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

Vypracovala: **Erika Szakálová**

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

Autorka práce:	**Erika Szakálová (26486)**
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 elektronického 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 Bratislava - 5 -

Szakálová
Erika Szakálová
riešiteľka bakalárskej práce

prof. Ing. Dr. Miroslav Fikar
vedúci pracoviska

prof. Ing. Dr. Miroslav Fikar
garant študijného programu

Poďakovanie

Rada by som poďakovala Ing. Kataríne Vanekovej za odborné vedenie a rady pri realizácii bakalárskej práce. Chcem tiež poďakovať rodine za morálnu i hmotnú podporu počas štúdia.

Abstrakt

Bakalársky projekt sa zaoberá identifikáciou laboratórneho zariadenia predstavujúceho 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

Práca sa zaoberá identifikáciou systému a návrhmi regulátorov pre laboratórne zariadenie sústavy III. rádu. Počas vytvárania projektu som aktívne využívala priemyselný riadiaci systém SIMATIC, MATLAB a Microsoft Excel. Tvorbu samotného projektu som uskotočnila v programe STEP7 a vytváranie vizualizačnej obrazovky som realizovala v prostredí WinCC.

Posledné dva spomenuté programy (MATLAB, Microsoft Excel) slúžili na zber dát cez DDE rozhranie, čo bolo nevyhnutnou podmienkou pre realizáciu toho projektu.

Práca sa skladá zo šiestich hlavných častí.

Prvá časť práce sa zaoberá samotným opisom riadiaceho systému SIMATIC S7– 300. Druhá časť práce uvádza základné vlastnosti použitých programov. Tretia časť práce opisuje postup pri vytváraní projektu v programe STEP7. Štvrtá časť sa zaoberá s problematikou DDE komunikácie. Opisuje možnosti MATLABu pri komunikácii a prácu s časovačom v MATLABe. V piatej časti sa nachádza postup pri vytváraní vizualizačnej obrazovky a prepojenie príslušných programov pomocou tagov. Posledná časť mojej práce zahŕňa v sebe identifikáciu laboratórneho zariadenia, syntézy regulátorov, otestovanie vhodnosti navrhnutých regulátorov, spracovanie a vyhodnotenie získaných výsledkov.

1. Riadiaci systém SIMATIC S7 – 300

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

I. 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 štandartných PC kvôli požiadavkám na odolnosť voči teplotám vibráciám atď.

II. 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)

III. Prepojenie PC a pracovnej stanice **SIMATIC S7-300** 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.

IV. **STEP7** a **WinCC** je programátorský a vizualizačný software, ktorý slúži k programovaniu PLC, sprístupňuje dáta užívateľovi a je komfortnou alternatívou monitoringu a riadenia reálnych procesov [1].

2. Charakteristika použitých programov

Počas práce som využila programy **STEP7**, **WinCC**, **MATLAB** a **Microsoft Excel**. **STEP7** a **WinCC** sú súčasťou riadiaceho systému **SIMATIC S7-300**. **MATLAB** a **Microsoft Excel** slúžili na vytvorenie Dynamic Data Exchange (DDE) rozhrania, ktoré slúži na prenos dát z **WinCC** do **MATLAB**u, kde som pozbierala potrebné dáta.

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, Industrial Ethernet, resp. PTP.

2. 3. 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.

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

2. 4. 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 v takmer vo všetkých druhoch obchodných spoločností a vo všetkých odvetviach priemyslu [2].

3. Hlavné kroky pri vytváraní projektu

Pri vytváraní riadiaceho projektu, je dôležité prihliadať na jeho kompaktnosť, tak aby plnil požadované kritéria a ciele. Pri tvorbe riadiaceho projektu treba splniť nasledovné:

- *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 S7-300** 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 [3].

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** [4].

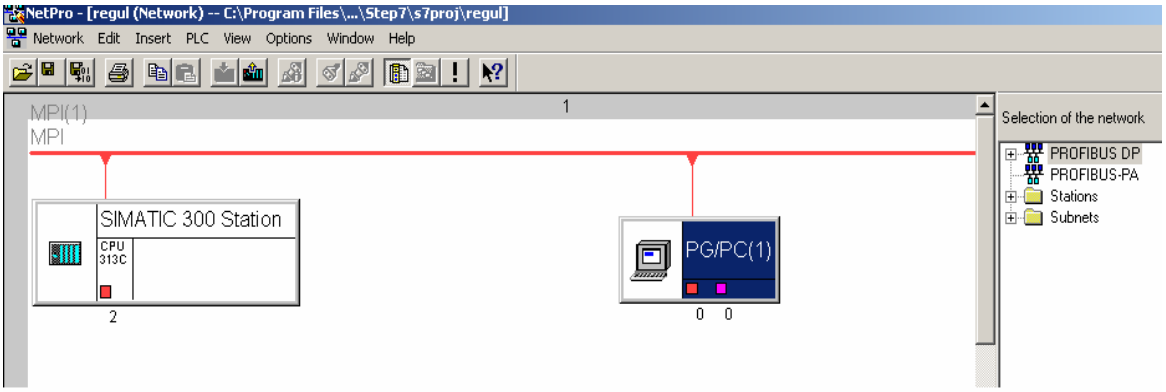
3. 2. Konfigurácia siete

V tejto časti sa vytvorí prepojenie počítača s pracovnou stanicou, zvolilo sa prepojenie cez MPI rozhranie. 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 [3].

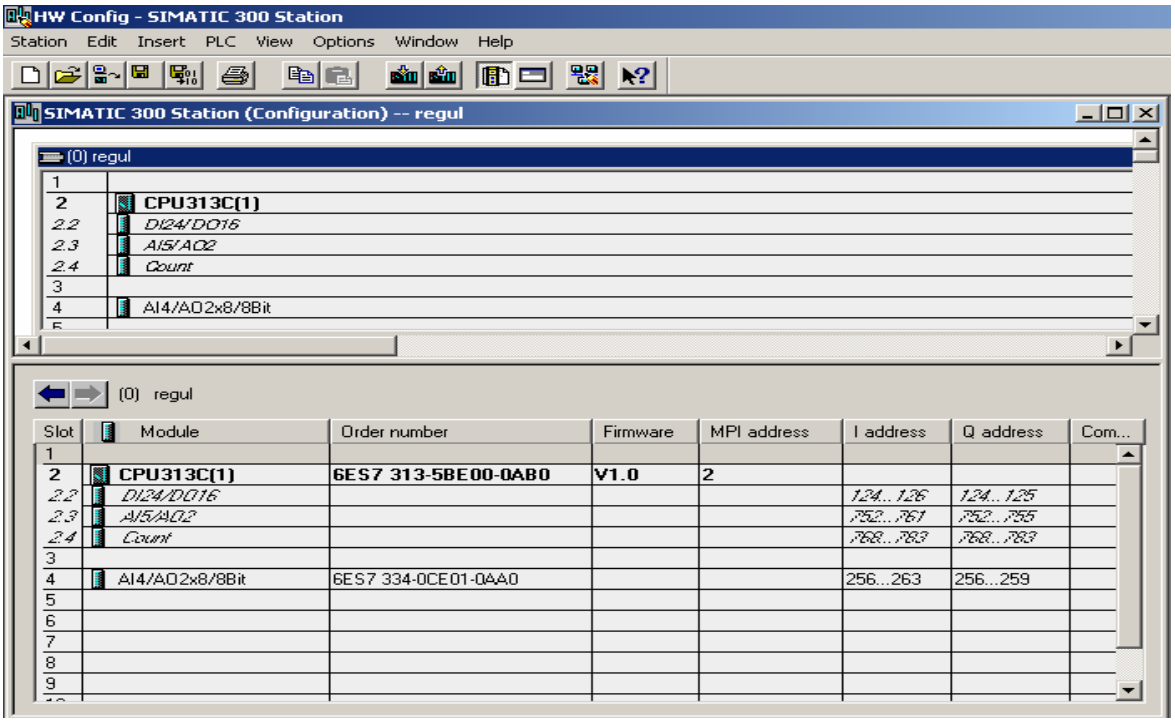
Základné spojenie pomocou MPI je možné vidieť na obr. 1.

3. 3. Konfigurácia I/O modulov

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 a tretí pre IM. Ak IM modul chýba, ostáva tretí slot neobsadený. Ostatné sloty slúžia na vstupno-výstupné (I/O)moduly.

3. 4. Programovanie logických blokov pomocou FBD

Na programovanie v **STEP7** som využívala **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 [3].
V programe **STEP7** sa využívajú nasledovné bloky:

- **OB** – organizačné bloky. Udávajú štruktúru užívateľského programu. Používala 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 [3].

Funkčné bloky

Funkčné bloky sa nachádzajú v knižnici blokov v programe **STEP7**. V práci som využila 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 je funkčný blok súčasťou organizačného bloku OB35 [5].

