SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE

Fakulta chemickej a potravinárskej technológie

Ústav informatizácie, automatizácie a matematiky
Oddelenie informatizácie a riadenia procesov

BAKALÁRSKY PROJEKT

Riadenie pomocou priemyselného riadiaceho systému SIMATIC

Vypracoval: **Alexander Szűcs**

Vedúca bakalárskej práce: **Ing. Katarína Vaneková**

B r a t i s l a v a 2008

Slovenská technická univerzita v Bratislave
Oddelenie informatizácie a riadenia procesov

Fakulta chemickej a potravinárskej technológie
Akademický rok: 2007/2008

S T U
F C H P T

ZADANIE BAKALÁRSKEJ PRÁCE

Autor práce:	**Alexander Szücs (26488)**
Študijný program:	automatizácia, informatizácia a manažment v chémii a potravinárstve
Kombinácia študijných odborov:	5.2.14 automatizácia, 5.2.52 priemyselné inžinierstvo
Vedúca práce:	Ing. Katarína Vaneková
Miesto vypracovania:	Bratislava

Názov témy: **Riadenie pomocou priemyselného riadiaceho systému SIMATIC**

Špecifikácia zadania:

Využitie priemyselného riadiaceho systému Simatic S-7 300 na riadenie procesu vo forme elektrického modelu. Definovanie siete, konfigurácia vstupno-výstupných modulov. Programovanie blokov, vytvorenie programu v Step 7. Vytvorenie vizualizačného rozhrania a riadenie pomocou PID regulátora.

Rozsah práce: 40

Riešenie zadania práce od: 07. 03. 2008
Dátum odovzdania: 23. 05. 2008

Slovenská technická univerzita v Bratislave Fakulta chemickej a potravinárskej technológie dekanát - 5 -

Alexander Szücs
riešiteľ bakalárskej práce

prof. Ing. Dr. Miroslav Fikar
vedúci pracoviska

prof. Ing. Dr. Miroslav Fikar
garant študijného programu

Poďakovanie:

Rád by som poďakoval Ing. Kataríne Vanekovej za odborné vedenie a rady pri realizácii bakalárskej práce. Taktiež by som chcel poďakovať rodine za morálnu i hmotnú podporu počas štúdia.

Abstrakt

Bakalársky projekt sa zaoberá identifikáciou laboratórneho zariadenia predstavujúce elektronický model a návrhom PI regulátorov. Riadenie modelu sa uskutočnilo v programe SIMATIC a zber dát cez DDE rozhranie. Ide o rozhranie, ktoré vyvinula spoločnosť Microsoft na nadviazanie vzájomnej komunikácie medzi aplikáciami na báze operačného systému MS Windows. Práca tiež obsahuje vyhodnotenie regulátorov.

Abstract

The bachelor project deals with an identification of a laboratory model representing an electronic model and with proposition of the PI controllers. Control of the laboratory model was realized by programme SIMATIC and gathering of the data by DDE interface. It‘ s an interface, which was made by Microsoft company for making interactive communication among aplications, on the base of operating system MS Windows. The work contains evaluating of the controllers.

OBSAH

ÚVOD

Cieľom práce bolo navrhnúť riadenie pre laboratórne zariadenie predstavujúce elektronický model sústavy III. rádu. Práca sa zaoberá identifikáciou laboratórneho zariadenia v prostredí MATLAB, návrhom regulátorov pre identifikovaný laboratórny proces a riadením pomocou riadiaceho systému SIMATIC.

Práca sa pozostáva z niekoľkých častí. V prvej časti práce sa nachádza opis riadiaceho systému SIMATIC S7-300. V druhej časti nájdeme charakteristiky použitých programov. Tretia časť sa zaoberá samotným vytváraním programu v riadiacom systéme SIMATIC konfiguráciou I/O modulov a konfiguráciou premenných. V ďalšej časti práce nájdeme opis programov MATLAB a Microsoft Excel, ktoré sú aktívnymi súčasťami pri komunikácii cez DDE (Dynamic Data Exchange) rozhranie. Záverečná časť sa zaoberá samotnou identifikáciou systému, návrhmi regulátorov, otestovaním ich funkčnosti a následným spracovaním a vyhodnotením výsledkov riadenia laboratórneho zariadenia.

1. Riadiaci systém SIMATIC S7-300

Na riadenie laboratórneho zariadenia sústavy III. rádu bol použitý riadiaci systém **SIMATIC S7-300**, ktorý pozostáva zo štyroch hlavných častí, ktoré spolu tvoria kompaktný celok.

1. **Priemyselný Rack PC 830** patrí do strednej triedy priemyselných PC, postavených na báze technológie od Intelu, vhodný pre používanie v priemyselnom prostredí, ktoré znemožňuje použitie štandardných PC kvôli požiadavkám na odolnosť voči teplotám vibráciám a ďalším vplyvom.

2. Pracovná stanica SIMATIC S7-300 je univerzálny PLC schopný pôsobiť v širokom spektre aplikácií automatizačného inžinierstva s dôrazom na produkciu, ekonomické a praktické riešenia v rôznych odvetviach priemyslu.
Pracovná stanica obsahuje:

- Napájací modul PS 307 5A
- CPU 313 je centrálna procesorová jednotka s prostredím pre veľkú programovú pamäť s MPI, PROFIBUS interfacom
- Zásuvné vstupno – výstupné (I/O) moduly (maximálny počet I/O modulov na jednu pracovnú stanicu S7 – 300 je osem)

3. Prepojenie PC a pracovnej stanice SIMATIC pomocou MPI (Multi Point Interface) kábla, PROFIBUS, INDUSTRIAL ETHERNET, resp. PTP. Taktiež je možné kombinovať jednotlivé spôsoby a tým vytvoriť sieť, ktorá umožní prenos dát medzi jednotlivými pracovnými stanicami a PC.

4. STEP7 a WinCC sú softwarové súčasti riadiaceho systému, ktoré slúžia k programovaniu PLC, sprístupňujú dáta užívateľovi. Sú vhodné na monitoring a riadenie reálnych procesov [1].

2. Charakteristika použitých programov

2. 1. STEP7

STEP7 je program od firmy SIEMENS, ktorý sa používa na realizáciu relatívne rozsiahlych a komplexných aplikácií, kde sa vyžaduje pre programovanie, funkcie alebo komunikačné moduly použitie jazykov vyššej úrovne a jazykov grafických návrhov. Slúži na naprogramovanie riadiaceho algoritmu, ktorý je následne prenesený cez rozhranie (MPI, PROFIBUS, INDUSTRIAL ETHERNET, resp. PTP) do PLC.

2. 2. WinCC

Program **WinCC** slúži na vytvorenie vizualizačných okien, pomocou ktorých je možné bežiaci proces monitorovať a aktívne do neho zasahovať. Všetky dôležité dáta čerpá z programu **STEP7**, s ktorým je prepojený prostredníctvom tagov (dátové kanály). Počítač, na ktorom je nainštalovaný **WinCC** musí byť pripojený k PLC cez rozhrania MPI, PROFIBUS, INDUTRIAL ETHERNET, resp. PTP.

2. 3. Microsoft Excel

Aplikácia **Microsoft Excel** je obsiahnutá v kancelárskom balíku Microsoft Office. Slúži na vytváranie a správu tabuliek, na riešenie rôznych funkčných závislostí. Využíva sa takmer vo všetkých druhoch obchodných spoločností a vo všetkých odvetviach priemyslu.

2. 4. MATLAB

MATLAB je vysokovýkonný integrovaný prostriedok pre technické výpočty. Je charakterizovaný integráciou výpočtov, vizualizáciou a programovaním v jednoduchom užívateľskom prostredí, kde problémy a riešenia sú vyjadrené bežnými matematickými zápismi [2].

Zahŕňa nasledovné oblasti:

- Matematické výpočty
- Vývojové prostriedky pre tvorbu algoritmov
- Modelovanie a simulácia
- Vizualizácia a analýza dát
- Vysoko výkonná 2D a 3D grafika
- Aplikačné prostriedky pre vytváranie grafického užívateľského prostredia

3. Postup pri vytváraní projektu v programe STEP7

Pri vytváraní riadiaceho projektu, je dôležité prihliadať na jeho kompaktnosť, tak aby plnil požadované kritéria a ciele.

Pri tvorbe riadiaceho projektu treba splniť nasledovné kritéria:

- *Vytvorenie nového projektu v programe* **STEP7**, v ktorom sú uložené všetky funkcie a dáta potrebné k riadeniu.
- *Konfigurácia I/O modulov*, ktorá umožňuje prepojenie medzi pracovnou stanicou a reálnym procesom.
- *Konfigurácia siete*, ktorá umožňuje prepojenie PC s pracovnou stanicou **SIMATIC** pomocou zvoleného rozhrania.
- *Vytvorenie riadiaceho programu* pomocou programovacieho jazyku **STEP7**, ktorý umožňuje riadiaci program napísať pomocou klasických príkazov, programovaním funkčných blokov alebo kombináciou obidvoch uvedených typov. Vytvorený program po nakopírovaní do RAM pamäte CPU možno kedykoľvek diagnostikovať a logicky testovať aj bez priebehu reálneho procesu.
- *Vytvorenie vizualizačného prostredia* umožňuje užívateľovi definovať si vlastné vizualizačné prostredie pre daný proces pomocou programu **WinCC**, ktorého súčasťou je grafický editor. **WinCC** umožňuje rozmiestnenie jednotlivých ovládacích prvkov, vstupno – výstupných polí a monitorovacích okien na obrazovke užívateľa tak, aby mal neustály prehľad nad priebehom procesu a rýchlymi zásahmi ho mohol aktívne ovplyvňovať.
- *Prepojenie riadiaceho programu a vizualizačného prostredia.* Vizualizačné prostredie a riadiaci program je potrebné prepojiť. Prepojenie sa uskutočňuje prostredníctvom tagov. Tag je virtuálny dátový kanál, cez ktorý prechádzajú dáta. Jeden „koniec“ tagu je napojený na určitú pamäťovú adresu (tá slúži ako zásobník dát) a druhý koniec tagu tieto dáta sprístupňuje užívateľovi [1].

3. 1. Vytvorenie nového projektu v programe STEP7

Tu sa nachádzajú všetky funkcie a dáta potrebné k riadeniu.Vytvorenie nového programu sa uskutočňuje cez položky *File – "New Project Wizard"*. Otvorí sa okno **STEP7 Wizard: "New project"** ponuka **Introduction**. Objaví sa ponuka **Which CPU are you using in your project?**. V tejto ponuke je nutné vybrať typ používaného CPU. V mojom prípade sa jednalo o CPU313(C1**)**. Ďalej je potrebné priradiť **MPI** (Multi Point Interface) adresu. V ponuke **Which blocks do you want to add?**. Pre začiatok postačuje vybrať blok **OB35**. Tieto bloky je možné pridávať aj počas programovania. Nie je nutné ich vybrať hneď na začiatku. Ďalej si treba vybrať jazyk v ktorom chceme príslušné bloky programovať **Language for Selected Blocks**:

