

Praca dyplomowa inżynierska

Krzysztof Ślusarczyk

**Opracowanie, wykonanie i zbadanie środowiska do prezentacji
przemysłowego regulatora *fuzzy-logic*.**

Opiekun naukowy:

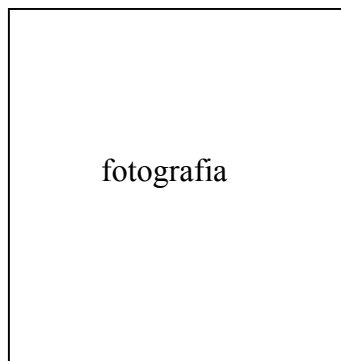
dr inż. Zygmunt Komor

Ocena _____

Podpis Przewodniczącego
Komisji Egzaminu Dyplomowego

Warszawa, wrzesień 2004

Krzysztof Ślusarczyk



Specjalność podstawowa: Komputerowe Systemy Sterowania

Data urodzenia: 29 lipca 1981

Data rozpoczęcia studiów: 1 października 2000

Życiorys

Urodziłem się 29 lipca 1981 roku w Nisku (woj. podkarpackie). W 1988 roku rozpocząłem naukę w Szkole Podstawowej nr 11 w Stalowej Woli. W okresie ośmiu lat nauki uczestniczyłem w olimpiadach: historycznej, fizycznej i matematycznej.

Kolejnym krokiem mojej edukacji było rozpoczęcie w 1996 roku nauki w Liceum Ogólnokształcącym im. Komisji Edukacji Narodowej w Stalowej Woli na kierunku matematyczno – fizycznym. Szkołę ukończyłem z wyróżnieniem.

Po zdaniu egzaminów maturalnych dostałem się na studia dzienne I stopnia, na Wydział Elektroniki i Technik Informatycznych Politechniki Warszawskiej na drugą grupę specjalności. Po dwóch latach studiów wybrałem specjalność Komputerowe Systemy Sterowania w Instytucie Automatyki i Informatyki Stosowanej gdzie obecnie kończę pracę dyplomową.

Interesuję się kulturą Stanów Zjednoczonych, czego wynikiem są dwie podróże do Nowego Jorku w 2002 i 2003 roku.

EGZAMIN DYPLOMOWY

Złożył egzamin dyplomowy w dniu: _____

z wynikiem: _____

Ogólny wynik studiów: _____

Dodatkowe wnioski i uwagi Komisji: _____

Spis Treści

1. Wstęp	5
2. Układ regulacji	6
2.1 Model obiektu.....	6
2.2 Połączenie z regulatorem.....	8
2.3 Regulator.....	8
3. Budowa regulatora LB – 600	9
4. Programowanie regulatora LB – 600	11
3.1 Zaprogramowanie regulatora do pracy z komputerem PC.....	11
3.2 Sposób połączenia funkcyj.....	11
3.3 Wykorzystanie najważniejszych zmiennych.....	12
5. Specyfikacja programu komputerowego	14
Wymogi programu komputerowego.....	14
6. Opis wykorzystanych funkcji regulatora	15
6.1 Samostrojenie.....	15
6.2 Regulacja rozmyta (ang. <i>fuzzy - logic</i>).....	17
7. Wybór najlepszego środowiska pracy	19
Środowiska brane pod uwagę.....	19
8. Uruchomienie węzła iFix	20
9. Konfiguracja programu iFix	23
10. Omówienie programu Symulator	28
10.1 Uruchomienie drajwera.....	28
10.2 Ekran główny.....	28
10.3 Opis przycisków znajdujących się w programie.....	29
10.4 Opis poszczególnych funkcji programu <i>Symulator</i>	32
10.5 Uruchomienie programu.....	34
10.5.1 Przygotowanie danych.....	34
10.5.2 Regulacja automatyczna.....	35
10.5.3 Regulacja ręczna.....	35
10.5.4 Regulacja rozmyta.....	36
10.5.5 Samostrojenie.....	36

11. Opis wybranych fragmentów kodu źródłowego	38
11.1 Opis głównej funkcji programu.....	38
11.2 Implementacja modelu obiektu.....	39
11.3 Funkcje odpowiedzialne za wciśnięcie przycisku.....	39
11.4 Opis testowania szybkości obliczeń.....	40
11.5 Implementacja mechanizmu opóźnienia.....	40
12. Testowanie programu	42
13. Współpraca z firmą LAB – EL	45
14. Zakończenie	45
15. Literatura	46
16. Załączniki	46
16.1 Przykładowe skrypty w Visual Basicu.....	46
16.2 Lista zmiennych użytych w programie <i>Symulator</i>	47
16.3 Spis rysunków.....	53
16.4 Zdjęcie stanowiska pracy.....	54
16.5 Oprogramowanie na płycie CD-ROM.....	54

Wstęp

Rosnąca konkurencja na rynku usług i rozwiązań z dziedziny automatyki wymusza na przedstawicielach firm już nie tylko zaprojektowanie dobrego urządzenia lub satysfakcjonującego rozwiązania danego problemu, lecz obejmuje również wyczerpujące zaprezentowanie i pokazanie wyższości swojego rozwiązania nad innymi.

Celem pracy inżynierskiej jest wykonanie środowiska do prezentacji przykładowego urządzenia z dziedziny automatyki jakim jest mikroprocesorowy regulator LB – 600 firmy LAB-EL. Umożliwia ona poznanie budowy urządzenia, sposobu jego programowania oraz uczy implementacji obiektów nieliniowych w wybranym języku programowania. Na potrzeby pracy stworzono program *Symulator*, który umożliwia pokazanie interesujących cech i właściwości urządzenia, stając się tym samym przydatnym narzędziem przy prezentacjach i pokazach reklamowych.

Praca daje możliwość poznania algorytmu regulacji rozmytej, wciąż niedocenianego w aplikacjach przemysłowych. Przeprowadzone symulacje potwierdzają jej wyższość nad zwykłymi algorytmami PID (por. rozdział 12: Testowanie programu). Najważniejszą jej cechą jest jednak możliwość przetestowania na dowolnej dynamice i nieliniowości obiektu. Użytkownik ma możliwość sprawdzenia zachowania się układu dla wybranych przez siebie wartości opisujących obiekt (ustawianych w programie lub zapisanych w pliku). Dodatkowo funkcja automatycznego doboru nastaw (samostrojenie regulatora) zaimplementowana w programie, pozwala na dobór parametrów regulatora nawet osobom bez specjalistycznej wiedzy z dziedziny automatyki. Program można również wykorzystać do badania odpowiedzi obiektu na skok wartości sterowania.

W pracy opieram się na regulacji rozmytej (ang. *fuzzy*) nieliniowego obiektu dwuinercyjnego, z opóźnieniem, zaimplementowanego w komputerze PC.

Abstract

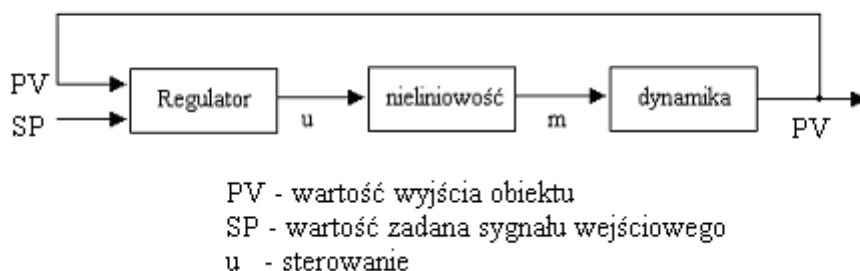
Rising competition on services market and solutions in automation technology makes company's representatives not only project good device or satisfuctional solution of a specific problem but also includes exhaustive presentation and showing superiority of our solution over elses.

The main aim of B.SC. thesis is to make an environment for presentation a hypothetical device from automation technology, such as LABEL's microchip controller LB –600. The application should make possible (should enable to show) showing some interesting features and characteristics of the device, thereby (thus) becoming useful tool in presentations and advertisement shows.

My work is based on *fuzzy – logic* control with nonlinear, two-inertia object with delay implemented on PC computer.

2. Układ regulacji

Układ regulacji składa się z regulatora LB – 600 firmy LAB-EL, modelu nieliniowości oraz modelu dynamiki obiektu. Warto zaznaczyć, że modele: nieliniowości oraz dynamiki zaimplementowane są w komputerze PC gdzie można dowolnie zmieniać wartości ich parametrów.



Rys. 1.1 Układ regulacji

2.1 Model obiektu

- dynamika

Dynamikę obiektu założono w postaci dwuinercyjnej z opóźnieniem. Transmittancja takiego obiektu wynosi:

$$G(s) = \frac{K_{ob} \cdot e^{-T_0 s}}{(1 + T_1 \cdot s) \cdot (1 + T_2 \cdot s)}, \text{ gdzie}$$

K_{ob} – wzmacnienie obiektu

T_0 – opóźnienie

T_1 – stała czasowa (1)

T_2 – stała czasowa (2)

(nazewnictwo zmiennych jest zgodne z nazwami w programie komputerowym).

Wybrany model oddaje charakter zmian dużej klasy rzeczywistych obiektów. Do dalszych rozważań wybrano przykładowo następujące wartości stałych czasowych:

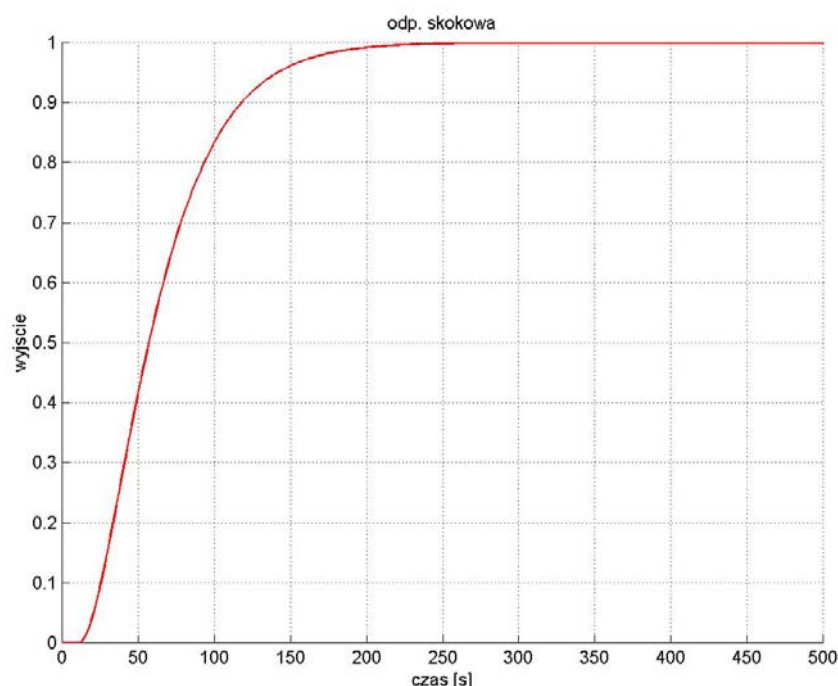
$$K_{ob} = 1$$

$$T_0 = 10s$$

$$T_1 = 25s$$

$$T_2 = 30s$$

Odpowiedź skokowa dla podanych wartości jest pokazana na Rys. 2.1



Rys. 2.1 Odpowiedź skokowa części dynamicznej obiektu

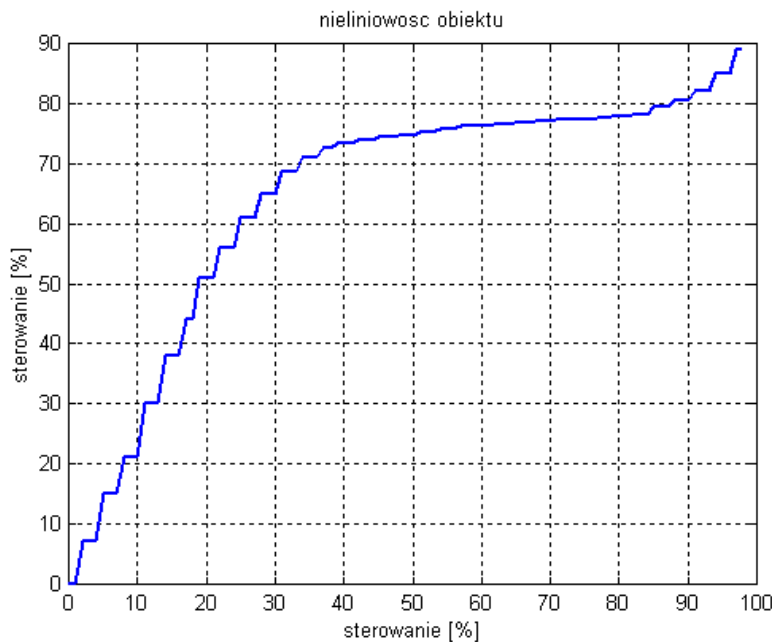
- **nieliniowość**

Użytkownik ma możliwość wprowadzenia dowolnej nieliniowości, którą zapisuje w pliku. Musi jednak pamiętać o dwóch warunkach:

1. dane muszą być zapisane w porządku niemalejącym
2. danych musi być dokładnie 35

Pierwszy warunek jest podyktowany warunkami poprawnej pracy regulatora. Warunek drugi służy uproszczeniu i ujednoczeniu wprowadzania danych. Spełnienie drugiego ograniczenia wprowadza możliwość powstawania błędów dokładności odtworzenia nieliniowości obiektu, lecz ilość 35-ciu zakresów jej zmian, jest zdaniem autora pracy wystarczająca do przybliżenia rzeczywistej nieliniowości.

Przykładowa część nieliniowa może wyglądać jak z Rys 2.2



Rys. 2.2. Przykładowa nieliniowość obiektu

Na osiach: poziomej i pionowej widnieje procentowy zakres sterowania. Użytkownik jest zobligowany do podania wartości sterowania w 35 równoodległych punktach na charakterystyce nieliniowości. Pomiędzy tymi punktami program dokonuje aproksymacji zerowego rzędu.

2.2 Połączenie z regulatorem

Połączenie komputera klasy PC z regulatorem jest realizowane przez pakiet łącza szeregowego. Wymiana danych odbywa się bez konieczności instalowania dodatkowego sprzętu w postaci przetworników c/a i a/c.

2.3 Regulator

Zadaniem regulatora jest odbieranie sygnałów z komputera (traktowanie ich jako wyjście modelu obiektu) a następnie na ich podstawie generowanie sygnału sterującego.