Funkčný blok FB41

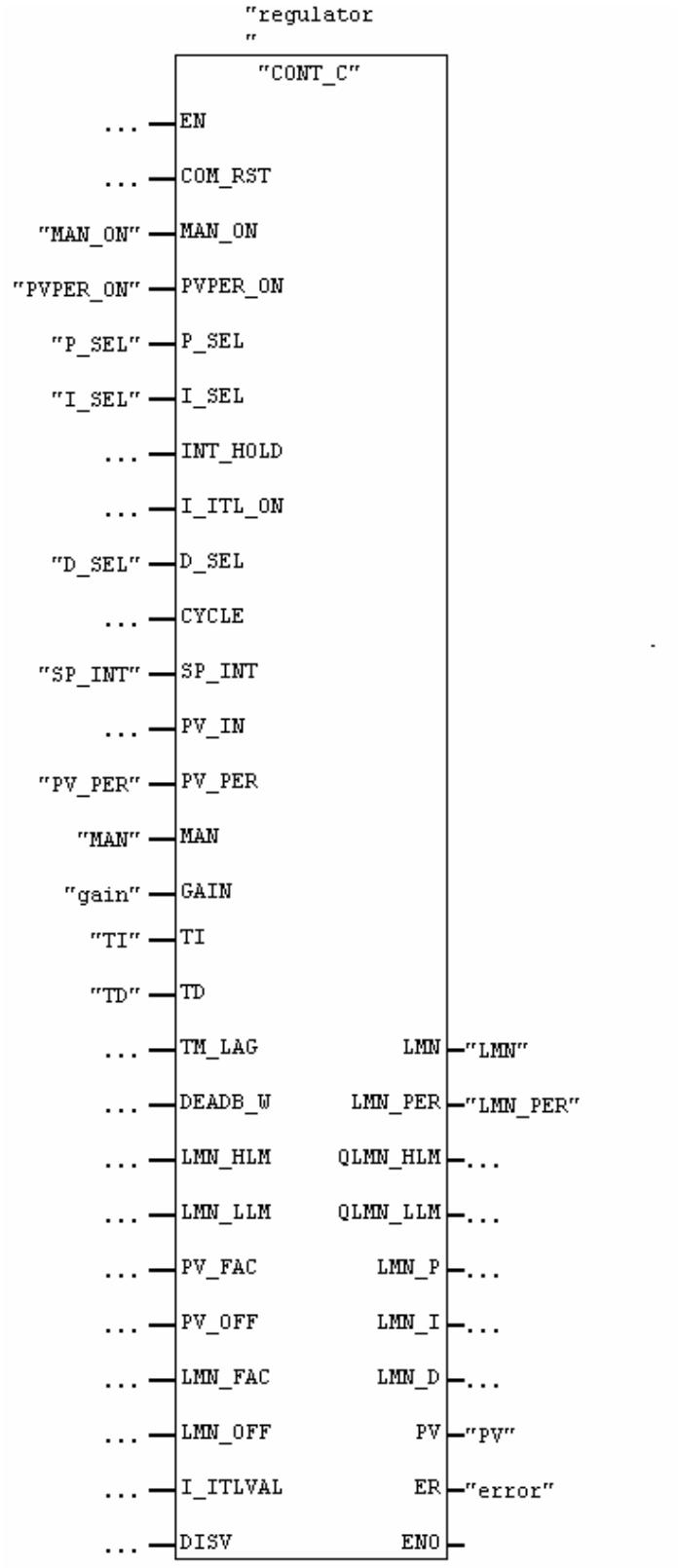
Na obr. 4 je zobrazený tvar PID regulátora vo forme funkčného bloku FB 41, ktorý sa nachádza v knižnici blokov programu **STEP7**, s jednotlivými vstupmi a výstupmi [6].

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:

- 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. 3 FB41

3. 5. 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 adresy typu WORD a v prípade boolovskej premennej Q 0.1 - 0.5. Symbolickú tabuľku je možné dopĺňať aj počas programovania jednotlivých blokov. Nie je potrebné pred programovaním zadefinovať všetky symbolické premenné, ktoré budú následne použité [3]. Symbolická tabuľka je zobrazená na obr. 4.

Symbol Editor - S7 Program(1) (Symbols)

Symbol Table Edit Insert View Options Window Help

All Symbols

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

	Status	Symbol	Address	Data type	Comment
1		CONT_C	FB 41	FB 41	Continuous Control
2		Cycle Execution	OB 1	OB 1	
3		Cyclic Interrupt 5	OB 35	OB 35	
4		D_SEL	Q 0.5	BOOL	
5		gain	MD 40	REAL	
6		I_SEL	Q 0.4	BOOL	
7		LMN	MD 10	REAL	
8		LMN_PER	PQW 752	WORD	
9		MAN	MD 20	REAL	
10		MAN_ON	Q 0.1	BOOL	
11		P_SEL	Q 0.3	BOOL	
12		PV	MD 30	REAL	
13		PV_PER	PIW 752	WORD	
14		PVPER_ON	Q 0.2	BOOL	
15		regulator	DB 1	FB 41	
16		SP_INT	MD 70	REAL	
17		TD	MD 60	TIME	
18		TI	MD 50	TIME	
19		error	MD 80	REAL	
20					

Obr. 4 Symbolická tabuľka pre sústavu III. rádu

3. 6. 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

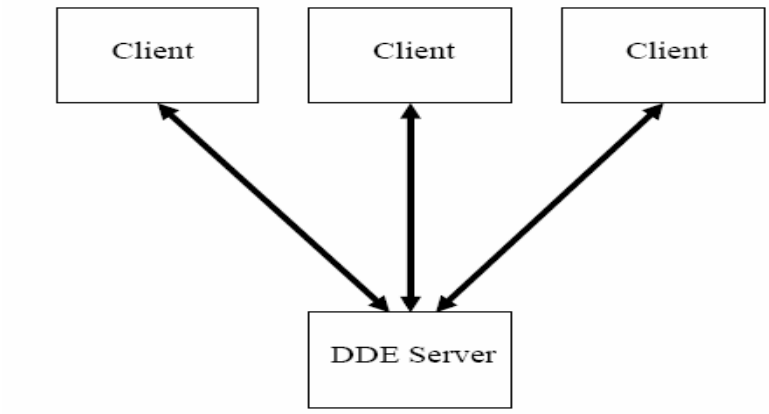
Pred spustením programu treba zapnúť zdroj pracovnej stanice do polohy ON, nastaviť kľúč CPU313 pracovnej stanice do polohy STOP. Pred nahraním programu do pracovnej stanice treba skontrolovať kontrolné LED diódy. Ak je všetko v poriadku tak by mala svietiť zelená LED dióda zapnutého zdroja, ďalej zelená LED (DC5V) a oranžová LED dióda pre nastavenie kľúča do polohy STOP. Ak sa teraz otočí kľúč do polohy RUN program sa spustí [6].

4. Opis protokolu Dynamic Data Exchange

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 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.

4. 1. Klient/Server princíp

- **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 (obr. 5).



Obr. 5 Možné spojenia medzi klientom a serverom

- **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.

V mojom prípade **MATLAB** a **WinCC** som využila ako Klient, **Microsoft Excel** ako Server. Použitý program pre spustenie DDE komunikácie medzi **MATLAB**om a **Microsoft Excel**om je uvedený v prílohe A a zobrazenie okna **Microsoft Excel** v prílohe C.

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

Sú tri rozdielne typu komunikačných liniek: **cold link**, **warm link** a **hot link.**

- **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.

- **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.

- **Warm link**

Priebežný príjem odkazov (ako opak k dátam) je spôsobený jednorázovou požiadavkou od Klienta. 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 [2].

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.

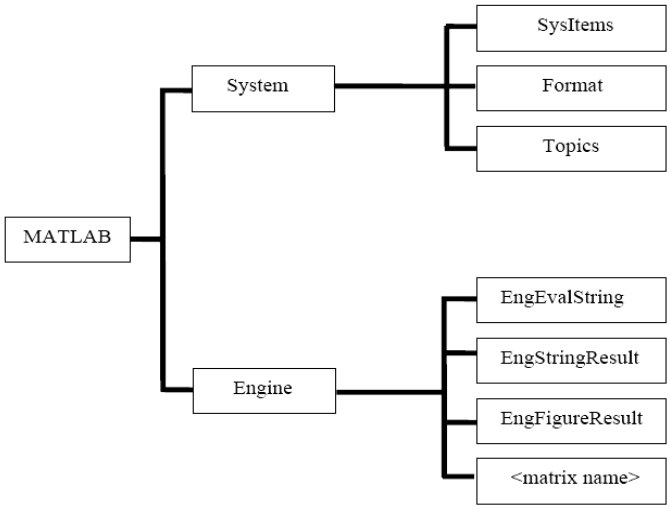
- **Služba** - Každá aplikácia, ktorá môže byť DDE serverom, má jednoznačný *Názov služby*. Zvyčajne je to názov spúšťacieho súboru aplikácie bez prípony.
- **Téma -** Definuje subjekt DDE konverzácie.
- **Prvok -** Každá téma obsahuje jeden alebo viac prvkov. Prvok identifikuje dáta posielané počas DDE konverzácie [3].