STL – Statment list. Umožňuje programovanie pomocou príkazov

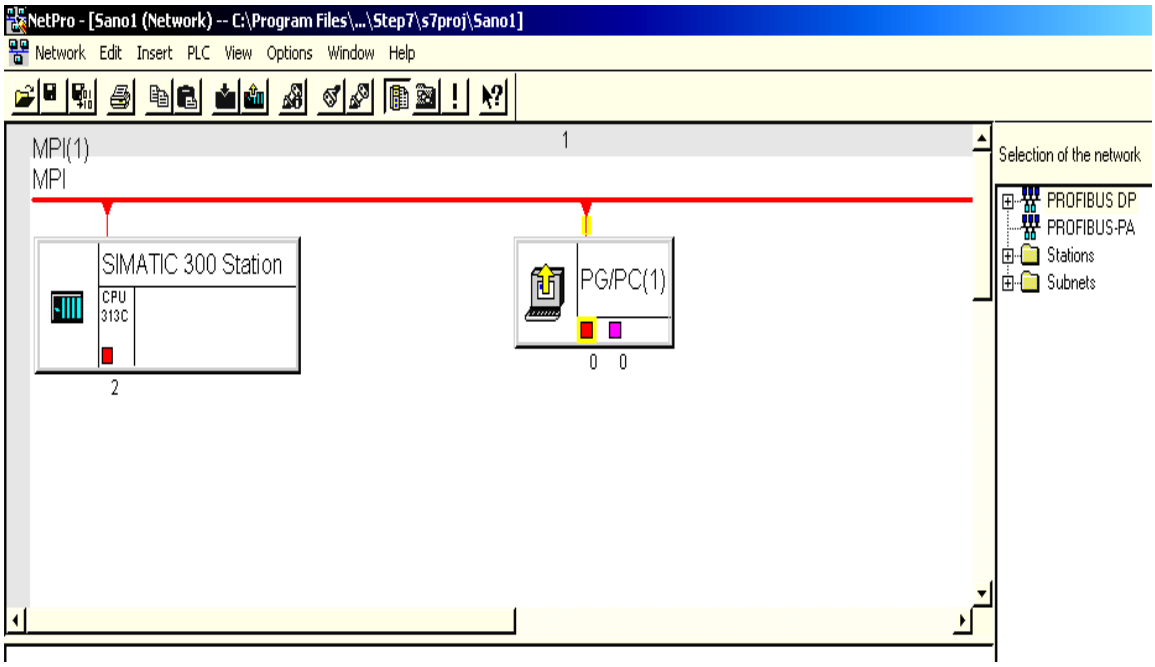
LAD – Lader logic. Umožňuje programovanie pomocou schematického zapojenia

FBD – Function block diagram. Umožňuje programovanie pomocou blokovej schémy V poslednej ponuke **What do you want to call your project?** Je potrebné zvoliť názov celého projektu. LK na **Finish** [3].

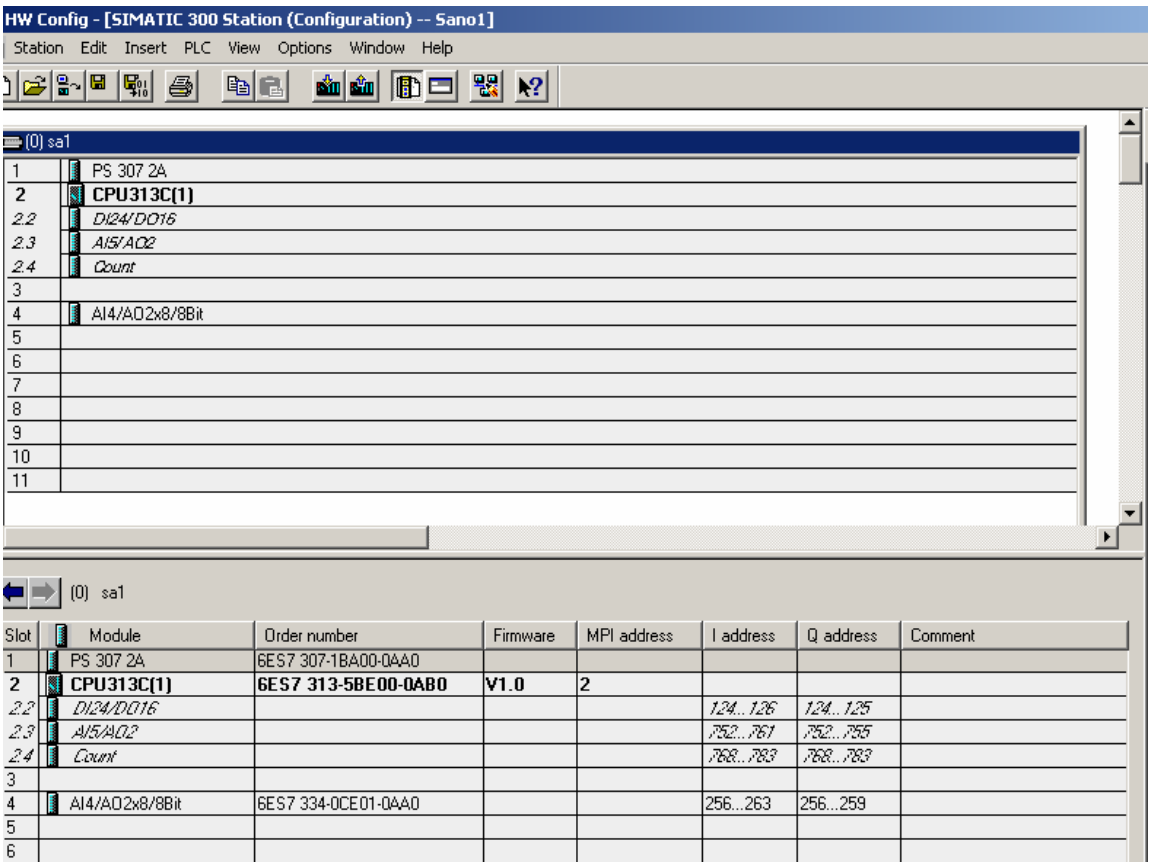
3. 2. Konfigurácia siete a I/O modulov

Umožňuje prepojenie PC s pracovnou stanicou **SIMATIC** pomocou zvoleného rozhrania. Pri konfigurácii siete sa vytvára komunikačná sieť medzi PC a príslušnými pracovnými stanicami. Základné spojenie pomocou MPI je možné vidieť na obr. 1.

Pri konfigurácii siete musí mať každý objekt rozdielny NOD. CPU má rezervovaný NOD 2 a PG/PC má rezervovaný NOD 0. Pomocou riadiacej stanice **SIMATIC S7-300** je zabezpečené prepojenie snímačov a akčných členov s PC [1]. Prepojenie snímačov a akčných členov s PC je zabezpečené pomocou riadiacej stanice **SIMATIC S7-300**, ktorou sú prepojené analógové vstupno – výstupné (I/O) moduly schopné spracovávať prúdový alebo napäťový signál.



Obr. 1 Zobrazenie nakonfigurovanej siete v **STEP7**



Obr. 2 Okno konfigurácie hardwaru

Na konfiguračnom okne hardware (obr. 2) možno vidieť obsadenie jednotlivých slotov daného railu. Rail predstavuje celú pracovnú stanicu a jeho riadky jednotlivé sloty, v ktorých sú definované jednotlivé moduly. Prvé tri sloty sú rezervované. Prvý pre napájací modul PS, druhý pre CPU tretí pre IM a štvrtý pre vstupno-výstupný (I/O) modul. Ak IM modul chýba, ostáva tretí slot neobsadený [4].

3. 3. Programovanie logických blokov pomocou FBD

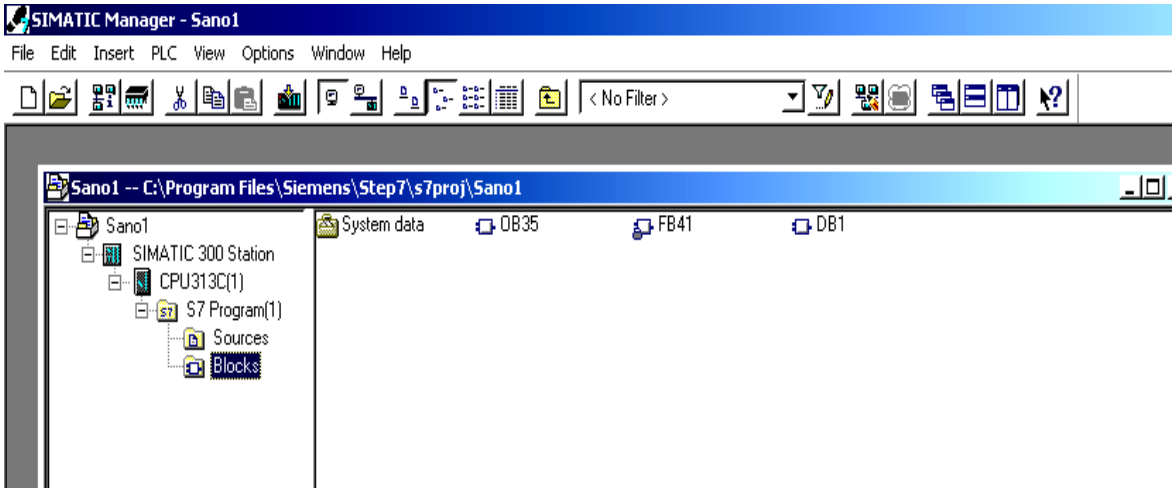
Na programovanie v **STEP7** som využíval programovací jazyk **FBD** (function block diagram), ktorý umožňuje programovanie pomocou blokovej schémy. Ďalšie možnosti programovania sú pomocou **STL** (statment list), kde sa programuje pomocou príkazov a **LAD** (lader logic), ktorý pri programovaní využíva schematické zapojenie. V prípade potreby možno jednotlivé programovacie jazyky vzájomne kombinovať. Všetky tri jazyky sú rovnocenné [1].

Pomocou logických blokov je možné naprogramovať všetky potrebné logické operácie ako aj riadenie daného reálneho procesu. V prípade, že nie je dostatok CPU pamäte na nahranie všetkých dát, je možné tieto dáta ukladať v spoločných dátových blokoch (DB). Údaje o týchto DB sú okamžite prístupné a použiteľné pre všetky FB bloky. Naopak DB asociovaný s príslušným FB blokom poskytuje údaje len pre daný FB blok [5].Vytvorené bloky v **STEP7** sú zobrazené na obr. 3.

V programe **STEP7** sa využívajú nasledovné bloky:

- **OB** – organizačné bloky. Udávajú štruktúru užívateľského programu. Využíval som blok OB35, ktorý reprezentuje hlavný program, a pracuje v cyklickom režime tak, že si načíta jednotlivé funkčné bloky alebo funkcie.
- **FB** – funkčné bloky, sú samostatne programovateľné a ukladajú sa v nich vnútorné premenné. Logická operácia obsiahnutá v FB bloku sa vykonáva nezávisle na ostatných blokoch.
- **FC** – funkcie, obsahujú algoritmy pre najčastejšie používané funkcie.

- **DB** – dátové bloky, slúžia na uloženie užívateľských dát. Pre každý naprogramovaný FB blok je nutné vytvoriť asociovaný DB blok.
- **SFB, SFC** – systémové funkčné bloky a systémové funkcie. Sú integrované priamo v CPU a umožňujú vstup do niektorých dôležitých systémových funkcií [1].



Obr. 3 Okno vytvorených blokov v **STEP7**

3. 3. 1. Funkčné bloky

Funkčné bloky sa nachádzajú v knižnici blokov v programe **STEP7**. V práci som využil funkčný blok FB41. Tento blok je samostatne programovateľný a ukladajú sa v ňom vnútorné premenné. K funkčnému bloku FB41 je nevyhnutné vytvoriť dátový blok, ktorý slúži na uloženie užívateľských dát. Pri obsadení jednotlivých vstupov a výstupov funkčný blok je súčasťou organizačného bloku OB35 [5].

