulatedVar_PosCo := rUp + rUi + rUd;
ELSE
    rUp := 3.0 * rDeviation;
    rManipulatedVar_PosCo := rUp;

END_IF;

```

5.3.5 Regulátor rychlosti

Nejprve je třeba určit aktuální natočení hřídele motoru. To je zjištěno přečtením údaje z absolutního snímače natočení.

```

wPosition := ONBOARD_INPUT;
wPosition := ROL_WORD(wPosition,2);
wPosition.X0 := wPosition.X10;
wPosition.X1 := wPosition.X11;
wPosition := wPosition & WORD#16#3FF;

```

Poté je hodnota znormována na velikost 0 až 1, aby byl výpočet akční veličiny jednodušší a přehlednější.

```

rActualPosition := WORD_TO_REAL(wPosition)/1024.0;

```

Podle přijatého pokynu od nadřazeného automatu se nastaví žádaná hodnota polohy, popřípadě se povolí zjišťování hmotnosti a vynulují se příznaky provedeního výpočtu hmotnosti, aby mohl být proveden výpočet nový.

```

IF stReceivedData.wCommand = WORD#02
    THEN
        CASE WORD_TO_INT(stReceivedData.wValue) OF
            1:rRequiredPosition := aPosition[1]; xCalculate_DONE := FALSE;
            xWeightEnable := FALSE; xSend_DONE := FALSE;
            2:rRequiredPosition := aPosition[2]; xCalculate_DONE := FALSE;
            xWeightEnable := FALSE; xSend_DONE := FALSE;
            3:rRequiredPosition := aPosition[3]; xCalculate_DONE := FALSE;
            xWeightEnable := FALSE; xSend_DONE := FALSE;
            4:rRequiredPosition := aPosition[4]; xCalculate_DONE := FALSE;
            xWeightEnable := FALSE; xSend_DONE := FALSE;
            5:rRequiredPosition := aPosition[5]; xCalculate_DONE := FALSE;
            xWeightEnable := FALSE; xSend_DONE := FALSE;
        END_CASE;
    END_IF;

```

```

IF (stReceivedData.wCommand = WORD#01)&(stReceivedData.wValue = WORD#00)
  THEN xWeightEnable := TRUE; xSend_DONE := FALSE;
  ELSE xWeightEnable := FALSE; xSend_DONE := FALSE; xCalculate_DONE
       := FALSE;
END_IF;

```

Nyní následuje samotný výpočet akčního zásahu regulátoru rychlosti, jehož žádanou hodnotou je akční veličina předchozího stupně, tedy regulátoru polohy.

```

(* ---- vypocet rozdilu dvou sousednich poloh ---- *)
iPositionDifference := iActualPosition - iActualPosition_1;
iActualPosition_1 := iActualPosition;

(* ---- osetreni prechodu snimace natoceni ---- *)
IF iPositionDifference < -512 THEN iPositionDifference := iPositionDifference + 1024;
END_IF;
IF iPositionDifference > 512 THEN iPositionDifference := iPositionDifference - 1024;
END_IF;

(* ---- vypocet rychlosti ---- *)
rActualSpeed := (INT_TO_REAL(iPositionDifference)/1024.0) / rTime;

(* ----- vypocet akcni veliciny ----- *)
rDeviation := rManipulatedVar_PosCo - rActualSpeed;
rManipulatedVar_SpeedCo := rProportSpeed * rDeviation;

```

Akční veličina regulátoru rychlosti je žádanou hodnotou následujícího stupně regulace, tedy regulátoru proudu.

Pokud je přijata žádost na zjištění hmotnosti předmětu umístěného na rameni motoru, je nastaven mód 2 regulátoru. Servozesilovač začne dodávat do motoru maximální možnou hodnotu proudu (programově omezeno na 9A) a spustí se zapisování měřených veličin do tabulky.

```

IF (rActualPosition < rSwitchingPos23)&(xWeightEnable =
  TRUE)&(xCalculate_DONE = FALSE)
  THEN iRegulationMod := 2; (*zrychlení*)
       arRecordAcceleration[1][iRowAcceleration] := rActualPosition;
       arRecordAcceleration[2][iRowAcceleration] :=
       WORD_TO_REAL(wActualValue);
       iRowAcceleration := iRowAcceleration + 1;
       wSetPoint := REAL_TO_WORD(rMaxCurrent);
       xMeasuring := TRUE;

```

Jakmile je dosaženo první přepínací úrovně, regulace se přepne do 3. módu. Polarita proudu se obrátí (je tedy -9A) a motor začne působit kroutícím momentem působícím v opačném směru k probíhající rotaci. Tím je dosaženo brzdění ramene. Při brzdění jsou stejné hodnoty zaznamenávány do další tabulky, aby se později dala snadněji odstranit nepotřebná data, tedy taková, která jsou zatížena chybou způsobenou momenty tření.

```

IF (rRequiredPosition = aPosition[3])&(rActualPosition >
  rSwitchingPos23)&(rActualPosition < rSwitchingPos31)
  THEN iRegulationMod := 3; (*zrychlení*)
       arRecordDeceleration[1][iRowDeceleration] := rActualPosition;
       arRecordDeceleration[2][iRowDeceleration] :=
       WORD_TO_REAL(wActualValue);

```

```
iRowDeceleration := iRowDeceleration + 1;
xMeasuring := TRUE;
xCalculate_DONE := FALSE;
wSetPoint := REAL_TO_WORD(- rMaxCurrent);
```

Když rameno dosáhne poslední přepínací úrovně, je navolen 1. mód, který využívá výše zmíněný výpočet akčního zásahu, který nebyl v předešlých módech brán v potaz.

```
iRegulationMod := 1; (* mod standard *)
wSetPoint := REAL_TO_WORD(rManipulatedVar_SpeedCo);
```

Jestliže je dosaženo žádané polohy, je odeslána odpověď nadřazenému PLC podle předem stanovených pravidel. Pokud byl přijat povel na zjišťování hmotnosti, odešle se zjištěná hmotnost:

```
stSendData.wCommand := WORD#01;
IF rWeight < 0.0 THEN rWeight := 0.0; END_IF;
stSendData.wValue := REAL_TO_WORD(rWeight);
```

Pokud byl povel na dosažení nějaké polohy, odešle se stejná hodnota zpět:

```
stSendData := stReceivedData;
```

5.3.6 Výpočet hmotnosti

Jakmile je přepnuto z módu 2 regulátoru rychlosti do módu 1, spustí se výpočet hmotnosti ze zaznamenaných hodnot. Nejprve se vynulují pomocné proměnné a zjistí se, kolik bylo načteno hodnot.

```
iRowsAcceleration := iRowAcceleration - 1;
iRowsDeceleration := iRowDeceleration - 1;
rSumCurrentAcceleration := 0.0;
rSumCurrentDeceleration := 0.0;
```

Následuje vypočtení sum proudů samostatně pro urychlování a zpomalování pohybu, přičemž jsou vynechány dvě první hodnoty v obou tabulkách, které jsou poznamenány chybou způsobenou třecími momenty a zpožděním komunikace na sběrnici.

```
FOR i := 3 TO iRowsAcceleration DO
    rSumCurrentAcceleration := rSumCurrentAcceleration +
        arRecordAcceleration[2][i];
END_FOR;

FOR j := 3 TO iRowsDeceleration DO
    rSumCurrentDeceleration := rSumCurrentDeceleration +
        arRecordDeceleration[2][j];
END_FOR;
```

Dále je vypočtena průměrná hodnota obou proudů a změna rychlostí.

```
rAverageCurrentAcceleration := rSumCurrentAcceleration /
    INT_TO_REAL(iRowsAcceleration - 3);
```

```
rAverageCurrentDeceleration := rSumCurrentDeceleration /
INT_TO_REAL(iRowsDeceleration - 3);
```

```
rDeltaSpeedAcceleration := (arRecordAcceleration[1][iRowsAcceleration] -
arRecordAcceleration[1][3]) / (rTime * INT_TO_REAL(iRowsAcceleration - 3));
rDeltaSpeedDeceleration := (arRecordDeceleration[1][iRowsDeceleration] -
arRecordDeceleration[1][3]) / (rTime * INT_TO_REAL(iRowsDeceleration - 3));
```

Závěrem jsou vypočteny dvě hmotnosti, přičemž jedna je dána hodnotami z rozběhu a druhá z brzdění. Tyto jsou následně zprůměrovány a výsledek je považován za skutečnou hmotnost váženého předmětu.