Podstawowym zadaniem przy wysyłaniu i odbieraniu sygnałów jest "ominięcie " części analogowej regulatora. W praktyce obliczone przez regulator sterowanie jest przetwarzane na postać analogową i dopiero wtedy pojawia się na wyjściu regulatora (przy czym wyjść może być kilka). W przyjętym przeze mnie rozwiązaniu zakładam, że przy połączeniu z komputerem wykorzystywany jest jedynie port RS – 232, nie będzie więc potrzeby odbierania sygnału analogowego i przekształcania go na cyfrowy.

Problem ten rozwiązano przez umiejętne wykorzystanie danych o oprogramowaniu regulatora (więcej na ten temat w rozdziale 4: Oprogramowanie regulatora LB – 600).

3. Budowa regulatora LB – 600

Regulator LB – 600 firmy LAB – EL stanowi zbiór swobodnie programowalnych bloków funkcjonalnych zwanych dalej *funktorami*. Każdy z funktorów jest elementem posiadającym wiele wejść oraz tylko jedno wyjście. Niektóre funktry mogą generować binarne sygnały alarmowe **AL** i **AH**. Funktory mogą realizować różne funkcje w zależności od potrzeby wynikającej ze specyfiki sterowanego procesu. Stanowią one swego rodzaju elementy macierzy, gdzie kolumny macierzy to warstwy, a wiersze macierzy to tory (kanały). Identyfikacja lokalizacji funktra polega na podaniu jego "adresu", czyli numeru warstwy oraz numeru toru (kanału).

Poszczególne warstwy związane są z obsługą pewnych funkcji regulatora; i tak:

- warstwa 0 – parametry generalne (hasła, adresy, zegary, alarmy czasowe, itp.)
- warstwa 1 – obsługa wejść analogowych
- warstwa 2 – obsługa wejść binarnych
- RSB (Rejestr Stanów Binarnych) – między innymi generowanie alarmów
- warstwy 3, 4, 5, 6 – obsługa funkcji arytmetycznych, logicznych i czasowych wielu zmiennych wejściowych
- warstwa 7 – obsługa właściwej funkcji regulatora (tryby i algorytmy regulacji)
- warstwa 8 – dodatkowy zbiór funktrów arytmetycznych dla sygnałów analogowych
- warstwa 9 – obsługa wyjść analogowych
- warstwa A – obsługa wyjściowych funktrów binarnych
- warstwa b – obsługa skalowania wejść i wyjść analogowych

Oznaczenie **31** określa funktra warstwy 3 z kanału (toru) 1 (Rys. 3.1)

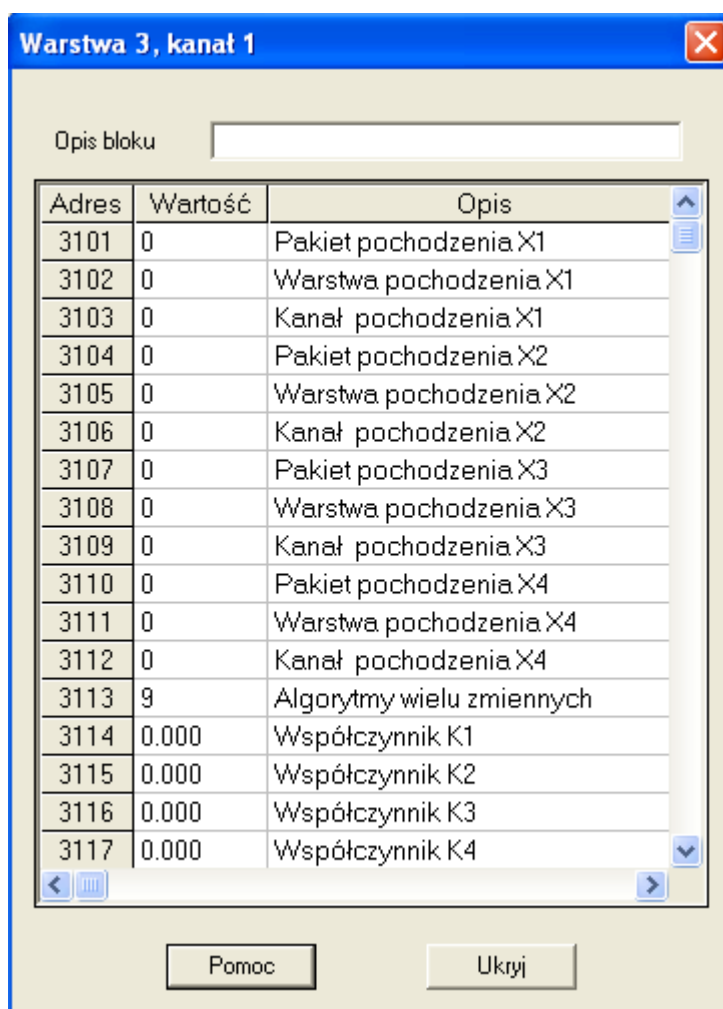


Rys. 3.1 Przykładowe oznaczenie funktra

Każdy parametr funktra można określić poprzez podanie numeru warstwy i kanału, z którego pochodzi dany funktra, oraz numeru tej zmiennej, np. współczynnik K1 z funktra **31** ma adres **3114**, a współczynnik K1 funktra **32** ma adres **3214**. Każdy adres ma przyporządkowany numer rejestru w pamięci regulatora. Spis wszystkich parametrów poszczególnych funktrów oraz odpowiadające im wartości adresów i rejestrów można odczytać z dokumentacji technicznej regulatora LB – 600.

Dla przykładu adres współczynnika K1 z warstwy 1 wynosi **3114** a jego numer rejestru **2913**.

Wartości parametrów regulatora można zapisywać i odczytywać przy pomocy programu *LB600mod* autorstwa firmy LAB-EL. Jest to szybki i wygodny sposób zaprogramowania całej struktury urządzenia. Innym sposobem zapisania parametrów do regulatora jest program *Diagram*. Przykładowy ekran wprowadzania danych z tego programu jest pokazany na *Rys. 3.2*



Warstwa 3, kanał 1

Opis bloku

Adres	Wartość	Opis
3101	0	Pakiet pochodzenia X1
3102	0	Warstwa pochodzenia X1
3103	0	Kanał pochodzenia X1
3104	0	Pakiet pochodzenia X2
3105	0	Warstwa pochodzenia X2
3106	0	Kanał pochodzenia X2
3107	0	Pakiet pochodzenia X3
3108	0	Warstwa pochodzenia X3
3109	0	Kanał pochodzenia X3
3110	0	Pakiet pochodzenia X4
3111	0	Warstwa pochodzenia X4
3112	0	Kanał pochodzenia X4
3113	9	Algorytmy wielu zmiennych
3114	0.000	Współczynnik K1
3115	0.000	Współczynnik K2
3116	0.000	Współczynnik K3
3117	0.000	Współczynnik K4

Pomoc Ukryj

Rys. 3.2 Zmienne funkтора 31

4. Programowanie regulatora LB – 600

4.1 Programowanie regulatora do pracy z komputerem PC

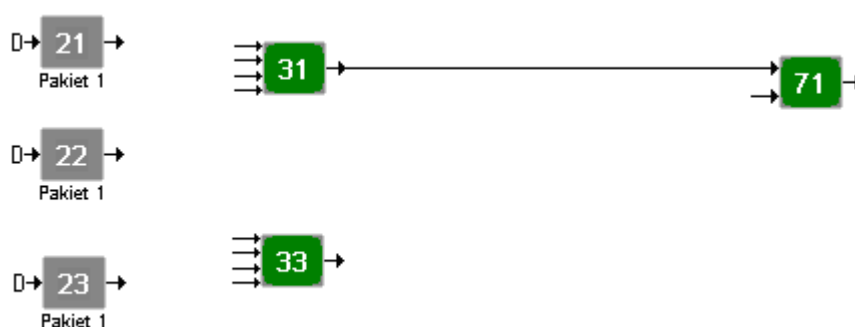
Program Symulator (wspomniany we Wstępie) umożliwia pracę z dowolnym regulatorem LB – 600 firmy LAB-EL. Konieczne jest jednak uprzednie zaprogramowanie go, tak aby stworzyć z funktorów strukturę regulacji. Służy do tego plik *struktura_lb600*. Przy pomocy oprogramowania firmy LAB – EL, użytkownik ma możliwość załadowania struktury do dowolnego regulatora, a następnie wykorzystanie go w pracy. Można również ręcznie programować regulator, lecz ze względu na liczbę zmiennych (szacunkowo około 90) wydaje się to mało praktyczne.

4.2 Sposób połączenia funktorów

Sposób połączenia funktorów odgrywa zasadnicze znaczenie przy programowaniu regulatora. Wynika to z faktu, że do regulatora nie są przyłączone żadne wejścia ani wyjścia (analogowe lub binarne). W standardowej konfiguracji regulatora, posiada on pakiet wejść/wyjść umieszczony w slotach obudowy regulatora. W przypadku wykorzystania do komunikacji jedynie portu RS – 232 komputera, zachodzi konieczność przekazywania sygnału wejściowego do regulatora w inny sposób – przez zaprogramowanie niżej opisanej struktury.

Wejściem dla funktora (71 – odpowiedzialnego za algorytm regulacji) jest sygnał przesyłany z wyjścia funktora 31. Jedną ze zmiennych funktora 31 jest współczynnik K_1 , który przyjmuje wartość obliczoną przez program *Symulator*, czyli wartość wyjścia obiektu dynamicznego. Nadając funktorowi 31 algorytm *wyjście = K_1* , wprowadzamy na wejście funktora 71 wyjście obiektu.

Ilustruje to rysunek 4.1.

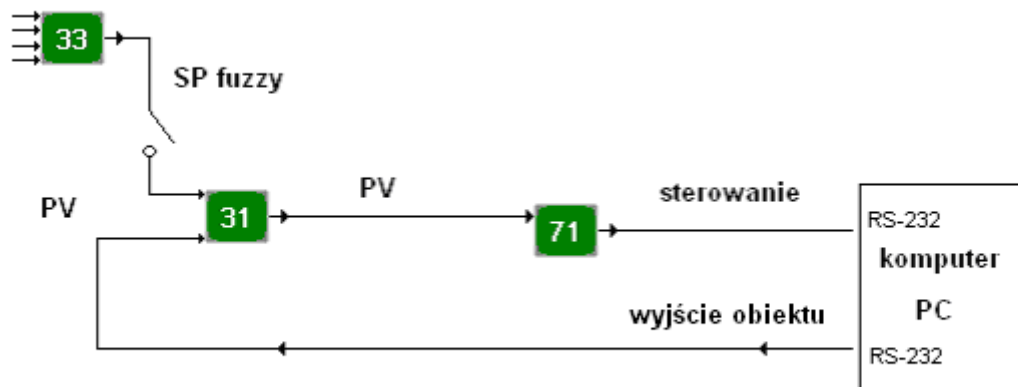


Rys. 4.1 Struktura funkcjonalna zaprogramowanego regulatora LB – 600

Wyjście funktora 33 stanowi wartość zadaną przy regulacji rozmytej.

Pojawienie się wartości 1 na wyjściu funktora 21 uruchamia, na wyjściu 22 wyłącza a na wyjściu 23 przerywa regulację rozmytą.

Schemat układu regulacji z wykorzystaniem funktorów przedstawiony jest na rysunku 4.2



Rys 4.2 Schemat układ regulacji rozmytej z wykorzystaniem funktorów

W programie *Symulator* wykorzystano jedynie funktory warstwy 2, 3 i 7. Dopiero zaprogramowane i połączone ze sobą stanowią strukturę umożliwiającą poprawne uruchomienie programu i przeprowadzenie prezentacji.

4.3 Wykorzystanie najważniejszych funktorów

Funktory warstwy 2 odpowiadają za uruchomienie i zatrzymanie regulacji rozmytej oraz samostrojenia (por. 6.1, 6.2). Odpowiednia akcja jest przeprowadzana gdy na wyjściu danego funktora pojawia się logiczna jedynka.

Funktor 21 jest odpowiedzialny za **START** wyżej wspomnianych procedur. Opis ważniejszych zmiennych funktora, wraz z ich wartościami, jest przedstawiony poniżej:

adres 2102	(logika wejścia dyskretnego)	: wartość 1
adres 2104	(aktywność bloku)	: wartość 1

Funktor 22 jest odpowiedzialny za **STOP** procedur.

adres 2202	(logika wejścia dyskretnego)	: wartość 1
adres 2204	(aktywność bloku)	: wartość 1

Do funktorów warstwy 3 zapisywane jest wyjście obiektu oraz wartość zadana dla regulacji rozmytej. Zapisu dokonuje program *Symulator*.

Wartość na wyjściu **funktora 31** to wyjście układu regulacji (obiektu)

adres 3113	(alg. wielu zmiennych)	: wartość 9
adres 3114	(współczynnik K_1)	: wartość zapisywana przez program <i>Symulator</i> (wyjście układu regulacji)

Wartość wyjścia **funktora 33** spełnia rolę wartości zadanej dla regulacji rozmytej.

adres 3313 (alg. wielu zmiennych) : wartość 9
adres 3314 (współczynnik K_1) : wartość zapisywana przez program *Symulator* (wartość zadana w regulacji rozmytej)

Funktor warstwy 7 odpowiada za algorytmy regulacji. Jako wejście należy mu podać miejsce pochodzenia (numer warstwy i kanału) wyjścia obiektu (wspomniany wyżej funktor **31**).

Poprawne zaprogramowanie **Funktora 71** odgrywa zasadniczą rolę w poprawnej pracy urządzenia. Funktor można niejako traktować jak regulator. Na wejście podaje się wartość zadaną oraz wyjście obiektu, a algorytm regulacji (ustawiany jako jedna ze zmiennych funktora) generuje wartości sterowania.

adres 7101 (pakiet pochodzenia PV) : wartość 1
adres 7102 (warstwa pochodzenia PV) : wartość 3
adres 7103 (kanał pochodzenia PV) : wartość 1
adres 7104 (pakiet pochodzenia SP) : wartość 1
adres 7105 (warstwa pochodzenia SP) : zależne od rodzaju regulacji
adres 7107 (wartość min PV) : wartość 0,0
adres 7108 (wartość max PV) : wartość 100,0
adres 7109 (rodzaj regulacji) : wartość 0
adres 7110 (algorytm regulacji) : wartość 1
adres 7120 (wartość zadana) : ustawiana przez użytkownika w regulacji ręcznej oraz PID
adres 7138 (początkowy nr tablicy) : wartość 1
adres 7142 (numer RSB startu) : wartość 320
adres 7143 (numer RSB stopu) : wartość 321
adres 7144 (numer RSB pauzy) : wartość 322

5. Specyfikacja programu komputerowego

5.1 Wymogi programowe

Działanie napisanego programu powinno przekonać potencjalnego nabywcę, że regulator rozmyty jest godny uwagi przy regulacji nieliniowych obiektów. Powinien pokazać przewagę nad innymi typami regulatorów m.in. zwykłym PID. Dzieje się to przez zmianę nastaw w zależności od nieliniowości obiektu. Program powinien być również "przyjazny" dla użytkownika.