Funkčný blok FB41

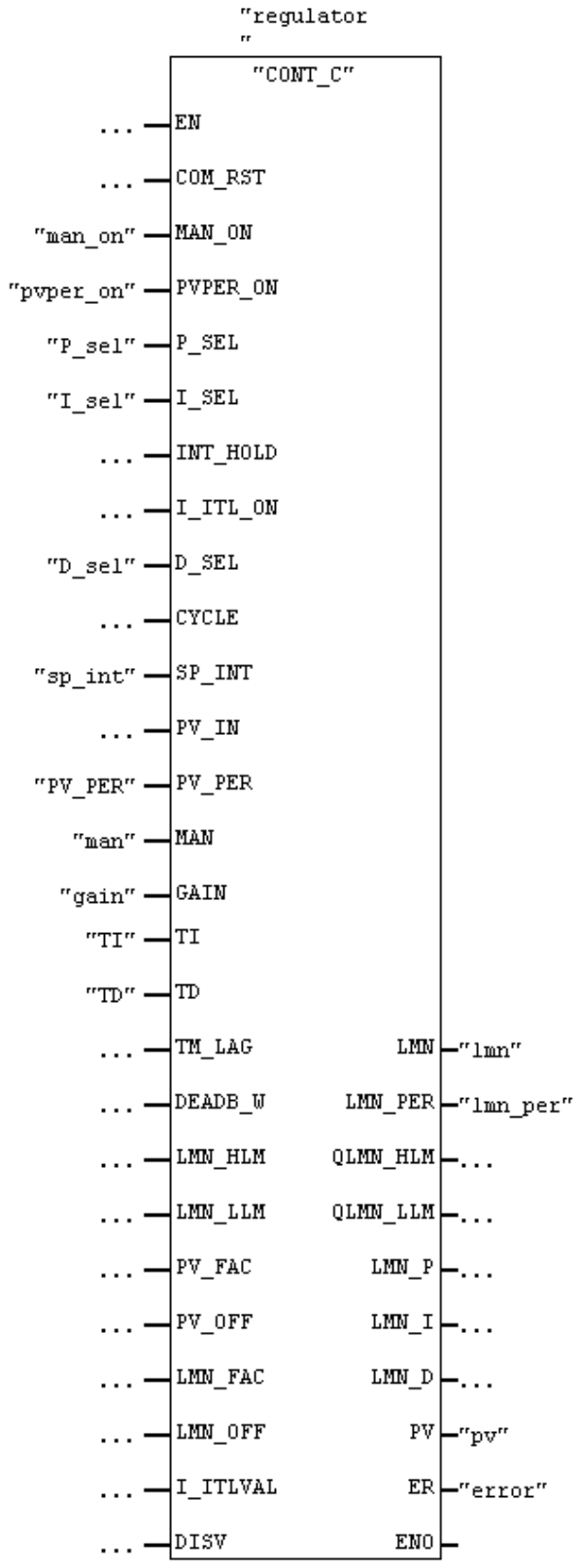
Na obr. 4 je zobrazený tvar PID regulátora vo forme funkčného bloku FB 41, ktorý sa nachádza v knižnici blokov programu **STEP7**, s jednotlivými vstupmi a výstupmi.

Vstupné parametre regulátora s opisom:

- MAN_ON - zapnutie / vypnutie zložky pre zadávanie „manual value“ (pri zapnutí zložky pre zadávanie „manual value“ sa vypne blok regulátora a regulátor sa pri riadení nepoužíva)
- PVPER_ON - zapnutie / vypnutie zložky pre zadávanie „process variable peripherial“, slúži na snímanie signálov z periférnych adries
- P_SEL – zapnutie / vypnutie zosilnenia regulátora (pri aktivácii regulátor pracuje s touto zložkou)
- I_SEL - zapnutie / vypnutie integračnej časovej konštanty regulátora (pri aktivácii regulátor pracuje z touto zložkou)
- D_SEL – zapnutie/vypnutie derivačnej časovej konštanty regulátora (pri aktivácii regulátor pracuje s touto zložkou)
- SP_INT – vstup žiadanej veličiny (sem sa zadáva hodnota žiadanej veličiny)
- PV_PER – výstupná hodnota z modelu a vstupná hodnota do regulátora v percentách
- MAN – akčná veličina (sem sa zadáva hodnota akčnej veličiny - „manual value“ pri aktivácii zložky MAN_ON)
- GAIN – hodnota zosilnenia regulátora
- TI – hodnota integračnej časovej konštanty
- TD – hodnota derivačnej časovej konštanty

Výstupné parametre regulátora s opisom[6]:

- LMN - výstup z regulátora (hodnota akčnej veličiny vystupujúcej z regulátora)
- LMN_PER – výstupná hodnota z regulátora v percentách a vstupná hodnota do modelu
- PV – meraná veličina (zobrazuje sa tu hodnota meranej veličiny)
- ER – regulačná odchýlka (rozdiel žiadanej a meranej veličiny)


"regulator
"
"CONT_C"
... EN
... COM_RST
"man_on" MAN_ON
"pvper_on" PVPER_ON
"P_sel" P_SEL
"I_sel" I_SEL
... INT_HOLD
... I_ITL_ON
"D_sel" D_SEL
... CYCLE
"sp_int" SP_INT
... PV_IN
"PV_PER" PV_PER
"man" MAN
"gain" GAIN
"TI" TI
"TD" TD
... TM_LAG
... DEADB_W
... LMN_HLM
... LMN_LLM
... PV_FAC
... PV_OFF
... LMN_FAC
... LMN_OFF
... I_ITLVAL
... DISV
LMN "lmn"
LMN_PER "lmn_per"
QLMN_HLM ...
QLMN_LLM ...
LMN_P ...
LMN_I ...
LMN_D ...
PV "pv"
ER "error"
ENO

Obr. 4 FB41

3. 4. Konfigurácia premenných

Pre ľahšiu orientáciu medzi použitými typmi blokov sa používa *Symbol editor* (symbolická tabuľka). V okne symbolickej tabuľky je možné priradiť symbolickú premennú nielen jednotlivým I/O adresám, ale aj použitým blokom.

V stĺpci *Address* je ako absolútna adresa uvedený typ a číslo použitého bloku, prípadne pamäťovej adresy typu REAL, vstupnej a výstupnej periférnej adresy typu WORD a v prípade boolovskej premennej Q 0.1 - 0.5.

Symbolickú tabuľku (obr. 5) je možné dopĺňať aj počas programovania jednotlivých blokov. Nie je potrebné pred programovaním zadefinovať všetky symbolické premenné, ktoré budú následne použité.

S7 Program(1) (Symbols) -- Sano1\SIMATIC 300 Station\CPU313C(

	Status	Symbol	Address	Data type	Comme
1		CONT_C	FB 41	FB 41	Continu...
2		Cyclic Interrupt 5	OB 35	OB 35	
3		D_sel	Q 0.5	BOOL	
4		gain	MD 40	REAL	
5		I_sel	Q 0.4	BOOL	
6		Imn	MD 10	REAL	
7		Imn_per	PQW 752	WORD	
8		man	MD 20	REAL	
9		man_on	Q 0.1	BOOL	
10		P_sel	Q 0.3	BOOL	
11		pv	MD 30	REAL	
12		PV_PER	PIW 752	WORD	
13		pvper_on	Q 0.2	BOOL	
14		regulator	DB 1	FB 41	
15		sp_int	MD 70	REAL	
16		TD	MD 60	TIME	
17		TI	MD 50	TIME	
18		error	MD 80	REAL	
19					

Obr. 5 Symbolická tabuľka pre sústavu III. rádu

3. 5. Spustenie programu a jeho diagnostika

Na spustenie vytvoreného projektu je potrebné vytvoriť on-line spojenie medzi pracovnou stanicou **SIMATIC S7-300** a PC.
Pracovná stanica má 3 pracovné módy:

- **M RES** – tento mód je určený na resetovanie pamäte procesorovej jednotky
- **RUN** – pracovný režim
- **STOP** – slúži na zastavenie bežiaceho programu

Pred spustením programu treba zapnúť zdroj pracovnej stanice do polohy ON, nastaviť kľúč CPU313 pracovnej stanice do polohy STOP. Pred nahraním programu do pracovnej stanice treba skontrolovať kontrolné LED diódy [1]. Ak je všetko v poriadku tak by mala svietiť zelená LED dióda zapnutého zdroja, ďalej zelená LED (DC5V) a oranžová LED dióda pre nastavenie kľúča do polohy STOP [4].

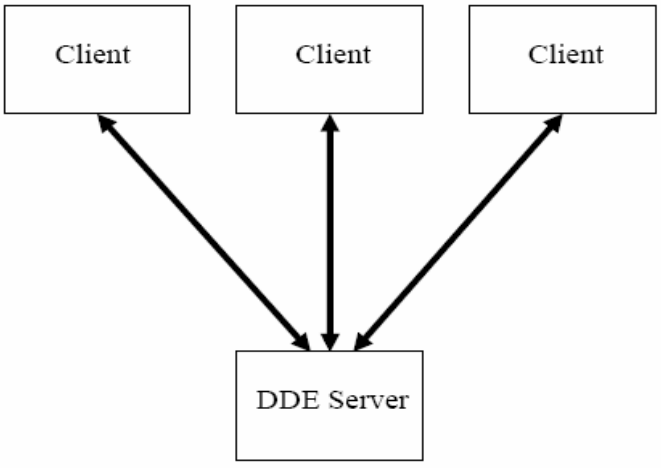
4. DDE komunikácia

Dynamic Data Exchange (DDE) je protokol, ktorý vyvinula spoločnosť Microsoft pre vzájomnú komunikáciu aplikácií na báze operačného systému MS Windows. Umožňuje komunikáciu v reálnom čase, či už medzi aplikáciami na jednom počítači, alebo v počítačovej sieti. Protokol je vytvorený na báze klient/server. DDE používa na výmenu dát medzi aplikáciami zdieľanú pamäť. Aplikácie môžu využívať DDE na jednorázový prenos údajov, ale aj na kontinuálny prenos a aktualizáciu dát. Každá aplikácia môže pôsobiť ako server pre viac klientov alebo ako klient pre viac serverov a zároveň môže byť serverom aj klientom.

4. 1. Klient/Server princíp

Možné spojenia medzi klientom a serverom sú zobrazené na obr.6.

- **Klient -** Klient je aplikácia, ktorá inicializuje DDE konverzáciu a riadi výmenu dát. Klient riadi logický komunikačný kanál, otvára ho, posiela požiadavky a opäť ho zatvára. Klient môže otvoriť viac ako jedno spojenie so Serverom a môže pristupovať viacerými premennými na spojenie.
- **Server** - Server je aplikácia, ktorá odpovedá na žiadosti o informácie od Klienta. Server je pasívny partner v konverzácii a odpovedá iba žiadostiam o informácie od aktívneho partnera t.j. Klienta.



Obr. 6 Možné spojenia medzi klientom a serverom

4. 2. DDE konverzácia

Aplikácie komunikujú medzi sebou po nadviazaní DDE konverzácie. Typický postup krokov v DDE konverzácii je nasledovný:

- DDE Klient spustí DDE konverzáciu otvorením spojenia so Serverom. DDE Server potvrdí otvorenie DDE spojenia
- Výmena dát s aplikáciami
- Klient alebo Server ukončí DDE komunikáciu

4. 2. 1. Typy komunikačných liniek

Poznáme tri typy komunikačných liniek medzi Serverom a Klientom.
Rozlišujú sa podľa spôsobu sprístupňovania dát na:

Cold link

Jedná sa o jednorázový príjem dát zo Serveru pri jednorázovej požiadavke Klienta. Klient pošle žiadosť Serveru pre prenos zmenených dát. Pri zmene dát na Serveri Klient nie je upozornený a je plne zodpovedný za aktualizáciu svojich dát. V tomto prípade komunikácie nie je záruka, že všetky zmeny dát budú zachytené.

- **Príjem** - Klient pošle požiadavku o informácie Serveru. Ak je Server v pozícii poskytnúť tieto dáta, pošle ich Klientovi. Ak nie, pošle negatívne potvrdenie. Žiadosti Klienta o informácie môžu byť opakované tak často ako je potrebné, až kým Klient alebo Server uzavrie DDE spojenie.
- **Odoslanie -** Klient jednorázovo posiela dáta Serveru [2].

Hot link

Spojenie v prípade, že sa dáta na Serveri zmenia, zmena sa automaticky prenesie na Klienta. To znamená, že Server oznámi každú zmenu hodnoty Klientovi. Zmeny sú posielané dovtedy, kým Klient neukončí dátové spojenie so Serverom [1].