```
rWeightAcceleration := rK * (rAverageCurrentAcceleration - rCurrentFriction) * rTime /
rDeltaSpeedAcceleration - rResidualWeight;
rWeightDeceleration := rK * (ABS(rAverageCurrentDeceleration) + rCurrentFriction) *
rTime / rDeltaSpeedDeceleration - rResidualWeight;
```

```
rWeight := (rWeightAcceleration + rWeightDeceleration) / 2.0;
```

5.4 Poznátky z řešení

Prvním problémem, se kterým se bylo třeba vypořádat již v počátku řešení, byla parametrizace a ovládání servozesilovače. Z manuálu bylo zřejmé, že k nastavení se používá acyklická komunikace PCP, kdežto k procesní komunikaci jsou určena dvě slova cyklické komunikace. Bohužel ale nebylo z dostupných pramenů zřejmé, jakým způsobem komunikaci navázat a informace předávat. To velice zdrželo průběh řešení práce. Nakonec byly nalezeny bloky sloužící pro PCP komunikaci (*viz kap. 5.3.1*). Je třeba dávat pozor, protože některé hodnoty pro parametrizaci jsou odesílány v datovém typu byte. Zkouškami se ověřilo, že servozesilovač nemá paměť pro zápis změněných parametrů, a proto je po každém zapnutí uveden do stavu daného výrobcem. Z toho vyplývá nutnost při každém zapnutí automatu provést inicializaci, ve které se provede nastavení všech potřebných parametrů.

Procesní komunikace, tedy ta, která slouží k zadávání okamžitých žádaných hodnot a ke čtení aktuálních veličin ze servozesilovače, se provádí prostřednictvím dvou registrů, a to SetPoint a ActualValue. Jsou to registry typu word. Při parametrizaci servozesilovače je třeba nastavit, jaká hodnota se bude do registru SetPoint zapisovat (v závislosti na použité regulaci) a jaká se bude z registru ActualValue číst. Možnosti nastavení je možné najít v manuálu k servozesilovači.

Dalším velkým problémem bylo čtení aktuální polohy hřídele motoru ze snímače natočení. Nepřesnost čtení se projevila při nastavování regulátorů, kde byly v průbězích proudů a následně i rychlostí zaznamenány chybné hodnoty. Podrobným rozбором bylo zjištěno, že jsou data čtena s chybou. Varianta, že by byl vadný kotouč snímače byla vyloučena, protože provedení je velmi precizní. Zbyla tedy možnost chyby na straně automatu, a nebo výstupním obvodu snímače. Pro eliminaci zpožděného čtení některých bitů jsem se pokoušel využít vstupu snímače LATCH, o němž jsem se domníval, že po přivedení logické „1“ podrží všech deset bitů na aktuální hodnotě, aby je automat mohl přečíst. Aby mohl být LATCH použit, bylo by třeba, posílat velice rychle impulsy na tento vstup, a to tak, aby se opakovaly v pravidelných časových intervalech a zadržení vždy předcházelo okamžiku, ve kterém se má provést čtení. Po dlouhém hledání možnosti, jak tyto impulsy z automatu vysílat, bylo zjištěno, že automat neumí vytvořit dostatečně rychlý a časově přesný spouštěný program, který by výstup obsloužil. Navíc doba, která je potřeba k přepnutí

výstupu mezi logickými úrovněmi, není zanedbatelná. Nesprávné načasování zadržení údaje by znamenalo přečtení údaje, který by nebyl v daném okamžiku aktuální, a tím by byla regulace velice nepřesná a možná i nestabilní. Funkce vstupu LATCH byla proto testována pomocí generátoru pulsů, avšak nebylo zcela zjištěno, s jakou frekvencí lze daný vstup přepínat.

Prvním pokusem, jak problém vyřešit, bylo posunutí všech deseti bitů na vstupech automatu, přičemž dva s nejnižší vahou byly připojeny na vstupy, o kterých byla domněnka, že nemají dostatečnou rychlost čtení. Tuto změnu bylo ale potřeba zohlednit i v části programu, kde se hodnota polohy načítala, a musela se provést rotace a přesun všech deseti bitů. Tímto problém nebyl zcela vyřešen, ale chyba čtení se podstatně zmenšila. Původně se jednalo o chyby na hranici poloviny a čtvrtiny rozsahu, což odpovídalo bitům s nejvyšší a druhou nejvyšší vahou. Nyní tedy bylo dosaženo relativně malé chyby, protože šlo jen o občasné chybné přečtení dvou bitů s nejnižší vahou.

Jak již bylo zmíněno výše, při řešení se potvrdilo, že pouze osm binárních vstupů automatu je schopno zaznamenat rychlé změny. U zbývajících čtyř vstupů jsou časové filtry s delšími periodami (viz *kap. 3.1.7*).

Během nastavování parametru regulátoru proudu se objevilo nepříjemné zpoždění interní sběrnice interbus. To se projevilo na přechodové charakteristice, kde byla odeslána žádaná hodnota proudu a změna se začala projevovat až po několika milisekundách. S ohledem na tento problém se tedy muselo přistupovat i k nastavování regulátoru, kde nebylo možné dosáhnout rychlejší reakce na změnu žádané hodnoty.

Pro ethernetovou komunikaci mezi PLC jsou v programovacím prostředí PC WorX vytvořeny bloky, které usnadňují výměnu dat. Jejich využití ale vyžaduje nastavit pro program, kde mají být použity, typ procesoru, jinak jsou nedostupné. Tyto jsou blíže popsány v *kap. 5.3.3*.

Jelikož byl použit desetibitový absolutní snímač natočení, v konečné poloze ramene, kdy již mělo být v klidu, docházelo k pohybu v rozmezí daném citlivostí snímače. Aby se tento efekt zmírnil, je v případě dosažení takového stavu regulátor polohy vypnut. To zajistí, že rameno zůstane v klidu, dokud se nevychýlí mimo toto toleranční pole.

6 Závěr

Diplomová práce byla řešena na prostředcích od společnosti Phoenix Contact, které jsou teprve několik měsíců součástí vybavení laboratoří ústavu automatizace. Z tohoto důvodu ještě nebyly ani zařazeny do výuky, a tak bylo třeba se seznámit s těmito automaty a s programovacím prostředím PC WorX formou samostudia.

Během řešení se vyskytlo i několik problémů, které bylo třeba řešit. Jednou z obtížných částí bylo pochopení a zvládnutí PCP komunikace mezi programovatelným automatem a rozšiřujícím modulem servozesilovače.

Regulace ramene motoru byla řešena jako blok tří na sebe navazujících regulátorů, což umožňuje přesnější a rychlejší regulaci. Bylo by možné použít jen regulátor proudu a polohy, ale nebylo by potom možné zajistit potřebnou dynamiku, kterou navržená regulace umožňuje.

Regulátor rychlosti byl řešen jako vícemódový, kde k přepínání dochází na základě právě probíhajícího procesu. Tím je myšlena buď klasická regulace, a nebo proces zjišťování hmotnosti. Regulátor polohy je také řešen s přepínáním, ale tentokrát lze říci, že se jedná o adaptivní řízení. Přepínání je řízeno aktuální hodnotou regulační odchylky. Toto řešení umožnilo přesnější a vzhledem k celému projektu i rychlejší řízení ramene motoru.

Problematika zjišťování hmotnosti na základě dynamiky pohybu byla úspěšně zvládnuta na použitých prostředcích. Díky vyhodnocování hmotnosti z rozběhu a brzdění byly eliminovány vlivy tření v motoru a částečně i nutnost absolutní vodoroviny pod celým zařízením. Odchylky hmotností zjištěných touto metodou od skutečných hodnot odpovídají předpokladům. Celé řešení bylo navrženo pro zjišťování hmotností předmětů od přibližně 30 g do cca 200 g, přičemž chyba nepřesahuje 5 g.

Řešení nebylo provedeno pouze teoreticky, ale také v praktické podobě. Bylo zrealizováno zařízení sloužící k navažování citrónů do zásobníků podle daných kritérií. Dané zařízení a především poznatky z řešení bude možné využít ve výuce.

Diplomová práce se v rámci týmového projektu účastnila mezinárodní soutěže Xplore New Automation Award 2008. Tým sice nebyl vybrán do finálového kola, ale i přesto si myslím, že se jednalo o úspěšnou reprezentaci Ústavu automatizace a informatiky FSI VUT v Brně.

Seznam použité literatury

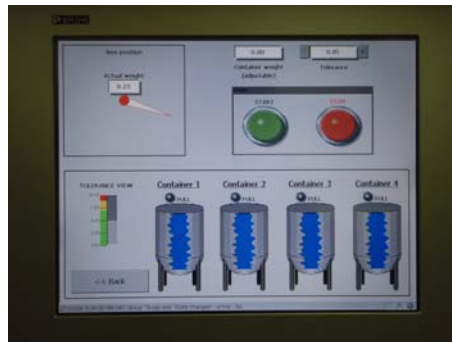
- [1] Němec, Z.: *Vyhodnocování hmotnosti předmětu z dynamiky pohybu otáčivého ramene. Návrh metody a výsledky předběžných zkoušek*. Pracovní materiál FSI VUT v Brně ze dne 3.3.2007.
- [2] Šmejkal, L.-Martinásková, M.: *PLC a automatizace*. Praha: BEN, 1999.
- [3] Švarc, I.: *Automatizace: Automatické řízení*. Brno: VUT Brno, 2005.
- [4] *um_en_ilc_330_350_6959_en_05.pdf* (20.9.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=830086&name=um_en_ilc_330_350_6959_en_05.pdf&UID=2737203&prodid=ILC%20350%20ETH&tab=1&qprodid=ILC%20350%20ETH&lang=en&f=me_doku/trans/english/5300/um/6959_en_05.pdf
- [5] *ah_en_ilc_350_eth_update_4_6f_7096_en_00.pdf* (25.9.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=604097&name=ah_en_ilc_350_eth_update_4_6f_7096_en_00.pdf&find.y=0&UID=7096&find.x=0&prodid=7096_en_00&tab=1&qprodid=7096_en_00&lang=en&f=me_doku/trans/english/5300/ah/7096_en_00.pdf
- [6] *ah_en_fb_tcp_ip_communication_6982_en_03.pdf* (20.12.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=758388&name=ah_en_fb_tcp_ip_communication_6982_en_03.pdf&UID=6982&prodid=6982_en_03&tab=1&qprodid=6982_en_03&lang=en&f=me_doku/trans/english/5300/ah/6982_en_03.pdf
- [7] *ibs_sys_fw_g4_um_e_5150d_e.pdf* (20.11.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=536907&name=ibs_sys_fw_g4_um_e_5150d_e.pdf&UID=2745185&prodid=5150D&tab=1&qprodid=5150D&lang=en&f=me_doku/trans/english/5300/um/5150d_e.pdf
- [8] *ibs_sys_pcp_g4_um_e_5334b_e.pdf* (20.11.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=536901&name=ibs_sys_pcp_g4_um_e_5334b_e.pdf&UID=2745169&prodid=5334B&tab=1&qprodid=5334B&lang=en&f=me_doku/trans/english/5300/um/5334b_e.pdf
- [9] *um_qs_en_pc_worx_7127_en_02.pdf* (18.8.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=846201&name=um_qs_en_pc_worx_7127_en_02.pdf&UID=2699862&prodid=7127_en_02&tab=1&qprodid=7127_en_02&lang=en&f=me_doku/trans/english/5300/um/7127_en_02.pdf

- [10] *db_en_ib_il_dc_ar_48_10a_2mbd_pac_6455_en_02.pdf* (8.9.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=903390&name=db_en_ib_il_dc_ar_48_10a_2mbd_pac_6455_en_02.pdf&find.y=16&UID=2897677&find.x=3&prodid=IB%20IL%20DC%20AR%2048%2f10A&tab=1&qprodid=IB%20IL%20DC%20AR%2048%2f10A&lang=en&f=me_doku/trans/english/5300/db/6455_en_02.pdf
- [11] *um_en_ib_il_dc_ar_48_10a_6949_en_00.pdf* (8.9.2007) Dostupné z:
http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=505774&name=um_en_ib_il_dc_ar_48_10a_6949_en_00.pdf&find.y=16&UID=2897677&find.x=3&prodid=IB%20IL%20DC%20AR%2048%2f10A&tab=1&qprodid=IB%20IL%20DC%20AR%2048%2f10A&lang=en&f=me_doku/trans/english/5300/um/6949_en_00.pdf
- [12] *2819286 IB IL DC AR 48/10A* [online] (10.10.2007) Dostupné z:
<http://eshop.phoenixcontact.com>
- [13] *TR-Electronic GmbH* [online] (10.10.2007) Dostupné z:
<http://www.tr-electronic.de/trgroup/index.html>
- [14] Němec, Z.: *XPlore BUT Brno.pdf*: „XPLORE NEW AUTOMATION AWARD 2008“ COMPETITION. Brno: VUT v Brně, 2008.

Seznam příloh

- Příloha 1: Fotografie celého zařízení
- Příloha 2: Výpis programu v prostředí PC WorX
- Příloha 3: Zjednodušený překlad manuálu pro servozesilovač IB IL DC AR 48/10A
- Příloha 4: CD, které obsahuje:
- Pour_Řízení ramene motoru a určení hmotnosti předmětu z dynamiky pohybu_2008.pdf (elektronická podoba diplomové práce)
 - Popisný soubor závěrečné práce (metadata)
 - Přílohy
 - o Dynamická sestava ramene (vytvořeno v programu Autodesk Inventor)
 - o Navržená mechanická konstrukce (vytvořeno v programu Autodesk Inventor)
 - o Program řízení ramene motoru pro PLC 350ETH (vytvořeno v programu PC WorX)
 - o Simulace otáčení ramene.avi (video soubor)

Příloha 1



Příloha 2

INICIALIZACE