W programie można podejrzeć między innymi:

- wyjście układu
- aktualną wartość zadaną
- typ regulacji, aktualne nastawy itp.

Ważną cechą programu musi być przenośność pomiędzy różnymi komputerami. Istotnego znaczenia nabiera szybkość wykonywania obliczeń przez jednostkę centralną komputera. W takim przypadku należy zastosować zmianę skali czasu. Ma ona na celu generowanie tych samych wartości sygnałów w jednakowych chwilach czasu.

Walory rynkowe, badawcze i dydaktyczne uzyskuje się poprzez możliwość wprowadzenia dowolnej (jednak zgodnej z założeniami projektowymi) nieliniowości modelu, odczytywanej z pliku. Odczytane dane można przedstawić na wykresie. W ten sposób prezentacja regulatora staje się interaktywna. Podobnie jak w przypadku testu szybkości, wczytanie danych o nieliniowości modelu powinno być warunkiem koniecznym uruchomienia całego programu.

W przypadku, gdy użytkownik nie zna modelu na tyle dobrze, aby sam potrafił określić poprawne nastawy regulatora, może uruchomić procedury samostrojzenia w regulatorze LB-600. Wysilek użytkownika systemu sprowadza się wtedy do przemyślanego wyboru punktów pracy i starannego przeprowadzenia eksperymentu samostrojzenia.

Do regulacji rozmytej w regulatorze LB – 600 wykorzystana jest tablica z wartościami zadanymi i nastawami dla poszczególnych punktów pracy. Użytkownik może skompletować te wartości poprzez własne eksperymenty (zaprogramowanie regulatora), samostrojzenie bądź wczytanie z pliku, a następnie wysłanie danych do regulatora.

Zasadniczą częścią przedstawiania wyników będzie rysowanie na bieżąco wykresu wartości wyjścia wraz z wartością zadaną i sterowaniem. Jest to podstawowe zadanie, które musi realizować program.

Jednocześnie wymóg bieżącego rysowania wartości sygnałów powodował znaczne zawężenie poszukiwanych środowisk programistycznych. W swoich poszukiwaniach kierowano się prostotą implementacji takiego wykresu oraz wynikiem końcowym, pod względem estetyki prezentowanego wyniku.

6. Opis wykorzystanych funkcji regulatora

W pracy wykorzystano najważniejsze funkcje regulatora LB – 600, świadczące o jego dużych możliwościach. Opisane funkcje zostały przetestowane z przykładowym modelem nieliniowym. Dokładny opis przeprowadzonych testów można znaleźć w punkcie 12: Testowanie programu.

6.1 Samostrojenie

Regulator LB – 600 posiada funkcję samostrojenia, pozwalającą na identyfikację obiektu i dobranie optymalnych nastaw regulatora PID wg:

- metody Cohena-Coona,
- wyboru jednego z trzech kryteriów jakości regulacji: (5% przeregulowania, 20% przeregulowania, minimum całki z kwadratu uchybu regulacji),
- zasady Zieglera-Nicholsa

W regulatorze LB-600 nastawy obliczane są automatycznie wg parametrów eksperymentu identyfikacyjnego. Po udanym eksperymencie identyfikacyjnym użytkownik otrzymuje obliczony zestaw nastaw PID, które automatycznie lub po uprzedniej akceptacji zapisywane są do tablic struktury regulatora. Po udanym eksperymencie identyfikacyjnym uzyskuje się nie nastawy regulatora, lecz parametry identyfikacyjne regulowanego obiektu (procesu):

- T - stała czasowa obiektu,
- k_o - wzmacnienie obiektu,
- T_o - czas opóźnienia obiektu

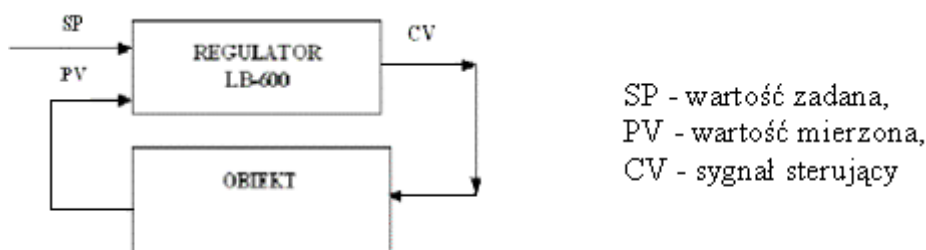
W ten sposób obiekt opisany równaniem transmitancji w postaci dwóch stałych czasowych zostaje opisany jedną stałą czasową wynikającą z przeprowadzonego eksperymentu samostrojenia

Na podstawie parametrów identyfikacyjnych oraz wyboru odpowiedniego kryterium jakościowego użytkownik współdecyduje o nastawach PID, jakie zostaną wprowadzone do struktury regulacyjnej.

Samostrojenie przeznaczone jest przede wszystkim dla obiektów statycznych, dla których spełniony jest warunek:

$$0,1 \leq \frac{\text{Opóźnienie obiektu}}{\text{Zastępcza stała czasowa}} \leq 0,6$$

Stosunek opóźnienia obiektu do jego zastępczej stałej czasowej mieści się w granicach od 0,1 do 0,6. Samostrojenia nie należy stosować dla obiektów o stałych czasowych mniejszych od kilkunastu sekund. Samostrojenie przeprowadza się w układzie zamkniętym tak jak na rys. 6.1



Rys 6.1 Schemat blokowy podstawowego układu regulacji

Eksperyment samostrojzenia polega na wprowadzeniu do sygnału sterującego (CV) okresowego zaburzenia o określonej amplitudzie. W trakcie eksperymentu sygnał sterujący będzie generowany symetrycznie wokół punktu pracy. Na podstawie zmian w sygnale regulowanym (PV), wywołanych zaburzeniem, algorytm samostrojzenia ustala nowe nastawy.

Warunkiem przeprowadzenia eksperymentu samostrojzenia jest doprowadzenie obiektu do stanu równowagi wokół punktu pracy, tzn. różnica PV-SP musi być mniejsza od ustalonej wartości. Doprowadzenie do stanu równowagi może być dokonane automatycznie (przy pomocy dotychczasowych nastaw) lub ręcznie.

Samostrojzenie składa się z trzech etapów:

- doprowadzenie do stanu równowagi,
- pomiaru poziomu zakłóceń,
- właściwego eksperymentu

Czas trwania pierwszego etapu zależy od szybkości osiągnięcia stanu równowagi. Czas trwania drugiego etapu jest stały i wynosi 2 minuty. Czas trwania trzeciego etapu, czyli właściwego eksperymentu zależy od dynamiki obiektu i nie przekracza wartości $6 \cdot T_z$, gdzie: T_z - zastępcza stała czasowa obiektu.

W przypadku stwierdzenia przez procedurę niemożliwości dobrania nastaw, algorytm się wyłącza i sygnalizuje odpowiednim kodem błędu. Eksperyment samostrojzenia można wyłączyć w dowolnym momencie. W przypadku pomyślnego zakończenia eksperymentu na wyświetlaczu można podejrzeć nowe nastawy.

Jeśli eksperyment zakończy się niepowodzeniem, pojawi się napis *Err*, a zamiast wartości nowych nastaw będą wyświetlone numery błędów.

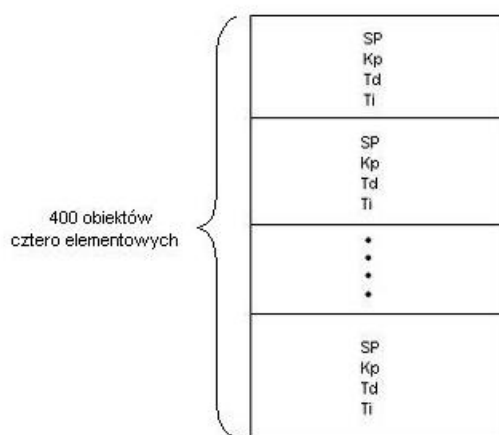
Warunkiem koniecznym przeprowadzenia eksperymentu identyfikacyjnego, a co za tym idzie procedury samostrojzenia (obliczenie optymalnych nastaw PID regulatora w punkcie pracy), jest wybranie jednej z metod identyfikacyjnych czyli zaprogramowanie adresów: **7-x-45 ≠ 0** oraz w **7-x-46** należy ustawić wartość czasu Δt dla eksperymentu Zieglera-Nicholsa, w **7-x-48** należy ustawić czas stabilizacji procesu (dotyczy to wszystkich procedur identyfikacyjnych), w **7-x-49** należy ustawić w % dopuszczalny uchyb regulacji podczas procesu samostrojzenia, w **7-x-50** ustawia się w % skok sygnału identyfikacyjnego na wyjściu regulatora (na wejściu badanego obiektu).

Start procedury samostrojzenia następuje po pojawieniu się 1 logicznej w RSB adresowanym w **7-x-42**. Zatrzymanie procedury po ustawieniu 1 logicznej w RSB adresowanym w **7-x-43**.

6.2 Regulacja rozmyta (ang. fuzzy)

Regulacja rozmyta w regulatorze LB – 600 polega na pobieraniu przez algorytm PID zestawu nastaw (SP , k_p , T_d , T_i) z tablicy zapisanej w sposób :

- wartość zadaną SP_n (określającą punkt pracy n),
- współczynnik wzmocnienia k_{pn} (dla punktu pracy n),
- czas zdwojenia (całkowanie) T_{in} (dla punktu pracy n),
- czas wyprzedzenia (różniczkowanie) T_{dn} (dla punktu pracy n)



Rys. 6.2 Schemat tablicy wykorzystanej do regulacji rozmytej

Wartości te wpisuje się w kolejnych modułach $M_1 \dots M_n$, gdzie n – numer modułu w zakresie $1 \dots 400$. Uruchomienie regulacji tablicowej następuje po wpisaniu adresu początkowego tablicy do adresu **7x38** oraz wpisaniu wartości logicznej 1 do adresów **7x42**, **7x43** oraz **7x44** powodujących START, STOP i PAUZĘ w realizacji programu.

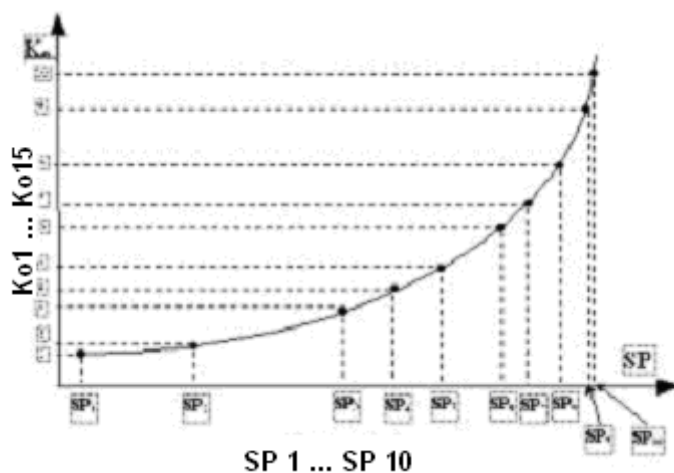
Nastawy dla poszczególnych punktów pracy mogą być wprowadzane ręcznie w trybie programowania regulatora lub jako przesyłka całej tablicy w trybie transmisyjnym. Określenie poszczególnych nastaw PID dla danego procesu może być realizowane w sposób empiryczny przez użytkownika znającego jego właściwości lub przez zbieranie nastaw określanych przez algorytm samostrojenia regulatora, pełniącego rolę identyfikatora procesu.

Jeśli właściwości dynamiczne nieliniowego procesu zależą od ustawionej wartości zadanej SP , wówczas dla scharakteryzowania dynamiki regulowanego nieliniowego obiektu można posłużyć się zbiorem kilkunastu lub kilkudziesięciu zestawów (k_p , T_d , T_i) nastaw, tzw. lokalnych regulatorów PID, przy czym każdy zestaw nastaw powinien zapewniać odpowiednią regulację w małym otoczeniu danej wartości zadanej SP (otoczeniu danego punktu pracy).

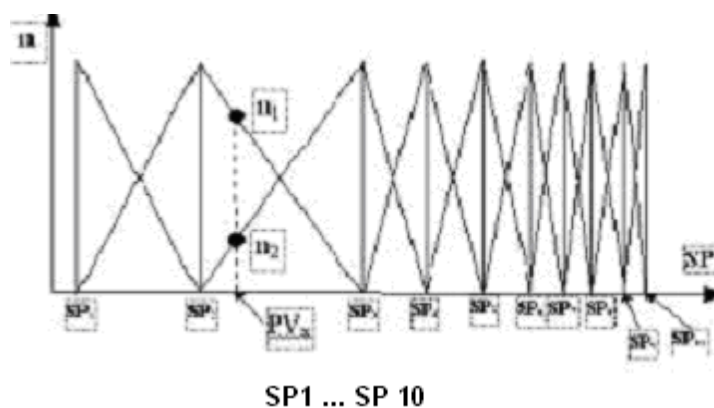
W danym, zdefiniowanym punkcie pracy regulator pobiera odpowiedni zestaw nastaw (identyfikatorem punktu pracy jest wartość zadana SP) i reguluje wg pobranych nastaw PID. W przypadku gdy rzeczywisty punkt pracy znajduje się pomiędzy punktami zdefiniowanymi, algorytm dokonuje korekty nastaw w oparciu o logikę rozmytą (*ang. fuzzy logic*).

Rys. 6.3a przedstawia przykładową nieliniową zależność z zaznaczonymi 10 punktami, w których dobrano nastawy k_p , T_d , T_i .

Rys. 6.3b przedstawia podział na poszczególne funkcje przynależności oraz zbiory rozmyte.