Warm link

Tento typ konverzácie je kombináciou cold a hot linky. Na rozdiel od hot linky, Klient neprijíma aktualizované dáta bez požiadania. Tento typ konverzácie sa používa, keď si Klient želá byť informovaný o aktualizovanej premennej, ale chce riadiť dátový príjem.

4. 3. DDE terminológia

Aplikácie komunikujú medzi sebou po nadviazaní DDE konverzácie. Pri nadviazaní DDE komunikácie, musí klient poslať dva parametre: *Názov služby* (Service name) a *Názov prvku* (Item name). Keď serverová aplikácia prijme požiadavku na konverzáciu, ktorá obsahuje podporovanú tému, potvrdí danú požiadavku a vytvorí DDE konverzáciu. Kombinácia *Názov služby* a *Názov témy* jednoznačne identifikujú konverzáciu. Ani jeden z týchto dvoch parametrov sa nesmie počas trvania konverzácie meniť. Počas DDE konverzácie si Server a Klient vymieňajú dáta súvisiace s prvkom. Prvok je odkaz na dáta, ktoré sú významné pri konverzácii pre obe aplikácie. Hociktorá z aplikácií môže prvok zmeniť.

Služba: Každá aplikácia, ktorá môže byť DDE Serverom, má jednoznačný *Názov služby*. Zvyčajne je to názov spúšťacieho súboru aplikácie bez prípony.

Téma: Definuje subjekt DDE konverzácie.

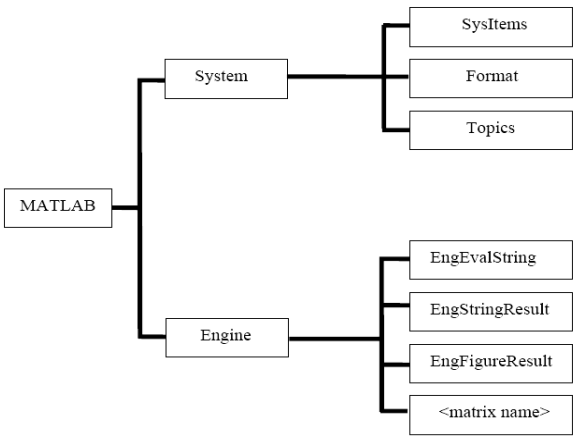
Prvok: Každá téma obsahuje jeden alebo viac prvkov. Prvok identifikuje dáta posielané počas DDE konverzácie.

4. 4. Možnosti MATLABu pri DDE komunikácii

MATLAB umožňuje komunikovať pomocou DDE rozhrania s ďalšími aplikáciami buď ako Server alebo Klient.

4. 4. 1. MATLAB ako DDE server

Klientská aplikácia môže pristupovať k DDE Serveru **MATLAB**u po zadaní *Názvu služby* „matlab“. **MATLAB** má v sebe zabudované dve témy. Systémová má názov „system“ a dátová má názov „engine“ [7]. Hierarchia DDE Servera v **MATLAB**e je zobrazená na obr. 7 a charakteristika jednotlivých pojmov je uvedená v tab.1.



Obr. 7 Hierarchia DDE Servera v **MATLAB**e

Tab. 1 Charakteristika použitých pojmov horeuvedenej hierarchie

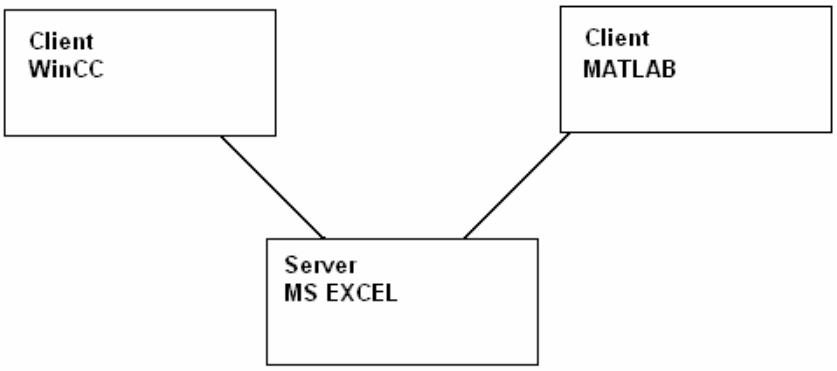
Prvok	**Opis**
SysItems	Zoznam prvkov podporovaných témou „system“.
Format	Zoznam formátov podporovaných serverom.
Topics	Zoznam tém podporovaných serverom.
EngEvalString	Príkaz posielaný **MATLAB**u na vykonanie.
EngStringResult	Výsledok vykonaného príkazu.
EngFigureResult	Grafický výsledok vykonaného príkazu.
<matrix name>	Premena, ktorej dáta klient požaduje.

4. 4. 2. MATLAB ako DDE klient

MATLAB je možné využívať pri DDE komunikácií ako Klienta. Pri tejto komunikácii sa využívajú príkazy, ktoré umožňujú pripojenie k zvolenému DDE Serveru [1].
Medzi hlavné využívané funkcie patria:

- ddeadv – nastavenie upozorňujúcej slučky so Serverom
- ddeexec – vykonanie príkazu Serverom
- ddeinit – inicializácia komunikácie medzi **MATLAB**om a DDE Serverom
- ddepoke – poslanie dát z **MATLAB**u Serveru
- ddereq – žiadosť o dáta zo Serveru
- ddeterm – ukončenie DDE komunikácie medzi **MATLAB**om a Serverom
- ddeaunadv – ukončenie upozorňujúcej slučky zo Serverom

MATLAB som využil ako Klient, **Microsoft Excel** ako Server a **WinCC** tiež ako Klient.



Obr. 8 Komunikačná štruktúra

4. 5. Práca s časovačom v MATLABe

MATLAB poskytuje objekt časovača, ktorý je možné použiť na plánovanie vykonania rôznych úloh. Spustenie načasovanej úlohy neznamená jej vykonanie, ale jej zaradenie do radu úloh čakajúcich na vykonanie [2].

Postup pre použitie časovača je nasledovný:

- Vytvorenie objektu časovača - pomocou funkcie *timer*
- Nastavenie parametrov časovača – definujú sa vlastnosti časovača, v mojom prípade išlo o periodické spúšťanie
- Spustenie časovača – pomocou funkcie *start*
- Zastavenie časovača – pomocou funkcie *stop*
- Vymazanie časovača – pomocou funkcie *delete*

4. 5. 1. Vlastnosti objektu časovača

Vlastnosti časovača možno rozdeliť do štyroch skupín:

- **Informačné parametre** – popisujú súčasný stav časovača. Tieto parametre užívateľ nemôže meniť. Uvedené sú v tab. 2.

Tab.2 Informačné parametre

Parameter	**Opis**
AveragePeriod	Priemerná perióda vykonávania obslužnej funkcie
InstantPeriod	Čas medzi poslednými dvomi vykonaniami obslužnej funkcie
Running	Súčasný stav časovača („on“ alebo „off“)
TaskExecuted	Počet vykonaní obslužnej funkcie od spustenia

- **Parametre správania** – udávajú časové správanie sa časovača. Parametre správania sú v tab. 3.

Tab. 3 Parametre správania

Parameter	**Opis**
BusyMode	Udáva spôsob ďalšieho spracovania obslužnej funkcie, ak posledné spracovanie nebolo ukončené
ExecutionMode	Spôsob zoraďovania obslužnej funkcie
Period	Perióda medzi dvoma vykonaniami obslužnej funkcie
StartDelay	Časový interval pred prvým vykonaním funkcie v sekundách
TasksToExecute	Zostávajúci počet vykonaní obslužnej funkcie

- **Parametre obslužných funkcií** – špecifikujú funkcie, ktoré sa majú vykonať pri danej udalosti. Parametre obslužných funkcií sú uvedené v tab. 4.

Tab. 4 Parametre obslužných funkcií

Funkcia	**Opis**
ErrorFcn	Funkcia vykonaná v prípade chyby
StartFcn	Funkcia vykonaná pri štarte časovača
StopFcn	Funkcia vykonaná pri zastavení časovača
TimerFcn	Funkcia vykonaná časovačom v každom kroku

- **Parametre definované užívateľom** – umožňujú pribaliť vlastné data, tieto parametre sú uvedené v tab. 5.

Tab. 5 Parametre definované užívateľom

Funkcia	**Opis**
Name	Textový reťazec identifikujúci časovač
Tag	Užívateľom definovaný popis časovača
UserData	Užívateľom definované dáta

Použitý program m-súboru a časovača na uskutočnenie DDE komunikácie sú uvedené v prílohe I. a II.

5. Vytvorenie vizualizačného prostredia

Program **WinCC** umožňuje užívateľovi definovať si vlastné vizualizačné prostredie pre daný proces pomocou grafického editora. **WinCC** umožňuje rozmiestnenie jednotlivých ovládacích prvkov, vstupno – výstupných polí a monitorovacích okien na obrazovke užívateľa tak, aby mal neustály prehľad nad priebehom procesu a rýchlymi zásahmi ho mohol aktívne ovplyvňovať.

5. 1. Vytvorenie projektu

Nový projekt sa vytvorí voľbou *File – New – Single-User Projekt*. Zadá sa meno projektu a nadefinuje sa rozhranie, pomocou ktorého bude **WinCC** komunikovať s PLC. Po tomto kroku **WinCC** automaticky vygeneruje okno s jednotlivými modulmi, ktoré tvoria jeho podzložky.
Najdôležitejšie moduly sú:

- *Tag Management* – slúži na definovanie a správu jednotlivých tagov
- *Graphics Designer* – grafický editor, ktorý umožňuje zobrazovať dané zariadenie a taktiež sledovať proces pomocou grafov alebo polí, ktoré ukazujú žiadané hodnoty
- *Global script* – slúži na kopírovanie jedného tagu do druhého pri DDE komunikácii

5. 2. Prepojenie programov WinCC, STEP7 a Microsoft Excel pomocou tagov

Keďže na používanom počítači nebolo možné vytvoriť archiváciu, pri práci som využil DDE komunikáciu. Bolo potrebné vytvoriť vo **WinCC** špecifické tagy umožňujúce tento typ komunikácie.