```

sPartnerETH := '/IP=192.168.0.11';
sPartnerPCP := 'D1.0.1';

(* ----- Nastavení parametru regulátoru polohy ----- *)

rProportPosition := 4.5;          (*proporc*)
rIntegrPosition := 3.0;           (*integr*)
rDerivPosition := 0.35;          (*derivacni*)
rDerivFiltPosition := 0.06;      (*derivacni filtracni*)

rSamplingTPos := 0.02;

(* ---- Vypocet konstant regulátoru ---- *)
rCi := rProportPosition * rSamplingTPos / rIntegrPosition;
rCd1 := rDerivFiltPosition / (rSamplingTPos + rDerivPosition);
rCd2 := rDerivPosition / (rSamplingTPos + rDerivPosition);

(* ----- Nastavení parametru regulátoru rychlosti ----- *)
rProportSpeed := 3000.0;          (*proporc*)
rIntegrSpeed := 0.0;              (*integr*)

(* ----- Nastavení parametru regulátoru rychlosti ----- *)
rProportCurrent := 800.0;         (*proporc*)

(* ----- Maximalní proud pro vážení ----- *)
rMaxCurrent := 900.0;

(* ---- Definování konstant --- *)
rK := 40.0;
rCurrentFriction := 0.5;
rResidualWeight := 150.0;

rTime := 0.01;

(* ---- Definování poloh ramene ---- *)
aPosition[1] := 0.6;
aPosition[2] := 0.65;
aPosition[3] := 0.8;              (*poloha konce vážení*)
aPosition[4] := 0.9;
aPosition[5] := 0.1;             (*poloha počátku vážení*)
rDelta := 0.09;

rSwitchingPos23 := aPosition[5] + (aPosition[3] - aPosition[5])/2.0 * (1.0 - rCurrentFriction /
rMaxCurrent)*(1.0 - rDelta / 2.0);
rSwitchingPos31 := aPosition[3] - (aPosition[3] - aPosition[5]) * 2.0 * rDelta;

xInitialization := TRUE;

```

MAIN

```
(* ----- provedeni Inicializace = nastaveni konstant ----- *)
IF Inicialization_1.xInicialization = FALSE
    THEN Inicialization_1();
    xInicialization_DONE := Inicialization_1.xInicialization;
END_IF;