4. 4. Funkcie MATLABu pre prácu s DDE a časovačom

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“. Hierarchia DDE servera v **MATLAB**e je na obr. 6.



Obr. 6 Hierarchia DDE servera v **MATLAB**e

Charakteristika použitých pojmov:

- **SysItems** – poskytuje zoznam prvkov podporovaných témou „system“
- **Format** – poskytuje zoznam všetkých formátov, ktoré **MATLAB** podporuje ako Server
- **Topics** – poskytuje zoznam tém podporovaných Serverom
- **EngEvalString** – príkaz posielaný **MATLAB**u na vykonanie
- **EngStringResult** – výsledok vykonaného príkazu

- **EngFigureResult** – grafický výsledok vykonaného príkazu
- **<matrix name>** - názov premennej pri posielaní a prijímaní dát **MATLAB**om

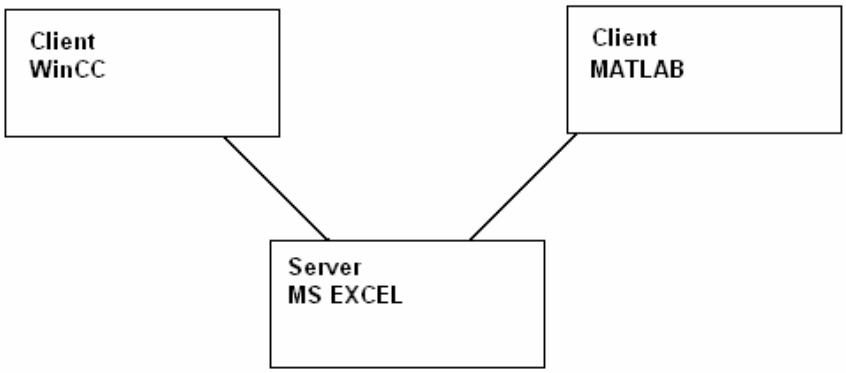
4. 4. 2. MATLAB ako DDE Klient

MATLAB je možné využívať pri DDE komunikácií ako Klienta. Pri tejto komunikácii sa využívajú príkazy, ktoré umožňujú pripojenie k zvolenému DDE Serveru.

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

V práci **MATLAB** som využila ako Klient, **Microsoft Excel** ako server a **WinCC** tiež ako klient. Komunikačná štruktúra je zobrazená na obr. 7.



Obr. 7 Komunikačná štruktúra

4. 4. 3. 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. 4. 4. Vlastnosti objektu časovača

Vlastnosti časovača možno rozdeliť do štyroch skupín:

- **Informačné parametre** – popisujú súčasný stav časovača. Tieto parametre užívateľ nemôže meniť.
  - **AveragePeriod** – určuje priemernú periódu vykonávania obslužnej funkcie
  - **InstantPeriod** – čas medzi poslednými dvomi vykonaniami obslužnej funkcie
  - **Running** – určuje aktuálny stav časovača („on“ alebo „off“)
  - **TaskExecuted** – určuje počet vykonaní obslužnej funkcie od spustenia
- **Parametre správania** – udávajú časové správanie sa časovača.
  - **TaskToExecute** – určuje koľko krát by mal časovač vykonať funkciu definovanú v TimerFcn
  - **ExecutionMode** – určuje spôsob zoraďovania obslužnej funkcie
  - **BusyMode** – určuje spôsob ďalšieho spracovania obslužnej funkcie, ak posledné spracovanie nebolo ukončené
  - **Period** – určuje periódu medzi dvoma vykonaniami obslužnej funkcie
  - **StartDelay** – určuje časový interval pred prvým vykonaním funkcie

v sekundách

- ❖ **Parametre obslužných funkcií** – špecifikujú funkcie, ktoré sa majú vykonať pri danej udalosti.
  - **ErrorFcn** – funkcia sa vykonáva v prípade chyby
  - **StartFcn** – funkcia sa vykonáva pri štarte časovača
  - **StopFcn** – funkcia sa vykonáva pri zastavení časovača
  - **TimerFcn** – obslužná funkcia sa vykonáva časovačom v každom kroku
- ❖ **Parametre definované užívateľom** – umožňujú pribaliť vlastné dáta.
  - **Name** – textový reťazec identifikujúci časovač
  - **Tag** – Užívateľom definovaný popis časovača
  - **UserData** – Užívateľom definované data

Použitý m-file pre spúšťanie časovača je uvedený v prílohe B.

5. Vytvorenie vizualizačného prostredia

Programu **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 komunikácií

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

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

Prepojenie **WinCC** s riadiacim programom **STEP7** sa robí 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**.

Tagy sú dvojakého typu:

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

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

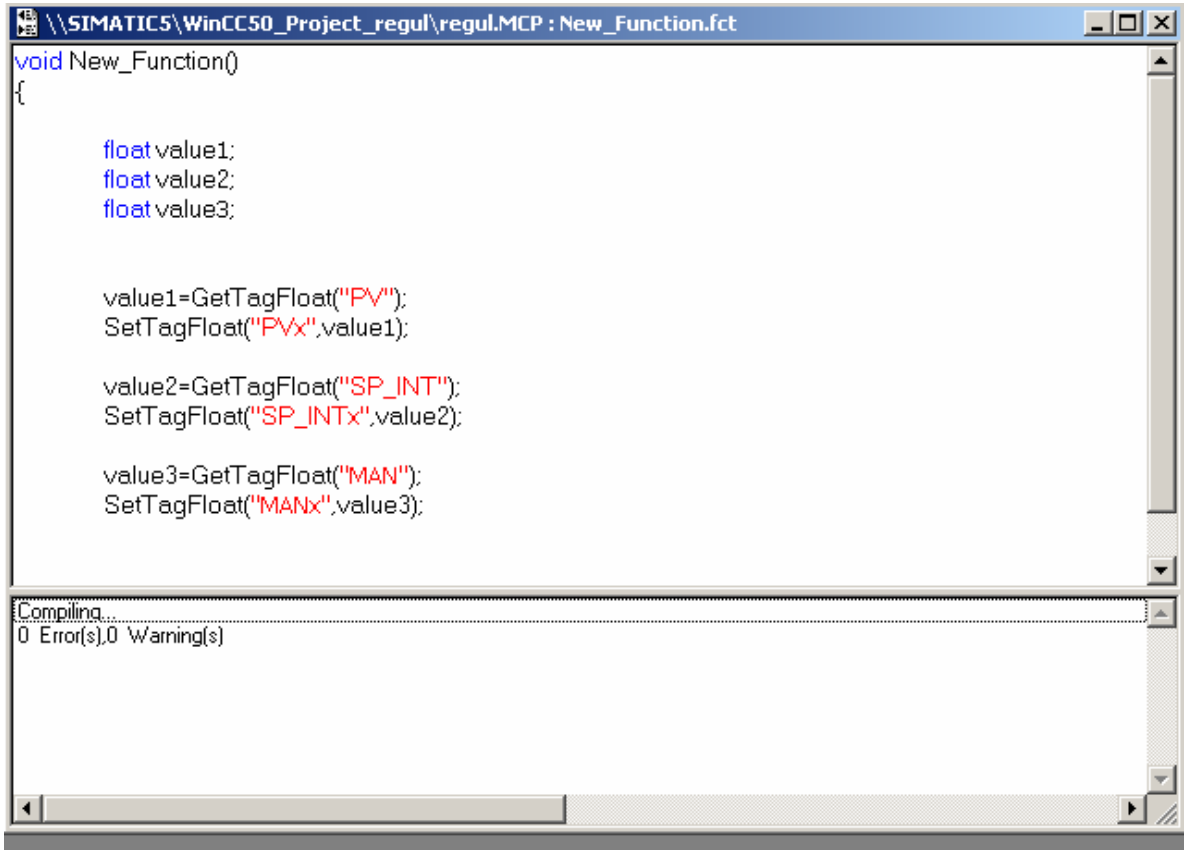
Prepojenie medzi **WinCC** a **Microsoft Excel** som uskutočnila pridávaním tagov v *Tag Managemen*te, kde sa pridá komunikačný driver *Windows DDE.CHN* a vytvorí sa rozhranie vo **WinCC** pre DDE komunikáciu. Pre DDE komunikáciu treba zadefinovať v *Connection Properties* aplikáciu. Keďže som **WinCC**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

Ešte bolo potrebné tagy spájajúce **WinCC** a **STEP7** s tagmi spájajúcimi **WinCC** a **Microsoft Excel** navzájom prepojiť. Bolo potrebné kopírovať hodnoty medzi tagmi. Kopírovanie jedného tagu do druhého bolo uskutočnené pomocou programu (*Project functions*), napísaného v module *Global Script*.