Rys. 6.3a. Przykładowa nieliniowa zależność



Rys. 6.3b. Podział na funkcje przynależności i zbiory rozmyte

7. Wybór najlepszego środowiska pracy

7.1 Środowiska brane pod uwagę

Skupiono się na dwóch rozwiązaniach:

- wykorzystaniu języka C++ i środowiska Visual Studio
- pracy w środowisku SCADA

Zaletą pierwszego rozwiązania jest powszechna dostępność narzędzi programowania w języku C++. Wysyłanie wartości sygnałów poprzez port RS – 232 (w celu komunikacji z regulatorem) zostało dobrze opisane i opracowane w literaturze.

Dodatkowo łatwa i szybka kompilacja do pliku wynikowego jest niewątpliwie dużym atutem.

Względne uniezależnienie programu od konkretnej konfiguracji komputera i systemu operacyjnego oraz postać pliku wykonywalnego to zasadnicze postulaty przemawiające za tym rozwiązaniem. Należy również dodać, że moduł komunikacyjny, poprzez port RS – 232, zobowiązała się dostarczyć firma LAB-EL.

Najistotniejszym minusem, przekreślającym możliwość wykorzystania środowiska Visual Studio C++, jest skomplikowane rysowanie wykresów na bieżąco. Dlatego też zdecydowano się wykorzystać środowiska SCADA (*ang. Supervisory Control And Data Acquisition*) do realizacji tematu pracy inżynierskiej.

Jednym z przedstawicieli tej klasy jest program iFIX firmy Intellution. Pozwala on na bardzo łatwe rysowanie wykresów w czasie rzeczywistym. Ponadto komunikację ze sprzętem automatyki umożliwiają specjalne moduły programowe, tzw. drajwery komunikacyjne. Spośród ponad 300 drajwerów oferowanych przez system iFIX, wykorzystałem, z uwagi na obsługę protokołu MODBUS RTU, drajwer MODICON, oznaczony w systemie iFIX, symbolem MB1.

Umożliwia on wymianę danych pomiędzy regulatorem LB-600 a komputerem poprzez złącze RS-232. W systemie iFIX występuje świetne narzędzie do tworzenia baz danych. Umożliwia ono wygodny zapis i odczyt (również na bieżąco) wartości rejestrów z regulatora, które będą wykorzystywane w programie komputerowym.

W dalszej części mojej pracy opisano dokładnie uruchomienie i możliwości, jakie daje program iFIX.

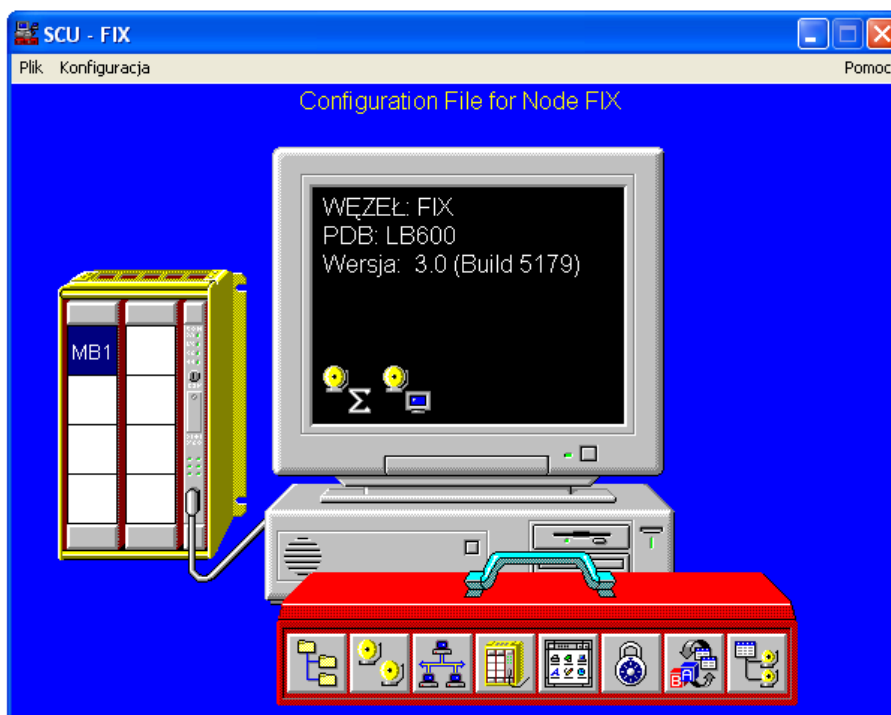
8. Uruchomienie węzła iFix

Podczas instalacji systemu iFIX, program Instalator tworzy automatycznie pliki konfiguracyjne. Węzeł iFIX wymaga dwóch następujących elementów w celu poprawnego uruchomienia:

- plik konfiguracyjny
- lokalna konfiguracja startowa

Plik konfiguracyjny zawiera informacje, których wymaga program startowy systemu iFIX w celu skonfigurowania węzła. System iFIX wykorzystuje plik konfiguracyjny jedynie podczas startu systemu. Wszelkie zmiany, jakie są wprowadzane do pliku konfiguracyjnego podczas pracy systemu, uaktywnią się dopiero po ponownym wystartowaniu. Program startowy musi znać nazwę węzła – lokalnego komputera – oraz ścieżkę i nazwę wykorzystywanego pliku konfiguracyjnego. Każda platforma, na której uruchamiany jest system iFIX, charakteryzuje się specyficznymi metodami określania lokalnej konfiguracji startowej.

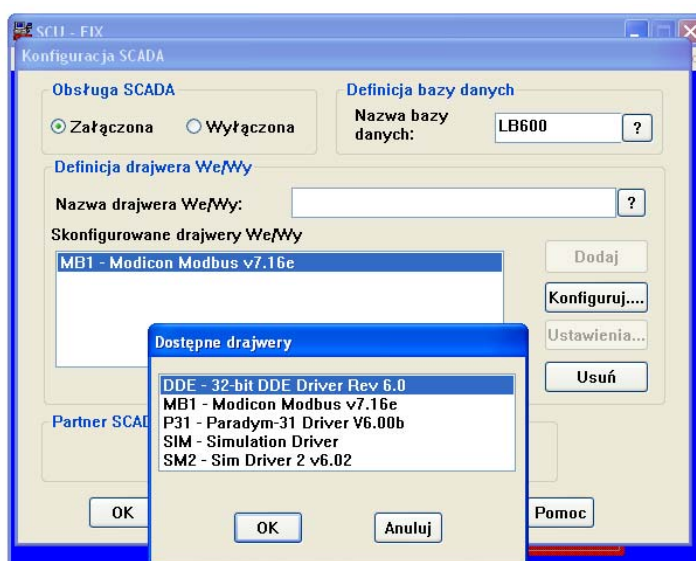
Na rys.8.1 pokazano przykładowe okno startowe programu SCU (*System Configuration Utility*). W oknie tym ustawia się wszystkie parametry węzła SCADA, które są niezbędne do prawidłowej pracy systemu.



Rys. 8.1 Plik konfiguracyjny programu iFIX

Pierwszym krokiem konfiguracji drajwera We/Wy jest konfiguracja serwera SCADA w programie SCU. Utworzony węzeł SCADA kojarzy się z odpowiednią bazą danych, przyporządkowuje odpowiedni drajwer komunikacyjny, itp.(rys. 8.2). Węzeł może być obsługiwany przez urządzenia podłączone przy wykorzystaniu odpowiednich drajwerów. Aby uaktywnić funkcjonalność SCADA należy wykonać następujące kroki:

- z menu *Konfiguracja* wybrać opcję *SCADA* (lub też kliknąć na czwartą od lewej ikonę znajdującą się w belce narzędziowej)
- kliknąć przycisk opcji *Załączona*, aby uaktywnić opcję SCADA
- w polu *Nazwa bazy danych* wpisz nazwę dostępnej bazy
- kliknąć „?” obok pola *Nazwa drajwera We/Wy* w celu wybrania drajwera. Lista dostępnych (zainstalowanych) drajwerów powinna się pojawić tak, jak to ilustruje poniższy rysunek:

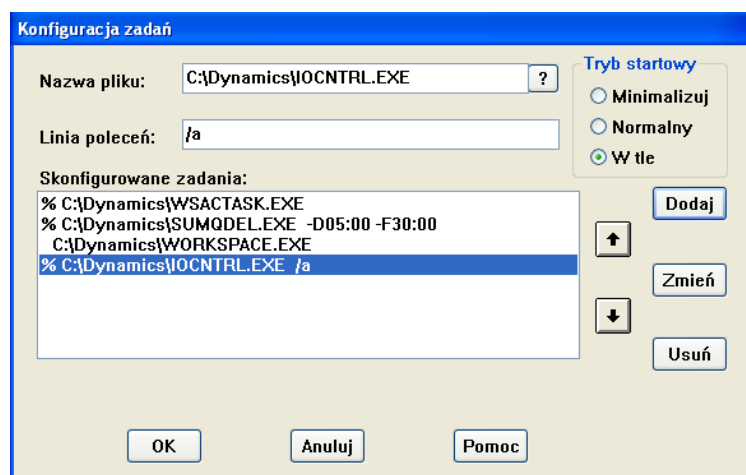


Rys. 8.2 Konfiguracja węzła SCADA: wybór bazy danych i drajwera

- Wybrać drajwer *SIM – Simulation Driver* a następnie kliknąć przycisk *OK*
- Wybrany drajwer powinien pojawić się w polu *Nazwa drajwera We/Wy*
- Kliknąć przycisk *Dodaj* aby dodać wybrany drajwer do listy aktywnych drajwerów SCADA w systemie iFIX
- Wybrać opcję *Zadania* z menu *Konfiguracja* i sprawdzić czy na liście zadań znajduje się linia:

C:\DYNAMICS\IOCNTL.EXE /a

Jeśli jej nie ma, to należy ją dodać wypełniając dialog tak, jak pokazuje Rys. 8.3, a następnie kliknąć na przycisk *Dodaj*



Rys. 8.3 Dodanie obsługi wejść/wyjść

- Zamknąć dialog przez naciśnięcie przycisku OK
- Potwierdzić konfigurację przyciskiem OK a następnie zapisać plik SCU korzystając z opcji *Zapisz* menu *Plik*

Po zamknięciu SCU i ponownym uruchomieniu aplikacji iFIX drajwer powinien być widoczny w systemie.

W celu uruchomienia węzła iFIX dla potrzeb pracy programu *Symulator*, stworzono plik konfiguracyjny *fix.scu*. W celu uruchomienia programu na komputerze z systemem iFIX trzeba załadować ten plik konfiguracyjny.

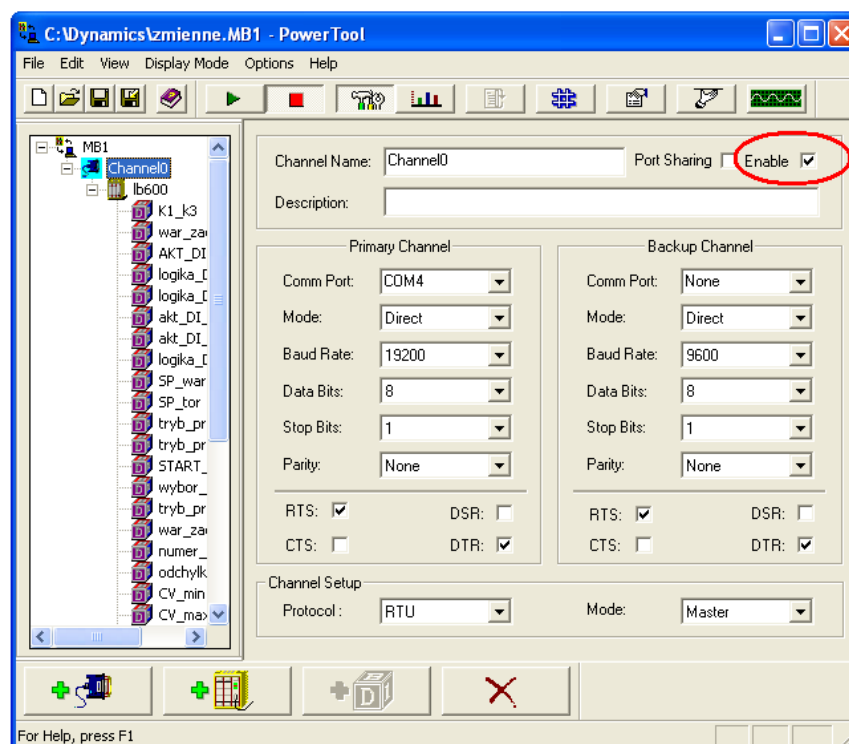
Wszystkie pliki konfiguracyjne dołączone do pracy dyplomowej sprawdzono na komputerze znajdującym się w laboratorium Instytutu Automatyki i Informatyki Stosowanej Wydziału Elektroniki i Technik Informacyjnych Politechniki Warszawskiej. Program nie wykazał żadnych błędów działania.

W ten sposób stworzone oprogramowanie może być wykorzystane do celów dydaktycznych w celu pokazania działania omawianego systemu.

9. Konfiguracja programu iFix

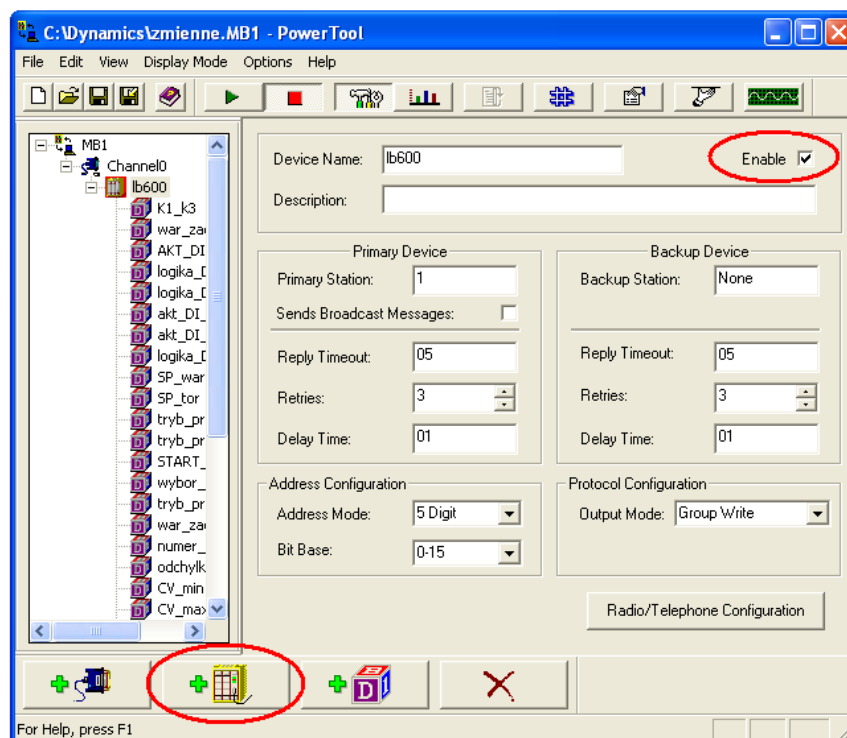
Komunikację z regulatorem LB – 600 zapewnia drajwer MODICON, oznaczony w systemie iFIX, symbolem MB1.