(* ----- PCP ----- *)

(* ----- navazani PCP komunikace se servozesilovacem ----- *)
IF (Inicialization_1.xInicialization = TRUE)&(PCP_connection_1.xPCPConnect_valid =
FALSE)
    THEN PCP_connection_1(sPartner := sPartnerPCP);
END_IF;

(* ----- nastaveni parametru servozesilovace pomoci PCP komunikace ---- *)
IF (PCP_connection_1.xPCPConnect_valid =
TRUE)&(Servoamplifier_settings_1.xServoSettings_DONE = FALSE)
    THEN Servoamplifier_settings_1();
END_IF;

(* ----- Servoamplifier enable ----- *)
IF (Servoamplifier_settings_1.xServoSettings_DONE =
TRUE)&(Servoamplifier_Enable_1.xServoamplifier_ENABLED = FALSE)
    THEN Servoamplifier_Enable_1();
END_IF;

(* ----- ETH ----- *)
(* ----- navazani ETH komunikace ----- *)
IF (Servoamplifier_Enable_1.xServoamplifier_ENABLED =
TRUE)&(IP_CONNECT_1.VALID = FALSE)
    THEN IP_CONNECT_1(EN_C:=TRUE,PARTNER:=sPartnerETH);
    xETHConnection_VALID := IP_CONNECT_1.VALID;
    iETH_ID := IP_CONNECT_1.ID;
END_IF;

(* ----- cteni dat z ETH ----- *)
IP_URCV_1(EN_R:=TRUE,ID:=iETH_ID,RD_1:=stReceivedData);
stReceivedData := IP_URCV_1.RD_1;
xETH_received := IP_URCV_1.NDR;

IF xTableAvailability = TRUE
    THEN Calculation_1();
END_IF;
```

CALCULATION

```
IF xCalculate_DONE = FALSE
```

```
  THEN
```

```
    IF xPassFirst = TRUE
```

```
      THEN iRowsAcceleration := iRowAcceleration - 1;
```

```
           iRowsDeceleration := iRowDeceleration - 1;
```

```
           rSumCurrentAcceleration := 0.0;
```

```
           rSumCurrentDeceleration := 0.0;
```

```
           xPassFirst := FALSE;
```

```
    END_IF;
```

```
(* --- vypocet stredniho proudu --- *)
```

```
FOR i := 3 TO iRowsAcceleration DO
```

```
  rSumCurrentAcceleration := rSumCurrentAcceleration + arRecordAcceleration[2][i];
```

```
END_FOR;
```

```
FOR j := 3 TO iRowsDeceleration DO
```

```
  rSumCurrentDeceleration := rSumCurrentDeceleration + arRecordDeceleration[2][j];
```

```
END_FOR;
```

```
rAverageCurrentAcceleration := rSumCurrentAcceleration /
```

```
INT_TO_REAL(iRowsAcceleration - 3);
```

```
rAverageCurrentDeceleration := rSumCurrentDeceleration /
```

```
INT_TO_REAL(iRowsDeceleration - 3);
```

```
rDeltaSpeedAcceleration := (arRecordAcceleration[1][iRowsAcceleration] -
```

```
arRecordAcceleration[1][3]) / (rTime * INT_TO_REAL(iRowsAcceleration - 3));
```

```
rDeltaSpeedDeceleration := (arRecordDeceleration[1][iRowsDeceleration] -
```

```
arRecordDeceleration[1][3]) / (rTime * INT_TO_REAL(iRowsDeceleration - 3));
```

```
(* --- vypocet hmotnosti --- *)
```

```
rWeightAcceleration := rK * (rAverageCurrentAcceleration - rCurrentFriction) * rTime /
```

```
rDeltaSpeedAcceleration - rResidualWeight;
```

```
rWeightDeceleration := rK * (ABS(rAverageCurrentDeceleration) + rCurrentFriction) * rTime /
```

```
rDeltaSpeedDeceleration - rResidualWeight;
```

```
rWeight := (rWeightAcceleration + rWeightDeceleration) / 2.0;
```

```
xCalculate_DONE := TRUE;
```

```
xTableAvailability := FALSE;
```

```
iRowAcceleration := 1;
```

```
iRowDeceleration := 1;
```

```
xMeasuring := FALSE;
```

```
END_IF;
```

POSITION CONTROLLER

```
IF rRequiredPosition = 0.0
    THEN rRequiredPosition := rActualPosition;
END_IF;

rDeviation_1 := rDeviation;
rDeviation := rRequiredPosition - rActualPosition;

(* ----- REGULATOR POLOHY ----- *)

IF (ABS(rDeviation) < 0.05)
    THEN
        (* --- integracni slozka --- *)
        rUi_1 := rUi;

        IF iRegulationMod <> 1
            THEN rUi := 0.0;
            ELSE rUi := rUi_1 + rCi * rDeviation;
        END_IF;

        IF rUi > 0.3
            THEN rUi := 0.3;
        END_IF;

        IF rUi < -0.3
            THEN rUi := -0.3;
        END_IF;

        IF ABS(rDeviation) < 0.001
            THEN rDeviation := 0.0;
        END_IF;

        (* --- proporcionalni slozka --- *)
        rUp := rProportPosition * rDeviation;

        (* --- derivacni slozka --- *)
        rUd_1 := rUd;
        rUd := rCd1 * rUd_1 + rCd2 * (rDeviation - rDeviation_1);

        (* --- akcni velicina --- *)
        rManipulatedVar_PosCo := rUp + rUi + rUd;
    ELSE
        rUp := 3.0 * rDeviation;
        rManipulatedVar_PosCo := rUp;
    END_IF;
```

SPEED CONTROLLER

(* ---- precteni aktualniho natoceni ze snimace polohy ----- *)

```
wPosition := ONBOARD_INPUT;
wPosition:=ROL_WORD(wPosition,2);
wPosition.X0 := wPosition.X10;
wPosition.X1 := wPosition.X11;
wPosition := wPosition & WORD#16#3FF;
```

```
iActualPosition := WORD_TO_INT(wPosition);
rActualPosition := WORD_TO_REAL(wPosition)/1024.0;
```

(* ---- nastaveni zadane hodnoty podle prijateho pokynu z ETH komunikace ---- *)

```
IF stReceivedData.wCommand = WORD#02
```

```
  THEN
```

```
    CASE WORD_TO_INT(stReceivedData.wValue) OF
```

```
      1      : rRequiredPosition := aPosition[1]; xCalculate_DONE :=
FALSE; xWeightEnable := FALSE; xSend_DONE := FALSE;
      2      : rRequiredPosition := aPosition[2]; xCalculate_DONE :=
FALSE; xWeightEnable := FALSE; xSend_DONE := FALSE;
      3      : rRequiredPosition := aPosition[3]; xCalculate_DONE :=
FALSE; xWeightEnable := FALSE; xSend_DONE := FALSE;
      4      : rRequiredPosition := aPosition[4]; xCalculate_DONE :=
FALSE; xWeightEnable := FALSE; xSend_DONE := FALSE;
      5      : rRequiredPosition := aPosition[5]; xCalculate_DONE :=
FALSE; xWeightEnable := FALSE; xSend_DONE := FALSE;
    END_CASE;
```

```
  END_IF;
```

```
IF (stReceivedData.wCommand = WORD#01)&(stReceivedData.wValue = WORD#00)
```

```
  THEN xWeightEnable := TRUE; xSend_DONE := FALSE;
```

```
  ELSE xWeightEnable := FALSE; xSend_DONE := FALSE; xCalculate_DONE :=
  FALSE;
```

```
END_IF;
```

(* osetreni stavu po zapnuti automatu -> akcni velicina reg. rychlosti bude nulova *)