Nasledovné príkazy boli použté:

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

Na začiatku programu bolo potrebné zadefinovať typ premennej. Keďže išlo o čísla s desatinnými miestami, typ premennej bol *float*. Na obr. 8 je vizualizovaný *Global Script* s napísaným programom. Spustenie programu sa vykonáva vo vizualizačnej obrazovke pomocou tzv. *C-Action*, ktorá vykonáva programy vložené do nej.



Obr. 8 *Global Script* s napísaným programom

5. 3. Vytvorenie vizualizačnej obrazovky a ovládacích prvkov

Na vizualizáciu som použila 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. 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. Periódu obnovovania údajov som nastavila na *250 ms*. Ďalšie nastavenie vlastnosti objektu sa robí voľbou *Properties*.

Medzi hlavné nastavenia vstupno-výstupného poľa patrí:

- ***Field Type*** – umožňuje nastaviť typ poľa na Input, Output, Both, kde:
  - *Input (vstupné)* – možno do poľa zapisovať hodnoty a tým riadiť proces 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
- ***Apply on Exit*** – aby bola vložená hodnota do daného poľa akceptovaná musí byť táto vlastnosť nastavená na *YES*

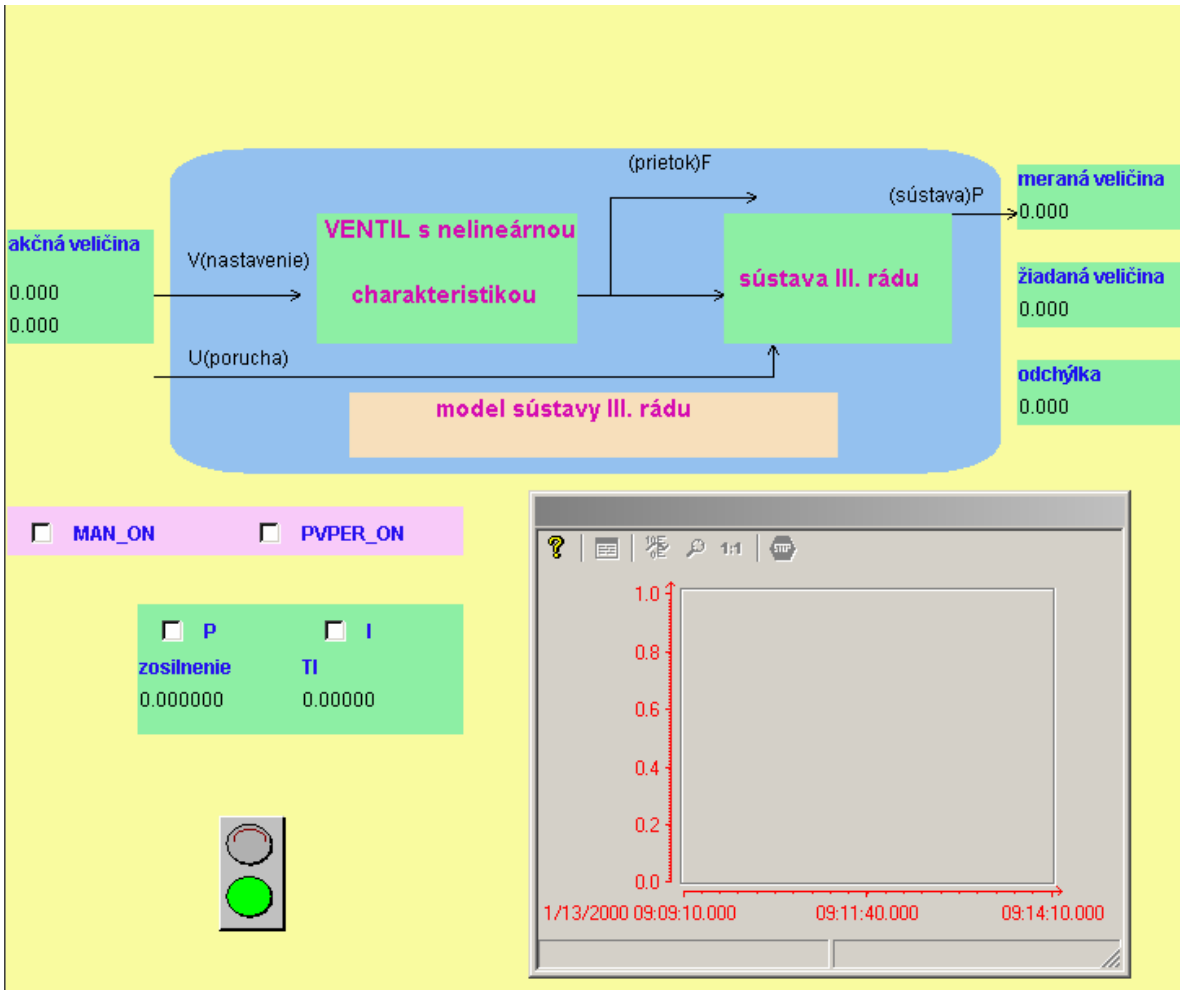
Každá vlastnosť môže byť statická alebo dynamická. Statická vlastnosť sa dá meniť iba v *Graphics Designer*i, hodnota dynamickej vlastnosti je daná hodnotou tagu.

Ďalším ovládacím prvkom, ktorý som využila 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 použila indicator kopírovania hodnôt *On_Off_3*. Indikátor som umiestnila kliknutím na ikonu *Display Library* . V položke Properties cez Tag Assigment vybrala som pre dynamickú vlastnosť položku *C-Action* v ktorej som definovala 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*. Po jeho umiestnení na pracovnú plochu je potrebné nastaviť pripojenie na archívny súbor. Ak sú nastavené všetky požadované vlastnosti, potom počas merania je

graf schopný vykresliť priebeh akčnej, meranej a žiadanej veličiny. Vytvorené vizualizačné okno možno vidieť na obr. 9.



.

Obr. 9 Vizualizačná obrazovka pre laboratórne zariadenie

6. Návrhy regulátorov

6. 1. Spracovanie prechodovej charakteristiky

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. Pred uskutočnením skokovej zmeny je nutné, aby bol skúmaný systém v ustálenom stave. 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 vymedzí dva časové údaje: čas prieťahu *tu* a čas nábehu *tn*. Okrem toho môžeme z PCH určiť hodnotu zosilnenia systému daného vzťahom:

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

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. Na základe tejto skutočnosti som v oblasti 0% - 20% akčného zásahu zistila odozvy na skokové zmeny akčnej veličiny (obr. 10). Systém som identifikovala pomocou Strejcovej metódy.

6. 2. Strejcova metóda identifikácie systémov

Dynamické vlastnosti identifikovaného systému aproximujeme pomocou náhradného prenosu v tvare

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

kde

Z - je zosilnenie

T - časová kosnštanta

D – dopravné oneskorenie systému

n – rád systému.

Systém som identifikovala na základe PCH na obr. 11 na 2. rádu s dopravným oneskorením, parametre procesu sú získané pomocou m-súboru uvedeného v prílohe D:

T = 31,6936 s

Z = 4,5741

D = 9,2953 s

Prenosová funkcia má potom tvar :

$$G(s) = \frac{4{,}5741}{(31{,}6936s + 1)^2} e^{-9{,}2953s} \tag{3}$$

6. 3. Metódy syntézy regulátora s využitím prechodovej charakteristiky

Pri návrhu regulátorov mojím cieľom bolo odstránenie trvalej regulačnej odchýlky, preto som navrhla PI regulátory.

A) Naslinova metóda syntézy regulátora

Regulátor sa navrhuje na základe požiadavky maximálneho preregulovania [7]. Vychádza z koeficientov CHR uzavretého obvodu tak, aby medzi všetkými 3 za sebou nasledujúcimi koeficientami CHR platila vzájomná súvislosť [8]:

$$a^2{}_i = \alpha a_{i-1} a_{i+1} \tag{4}$$

Maximálne preregulovanie som si zvolila 3%.

Vypočítané parametre regulátora:

Z_R = 0,1789

T_I = 34,5143 s

Funkčnosť regulátora, navrhnutý s Naslinovou metódou som vyskúšala pri nižšej hodnote žiadanej veličiny, ktorej priebeh je vykreslený na obr.12. a pri hodnote žiadanej veličiny nachádzajúcej sa mimo identifikovanej oblasti, ktorej priebeh je znázornený na obr. 14. Akčné veličiny sú zobrazené na obr. 13,15.