System iFIX umożliwia konfigurowanie poszczególnych urządzeń współpracujących z danym węzłem SCADA. Na rys. 9.1 przedstawiono program *Power Tool*, służący do konfiguracji drajwera MB1. Umożliwia on konfigurację do 15 urządzeń w każdym z 8 kanałów. Urządzenia skonfigurowane w odpowiednim kanale, podłączone są do portu COM komputera. Po kliknięciu na wybrany kanał, należy ustawić numer portu i odpowiednie parametry transmisji: prędkość transmisji (w regulatorze LB – 600 ustawienie fabryczne to 19200 b/s), liczbę bitów stopu oraz kontrolę parzystości lub nieparzystości. Nie należy zapomnieć o zaznaczeniu pola *Enable*, w prawym górnym rogu ekranu. W przeciwnym wypadku kanał nie zostanie uruchomiony.



Rys. 9.1 Konfiguracja drajwera MB1

Mając skonfigurowany kanał można przystąpić do dodania urządzenia pracującego w skonfigurowanym kanale (Rys 9.2).



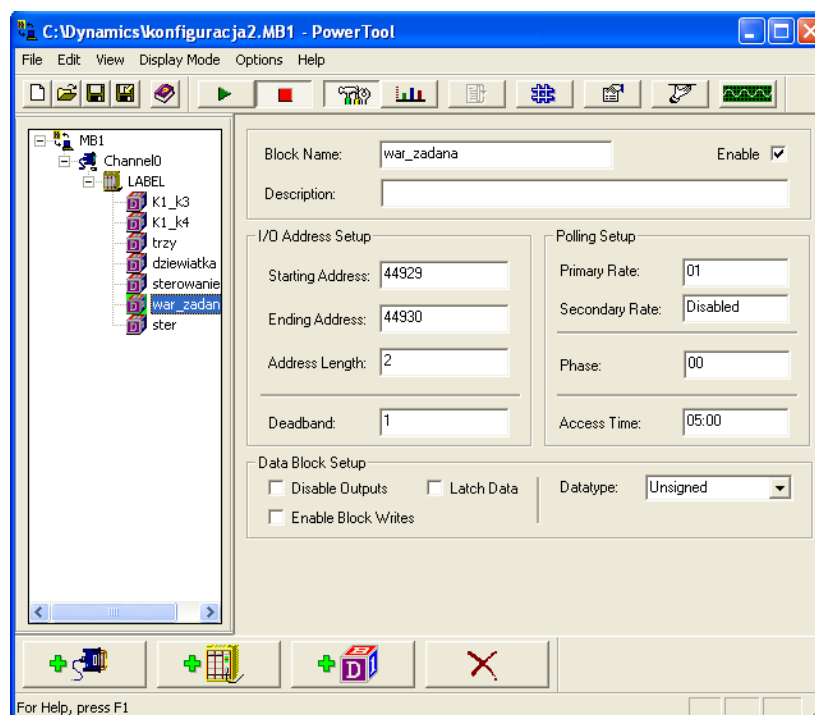
Rys. 9.2 Konfiguracja urządzenia

- *Device Name* – dowolna nazwa urządzenia,
- *Enable* – pozwolenie na działanie urządzenia,
- *Primary Station* – adres Modbus sterownika
- *Reply Timeout* – czas, po jakim nastąpi powtórne wysłanie danych do sterownika,
- *Retries* – liczba powtórzeń,
- *Delay Time* – czas, w którym dana paczka nie będzie odpytywana, gdy nastąpi utrata komunikacji

Opcje w ramce *Primary Device* dotyczą sterownika podstawowego, natomiast w ramce *Backup Device* - rezerwowego.

Po ustawieniu parametrów transmisji należy skonfigurowane urządzenie uruchomić poprzez zaznaczenie pole *Enable*.

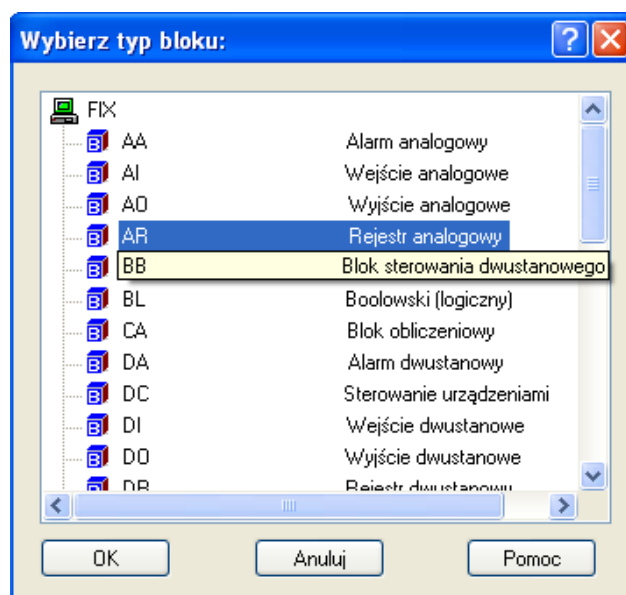
Zmienne z regulatora LB – 600 adresuje się poprzez dodanie do adresu bazowego drajwera MB1, który wynosi 40 001 numeru rejestru zmiennej z regulatora oraz typu zmiennej (Float lub Integer – zapisana w MB1 jako Unsigned). Ustawia się funkcję danego bloku skojarzonego ze zmienną jako "tylko do odczytu" lub "czytanie wraz z zapisem" (rys. 9.2).



Rys. 9.2 Konfiguracja bloków zmiennych

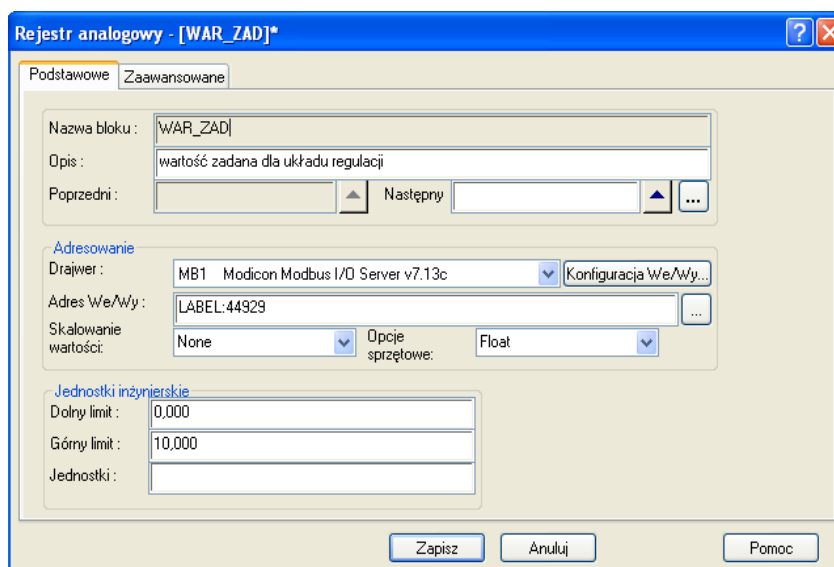
- *Enable* – pozwolenie na włączenie paczki danych,
- *Starting Address* – adres startowy paczki danych, zależny od drajwera,
- *Ending Address* – adres końcowy paczki danych, zależny od drajwera,
- *Address Length* – długość paczki danych,
- *Deadband* – strefa nieczułości,
- *Latch Data* – zatrzymywanie ostatnich poprawnych danych przy utracie komunikacji ze sterownikiem,
- *Disable Outputs* – zabronienie zapisu do paczki danych,
- *Enable Block Writes* – pozwolenie na blokowy zapis danych (cała paczka danych jest uaktualniana podczas zapisu nawet tylko jednej zmiennej),
- *Primary Rate* – podstawowy okres odpytywania paczki danych. Może być Disabled (paczka nigdy nie będzie czytana), 0 (paczka będzie odpytywana z maksymalną częstotliwością) lub wartość od 1 sekundy do 6:23:59:59.
- *Secondary Rate* – częstotliwość odpytywania po zadziałaniu czasu *Access Time*,
- *Phase* – przesunięcie odczytu dla paczki,
- *Access Time* – czas, po którym paczka przestanie być odpytywana z częstotliwością *Primary Rate* i zacznie być czytana z częstotliwością *Secondary Rate*, sytuacja ta wystąpi gdy system iFIX nie będzie potrzebował danych z tej paczki (np. gdy do wyświetlania danych z tej paczki na ekranie będziemy używać zmiennych typu *Analog Register* i ekran ten przez czas *Access Time* nie będzie oglądany),
- *Data Type* – typ danych dla paczki danych

Każda zmienna w węźle powinna być zdefiniowana nie tylko nazwą, ale również powinna mieć przypisany typ funkcji. Zmiennej typu Float powinna być przypisana funkcja, np. AI, AO, AR (wejście analogowe, wyjście analogowe, zmienna rejestrowa – zapisywana i odczytywana), zmiennej typu Integer (Unsigned) powinna być przypisana funkcja binarna, np. DI, DO, DR, BB, BL, DA, itp. (rys. 9.3)



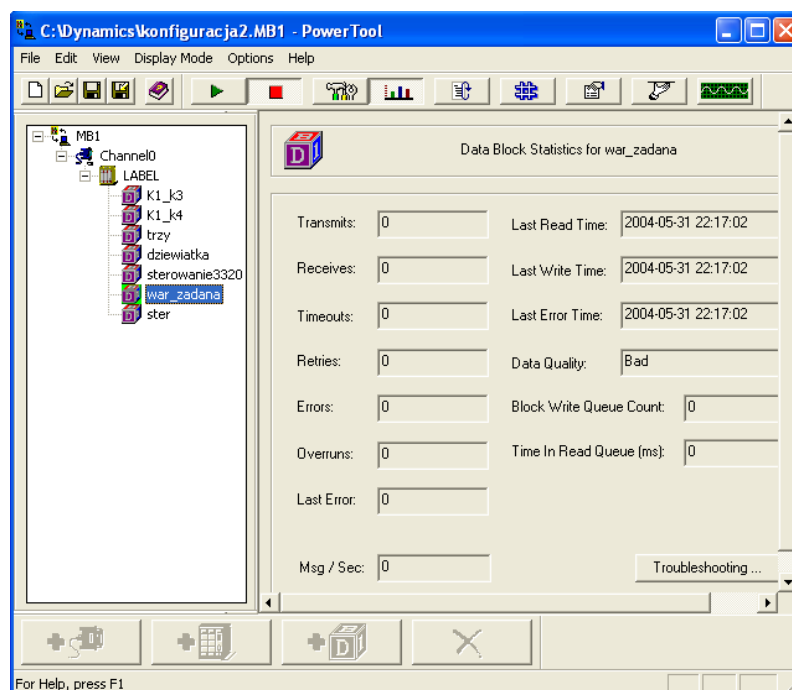
Rys. 9.3 Funkcje dostępne dla definicji zmiennych i bloków

Po przypisaniu zmiennej lub blokowi funkcjonalnemu odpowiedniej funkcji wejścia/wyjścia, należy ją skonfigurować. Rys. 8.4 przedstawia okno konfiguracji AR – rejestru analogowego.



Rys 9.4 Konfiguracja rejestru analogowego

Kontrolę zadań w węzle iFIX zapewnia okno (rys. 9.5) otwierane w danym węzle automatyzowanego procesu. Informacje udostępniane w tym oknie mogą pomóc użytkownikowi w diagnozowaniu potencjalnych problemów związanych z działaniem systemu iFIX.



Rys. 9.5 Okno kontrolne systemu iFIX

W systemie iFIX występuje narzędzie do tworzenia baz danych. Przykładowy arkusz bazy przedstawiono na rysunku 9.6. W kolumnie Nazwa bloku wpisuje się nazwę zmiennej. Nazwa ta występuje w zbiorze zmiennych węzła, a dalej typ zmiennej, czas skanowania, drajwer, adres zmiennej, wartość zmiennej (w przypadku braku aktywności połączenia pojawiają się ???).

	Nazwa bloku	Typ	Opis	Okres	Urz	Adr We/Wy	Wartość bież.
1	AKT_7	AR	aktywność bloku warstwy 7	—	MB1	lb600.44967	1
2	AKT_DI	AR	aktywność bloku. START fuzzy i samostroj	—	MB1	lb600.42604	0
3	AKT_DI_PAUSE	AR		—	MB1	lb600.42624	0
4	AKT_DI_STOP	AR	aktywność bloku. STOP fuzzy i samostroje	—	MB1	lb600.42614	0
5	AUTO_KP	AR	Kp przy regulacji PID	—	MB1	lb600.44917	1.400
6	AUTO_TD	AR	Td przy regulacji PID	—	MB1	lb600.44921	9.000
7	AUTO_TI	AR	Ti przy regulacji PID	—	MB1	lb600.44919	25.000
8	CV_MAX	AR	max wartość sterowania	—	MB1	lb600.44909	100.000
9	CV_MIN	AR	min wartość sterowania	—	MB1	lb600.44907	0.000
10	F_1_1	AR	parametry tablicy do fuzzy. zestaw 1	—	MB1	lb600.47101	????
11	F_1_2	AR		—	MB1	lb600.47103	????
12	F_1_3	AR		—	MB1	lb600.47105	????
13	F_1_4	AR		—	MB1	lb600.47107	????
14	F_2_1	AR		—	MB1	lb600.47111	????
15	F_2_2	AR		—	MB1	lb600.47113	????
16	F_2_3	AR		—	MB1	lb600.47115	????
17	F_2_4	AR		—	MB1	lb600.47117	????
18	F_3_1	AR		—	MB1	lb600.47121	????
19	F_3_2	AR		—	MB1	lb600.47123	????
20	F_3_3	AR		—	MB1	lb600.47125	????
21	F_3_4	AR		—	MB1	lb600.47127	????