```
IF (rRequiredPosition = 0.0)&(xInicialization_DONE = FALSE)
```

```
  THEN rManipulatedVar_SpeedCo := 0.0;
```

```
END_IF;
```

```
IF xInicialization_DONE = TRUE
```

```
  THEN
```

```
    (* ---- vypocet rozdilu dvou sousednich poloh ----- *)
```

```
    iPositionDifference := iActualPosition - iActualPosition_1;
```

```
    iActualPosition_1 := iActualPosition;
```

```
    (* ---- osetreni prechodu snimace natoceni ----- *)
```

```
    IF iPositionDifference < -512 THEN iPositionDifference := iPositionDifference +
    1024; END_IF;
```

```
    IF iPositionDifference > 512 THEN iPositionDifference := iPositionDifference -
    1024; END_IF;
```

```

(* ---- vypocet rychlosti ----- *)
rActualSpeed := (INT_TO_REAL(iPositionDifference)/1024.0) / rTime;

(* ----- vypocet akcni veliciny ----- *)
rDeviation := rManipulatedVar_PosCo - rActualSpeed;
rManipulatedVar_SpeedCo := rProportSpeed * rDeviation;

(* ----- omezeni akcni veliciny reg. rychlosti ----- *)
IF rManipulatedVar_SpeedCo > 1200.0
  THEN rManipulatedVar_SpeedCo := 1200.0;
END_IF;

IF rManipulatedVar_SpeedCo < -1200.0
  THEN rManipulatedVar_SpeedCo := -1200.0;
END_IF;

(* povoleni tabulky, pokud probihalo mereni a je standardni mod regulace *)
IF (xMeasuring = TRUE)&(iRegulationMod = 1)
  THEN xTableAvailability := TRUE;
       xPassFirst := TRUE;
END_IF;
END_IF;

IF xWeightEnable = TRUE
  THEN rRequiredPosition := aPosition[3];
       IF (rActualPosition < rSwitchingPos23)&(xWeightEnable =
          TRUE)&(xCalculate_DONE = FALSE)
         THEN iRegulationMod := 2;  (*zrychleni*)

arRecordAcceleration[1][iRowAcceleration] := rActualPosition;
arRecordAcceleration[2][iRowAcceleration] :=
WORD_TO_REAL(wActualValue);
iRowAcceleration := iRowAcceleration + 1;
wSetPoint := REAL_TO_WORD(rMaxCurrent);
xMeasuring := TRUE;

ELSE
  IF (rRequiredPosition = aPosition[3])&(rActualPosition >
     rSwitchingPos23)&(rActualPosition < rSwitchingPos31)
    THEN iRegulationMod := 3;  (*zrychleni*)
         arRecordDeceleration[1][iRowDeceleration] :=
            rActualPosition;
         arRecordDeceleration[2][iRowDeceleration] :=
            WORD_TO_REAL(wActualValue);
         iRowDeceleration := iRowDeceleration + 1;
         xMeasuring := TRUE;
         xCalculate_DONE := FALSE;
         wSetPoint := REAL_TO_WORD(- rMaxCurrent);
    ELSE iRegulationMod := 1;  (* mod standard *)
         wSetPoint := REAL_TO_WORD(rManipulatedVar_SpeedCo);
    END_IF;
  END_IF;
ELSE wSetPoint := REAL_TO_WORD(rManipulatedVar_SpeedCo);
     iRegulationMod := 1;

```


END_IF;

```

IF (stReceivedData.wCommand = WORD#01)&(stReceivedData.wValue =
WORD#00)&(xCalculate_DONE = TRUE)
    THEN stSendData.wCommand := WORD#01;
        IF rWeight < 0.0 THEN rWeight := 0.0; END_IF;
        stSendData.wValue := REAL_TO_WORD(rWeight);
        stReceivedData.wCommand := WORD#0;
        stReceivedData.wValue := WORD#0;
        xSendEnable := TRUE;
    ELSIF (ABS(rManipulatedVar_SpeedCo) < 78.0)&(xSend_DONE =
FALSE)&(rRequiredPosition <> 0.0)&(stReceivedData.wCommand =
WORD#02)&(stReceivedData.wValue > WORD#00)&(stReceivedData.wValue < WORD#06)
    THEN IF (rActualSpeed = 0.0)&(rActualPosition > rRequiredPosition -
0.002)&(rActualPosition < rRequiredPosition + 0.002)
        THEN xSendEnable := TRUE;
            stSendData := stReceivedData;
            iPass := 0;
            stReceivedData.wCommand := WORD#0;
            stReceivedData.wValue := WORD#0;
        ELSE iPass := iPass + 1;
            xSendEnable := FALSE;
        END_IF;
    END_IF;
END_IF;

```