B) Metóda umiestnenia pólov

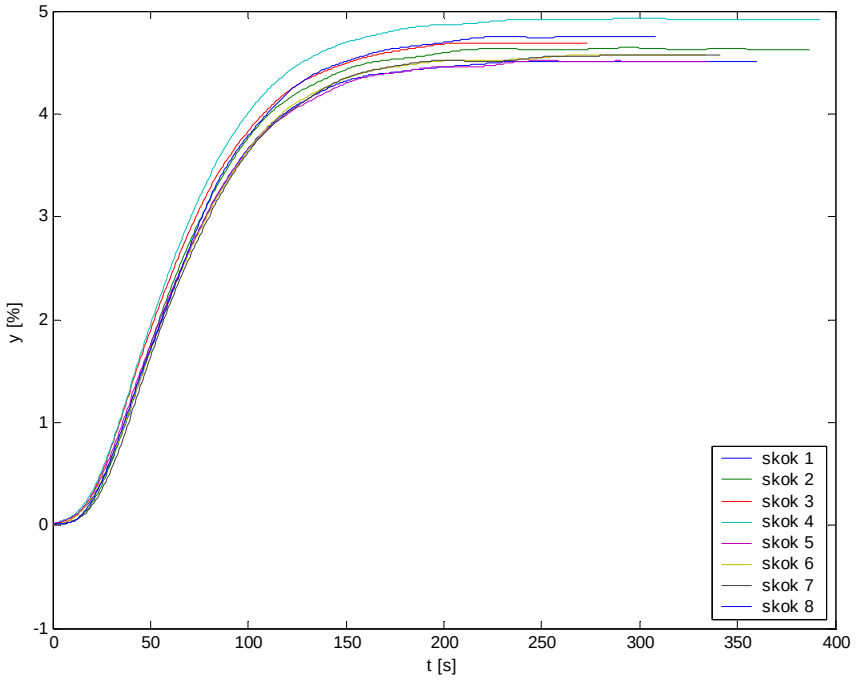
Hlavnou myšlienkou tejto metódy je voľba pólov CHR, čím sa predurčí dynamické správanie sa systému. Na základe vypočítaného koreňa CHR systému s = -0,0316 som si zvolila pól s1 = -0,033, ktorý je posunutý viac doľava od imaginárnej osi ako je pól riadeného systému.

Vypočítané parametre regulátora:

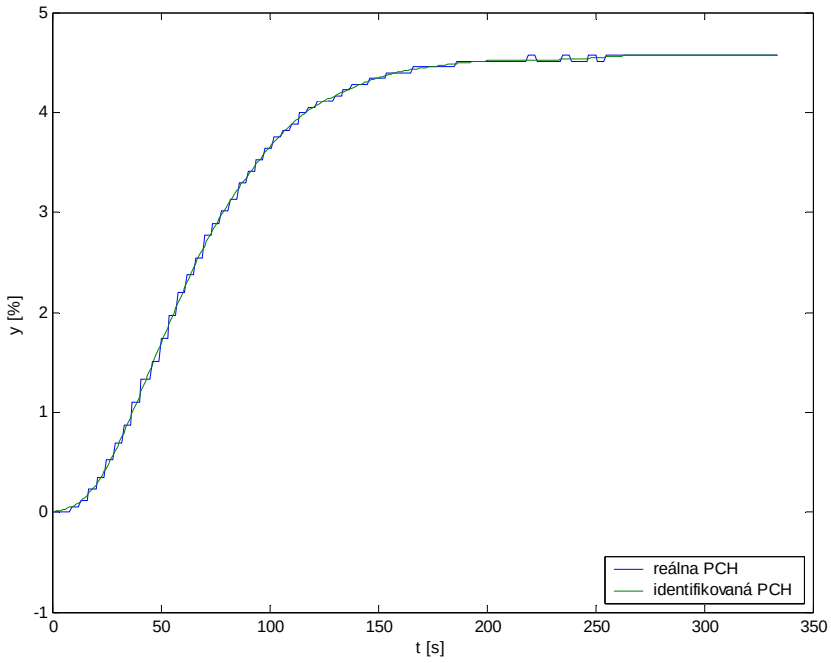
Z_R= 0,4988

T_I = 69,2044 s

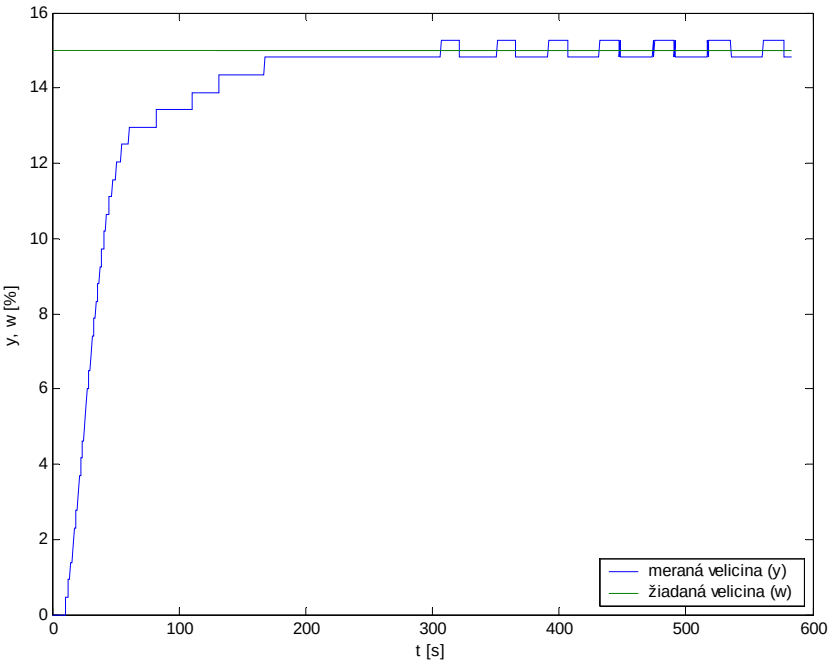
Priebehy riadenia elektronického modelu pomocou navrhnutého PI regulátora sú zobrazené na obr. 16,18 a príslušné priebehy akčných veličín sú na obr. 17,19.



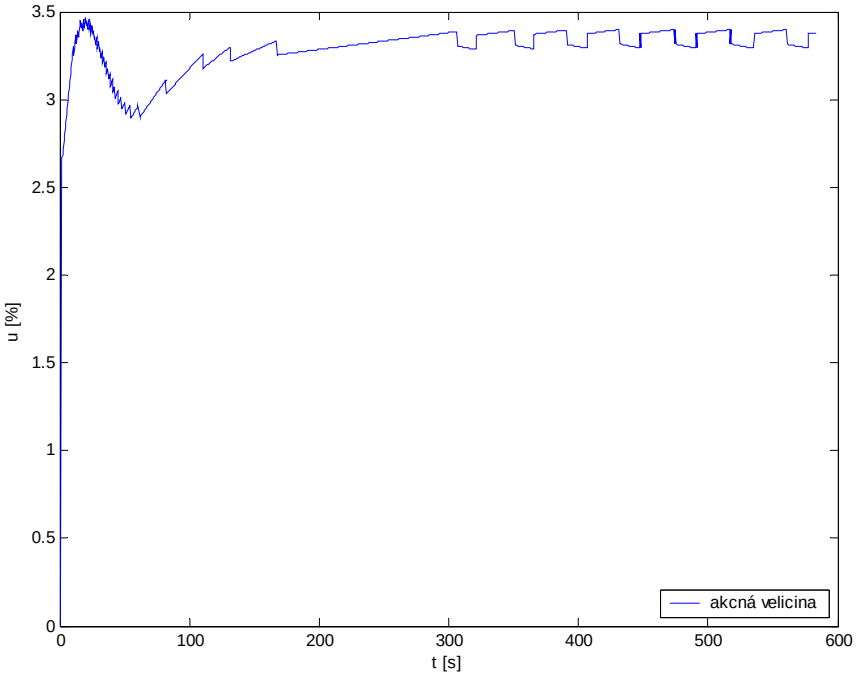
Obr. 10 Odozvy na skokové zmeny akčnej veličiny