Rys. 9.6 Arkusz przykładowej bazy danych

Na dołączonej płycie CD – ROM znajdują się 3 pliki umożliwiające natychmiastowe załadowanie ustawień drajwera. Program *Symulator* może być uruchomiony bez zbędnych zabiegów ze strony użytkownika.

10. Omówienie programu *Symulator*

Program *Symulator* spełnia wszystkie wymogi omówione w punkcie 5. Ponadto istnieje możliwość uruchomienia pliku pomocy. Znajduje się w nim opis uruchomienia większości funkcji programu oraz pomocne informacje przy pojawieniu się komunikatów o błędzie.

W pracy nie zastosowano zmiany skali czasu. Zmianę skali czasu trzeba uwzględnić wtedy, gdy symulację modelu obiektu wykonuje się na komputerze szybszym lub wolniejszym od komputera bazowego. Celem takiej zmiany jest generowanie w tych samych chwilach czasu jednakowych wartości sygnałów.

W programie *Symulator* zastosowano licznik, który co określony czas (zwany czasem odpytywania) wylicza wartość wyjścia obiektu. Czas ten wynosi 100ms. Oznacza to, że co 100ms komputer liczy wyjście "od nowa". Błędy wynikające z braku zastosowania zmiany skali czasu powstają w pojedynczym cyklu obliczeń (co wspomniane 100ms).

Aby wykluczyć ich występowanie należałoby przeprowadzić test szybkości obliczenia wyjścia układu w pojedynczym cyklu. Na jego podstawie (oraz znajomości wyniku szybkości komputera bazowego) należałoby pomnożyć wszystkie stałe czasowe modelu obiektu przez pewien współczynnik. Współczynnik ten jest stosunkiem wyniku obliczeń komputera na którym uruchomiono program *Symulator* do wyniku testu komputera bazowego.

Z uwagi na fakt, że program *Symulator* pracuje w środowisku Windows NT, szybkość procesora na którym pracuje ten system jest znaczna. Po przeprowadzonych testach szybkości stwierdzono, że wyniki testów wahają się w granicach możliwości obliczeniowych programu i systemu operacyjnego (na 10 prób uzyskano wszystkie wyniki równe 0) a wynik nie odzwierciedla rzeczywistej szybkości komputera.

Z tego względu zaniechano przeprowadzenia zmiany skali czasu. Wprowadzono natomiast test szybkości, polegający na obliczeniu wyjścia modelu w jednym cyklu. Ma to na celu sprawdzenie czy szybkość komputera pozwala na wykonywanie obliczeń w czasie mniejszym niż 100ms. Jeśli test szybkości zakończy się niepowodzeniem, ze względu na małą szybkość obliczeń, należy zmienić czas odpytywania na większy. Odbije się to na jakości regulacji, jednak umożliwi przeprowadzenie prezentacji na wolniejszym komputerze (por. rozdz. 11).

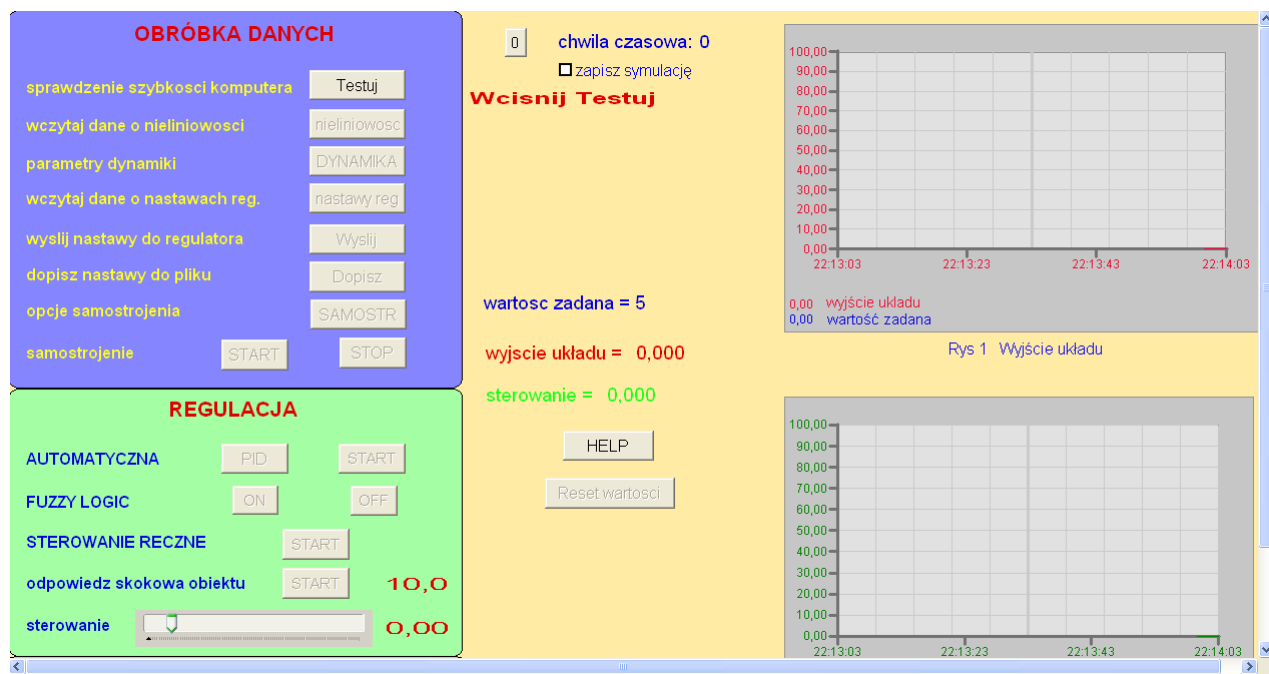
10.1 Uruchomienie drajwera

Przed uruchomieniem programu *Symulator* należy skonfigurować drajwer komunikacyjny MB1. Służy do tego program *Power Tool* i zamieszczone z programem pliki: ZMIENNE (ładowane są najważniejsze zmienne służące do przeprowadzenia symulacji) oraz TABLICA (ładowane są zmienne tablicowe – do wysłania przy regulacji rozmytej).

Plik TABLICA należy łączyć za każdym razem, gdy chcemy wysłać nastawy PID (odczytane uprzednio z pliku) do regulatora.

10.2 Ekran główny programu

Po załadowaniu ustawień drajwera i uruchomieniu programu pojawia się ekran główny (Rys. 10.1).



Rys. 10.1 Ekran główny programu Symulator

Po lewej stronie widnieją przyciski odpowiednich funkcji, po prawej rysowane są wykresy: wartości zadanej i wyjścia układu regulacji (górny wykres) oraz sterowania (dolny). Na środku ekranu wyświetlane są informacje o przebiegu regulacji oraz wyświetlane są wartości sygnałów: wartości zadanej, wyjścia i sterowania. Ponadto w przypadku wykorzystania regulacji automatycznej podawane są wartości nastaw, natomiast przy regulacji rozmytej dodatkowo numer aktualnie pobieranego zestawu nastaw wraz z ich wartościami.

10.3 Opis przycisków znajdujących się w programie

- **HELP**
Wyświetlany jest plik pomocy. Zawarte są w nim informacje dotyczące uruchamiania poszczególnych funkcji programu oraz najczęstsze błędy i sposoby ich eliminowania.
- **TESTUJ**
Przeprowadzenie testu szybkości obliczeń wykonywanych przez komputer. Ma to na celu stwierdzenie, czy szybkość obliczeń wykorzystanego komputera nie jest za wolna do przeprowadzenia symulacji, czy też należy (w celu kontynuowania pracy na tym komputerze) zmienić wartość timera (czasu odpytowania).

- **RESET WARTOŚCI**

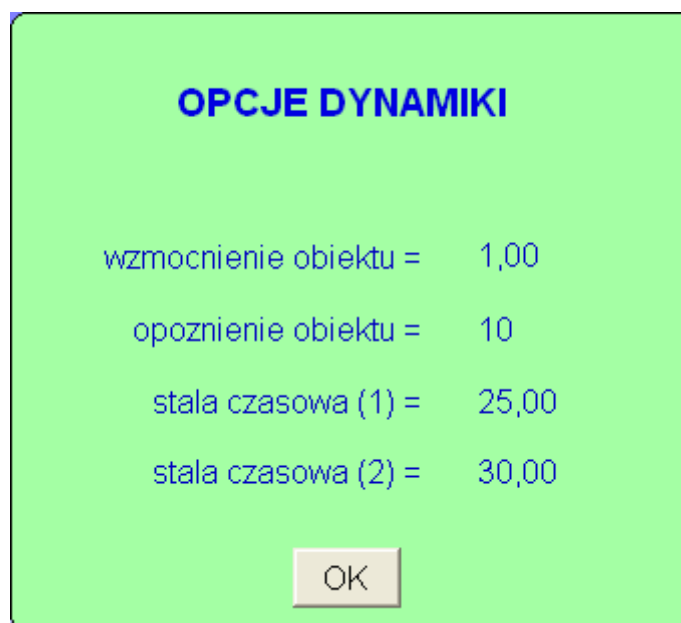
Warunkiem koniecznym uruchomienia programu jest wciśnięcie przycisku **RESET WARTOŚCI**, służącego do wyzerowania zmiennych i ustawienia wartości początkowych programu. Ustawiane są wartości domyślne: dynamiki obiektu, wyjścia, zmiennych pomocniczych. Ustawiane są napisy początkowe (podpowiadające użytkownikowi co robić) i uaktywniane przyciski wczytywania danych.

- **NIELINIOWOŚĆ**

Wczytanie danych o nieliniowości obiektu. Dane pobierane są z dowolnego pliku zapisanego w określony sposób: każda dana zapisana jest w nowej linijce oraz danych musi być dokładnie 35. Niespełnienie któregoś z tych warunków powoduje pojawienie się informacji o błędzie. Jednocześnie wczytane dane rysowane są na wykresie w programie Microsoft Excel. Można podejrzec, która dana jest niepoprawna.

- **DYNAMIKA**

Wpisanie danych o parametrach dynamiki obiektu. Po wybraniu tej funkcji pojawia się okno (Rys. 10.2)



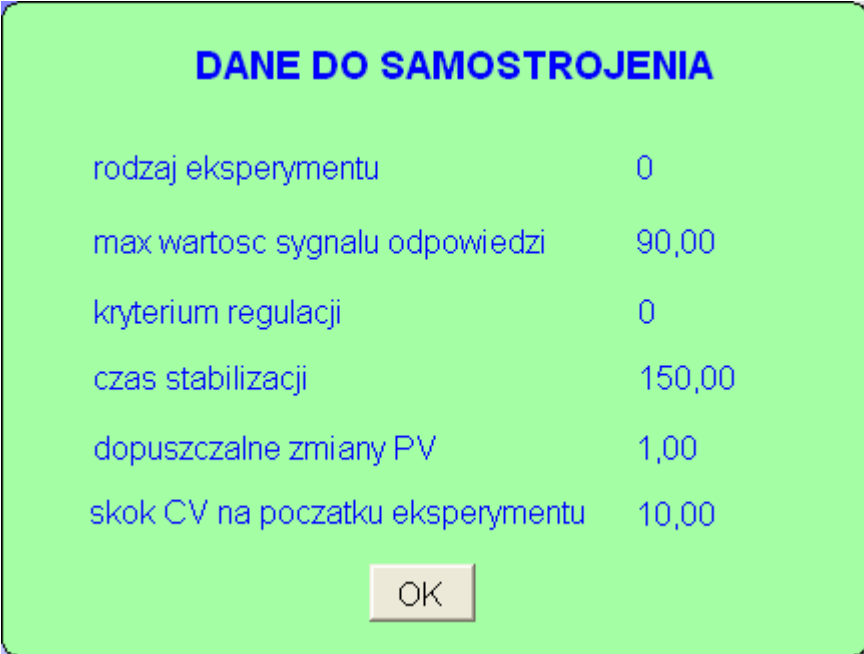
Rys. 10.2 Okno wprowadzania danych dynamiki obiektu

Wartości początkowe są wartościami domyślnymi. Program, przy każdym wprowadzaniu danej, sprawdza jej poprawność (zakres wartości, typ). Stałe czasowe dotyczą obiektu dwuinercyjnego.

- **NASTAWY REG.**

Wczytanie nastaw regulatora PID z pliku. Dane muszą być zapisane w odpowiedni sposób: każda dana musi znajdować się w nowej linijce i danych musi być dokładnie 140 (35 zestawów nastaw: SP, K_p, T_i, T_d). Wczytane nastawy będą wykorzystane do regulacji rozmytej.

- **WYŚLIJ**
Wysłanie uprzednio wczytanych nastaw do regulatora. Regulator przechowuje zestawy nastaw w tablicy, z której pobiera odpowiednie wartości, zgodnie z algorytmem regulacji rozmytej.
- **DOPISZ**
Dopisanie jednego kompletu nastaw do pliku. Dane uzyskane z eksperymentu samostrojzenia są dopisywane do pliku. W ten sposób użytkownik może przeprowadzić serię eksperymentów, zapisać ich wyniki do pliku a następnie wykorzystać je w innym czasie do regulacji.
- **SAMOSTR**
Wpisanie danych o eksperymencie samostrojzenia. Podobnie jak w przypadku przycisku **DYNAMIKA**, wartości początkowe są wartościami domyślnymi. Po wybraniu tej opcji pojawia się okno z rysunku 10.3



DANE DO SAMOSTROJENIA	
rodzaj eksperymentu	0
max wartosc sygnalu odpowiedzi	90,00
kryterium regulacji	0
czas stabilizacji	150,00
dopuszczalne zmiany PV	1,00
skok CV na początku eksperymentu	10,00

OK

Rys 10.3 Dane do eksperymentu samostrojzenia

Wartość z pola "rodzaj eksperymentu" musi mieć wartość różną od zera, jeśli chcemy uruchomić eksperyment. Ponadto wartość sygnału sterowania użytego do eksperymentu (w chwili skoku wartości CV) nie może mieć wartości większej niż maksymalna wartość tego sygnału.

- **NASTAWY**
Zapisanie do regulatora wartości nastaw służących do regulacji automatycznej.
- **0**
Przycisk ten służy do zerowania na wyświetlaczu minionego czasu.
- **zapisz symulację**
Zapisanie wartości wyjścia, wartości zadanej i chwili czasowej do pliku, w celu dalszej, ewentualnej, obróbki.