```

IP_USEND_1(REQ:=xSendEnable,ID:=iETH_ID,SD_1:=stSendData);
xSend_DONE:=IP_USEND_1.DONE;
stSendData:=IP_USEND_1.SD_1;
xSendEnable := FALSE;

```

Kompletní funkční verze programu je uložena na CD, které je Přílohou 4 této diplomové práce. Program byl vytvořen na PC s nainstalovaným operačním systémem Windows XP a aplikací PC WorX 5.00. Pro spuštění programu je třeba mít k dispozici programovatelný automat společnosti Phoenix Contact ILC 350ETH, který je doplněn servozesilovačem IB IL DC AR 48/10A.

Příloha 3

Servozesilovač

(IB IL DC AR 48/10A)

- **aplikace:**
 - o momentová/rychlostní regulace
 - o DC motory s permanentními magnety
 - o nominální napětí 12 V až 48 V DC
 - o výkon do 450 W
 - o proud motoru do 10 A
- možnost použití jako samostatný, nebo pro řízení ve více osách (ve spojení s IB IL POS 200)
- **servozesilovač umožňuje:**
 - o regulace rychlosti s IxR vyrovnáním (napěťové řízení)
 - o regulace rychlosti bez IxR vyrovnání
 - o regulace momentu (regulace proudu)
- **má bezpečnostní prostředky zajišťující prevenci proti:**
 - o překročení proudu
 - o překročení teploty
 - o proudu nakrátko v důsledku zkratování žil kabelu napájení motoru
 - o proudu nakrátko v důsledku zkratování žil kabelu napájecího zdroje
- napěťový zdroj 12 V až 48 V DC, 0 A až 10 A napojen na U_S

Regulace rychlosti bez IxR kompenzace (napěťová regulace)

- pracuje jako regulátor rychlosti bez zpětné vazby (tachogenerátor)
- využívá změny rychlosti otáčení motoru v důsledku změny napětí na jeho svorkách
- Servo Amplifier reguluje pouze napětí (kladné, záporné)
- deaktivace IxR kompenzace ("IxRCompensation" parameter = 0, index 010B_{hex})
- porovnává aktuální napětí na motoru s hodnotou napětí, které odpovídá požadované rychlosti otáčení
- napětí na zdroji musí být přibližně o 10% větší než napětí, které má jít na motor

Regulace rychlosti s IxR kompenzací

- rychlost motoru se mění v závislosti na měnícím se zatížení při použití napětí motoru pro jeho regulaci
- rychlost klesá se vzrůstajícím zatížením při konstantním napětí, protože odpor motoru nedovolí zvyšování proudu v závislosti na zvyšující se zátěži
- celková přesnost je limitována v důsledku nepřímé regulace rychlosti
- rozsah, ve kterém kolísá rychlost, což je způsobeno výkyvy zatížení mohou být eliminovány v závislosti na dynamické odezvě kolísání zátěže a struktuře motoru
- obecně může být kolísání zátěže redukováno až na 90% aktivací IxR kompenzace

Regulace momentu (proudová regulace)

- tuto funkci využívá k samostatnému nastavení momentu motoru na požadovanou hodnotu
- zajišťuje pomocí regulace proudu, pro kterou Servo poskytuje zpětnou vazbu měřením proudu do motoru

- používá se, pokud je požadována konstantní síla
- funkce: pokud mechanismus vykazuje menší moment, nežli je moment, se kterým Servo pracuje, stoupá rychlost otáčení motoru (je limitována napájecím napětím), pokud je moment mechanismu větší, rychlost klesá

Metody práce výstupního stupně

- Servo generuje požadované napětí nebo požadovaný proud motoru (pro regulaci momentu) s využitím PWM (Pulse Wide Modulation)
- výkon jen spínaný, průměrná hodnota napětí je řízena délkou pulzu
- pulsy jsou generovány frekvencí 20kHz

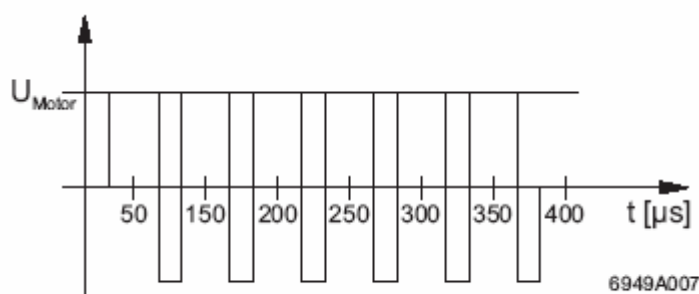


Figure 1-3 Pulse wide modulation (PWM)

Módy práce výstupního stupně

- pracuje s bipolárním PWM signálem
- větší efektivita se dosahuje při použití požadavků vyššího proudu

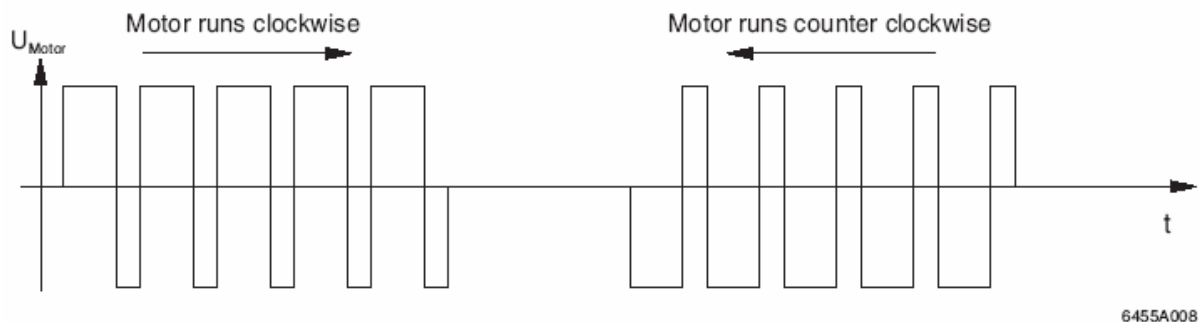
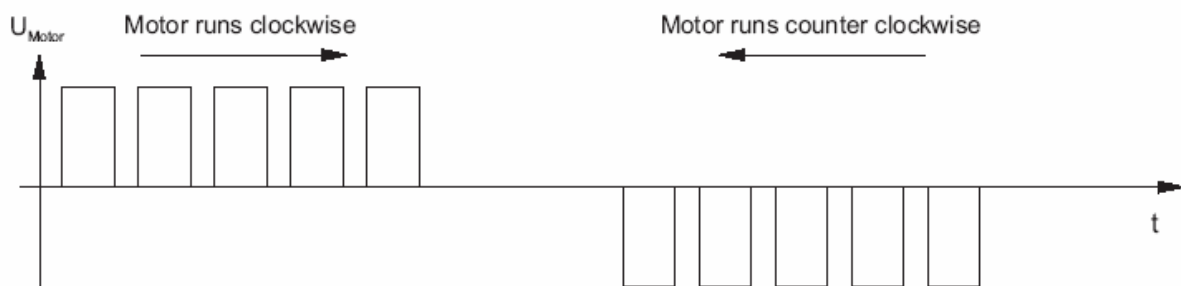


Figure 1-4 4T mode (4 transistor mode)

- použití motoru s malou induktancí a požadavkem na malý proud může vést k nepříjemnému zvyšování teploty motoru při nízké rychlosti otáčení
- v tomto případě může být motor přepnut do 2T módu, kde je výstup řízen unipolárně – pouze pozitivní nebo negativní pulsy jsou vysílány do motoru v závislosti na požadovaném směru otáčení

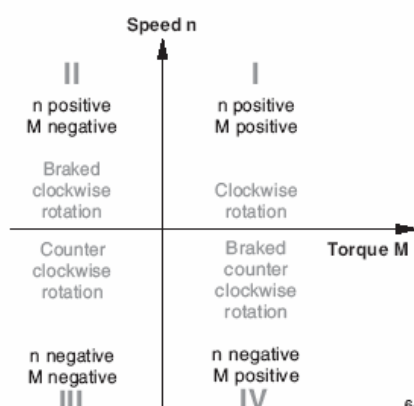


6455A009

Figure 1-5 2T mode (2 transistor mode)

4 kvadrantový mód

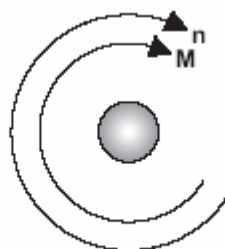
- název odvozen od možných pracovních stavů motoru při rychlostní/momentové regulaci



6455A010

Figure 1-6 Speed/torque/coordinate system

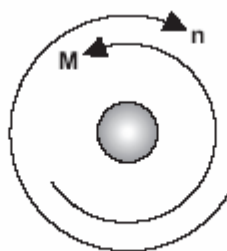
- counter a rotace ve směru hodinových ručiček, moment je ve směru rychlosti – motor běží (kvadrant I a III)