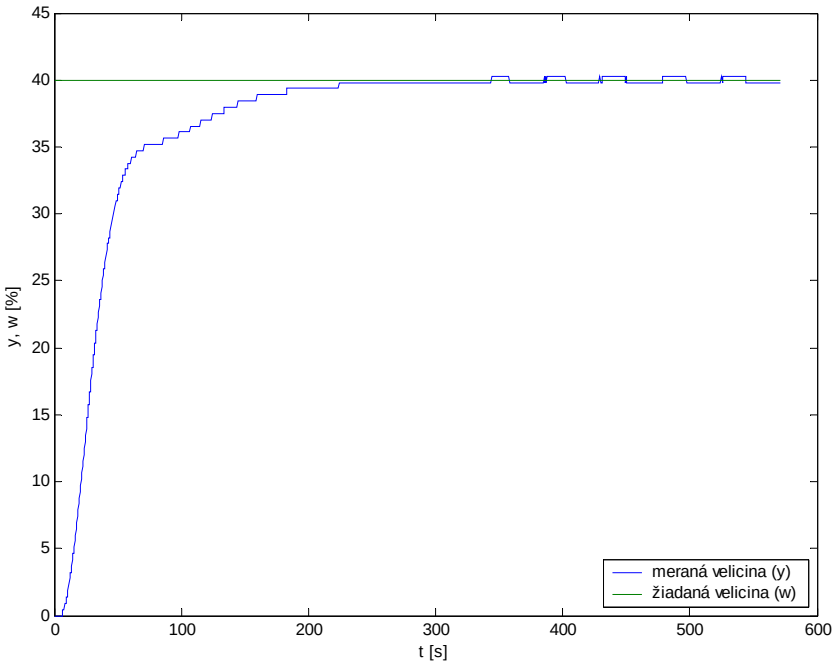
Obr. 11 PCH reálneho a identifikovaného systému



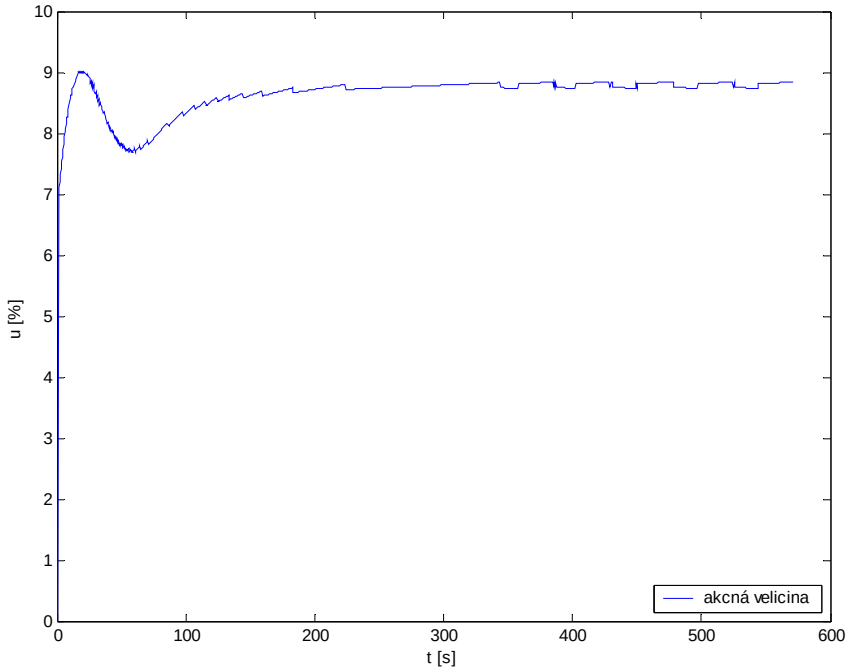
Obr. 12 Priebeh riadenia Naslinovou metódou v identifikovanej oblasti (w = 15%)



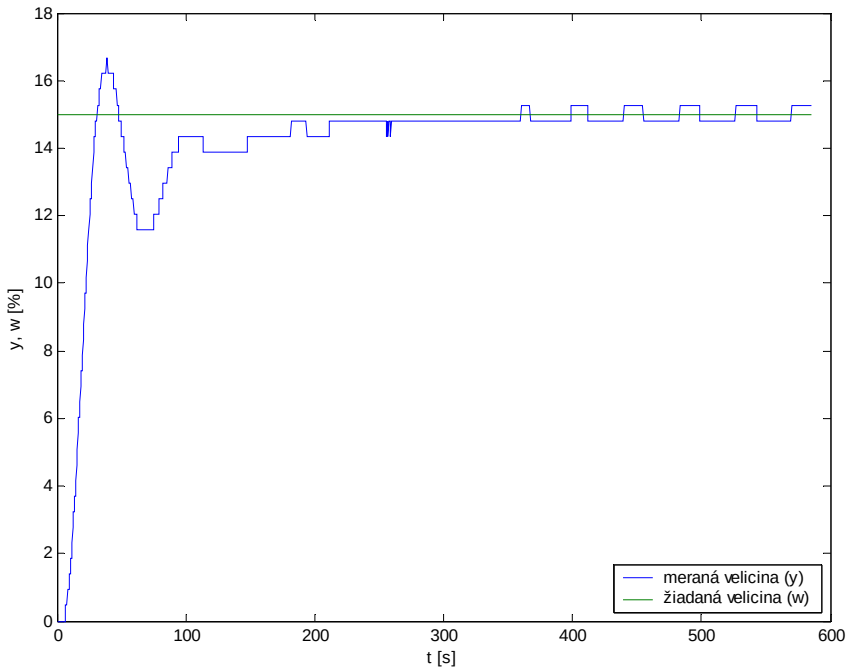
Obr. 13 Priebeh akčnej veličiny pre žiadanú veličinu w = 15% (Naslinova metóda)



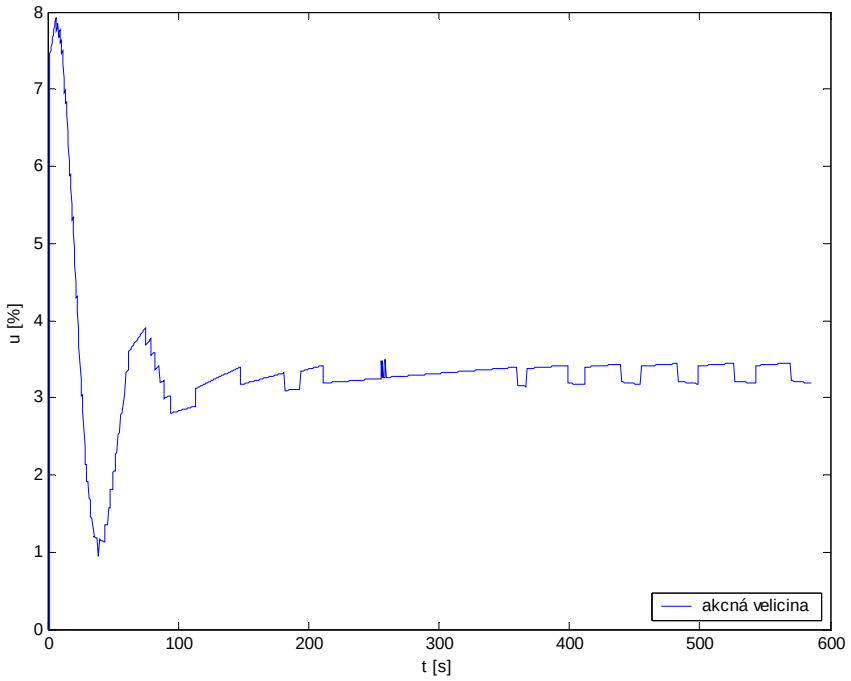
Obr. 14 Priebeh riadenia Naslinovou metódou mimo identifikovanej oblasti (w = 40%)



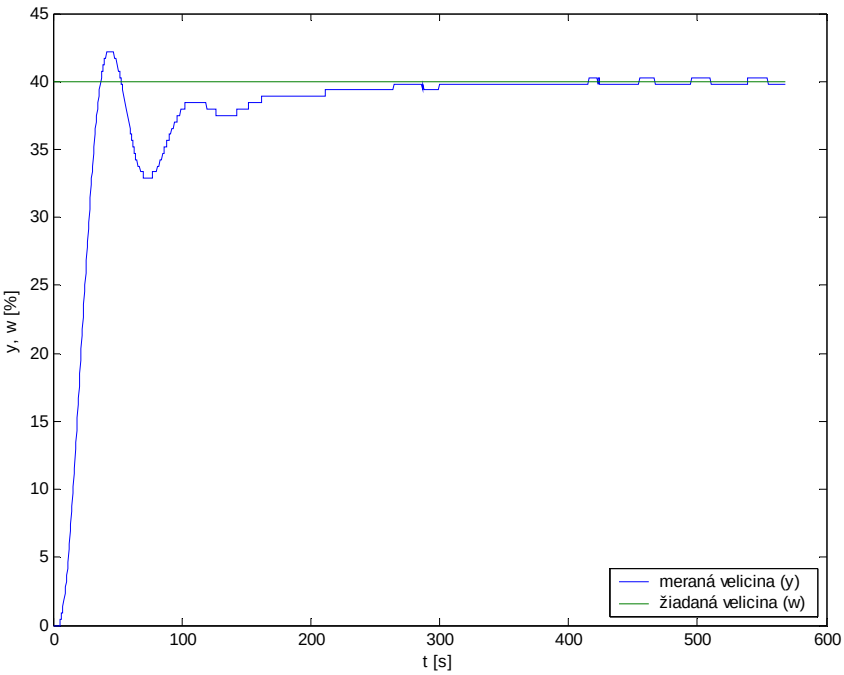
Obr. 15 Priebeh akčnej veličiny pre žiadanú veličinu w = 40% (Naslinova metóda)