Prepojenie **WinCC** s riadiacim programom **STEP7** je zabezpečované pomocou tagov. Tag je virtuálny dátový kanál, cez ktorý prechádzajú dáta. Tagy sa vytvárajú v *Tag Managmen*t po pridaní komunikačného driveru SIMATIC S7 PROTOCOL SUITE. Pri

vytváraní tagu je potrebné zadefinovať názov, typ premennej, ktorú tag využíva a adresu, pomocou ktorej bude daný tag spojený so **STEP7** [1].
Tagy sú dvojakého typu:

- *Interné tagy* – slúžia na uchovávanie interných premenných
- *Externé tagy* – slúžia na komunikáciu s PLC [3]

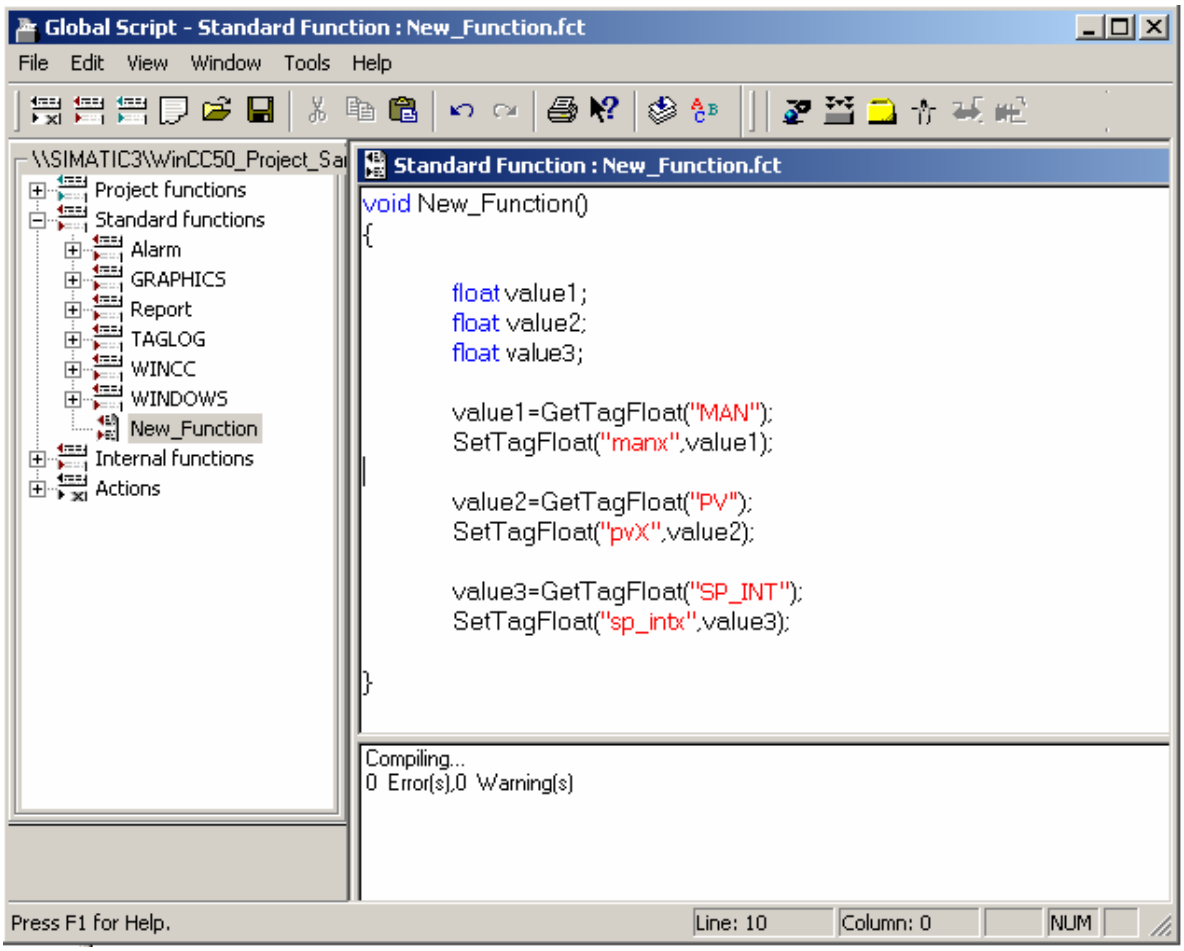
Pre riadenie elektronického modelu som vytvoril externé tagy pre akčnú, meranú, žiadanú veličinu, regulačnú odchýlku, P a I zložku PI regulátora.

Prepojenie medzi **WinCC** a **Microsoft Excel** som uskutočnil pridávaním tagov v *Tag Managemen*te, kde sa pridá komunikačný driver *Windows DDE.CHN* a vytvorí sa rozhranie vo **WinCC** pre DDE komunikáciu. Pre DDE komunikáciu treba zadefinovať v *Connection Properties* aplikáciu. Keďže som **WinCC** využíval ako *Klient*a, do názvu služby som zadával program, ktorý bol *Server*, "excel". *Názov témy* bol názov súboru v programe **Microsoft Excel** (Book.xls, príloha III) kde boli poslané **MATLAB**om žiadané hodnoty.

Po nastavení všetkých tagov bolo potrebné vytvoriť komunikáciu medzi tagmi spájajúce **WinCC** a **STEP7** s tagmi spájajúcimi **WinCC** a **Microsoft Excel**. Kopírovanie jedného tagu do druhého bolo uskutočnené pomocou programu (*Project functions*), napísaného v module *Global Script*.
Boli použité nasledovné príkazy:

- *GetTagFloat*("tag") – načíta hodnotu tagu do vybranej premennej
- *SetTagFloat*("tag", premenná) – nastaví hodnotu tagu na hodnotu premennej

Na začiatku programu som zadefinoval typ premennej. Keďže išlo o čísla s desatinnými miestami, typ premennej bol *float*. Na obr. 9 je vizualizovaný *Global Script* s napísaným programom. Spustenie programu sa vykonáva vo vizualizačnej obrazovke pomocou tzv. *C-Action*, ktorá vykonáva programy vložené do nej [2].



Obr. 9 *Global Script* s napísaným programom

5. 3. Vytvorenie vizualizačnej obrazovky a ovládacích prvkov

Na vizualizáciu je používaný grafický editor *Graphics Designer*, kde sú zobrazené aktuálne hodnoty meraných veličín, ako aj ich grafické znázornenie v čase. Jednotlivé objekty si môže užívateľ vytvoriť individuálne alebo môže použiť objekty nachádzajúce sa v knižnici [5]. Ovládacie prvky umožňujú ovplyvňovať vznikajúce udalosti v procese. Zaznamenávajú zmenu, ktorá vznikla v procese alebo zmenu, ktorú vykoná užívateľ pri zásahu do procesu. Ovládacie prvky možno prirovnať k dynamickým objektom. Nachádzajú sa v *Object Palette* v *Smart Objects*. Medzi najpoužívanejší ovládací vstupno-výstupný prvok patrí *I/O Field*. Je to objekt vo forme textového poľa, ktorého hodnota závisí od hodnoty na pripojenom tagu. Po umiestnení *I/O Field* na pracovnú plochu sa otvorí okno, ktoré umožňuje definovať *tag,* ktorého hodnoty bude objekt zobrazovať a tiež treba zvoliť periódu v akej budú hodnoty

obnovované. Nastavil som periódu obnovovania údajov na 250 ms. Ďalšie nastavenie vlastnosti objektu je možné voľbou *Properties*.

Medzi hlavné nastavenia vstupno-výstupného poľa patrí:

- ***Field Type*** – umožňuje nastaviť typ poľa na:
  - *Input (vstupné)* – možno do poľa zapisovať hodnoty a tým riadiť process napr. hodnotu žiadanej, alebo akčnej veličiny.
  - *Output (výstupné)* – zobrazuje aktuálne hodnoty namerané v systéme napr.meranú veličinu.
  - *Both (kombinované)* – umožňuje zapisovanie a zároveň aj sledovanie aktuálnej hodnoty procesu.
- ***Data format*** – možnosť voľby zobrazovaného formátu (Binary, Decimal, Hexidecimal, String)
- ***Output Format*** – umožňuje nastaviť koľko ciferné čísla má pole zobrazovať [1]
- ***Apply on Exit*** – aby bola vložená hodnota do daného poľa akceptovaná musí byť táto vlastnosť nastavená na *YES*

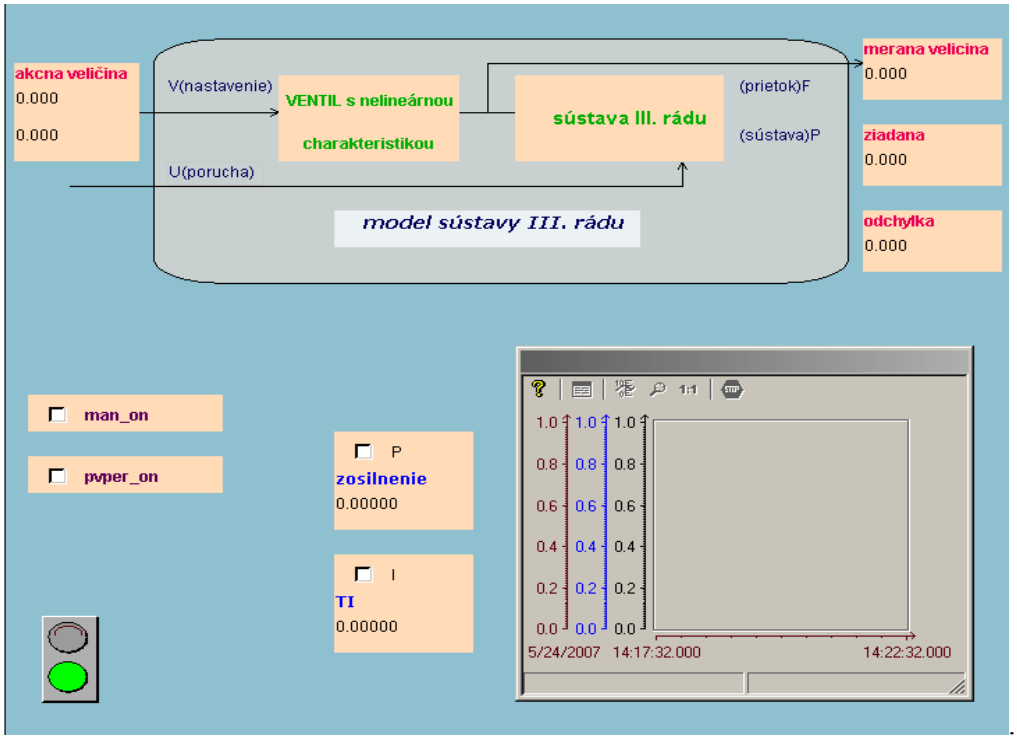
Každá vlastnosť môže byť statická alebo dynamická. Statická vlastnosť sa dá meniť iba v *Graphics Designer*i, hodnota dynamickej vlastnosti je daná hodnotou tagu [5].

Ďalším ovládacím prvkom, ktorý som využil pri práci v *Graphics Designer*i je *Check Box,* ktorý umožnil voľbu zapnutia, alebo vypnutia manuálneho a periférneho riadenia, a taktiež zapnutie a vypnutie zložiek príslušného regulátora.

Na indikáciu komunikácie medzi tagmi som použil indikátor kopírovania hodnôt *On_Off_3*. Indikátor som umiestnil kliknutím na ikonu *Display Library* . V položke Properties cez Tag Assigment vyberiem pre dynamickú vlastnosť položku *C-Action*, v ktorej definujem názov funkcie napísaný v *Global Script*e. Názov funkcie je *NewFunction* Pokiaľ indikátor zobrazuje zelenú farbu, tak prebieha funkcia napísaná v *Global Script*e. V opačnom prípade je v programe chyba a kopírovanie medzi tagmi nie je správne.

Ako vizualizačný prvok v prostredí *Graphics Designer*a slúži *WinCC Trend Control*, paleta *Controls*. Ak sú nastavené všetky požadované vlastnosti, potom počas merania je graf

schopný vykresliť priebeh akčnej, meranej a žiadanej veličiny [2]. Výsledné zobrazenie vizualizačného okna možno vidieť na obr. 10.



Obr. 10 Vizualizačná obrazovka pre elektronický model

6. Návrh regulátorov

Pre riadenie elektronického modelu som zvolil experimentálnu aj analytickú metódu syntézy regulátora. Najčastejšie používaným vstupným signálom pre návrh regulátora, prípadne pre približné určenie dynamických vlastností regulovaného objektu, je skoková zmena jednej zo vstupných veličín pri zachovaní ostatných vstupných veličín konštantných [7]. Zmena na vstupe sa vykoná, keď je systém v ustálenom stave a zaznamenajú sa hodnoty výstupnej veličiny [8]. Časový priebeh výstupnej veličiny, ktorý je odozvou na skokovú zmenu jednej zo vstupných veličín, voláme reálnou prechodovou charakteristikou (PCH). Na PCH určíme inflexný bod, preložíme ním dotyčnicu k PCH, ktorá na rovnobežkách s časovou osou prechádzajúcimi hodnotami y_0, y_∞ vymedzí dva časové údaje: čas prieťahu t_u a čas nábehu t_n.