Pozostałe przyciski służą do uruchomienia i zatrzymania funkcji: samostrojzenia, regulacji automatycznej i rozmytej a także odpowiedzi skokowej obiektu.

W programie znajdują się dwa suwaki służące do ustalania: wartości zadanej oraz sterowania. Uwidaczniają się one jednakże dopiero przy wybieraniu odpowiednich funkcji programu. Suwak wartości zadanej uruchamiany jest w momencie uruchomienia dowolnego rodzaju regulacji, natomiast suwak sterowania w momencie uruchomienia regulacji ręcznej.

Aby przejść do trybu regulacji automatycznej, ręcznej bądź rozmytej, trybu samostrojzenia lub odpowiedzi skokowej obiektu, należy wczytać dane o nieliniowości obiektu. Po poprawnym ich wczytaniu uaktywniają się przyciski odpowiednich funkcji.

Poprawnie wczytane dane o nastawach są warunkiem koniecznym uruchomienia regulacji rozmytej.

10.4 Opis poszczególnych funkcji programu *Symulator*

- **samostrojzenie**

Służy do wyliczenia nastaw do regulacji automatycznej bądź rozmytej (w tym przypadku dla jednego punktu pracy). Eksperyment przebiega bez ingerencji użytkownika. Można go jedynie wyłączyć.

Punkt pracy należy ustawić w panelu operatorskim regulatora. Wartość ta jest odczytywana w momencie uruchomienia eksperymentu. Układ będzie się starał doprowadzić do wyrównania wyjścia z wartością sterowania zmodyfikowaną przez nieliniowość obiektu. Gdy już się tak stanie, rozpocznie się właściwy eksperyment.

Jeżeli dane do eksperymentu samostrojzenia zostaną dobrane poprawnie (uwzględniające zachowanie się obiektu na skok sterowania) oraz obiekt jest sterowalny, to wynikiem będzie 6 parametrów. Trzy pierwsze z nich określają wynik przeprowadzonego eksperymentu. Są to:

opóźnienie obiektu τ_D , stała czasowa obiektu τ oraz wzmacnienie obiektu K . Trzy pozostałe są to obliczone (według podanego kryterium) wartości nastaw: współczynnik wzmacnienia regulatora k_p , czas zdwojenia (stała całkowania) regulatora T_i oraz czas wyprzedzenia (stała różniczkowania) regulatora T_d . Obliczone w ten sposób nastawy zostaną wyświetlone na ekranie komputera. Użytkownik ma możliwość zapisania ich do pliku wraz z wykorzystanym punktem pracy.

Jeżeli w czasie eksperymentu pojawi się błąd, eksperyment zostanie przerwany a na ekranie pojawi się komunikat o jego zakończeniu. Chcąc uruchomić go ponownie, należy najpierw skasować błąd z pulpitu operatorskiego regulatora.

- regulacja automatyczna (PID)

Regulator pobiera sygnał wyjściowy z komputera i na jego podstawie generuje sterowanie, które jest odczytywane przez program komputerowy. Na podstawie sterowania generowane jest wyjście układu (przesyłane do regulatora). Następnie cykl powtarza się do momentu wyłączenia regulacji. Wartość zadana pobierana jest z funkora **71**, podobnie jak nastawy. Uruchamiając regulację automatyczną wyświetlają się zarazem wartości nastaw użytych do regulacji. Jest to spowodowane zmianą tych wartości jeśli poprzednio była uruchomiona regulacja rozmyta lub uruchomiono eksperyment samostrojzenia. W pierwszym przypadku algorytm regulacji pobiera wartości nastaw z tablicy i zapisuje je w miejsce wartości z funkora **71**. W ten sposób poprzednie wartości ulegają bezpowrotnej stracie. Podobnie się dzieje z wartością zadaną. W drugim przypadku, po akceptacji wyniku samostrojzenia, wartości również zapisywane są do funkora **71**. Jeśli użytkownik stwierdzi, że chce je zmienić może to zrobić klikając na przycisk **NASTAWY** i wpisując swoje wartości.

- regulacja rozmyta (fuzzy PID)

Regulator pobiera sygnał wyjściowy układu z komputera, i w zależności od jego wartości pobierane są wartości nastaw z tablicy (uprzednio przesłanej do regulatora bądź zaprogramowanej przez użytkownika). Regulator przechodzi do trybu pracy komputerowego. W trybie tym należy podać zewnętrzną wartość zadaną, która pochodzi z funkora **33**. Wartość tą ustawia się z programu, bądź poprzez przesunięcie suwaka, bądź poprzez wpisanie wartości. O wartości nastaw aktualnie wykorzystywanych w regulacji informuje napis na ekranie monitora. Wyświetlany jest również numer zestawu, z którego pochodzą te wartości. Wraz ze zmianą aktualnie pobieranego numeru, wyświetlane wartości zmieniają kolor. Służy to lepszemu zobrazowaniu działania algorytmu.

- odpowiedź skokowa obiektu

Układ regulacji staje się układem otwartym. Na wejście obiektu podawany jest skok wartości sterowania, który generowany jest poprzez dodanie do wartości wyjścia układu (odczytanej w chwili uruchomienia funkcji) wartości skoku. Jego wielkość można podać wpisując ją w pole, obok przycisku uruchomienia funkcji. Wartością domyślną jest 10. Warto zaznaczyć, że aby uruchomić odpowiedź skokową obiektu, wartość sterowania powiększona o wartość skoku nie może przekraczać maksymalnej (ustawionej w regulatorze) wartości wyjścia.

- regulacja ręczna

W tym przypadku użytkownik ma możliwość zmiany wartości sterowania bezpośrednio z programu. Regulator w tym przypadku nie jest wykorzystywany. Wartość sterowania pochodzi z suwaka.

10.5 Uruchomienie programu

10.5.1 Przygotowanie danych

Po pojawieniu się okna (rys. 10.1) program oczekuje na wciśnięcie przycisku TESTUJ. Jest to sygnalizowane pojawieniem się na ekranie odpowiedniego napisu.



Rys. 10.4 Program oczekuje na wciśnięcie przycisku Testuj

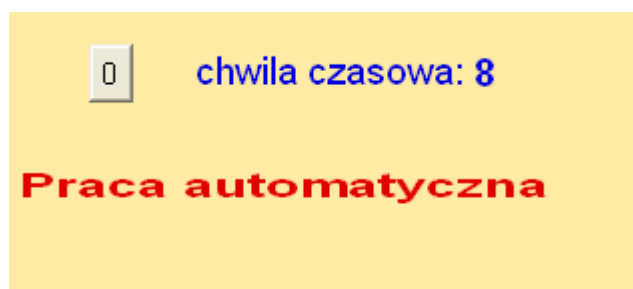
Po pomyślnym przejściu testu szybkości program czeka na wyzerowanie zmiennych przyciskiem RESET WARTOŚCI. Uaktywniają się wtedy przyciski wczytania danych z pliku oraz podania wartości parametrów dynamiki i samostrojenia. Wczytanie poprawnych danych jest warunkiem koniecznym dalszej pracy.



Rys. 10.5 Ustawienie parametrów dynamiki

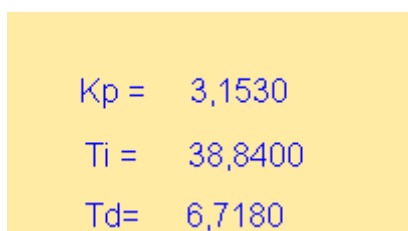
10.5.2 Regulacja automatyczna

Ustawienie parametrów dynamiki jest konieczne do uruchomienia m.in. regulacji automatycznej. Jako nastawy regulatora PID posłużyły wartości uzyskane z eksperymentu samostrojania dla punktu pracy równego 20.



Rys. 10.6 Program informuje o pracy automatycznej

Wartości te można zobaczyć u dołu ekranu (Rys. 10.7)



Rys. 10.7 Nastawy używane w regulacji automatycznej

10.5.3 Regulacja ręczna

Wybierając tę funkcję użytkownik ma możliwość ustawiania wartości sterowania. Służy do tego suwak zadawania wartości (Rys. 10.8). Regulator nie bierze udziału w wymianie danych.

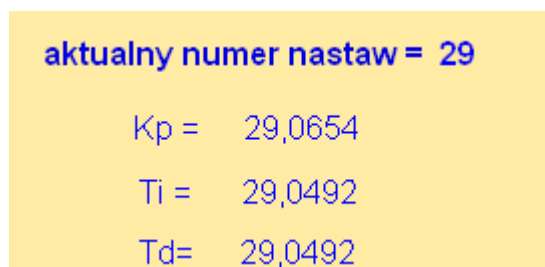


Rys. 10.8 Suwak zadawanie wartości sterowania

10.5.4 Regulacja rozmyta

Wczytanie poprawnych nastaw umożliwia uruchomienie regulacji rozmytej. Jako wartości nastaw dla poszczególnych punktów pracy posłużyły wartości uzyskane z eksperymentu samostrojenia.

W przypadku regulacji rozmytej wyświetlany jest dodatkowo numer aktualnie pobieranego zestawu nastaw.



aktualny numer nastaw = 29

$K_p = 29,0654$

$T_i = 29,0492$

$T_d = 29,0492$

Rys. 10.9 Numer aktualnie pobieranych nastaw

Przy zmianie pobieranego numeru nastaw pojawia się sygnał dźwiękowy.

10.5.5 Samostrojenie

Uruchomienie samostrojenia jest możliwe dopiero po ustawieniu rodzaju eksperymentu (wartość różna od 0). Wartości całkowite, przypisane kolejnym eksperymentom można odczytać z tablic konfiguracyjnych regulatora LB – 600.

Podczas eksperymentu na wykresie górnym rysowane jest wyjście układu oraz sterowanie. Wartość zadana w tym przypadku nie jest wykorzystywana.

W pierwszym etapie samostrojenia, układ należy doprowadzić do punktu równowagi, który jest wyświetlony na ekranie.

Po osiągnięciu stanu równowagi zaczyna się właściwy eksperyment. Trwa on do momentu wyliczenia nastaw, zatrzymania przyciskiem STOP, bądź pojawienia się błędu.

Po udanym zakończeniu się eksperymentu, pojawiają się wyliczone nastawy. Jeśli została wybrana funkcja akceptacji nastaw, należy je najpierw zaakceptować z pulpitu regulatora a następnie zatrzymać eksperyment. Zatwierdzone dane można dopisać do pliku klikając na przycisk DOPISZ.



Rys. 10.10a Ekran samostrojzenia – ustalanie równowagi



Rys. 10.10b Ekran samostrojzenia – właściwy eksperyment

Podczas samostrojzenia blokowana jest możliwość uruchomienia regulacji, zarówno automatycznej, ręcznej jak i rozmytej.

11. Opis wybranych fragmentów kodu źródłowego

11.1 Opis głównej funkcji programu

Do zrealizowania poszczególnych funkcji programu *Symulator* posłużono się językiem Visual Basic. Przyciśnięcie przycisku, przesunięcie suwaka, upływanie kwantu czasu (czasu odpytywania) powoduje wykonanie się odpowiedniej funkcji zapisanej w języku Visual Basic.

Funkcja:

```
Private Sub timer_OnTimeOut (ByVal lTimerId As Long)
```

jest główną funkcją w programie. Powoduje wyliczenie wyjścia układu regulacji i wysłanie tej wartości do regulatora LB – 600. Jest również odpowiedzialna za odświeżanie wykresów oraz napisów informujących np. o potrzebie wciśnięcia jakiegoś przycisku lub aktualnym stanie eksperymentu samostrojania. Funkcja zostaje uruchomiona z chwilą rozpoczęcia regulacji, odpowiedzi skokowej obiektu i eksperymentu samostrojania. Kończy działanie z chwilą wyłączenia programu *Symulator* lub przyciśnięcia przycisków STOP, zatrzymujących regulację i eksperyment samostrojania. Wywołuje się co czas odpytywania.

W pierwszym etapie sprawdzany jest stan, w którym znajduje się program, tzn. czy uruchomiono samostrojanie, czy regulację. W zależności od uruchomionej funkcji, sterowanie odczytywane jest z suwaka lub z rejestru regulatora (funktor 71).

```
If otwarty = False Then                                'czy układ regulacji jest otwarty
  If manual_komp = True Then                            'sterowanie odczytywane z komputera
    sterowanie2 = (inz2.SliderSTER.Value / 100 * (CV_max - CV_min)) + CV_min
  Else
    sterowanie2 = Fix32.Fix.ster.f_cv                 'sterowanie z wyjścia 7 warstwy
    sterowanie2 = CV_min + sterowanie2 * (CV_max - CV_min)
                                                    'przeskalowanie sterowania
  End If
Else
  sterowanie2 = zatrask + inz2.odp_skokowa.CurrentValue
                                                    'skok wartości sterowania
End If
```

Następnie wartość sterowania modyfikowana jest przez nieliniowość obiektu i zapisywana do tablicy przesuwanej na pierwszej pozycji (indeks 0). Z tak wypełnionej (w ciągu cykli pracy) tablicy odczytywana jest wartość sterowania z dowolnej chwili, nie większej jednak niż sprzed chwili określonej przez opóźnienie (por. 11.3).

Danych o nieliniowości jest 35. Stąd powstaje 35 zakresów sterowania. W zależności od tego, do którego zakresu należy sterowanie, generowana jest odpowiednia modyfikacja sterowania.

```
ind = (sterowanie2 - CV_min) / wspolczynnik           'przedział sterowania
If ind >= 35 Then
  ind_int = Fix(ind)                                  'czesc calkowita
Else
  ind_int = Fix(ind)                                  'ustalenie numeru przedziału
End If
za_NL = nielin(ind_int)                              'wartość sterowania w tym przedziale
```

Jeśli sterowanie znajduje się pomiędzy zakresami określonymi przez wczytane punkty, program dokonuje aproksymacji zerowego rzędu -

W kolejnym fragmencie kodu obliczane jest wyjście układu dla obiektu dwuinercyjnego.