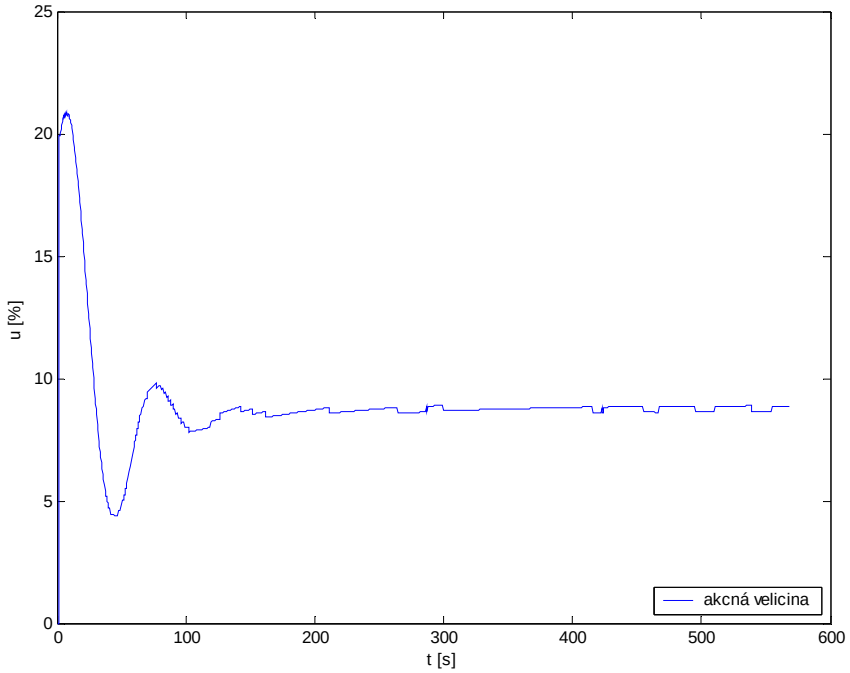
Obr. 16 Priebeh riadenia metódou umiestnenia pólov v identifikovanej oblasti(w = 15%)



Obr. 17 Priebeh akčnej veličiny pre žiadanú veličinu w = 15% (metóda umiestnenia pólov)



Obr. 18 Priebeh riadenia metódou umiestnenia pólov mimo identifikovanej oblasti (w = 40%)



Obr. 19 Priebeh akčnej veličiny pre žiadanú veličinu w = 40% (metóda umiestnenia pólov)

6. 4. Vyhodnotenie riadenia laboratórneho zariadenia

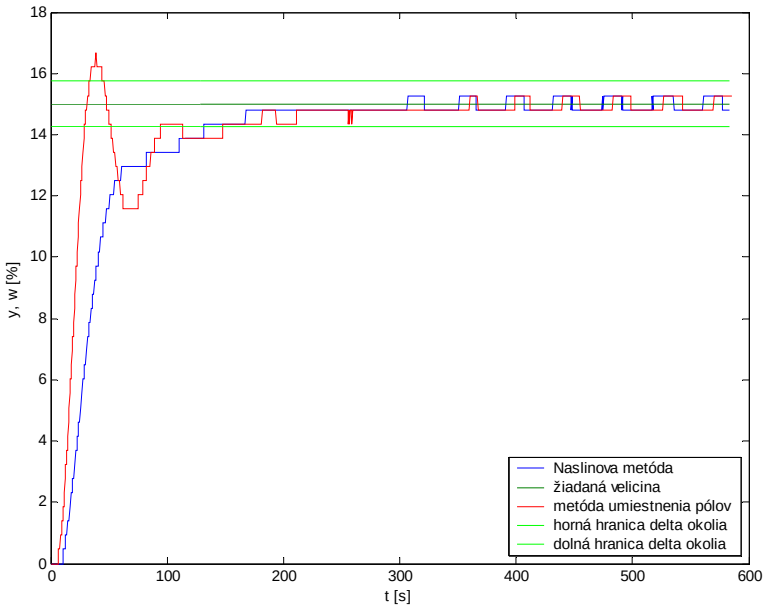
Pomocou získaných priebehov riadenia (obr. 20,21) som určila niektoré ukazovatele kvality riadenia a to:

- Čas regulácie
- Maximálne preregulovanie
- Čas maximálneho preregulovania

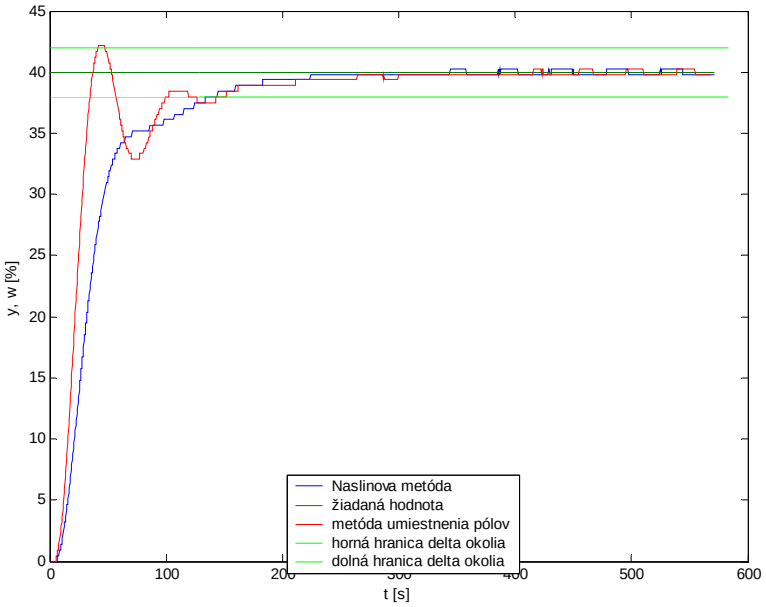
Pre určenie ukazovateľov kvality som si zvolila hodnotu δ okolia ± 5% žiadanej veličiny (w). Výsledky vyhodnotenia pre jednotlivé metódy sú zobrazené v tab. 1. Z uvedených výsledkov jednoznačne vyplýva, že Naslinovou metódou navrhnutý PI regulátor je vhodnejší.

Tab. 1 Ukazovatele kvality

Ukazovatele kvality	Naslinova metóda w = 15%	Naslinova metóda w = 40%	Metóda umiestnenia pólov w = 15%	Metóda umiestnenia pólov w = 40%
Čas regulácie [s]	131,89	144,04	147,89	151,4
Maximálne preregulovanie [%]	0	0	11,13	5,33
Čas maximálneho preregulovania[s]	–	–	38,75	44,5



Obr. 20 Priebeh riadenia Naslinovou metódou a metódou umiestnenia pólov v identifikovanej oblasti(w = 15%)



Obr. 21 Priebeh riadenia Naslinovou metódou a metódou umiestnenia pólov mimo identifikovanej oblasti (w = 40%)

Záver

Mojou úlohou bolo identifikovanie systému a následný návrh regulátorov. Aktívne som využívala vizualizačné prostredie WinCC, čo je neoddeliteľnou súčasťou tohto softwaru. Práve pomocou už spomenutého vizualizačného prostredia sa mi podarilo otestovať resp. vyskúšať vypočítané parametre mnou navrhnutých regulátorov. Celý proces návrhu regulátorov predbehla identifikácia systému, ktorú som uskutočnila pomocou Strejcovej metódy. Princíp Strejcovej metódy spočíva v nájdení inflexného bodu a následným nakreslením dotyčnice prechádzajúcej cez pracovný bod. Identifikáciu nasledoval návrh regulátorov. Funkčnosť obidvoch navrhnutých regulátorov som si overila aj v identifikovanej aj mimo identifikovanej oblasti. Na základe získaných priebehov riadenia, spomenutých v kapitole 6, som zistila, že navrhnuté regulátory odstránia trvalú regulačnú odchýlku pri zadaní žiadanej veličiny nachádzajúcej sa v identifikovanej aj mimo identifikovanej oblasti. Vyhodnotenie priebehov riadenia na základe ukazovateľov kvality je uvedené v kapitole 6. 4 v tab. 1. Pomocou týchto ukazovateľov sa dá usúdiť, že regulátor navrhnutý Naslinovou metódou je vhodnejší pre systém III. rádu.