Z PCH sa dá určiť hodnotu zosilnenia systému daného vzťahom:

$$Z = \frac{y_\infty - y_0}{u_\infty - u_0} \tag{1}$$

6. 1. Identifikácia elektronického modelu

Elektronický model som identifikoval Strejcovou metódou. Pri identifikácií Strejcovou metódou uvažujeme náhradu modelu prenosom *n*-tého rádu v tvare:

$$G(s) = \frac{Z}{(Ts+1)^n} e^{-Ds} \tag{2}$$

kde Z je zosilnenie, T časová konštanta, D dopravné oneskorenie a n rád systému, ktoré potrebujeme určiť. Keďže súčasťou modelu je ventil spôsobujúci nelinearity systému je pri získavaní prechodovej charakteristiky potrebné minimalizovať jeho vplyv, čím sa systém stáva v určitej oblasti lineárnym. Overenie jeho linearity je na obr. 11.

Elektronický model som identifikoval Strejcovou metódou ako systém 2. rádu v oblasti 0-20% akčnej veličiny.

Na základe PCH som získal nasledovné parametre

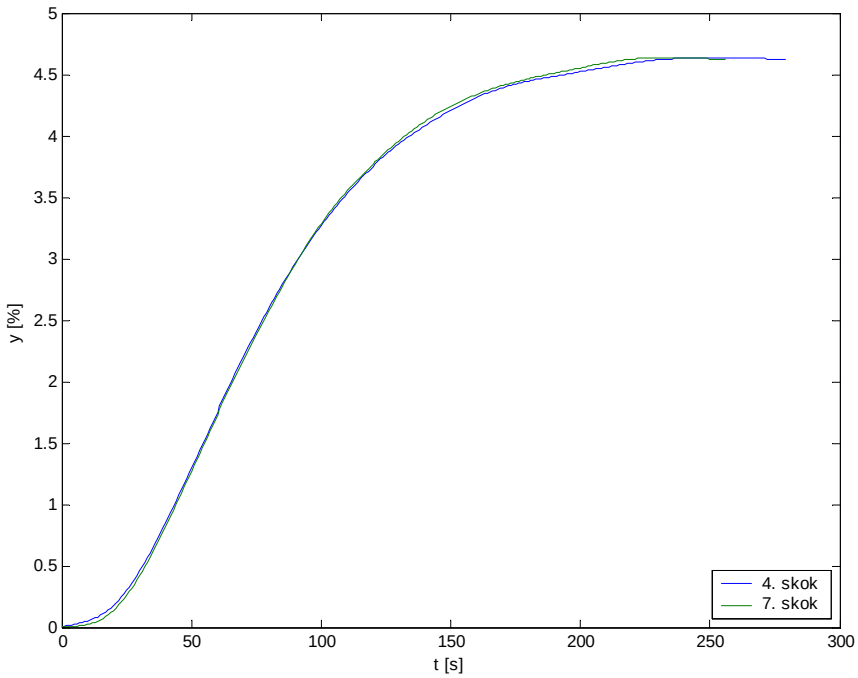
- Zosilnenie $Z = 4,6200$
- Časová konštanta $T = 37,0186\ s$
- Dopravné oneskorenie $D = 11,7578\ s$

Potom prenosová funkcia má tvar:

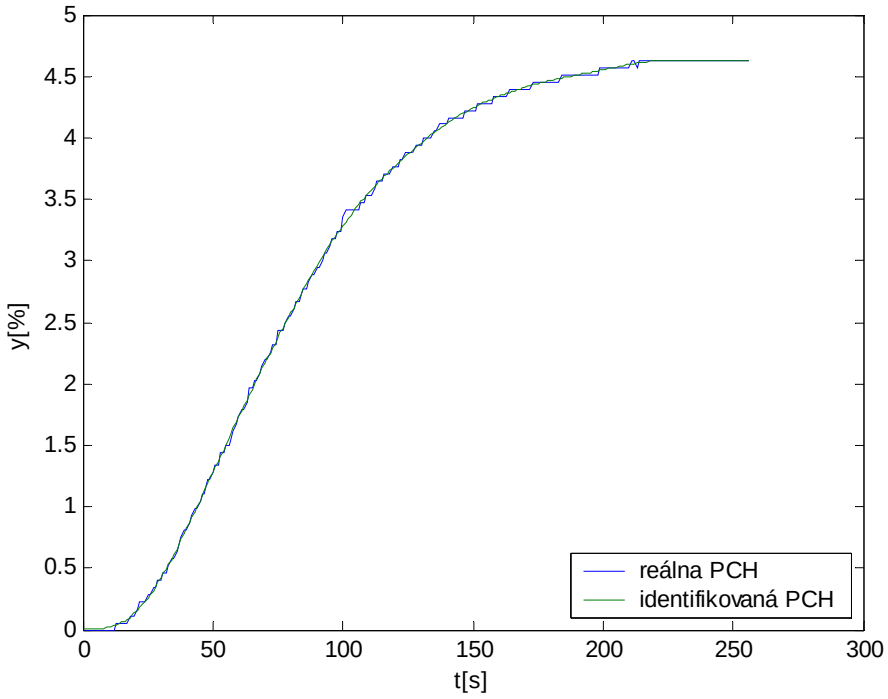
$$G(s) = \frac{4,6200}{(37,0186s+1)^2} e^{-11,7578s} \tag{3}$$

Použitý skript na identifikáciu je uvedený v prílohe IV.

Na základe reálnej a identifikovanej PCH (obr. 12) sa dá usúdiť, že elektronický model môžeme považovať ako systém 2. rádu s príslušnými parametrami.



Obr. 11 Overenie linearity



Obr. 12 Porovnanie identifikovanej a reálnej PCH

6. 2. Syntézy regulátorov

Pri návrhoch parametrov regulátora som využil experimentálne metódy využívajúce prechodovú charakteristiku ako aj identifikovaný prenos systému. Navrhoval som regulátory, ktoré nezanechávajú trvalú regulačnú odchýlku.

6. 2. 1. Strejcova metóda syntézy regulátora

Na základe identifikovaného prenosu (3) som vypočítal parametre PI regulátora podľa vzorcov:

$$Z_R = \frac{1}{Z}\frac{n+2}{4(n-1)} \tag{4}$$

$$T_I = T\frac{n+2}{3} \tag{5}$$

Vypočítané parameter regulátora sú:

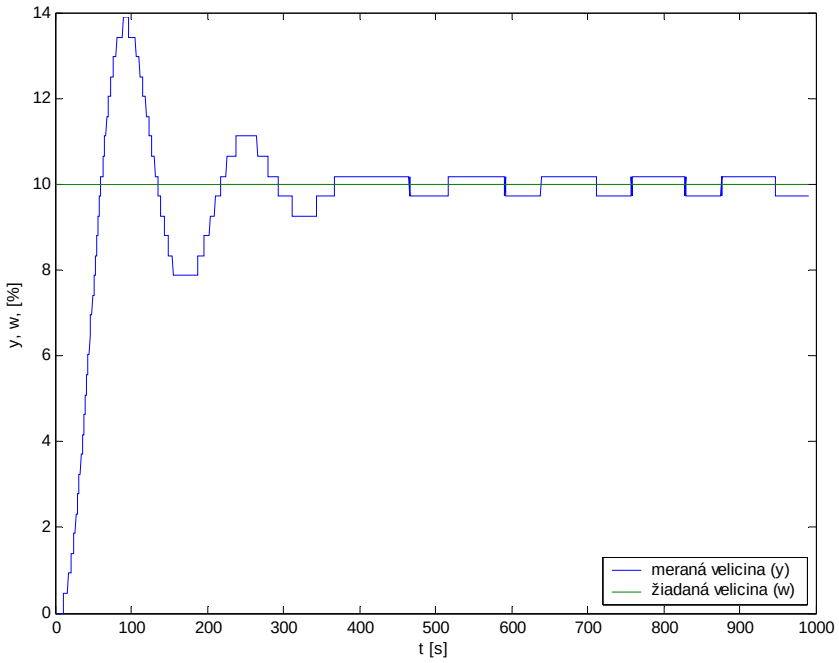
$Z_R = 0{,}2186$
$T_I = 42{,}2581$ s

Pre overenie funkčnosti navrhnutého regulátora som zvolil dve rôzne hodnoty žiadanej veličiny. Jednu z identifikovanej oblasti a druhú mimo tejto oblasti. Priebehy riadenia a priebehy príslušných akčných veličín sú znázornené na obr. 13 - 16. Z týchto priebehov riadenia vyplýva že regulátor je vhodný na riadenie laboratórneho zariadenia.

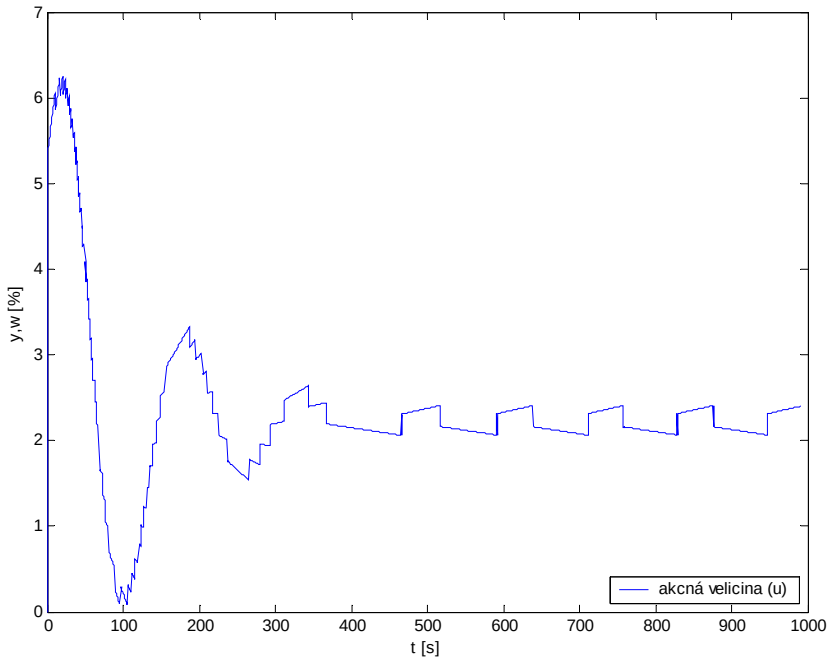
6. 2. 2. Syntéza regulátora metódou umiestnenia pólov

Metódou návrhu regulátora s umiestnením pólov nazývame postup, keď systému vnútime póly so známymi vlastnosťami, ktoré zaručia stabilitu a splnenie špecifikovaných ukazovateľov kvality regulácie [8]. Parametre regulátora som vypočítal na základe zvoleného pólu s = -0,029. Pól som si zvolil na základe vypočítaného koreňa CHR systému $s_1 = -0{,}027$. Zvolený pól je posunutý viac doľava od imaginárnej osi ako je pól riadeného procesu, aby uzavretý regulačný obvod bol rýchlejší, než riadený proces.
Vypočítané parametre regulátora:

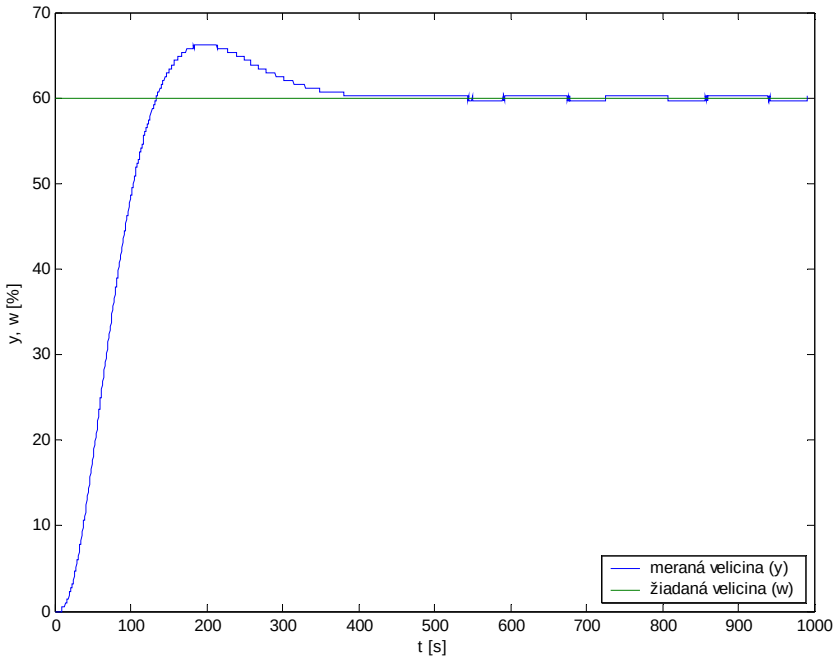
$Z_R = 0{,}5319$
$T_I = 73{,}5255$ s
Priebehy riadenia sú na obr.17, 19 a príslušné priebehy akčných veličín na obr. 18, 20.