- zastavený counter a rotace ve směru hodinových ručiček, moment působí proti směru rychlosti → motor brzdí
- *kvadrant I*: rotace ve směru hodinových ručiček, motor působí momentem ve směru rotace



6455A011

Figure 1-7 Clockwise rotation

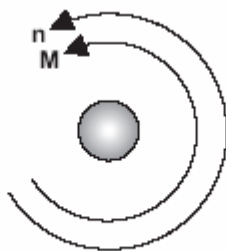
- *kvadrant II*: brzdění rotace ve směru hodinových ručiček, moment působí proti směru pohybu, motor brzdí



6455A012

Figure 1-8 Braked clockwise rotation

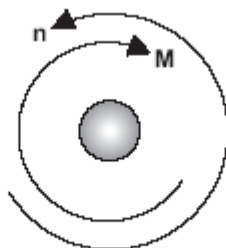
- *kvadrant III*: counter proti směru hodinových ručiček – moment působí ve směru rotace, opak kvadrantu I



6455A013

Figure 1-9 Counter clockwise rotation

- *kvadrant IV*: brzdění counteru v rotaci proti směru hodinových ručiček – moment působí proti směru rotace counteru



6455A014

Figure 1-10 Braked counter clockwise rotation

- při brzdění motor uvolňuje kinetickou energii, ve 4 kvadrantovém módu je tato energie dodávána zpět do zdroje jako elektrická energie; pokud není více zařízení zapojených na tento zdroj, může napětí narůst do hodnot, které mohou poškodit elektrický obvod nebo trigger
- Servo i zdroj musí odolat většímu výkonu nežli je maximum, které zdroj dodává
- Servo má funkci sledování napětí, která vypne motor, pokud napětí překročí svůj práh

Funkce regulátoru v zařízení

řízení rychlosti – 2 regulace v kaskádě

- regulátor rychlosti
- regulátor proudu

řízení momentu – v činnosti pouze regulace proudu

- oba regulátory jsou PI, nemají derivační složku
- parametry regulátorů jsou přednastaveny tak, že ve většině úloh pracují správně

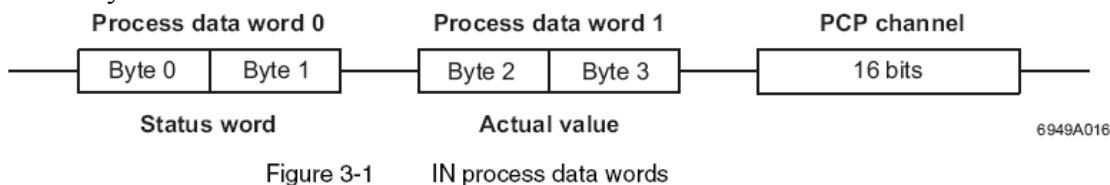
Instalace servozesilovače

- interbus drivecom profil 22
- 2 funkční skupiny:
 - rychlosti
 - momentu

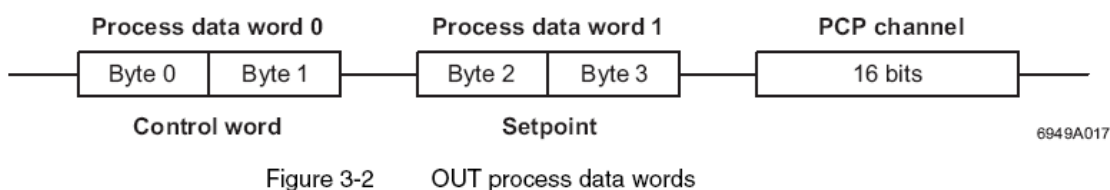
- parametr nastavení funkční skupiny je "ModeSelectionCode" (index 6060hex)
- defoltně nastaven mód rychlosti
- hodnota rychlosti se nastavuje v otáčkách za minutu pomocí process data word 1 (popřípadě procentně, což může být konvertováno na požadovanou hodnotu podle referenční hodnoty rychlosti)
- jakmile je nastavena rychlost, dochází k limitování rychlosti a zrychlení/zpoždění
- *požadované hodnoty motoru, které musí být zadány:*
 - o rychlost
 - o napětí
 - o proud
- napájecí napětí musí být přibližně o 10% větší, nežli napětí dodávané motoru
- maximální proud je 10 A, při připojení 3 A motoru může být odběr při rozběhu až 10 A, pokud je motor 10 A, startovací proud je pouze 10 A

Servo po připojení na sběrnici

- nelze nastavit odpor (resistance) ani další parametry
- je řízeno pomocí fast, cyclic process data, tato může být též použita pro provozní funkce:
 - o povolení
 - o povolení práce
 - o zakázání práce
 - o rychlý stop
- *pomocí tohoto kanálu lze nazpět číst aktuální data:*
 - o rychlost
 - o proud
 - o stav zařízení
- *s vyššími stupni komunikuje pomocí:*
 - o lokální sběrnice
 - o fast, cyclic process data channel
 - o acyclic parameter channel (PCP, **P**eripherals **C**ommunication **P**rotocol)
- process data jsou čtena cyklicky, přenos pomocí process data kanálu
- drive parameters (parametr data) mohou být čtena a zapisována acyklicky pomocí PCP service, tato permanentně neuchovává parametry v Servu
- v kruhu sběrnice využívá 1 slovo pro PCP kanál a 2 process data word pro každý datový směr



The contents of the IN process data words depend on the "INProcessDataDescription" parameter (index 6000_{hex}).



- control word – pro vzdálené řízení mezi jednotlivými stavy
- status word – pro čtení aktuálního pracovního stavu, další bity indikují:
 - o varování, chyba
 - o rychlostní, proudový limit je aktivován
 - o dosažení nastaveného bodu
- *PCP kanál* – parametrová data se mění je zřídka, jejich přenos je pře PCP kanál, index určuje, který parametr je adresován, přístupový atribut určuje, zda se bude číst nebo zapisovat

Řízení pomocí Control word/status word

- o Control word je použito pro změnu pracovního stavu pomocí sběrnice
- o další stavy mohou být dosaženy jen v daných sekvencích
- o jakmile se zapne, musí projít přes "Not ready to operate", "Start inhibit", "Ready to operate", a "ON" stavy, aby nastal "Operation enabled" stav
- o další pracovní stavy jsou dosaženy nastavením odpovídajících bitů control word
- o servo indikuje svůj pracovní stav v control word

Význam Process data word