Literatúra

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

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

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

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

[5] SIMATIC Manual 1999

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

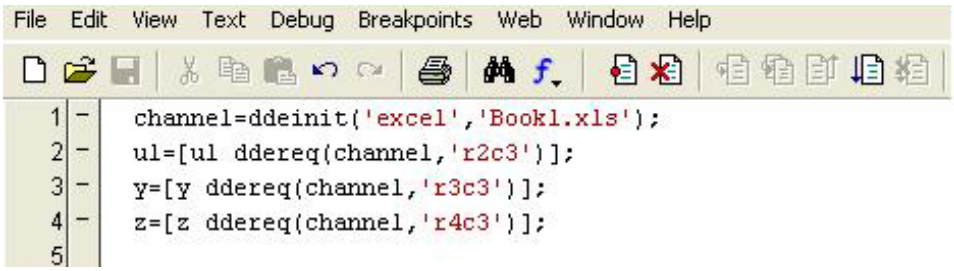
[7] Bakošová M., Fikar M., Čirka Ľ. :*Základy automatizácie*, STU, Bratislava 2003

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

Prílohy:

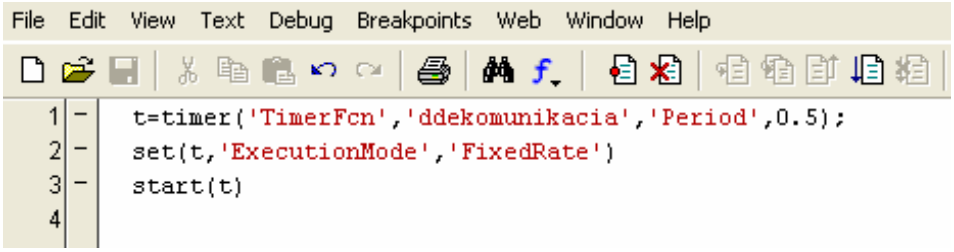
Príloha A

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



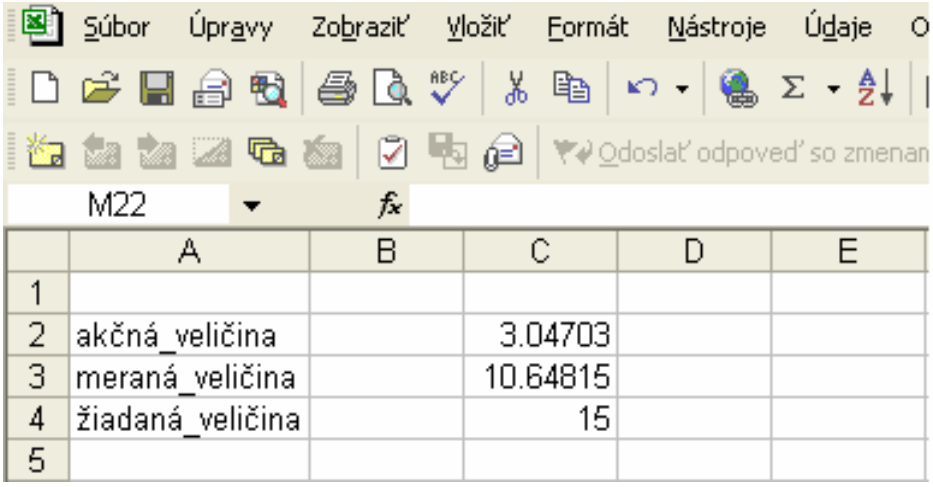
Príloha B

Výpis m-file súboru *timer.m*, pre prenos dát v požadovanom intervale.



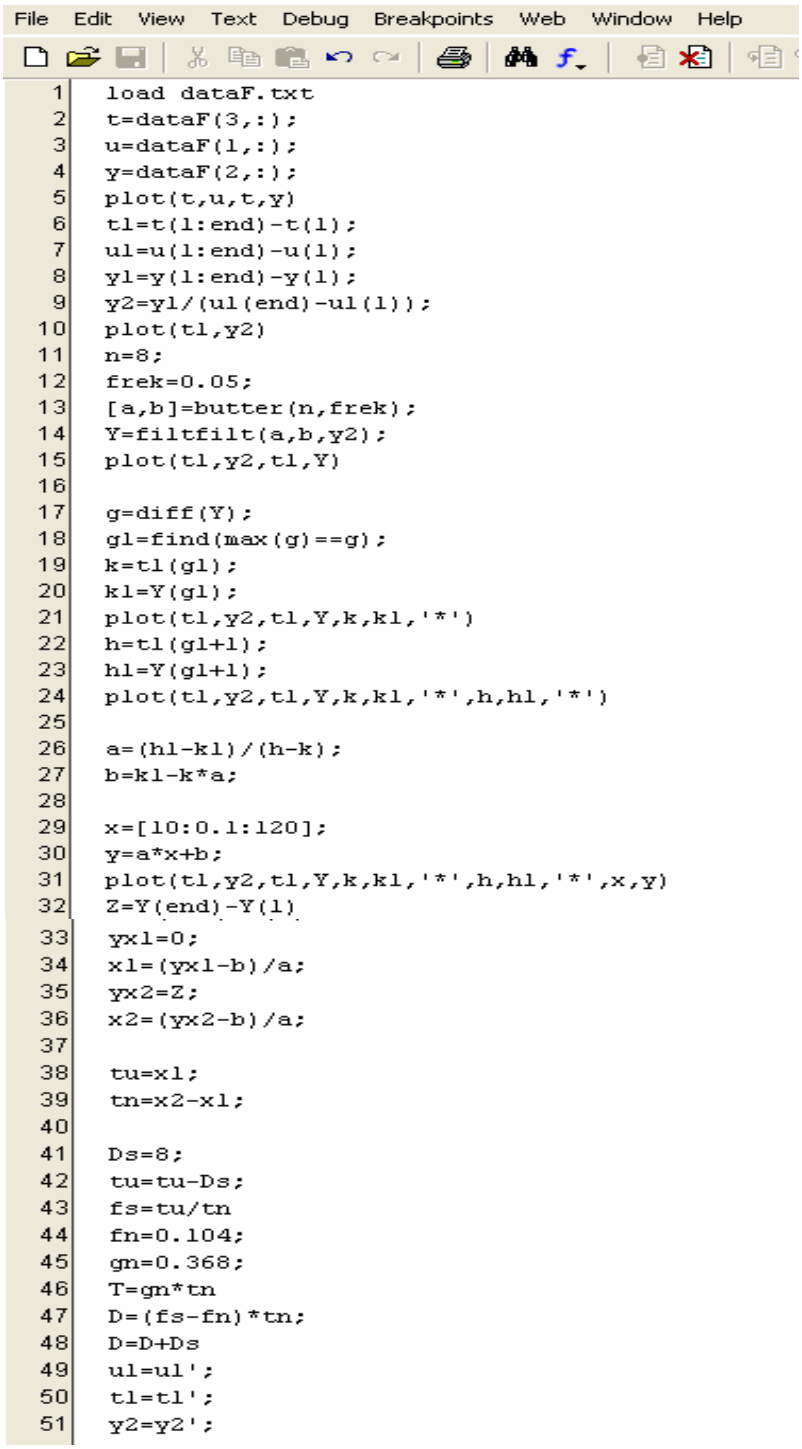
Príloha C

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



Príloha D

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



```
File  Edit  View  Text  Debug  Breakpoints  Web  Window  Help
 1  load dataF.txt
 2  t=dataF(3,:);
 3  u=dataF(1,:);
 4  y=dataF(2,:);
 5  plot(t,u,t,y)
 6  tl=t(1:end)-t(1);
 7  ul=u(1:end)-u(1);
 8  yl=y(1:end)-y(1);
 9  y2=yl/(ul(end)-ul(1));
10  plot(tl,y2)
11  n=8;
12  frek=0.05;
13  [a,b]=butter(n,frek);
14  Y=filtfilt(a,b,y2);
15  plot(tl,y2,tl,Y)
16
17  g=diff(Y);
18  gl=find(max(g)==g);
19  k=tl(gl);
20  kl=Y(gl);
21  plot(tl,y2,tl,Y,k,kl,'*')
22  h=tl(gl+1);
23  hl=Y(gl+1);
24  plot(tl,y2,tl,Y,k,kl,'*',h,hl,'*')
25
26  a=(hl-kl)/(h-k);
27  b=kl-k*a;
28
29  x=[10:0.1:120];
30  y=a*x+b;
31  plot(tl,y2,tl,Y,k,kl,'*',h,hl,'*',x,y)
32  Z=Y(end)-Y(1)
33  yx1=0;
34  x1=(yx1-b)/a;
35  yx2=Z;
36  x2=(yx2-b)/a;
37
38  tu=x1;
39  tn=x2-x1;
40
41  Ds=8;
42  tu=tu-Ds;
43  fs=tu/tn
44  fn=0.104;
45  gn=0.368;
46  T=gn*tn
47  D=(fs-fn)*tn;
48  D=D+Ds
49  ul=ul';
50  tl=tl';
51  y2=y2';
```

Príloha E

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.