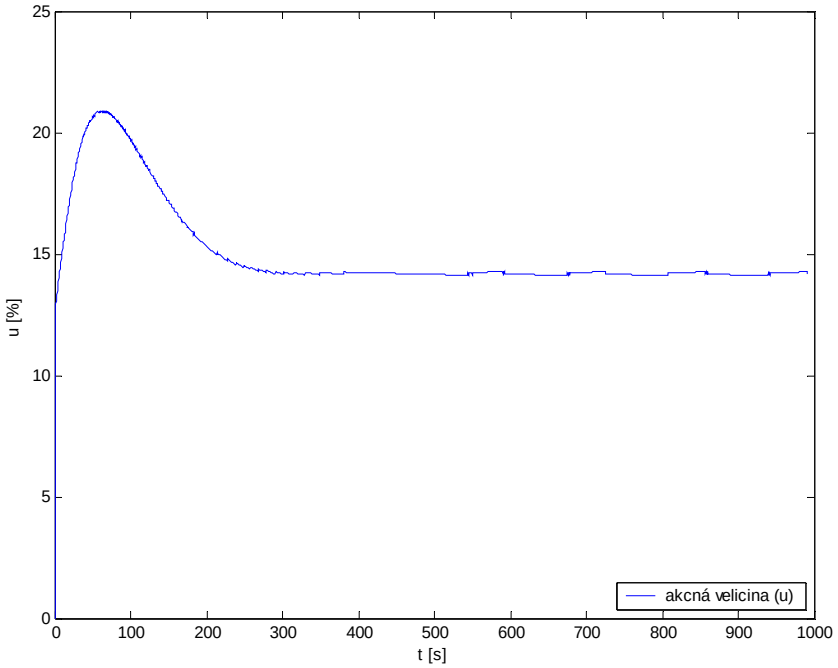
Obr. 13 Priebeh riadenia navrhnutým regulátorom Strejcovou metódou (w = 10%)



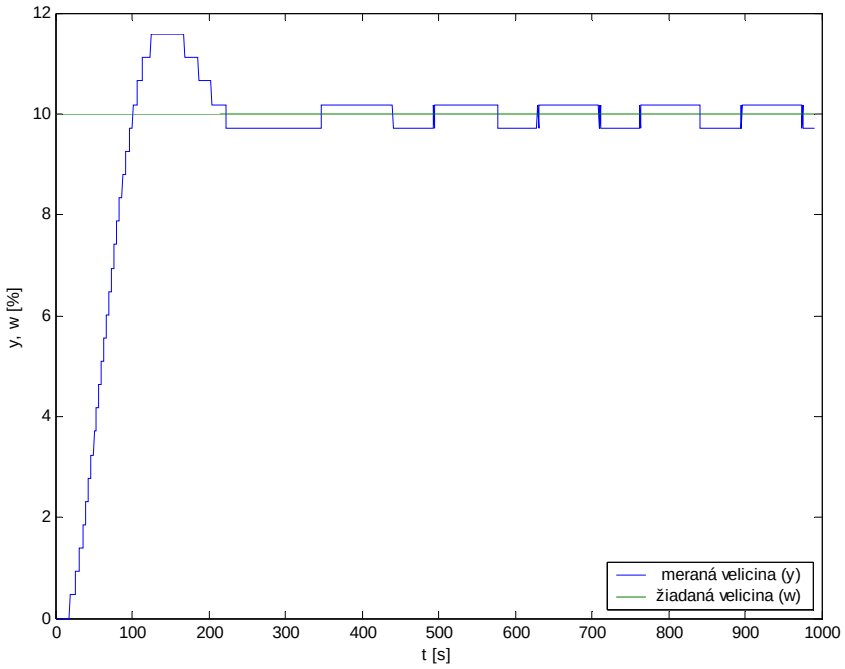
Obr. 14 Priebeh akčnej veličiny pre žiadanú veličinu 10% (Strejcova metóda)



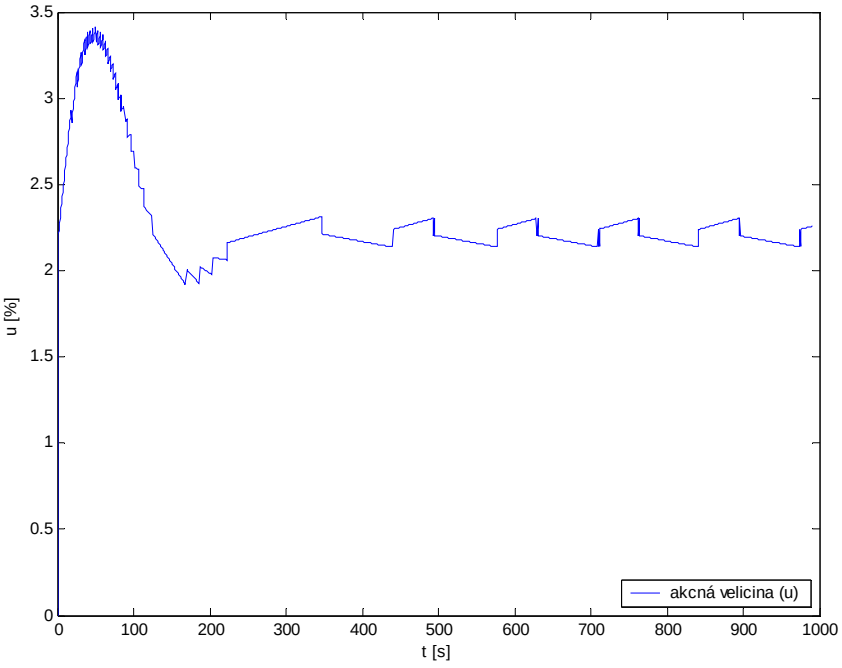
Obr. 15 Priebeh riadenia Strejcovou metódou mimo identifikovanej oblasti (w = 60%)



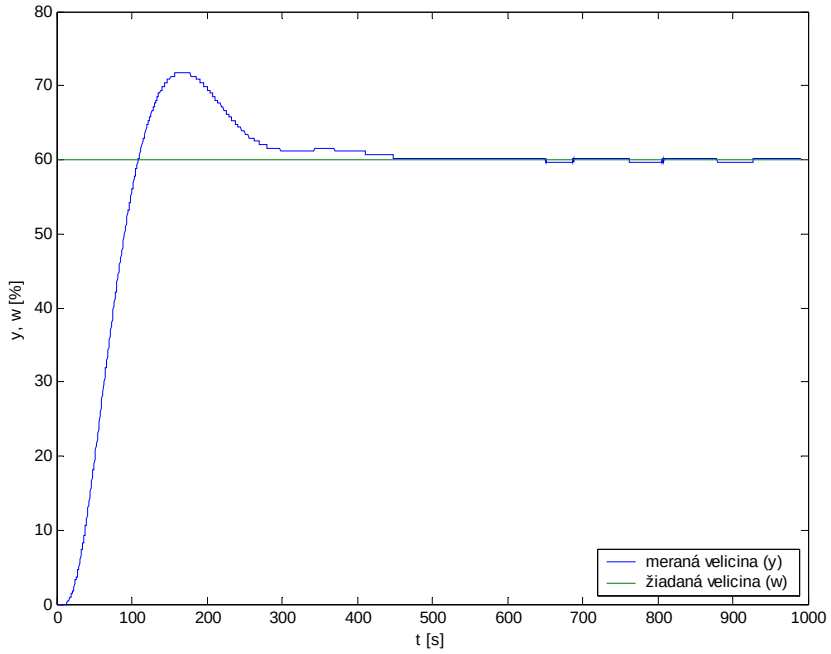
Obr. 16 Priebeh akčnej veličiny pre žiadanú veličinu 60% (Strejcova metóda)



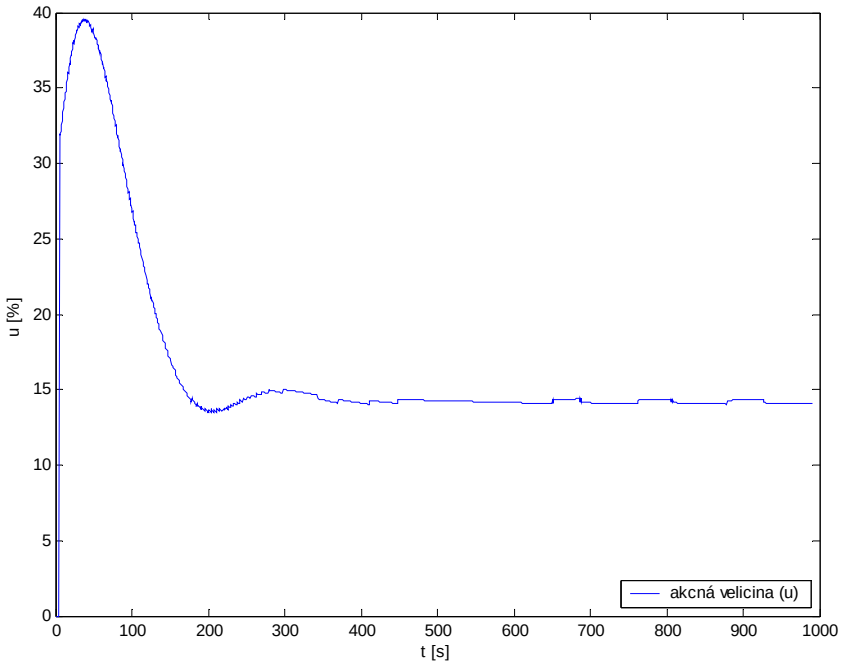
Obr. 17 Priebeh riadenia navrhnutým regulátorom metódou umiestnenia pólov (w = 10%)



Obr. 18 Priebeh akčnej veličiny pre žiadanú veličinu 10% metódou umiestnenia pólov



Obr. 19 Priebeh riadenia metódou umiestnenia pólov mimo identifikovanej oblasti (w = 60%)



Obr. 20 Priebeh akčnej veličiny pre žiadanú veličinu 60% (metóda umiestnenia pólov)

6. 3. Vyhodnotenie výsledkov

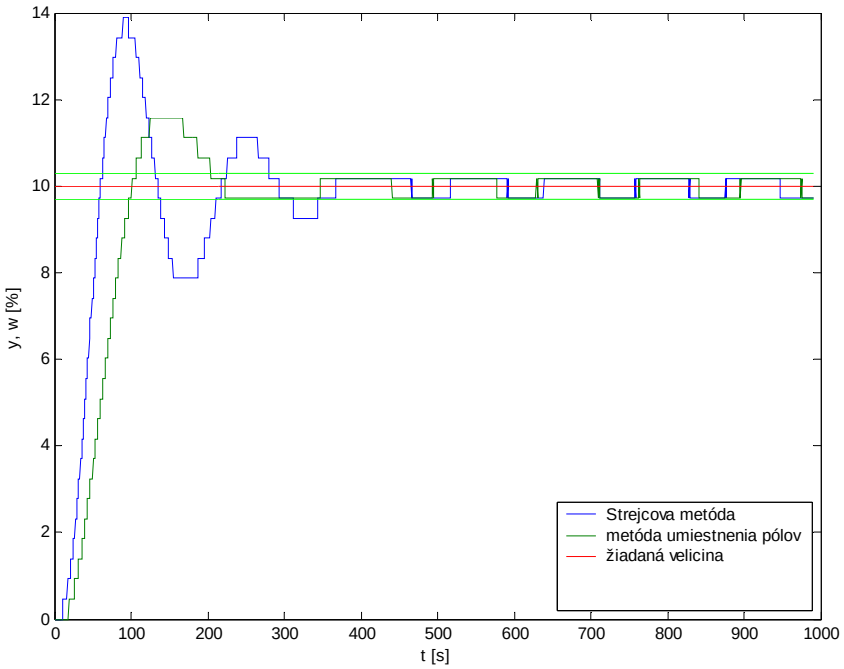
Vyhodnotenie riadenia elektronického modelu som uskutočnil pomocou kritérií kvality:

- čas regulácie t_{reg}
- maximálne preregulovanie σ_{max}
- čas maximálneho preregulovania tσ

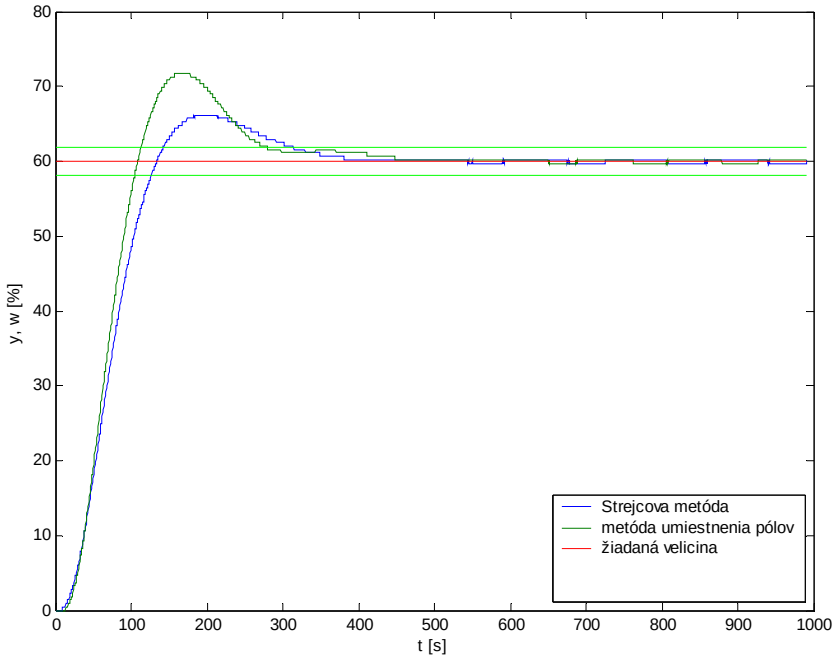
Pre výpočet jednotlivých kritérií som si zvolil hodnotu δ okolia, δ = ±3 % žiadanej veličiny. Výsledky vyhodnotenia jednotlivých kritérií kvality pre príslušnú zvolenú metódu návrhu regulátora a zvolenú žiadanú veličinu sú uvedené v tab. 6. Pri vyhodnotení som vychádzal z priebehov riadenia zobrazených na obr. 21 - 22. Z uvedených výsledkov jednoznačne vyplýva, že regulátor navrhnutý metódou umiestnenia pólov pre elektronický model je lepší.

Tab. 6. Ukazovatele kvality

Ukazovatele kvality	Strejcova metóda w = 10%	Strejcova metóda w = 60%	Metóda umiestnenia pólov w = 10%	Metóda umiestnenia pólov w = 60%
t_{reg} [s]	343,97	314,26	203,3	279,26
σ_{max} [%]	38,89	10,33	15,70	19,60
t_{σ} [s]	93	197,5	145	168



Obr. 21 Porovnanie priebehov riadenia pre žiadanú hodnotu 10%



Obr. 22 Porovnanie priebehov riadenia pre žiadanú hodnotu 60%

Záver

Hlavnou úlohou mojej práce bola navrhnúť funkčný regulátor pre riadenie elektronického modelu III. rádu. Pri tvorbe projektu som využíval program **STEP7**, ktorý som pripojil k pracovnej stanici. Nasledujúcim krokom mojej práce bola tvorba vizualizačného prostredia v programe **WinCC**, čo je neoddeliteľnou súčasťou **SIMATIC**u. Pri vytváraní projektu som aktívne využil výhody **MATLAB**u a **EXCEL**u. Nevyhnutnou podmienkou pre úspešnú komunikáciu medzi týmito programami bolo ich spojenie. Takto vytvorená tzv. DDE komunikácia umožnila zber dát v **MATLAB**e vo forme matíc, ktoré po uložení vo forme textových súborov boli vhodné na spracovanie. Z nameraných dát som vykreslil prechodové charakteristiku a následne som systém identifikoval pomocou Strejcovej metódy ako systém II. rádu s dopravným oneskorením. Princíp tejto metódy spočíva v nájdení inflexného bodu a preložení dotyčnice prechádzajúc cez ten bod. Proces identifikácie nasledoval výpočet parametrov regulátorov a otestovanie ich funkčnosti. Vhodnosť regulátorov bola overená už pomocou spomenutého vizualizačného prostredia, kde po zadaní žiadanej hodnoty do príslušného vstupno-výstupného ovládacieho prvku, som zistil, že regulátory úplne odstránia trvalú regulačnú odhcýlku. Ukazovatele kvality sú spracúvané v tab.6. Na základe týchto ukazovateľov PI regulátor navrhnutý metódou umiestnenia pólov je vhodnejší pre sústavu III. rádu.

Literatúra

[1] Palacka O.: Riadenie varáka rektifikačnej kolóny ako systému s integračnými vlastnosťami, FCHPT STU Bratislava 2005

[2] Vöröš J.: Inteligentné riadenie pomocou PLC SIMATIC S7 s podporou MATLABu, FCHPT STU Bratislava 2004

[3] Kožka Š., Kvasnica M.: Programovanie PLC SIMATIC 300 (Základná príručka), KIRP,Bratislava 2001

[4] Vaneková K.: Priemyselný riadiaci systém SIMATIC (Bakalárska práca), FCHPT STU Bratislava 2004

[5] Vaneková K.: Riadenie pomocou priemyselného riadiaceho systému Simatic(Diplomová práca), FCHPT STU Bratislava 2006

[6] SIMATIC Manual

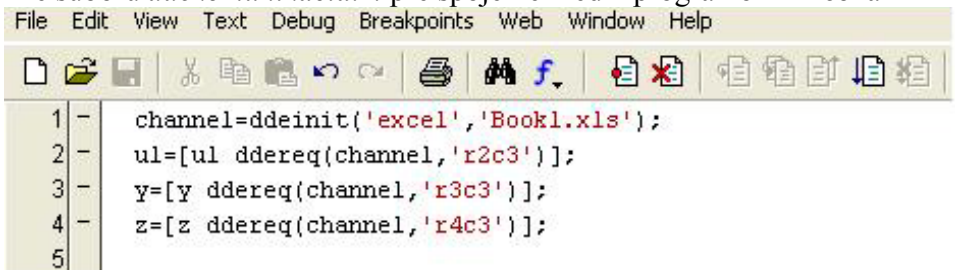
[7] Bakošová M., Fikar M., Čirka Ľ. :*Základy automatizácie*, STU, Bratislava 2003

[8] Mészáros A., Danko J., Mikleš J., Bakošová M.: *Základy automatizácie*, STU Bratislava 1997

Prílohy:

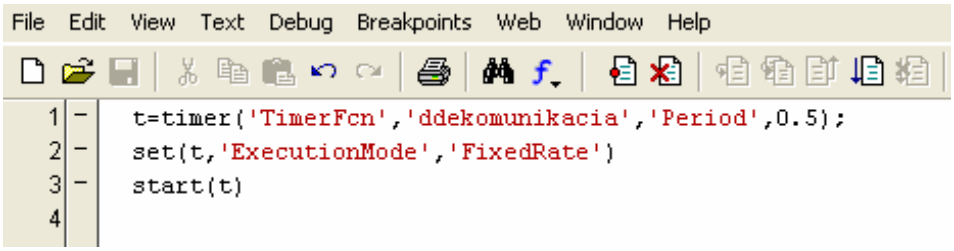
I.

Výpis m-file súboru *ddekomunikacia.m* pre spojenie medzi programom **Excel** a **MATLAB**om



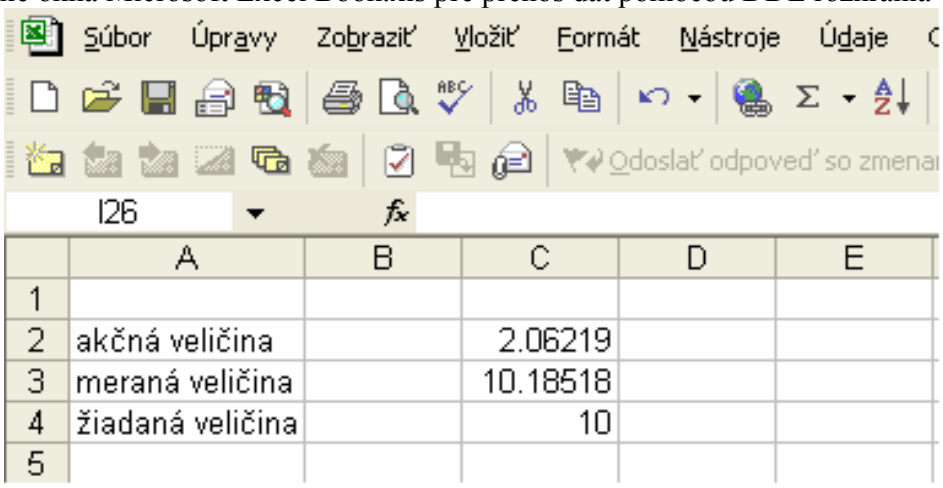
II.

Výpis m-file súboru *timer.m*, pre spúšťanie výpočtu akčnej veličiny v požadovanom intervale.



III.

Zobrazenie okna Microsoft Excel Book.xls pre prenos dát pomocou DDE rozhrania



	A	B	C	D	E
1					
2	akčná veličina		2.06219		
3	meraná veličina		10.18518		
4	žiadaná veličina		10		
5					

IV.

Výpis m-file súboru *identifikacia.m,* pre identifikáciu systému

File Edit View Text Debug Breakpoints Web Window Help

```
load dataE.txt
t5=dataE(3,:);
u5=dataE(1,:);
y5=dataE(2,:);
plot(t5,u5,t5,y5)

y5=y5(1:end)-y5(1);
n=8;
frek=0.05;
[a,b]=butter(n,frek);
Y5=filtfilt(a,b,y5);
plot(t5,u5,t5,y5,t5,Y5)

g=diff(Y5);
gl=find(max(g)==g);
k=t5(gl);
kl=Y5(gl);
plot(t5,y5,t5,Y5,k,kl,'*')
h=t5(gl+1);
hl=Y5(gl+1);
plot(t5,y5,t5,Y5,k,kl,'*',h,hl,'*')

a=(hl-kl)/(h-k);
b=kl-k*a;

x=[10:0.1:125];
y=a*x+b;
plot(t5,y5,t5,Y5,k,kl,'*',h,hl,'*',x,y)
Z=Y5(end)-Y5(1)
yx1=0;
x1=(yx1-b)/a;
yx2=Z;

tu=x1;
tn=x2-x1;

Ds=5;
tu=tu+Ds;
fs=tu/tn
fn=0.218;
gn=0.271;
T=gn*tn
D=(fs-fn)*tn
u5=u5';
t5=t5';
y5=y5';
```

V.

Príloha obsahuje CD so zdrojovými súbormi k vytvorenému projektu priemyselnom riadiacom systéme **SIMATIC**, všetky použité m-súbory na identifikáciu, DDE komunikáciu a vykreslenie priebehov riadenia.

.