- *IN process data words*
 - o IN process data word 0 – může být použito pro zobrazení různých parametrů, „INProcessDataDescription“ parametr (index 6000_{hex}) specifikuje, která data jsou zobrazována, Status word je zobrazeno standardně ("StatusWord" parameter, index 6041_{hex})
 - o IN process data word 1 – může být použito pro zobrazení různých parametrů, „INProcessDataDescription“ parametr (index 6000_{hex}) specifikuje, která data jsou zobrazována, aktuální rychlost je zobrazena standardně ("SpeedActualValue" parameter, index 6044_{hex})
- *OUT Process data words*
 - o OUT process data word 0 – může být použito pro vysílání různých parametrů, „OUTProcessDataDescription“ parametr (index 6001_{hex}) specifikuje, která data jsou vysílána, Control word je vysíláno standardně ("ControlWord" parameter, index 6040_{hex})
 - o OUT process data word 1 – může být použito pro vysílání různých parametrů, „OUTProcessDataDescription“ parametr (index 6001_{hex}) specifikuje, která data jsou vysílána, nastavení rychlosti je vysíláno standardně ("SpeedSetpoint" parameter, index 6042_{hex})

Parametrizace a čtení informací pomocí PCP

- všechny parametry pro jednotlivé funkce jsou zapisovány a čteny pomocí PCP kanálu podle daných pravidel
- parametry, které byly specifikovány pomocí OUT process data word nesmí být zapisována pomocí PCP kanálu
- parametrová data, informace na žádost jsou požadována jen zřídka, proto se pro jejich přenos využívá PCP kanál, tato data jsou přenášena jen na žádost
- data přenášena přes PCP jsou adresována pomocí indexu
- pro každý parametr „access atribut“ specifikuje, zda se bude parametr číst nebo zapisovat

- pro snížení počtu PCP zpráv na nastavení serva, které musí být posílány pomocí vyššího stupně řízení, je množství důležitých parametrů seskupeno do „ParametrGroup1“ (index E000_{hex})

Nastavení serva pomocí PCP kanálu

- *PMS Interface*
 - o servo obsahuje standardní PMS rozhraní
 - o tento komunikační kanál umožňuje plný přístup do všech řídicích parametrů serva
 - o komunikační protokol je založen na PCP komunikaci
- *objektový slovník OD*
 - o pro rozeznání mezi jednotlivými parametry během komunikace, má každý své unikátní číslo (index)
 - o index je uchováván spolu s popisem znaku parametru ve standardizovaném seznamu – object dictionary
 - o každé PCP zařízení, které vyměňuje informace pomocí datového kanálu má vlastní objektový slovník
 - o objektový slovník není implementován do Servo zesilovače
- *PMS služby*
 - o servo podporuje několik PMS služeb
 - o pouze následující jsou důležité pro nastavení serva zesilovače:
 - Initiate (connect)
 - Read
 - Write
 - Abort (disconnect)
 - o Další informace lze nalézt v "Peripherals Communications Protocol (PCP)" User Manual IBS SYS PCP G4 UM E, Order-No. 27 45 16 9
- *Zahájení PCP služby*
 - o zahajovací PMS služba může být použita pro komunikační spojení mezi ovladačem sběrnice a Servo zesilovačem
 - o ovladač sběrnice zahajuje spojení
 - o pokud je spojení navázáno úspěšně, Servo zesilovač odpoví na dotaz
- *PCP služba čtení*
 - o služba čtení umožňuje, ovladači sběrnice s právem pro čtení, čtení všech řídicích parametrů Serva
 - o všechny parametry a jejich popis jsou uvedeny na straně A-1 manuálu
- *PCP služba zápis*
 - o služba zápisu umožňuje, ovladači sběrnice s právem zápisu, zápis všech řídicích parametrů Serva, která mohou být zapsána
 - o v případě, že dojde k neoprávněnému přístupu do parametrů řízení, Servo generuje chybu zápisu s detailními informacemi
- *PCP služba zastavení komunikace*
 - o může být použita pro ukončení stávajícího komunikačního spojení
 - o ukončení je nepotvrzovaná služba a může být vyvolána kterýmkoliv z komunikujících členů

Structure of the Speed Function

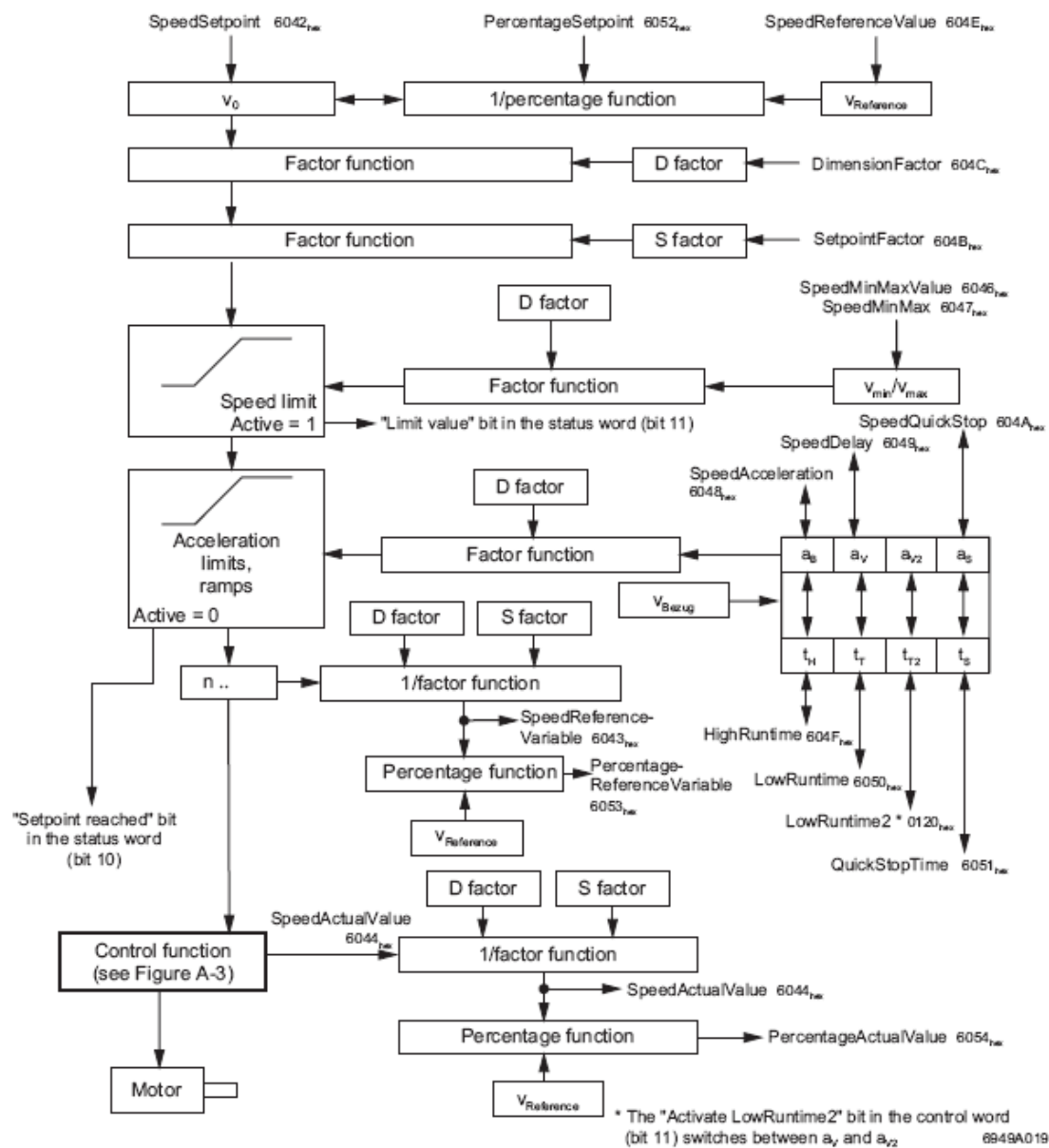


Figure A-1 Speed function

Structure of the Control Function

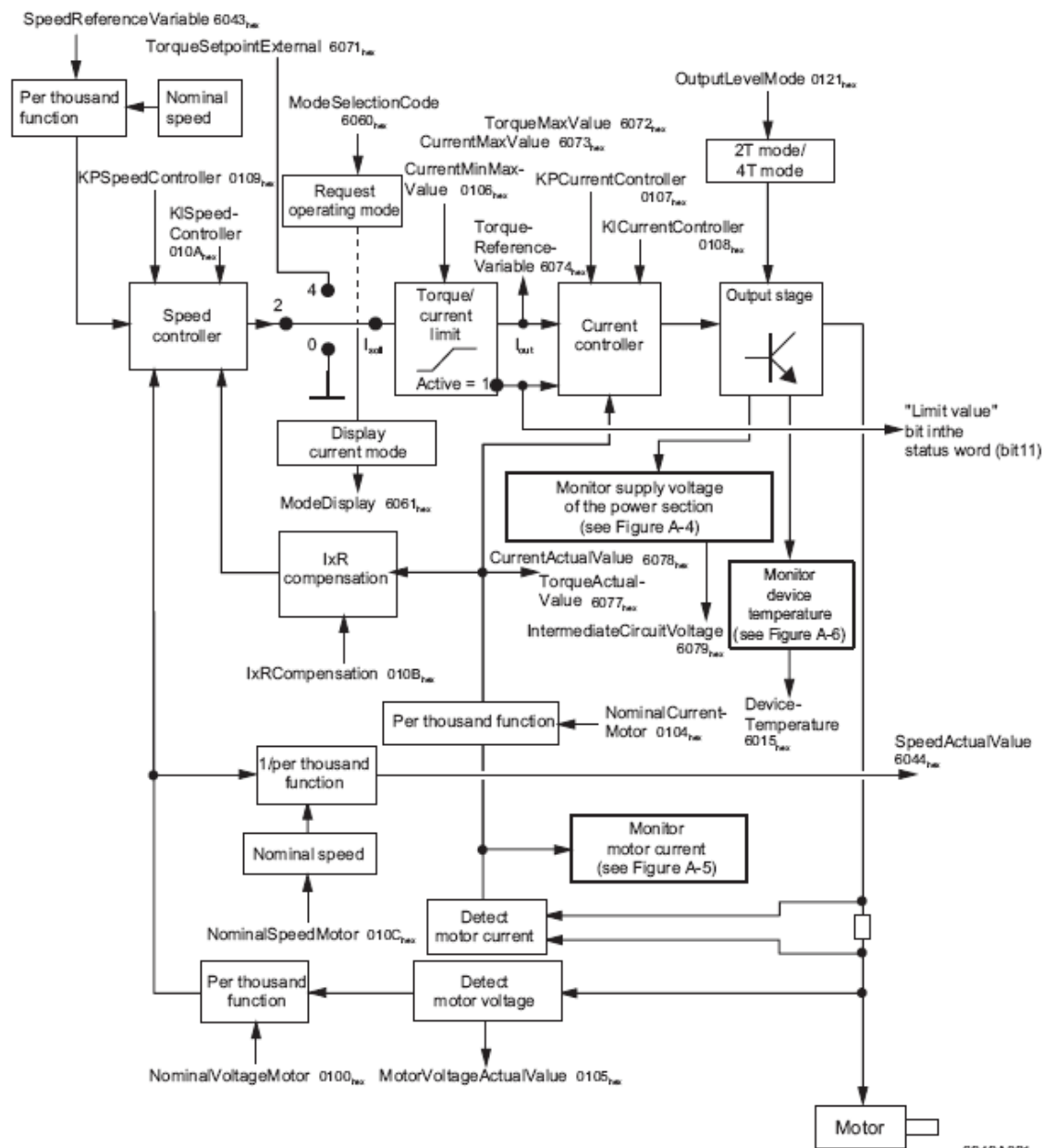


Figure A-3 Control function

Tento dokument je pouze orientační překlad manuálu

um_en_ib_il_dc_ar_48_10a_6949_en_00.pdf (8.9.2007), Dostupného z:

http://select.phoenixcontact.com/phoenix/dwl/dwl13a.jsp?fct=dwl&asid=505774&name=um_en_ib_il_dc_ar_48_10a_6949_en_00.pdf&find.y=16&UID=2897677&find.x=3&prodid=IB%20IL%20DC%20AR%2048%2f10A&tab=1&qprodid=IB%20IL%20DC%20AR%2048%2f10A&lang=en&f=me_doku/trans/english/5300/um/6949_en_00.pdf