11.2 Implementacja modelu obiektu

We fragmencie kodu odpowiedzialnym za wyliczenie wyjścia, zaimplementowano model obiektu dwuinercyjnego. Transmitancję $G(s) = \frac{K_{ob} \cdot e^{-T_0 s}}{(1 + T_1 \cdot s) \cdot (1 + T_2 \cdot s)}$ (por. 2.1) zapisano w

sposób: $G(s) = e^{-sT_0} \left(\frac{A}{1 + T_1 \cdot s} + \frac{B}{1 + T_2 \cdot s} \right)$. Porównując wielomiany w licznikach uzyskano

wartości parametrów A i B . Następnie zdyskretyzowano wartości w nawiasie, otrzymując $G(z)$, i tak wyliczono równanie wyjścia: $wjście = sterowanie \cdot G(z)$.

```

g1 = Exp(-inz2.Tp.InitialValue / inz2.T1.CurrentValue)
g2 = Exp(-inz2.Tp.InitialValue / inz2.T2.CurrentValue)
A = inz2.T1.CurrentValue / (inz2.T2.CurrentValue -
                             inz2.T1.CurrentValue)
B = inz2.T2.CurrentValue / (inz2.T2.CurrentValue -
                             inz2.T1.CurrentValue)
wp = 0          'wartość początkowa
vp = 0          'wartość początkowa
mp = u(inz2.opoznienie.CurrentValue * 1000 / inz2.timer.Interval)
                                     'sterowanie

v = g1 * vp + (1 - g1) * A * mp      ' równanie obiektu
w = g2 * wp + (1 - g2) * B * mp      ' równanie obiektu
inz2.wyjscie.CurrentValue = inz2.K.CurrentValue * (w - v)
vp = v                                'aktualizacja zmiennych
wp = w

```

W rozdziale 10 zwrócono uwagę aby nie przeprowadzać prezentacji programu *Model* na wolnych komputerach (z częstotliwością taktowania zegara poniżej 300 MHz). W przeciwnym wypadku spowolnieniu ulegnie m.in. rysowanie wykresów, a co ważniejsze wartość wyjścia modelu obiektu będzie wysyłana do regulatora w większych odstępach czasu. Może to powodować niekorzystne skoki wartości sterowania i pogorszenie jakości regulacji.

11.3 Funkcje odpowiedzialne za wciśnięcie przycisku

Za wykonanie odpowiedniej akcji przy kliknięciu przycisku odpowiada funkcja:

```

Private Sub nazwa_przyciskuClick()
...
End Sub

```

I tak, za wyświetlenie pliku pomocy odpowiada funkcja:

```
Private Sub help_Click()
    Dim help As New Word.Application
    help.Documents.Open ("C:\help.rtf")
    help.ActiveWindow.ActivePane.View.Zoom.Percentage = 100
    help.ActiveWindow.View.Type = wdPrintView
    help.Visible = True
End Sub
```

Przesunięcie suwakiem wartości sterowania powoduje wywołanie się funkcji:

```
Private Sub SliderSTER_Click()
inz2.sterowanie.CurrentValue=inz2.SliderSTER.Value * (CV_max-CV_min) + CV_min
End Sub
```

11.4 Opis testowania szybkości obliczeń

Funkcja:

```
Private Sub testuj_Click()
testuje szybkość wykonywania obliczeń przez komputer. W tym celu wykorzystywany
jest plik biblioteczny kernel32.dll.
```

Działanie testowania polega na zmierzeniu liczby wykonanych okresów zegarowych, trwających 1 ms, pomiędzy dwoma wywołaniami funkcji:

GetTickCount ()

Pomiędzy nimi umieszczono kod obliczający wyjście układu regulacji:

```
startTime = GetTickCount()
...
endTime = GetTickCount()
wynik = endTime - startTime
```

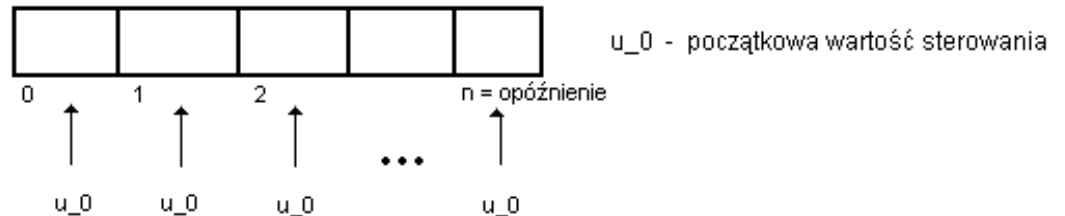
11.5 Opis mechanizmu opóźnienia

W celu realizacji mechanizmu opóźnienia zastosowano przesuwaną tablicę. Aktualna wartość sterowania wstawiana jest na początek tablicy, po uprzednim przesunięciu wszystkich poprzednich wartości sterowania na pozycje o jeden większą. W ten sposób chcąc odczytać wartość sterowania sprzed np. 7 sekund należy odczytać wartość zmiennej:

$$tablica \left[\frac{7 [s]}{\text{czas odpytywania [s]} } \right]$$

Ogólny schemat postępowania $k+1$ -ego kroku obrazuje rysunek 10.4 (przy założeniu, że opóźnienie $T_0 \geq k+1$).

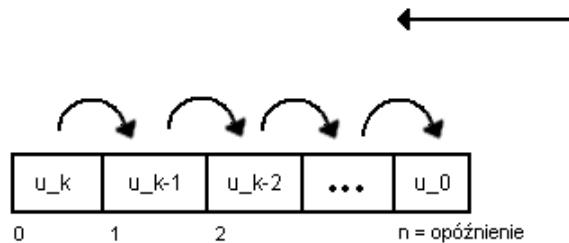
Na początku uruchomienia programu, tablica zostaje wypełniona wartością początkową sterowania równą 0. Ma to na celu zobrazowanie, że wyjście układu zmienia się po czasie równym czasowi opóźnienia.



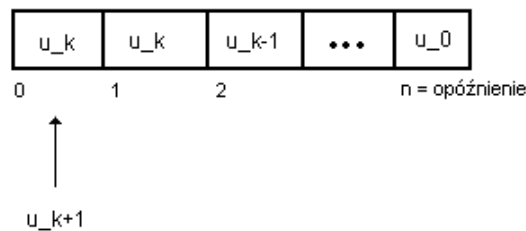
Rys. 11.1 Opis mechanizmu opóźnienia – wstawienie wartości początkowej

Następnie następuje przesunięcie wartości zmiennych na pozycje dalsze i wpisanie aktualnego sterowania na pierwszą pozycję.

I etap - przesunięcie zmiennych



II etap - wpisanie na pierwszą pozycję aktualnego sterowania



Rys. 11.2 Schemat działania mechanizmu opóźnienia

12. Testowanie programu

Wykorzystując funkcje regulatora LB – 600 oraz możliwości programu *Symulator* dobrano przykładowy model obiektu dwuinercyjnego z opóźnieniem (Rys. 1.1) oraz przykładową nieliniowość (Rys. 1.2). Posłużyły one do prezentacji zarówno pracy dyplomowej jak i możliwości samego regulatora.

Obiekt przechodzi do stanu ustalonego po ok. 200s. Taka wartość dobrze oddaje charakter rzeczywistych obiektów, jednocześnie pozwala na przeprowadzenie żmudnych eksperymentów samostrojenia w niedługim czasie.

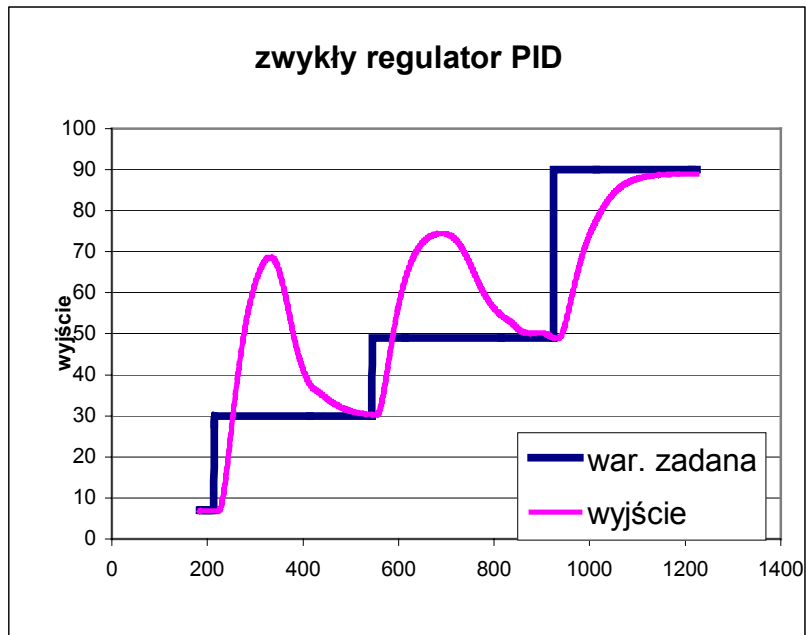
Dość pracochłonne okazało się przetestowanie mechanizmu opóźnienia. Aby stwierdzić czy obiekt rzeczywiście reaguje z podanym opóźnieniem, posłużono się programem *Matlab*. Przygotowano skrypt symulujący zachowanie się podanego obiektu na odpowiedź skokową. Celem przeprowadzonych symulacji było uzyskanie wartości wyjścia obiektu w kolejnych chwilach czasu. Następnie porównano je z wartościami generowanymi przez program *Symulator*. Nie zanotowano większych różnic.

Kolejnym krokiem testowania było porównanie nastaw otrzymanych procedurą samostrojenia z nastawami obliczonymi przez program *Matlab*. W tym celu stworzono *m-plik*, obliczający nastawy metodą Cohena – Coona. Strojenie starano się przeprowadzić jak najdokładniej w celu późniejszego porównania wyników. Wyniki testów nie były jednakowe. Wartości stałych całkowania i różniczkowania były zbliżone, podczas gdy wzmocnienie regulatora uzyskane samostrojeniem różniło się o 40 % w porównaniu do wyniku uzyskanego symulacjami w *Matlabie*. Było to spowodowane pewną niedokładnością w oszacowaniu czasów w eksperymencie uruchamianym w regulatorze. Ponadto eksperyment uruchamiany w regulatorze trwa ok. 800s. Z tego powodu wprowadzono pewne, niewielkie uchyby w ustalaniu równowagi. W ten sposób układ nie osiąga dokładnie stanu ustalonego, lecz eksperyment trwa zdecydowanie krócej. Oto uzyskane wyniki:

- dla programu *Symulator* : $K_p = 1.745$, $T_i = 64.50$, $T_d = 11.16$
- dla programu *Matlab* : $K_p = 2.99$, $T_i = 66.79$, $T_d = 11.01$

Mając dwa różne komplety nastaw postanowiono sprawdzić, jak zachowa się wyjście układu regulacji. Symulację przeprowadzono w programie *Symulator*. Wprowadzono otrzymane nastawy i zapewniono identyczne warunki regulacji. Wyniki były zbliżone. W przypadku nastaw uzyskanych programem *Symulator* układ wykazywał szybsze osiągnięcie stanu ustalonego.

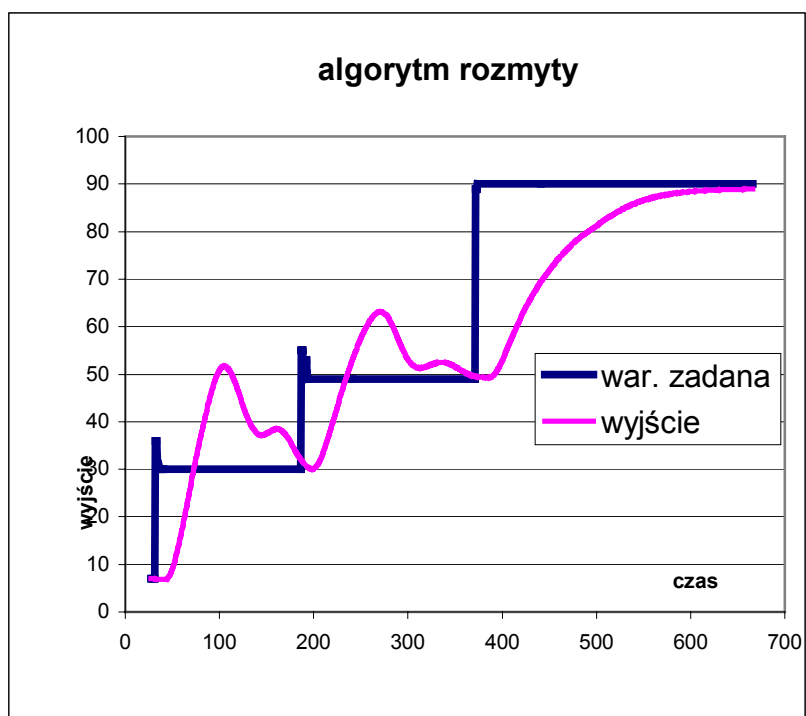
Najważniejszym etapem przeprowadzonych testów było porównanie regulacji automatycznej z rozmytą. Nastawy do regulacji automatycznej zostały wyznaczone przez eksperyment samostrojenia, dla punktu pracy równego 14,46. Symulację rozpoczęto gdy wartości wyjścia ustaliła się na poziomie 7. Gdy wyjście ustabilizowało się, ustawiono wartość zadaną równą 30 a gdy układ doszedł do stanu ustalonego zmieniono wartość zadaną na 49 a następnie po osiągnięciu stanu ustalonego na 90. Zwykły algorytm PID nie poradził sobie dobrze ze zmianą wielkości wartości zadanej. Ilustruje to rysunek 12.1



Rys. 12.1 Wyniki symulacji zwykłym regulatorem PID

Czas stabilizowania się wyjścia do wartości zadanej równej 30 wynosi około 370 s. Występują przeregulowania sięgające 100% ! Regulator reaguje za wolno na zmiany wartości zadanej.

Następnie uruchomiono tą samą symulację dla regulacji rozmytej. Przy dobieraniu nastaw do regulacji rozmytej posłużono się procedurami samostrojania zaimplementowanymi w regulatorze. W ten sposób uzyskano komplet 10 nastaw w najbardziej charakterystycznych punktach nieliniowości obiektu. Resztę punktów dobrano opierając się na wynikach poprzednich symulacji. Tak dobrane nastawy posłużyły do przeprowadzenia symulacji. Wyniki prezentuje rysunek 12.2



Rys. 12.2 Wyniki przeprowadzonej symulacji algorytmem rozmytym

Algorytm również nie poradził sobie najlepiej z takim przebiegiem zmian wartości zadanej ale przebiegi przejściowe są znacznie lepsze niż w przypadku zwykłego algorytmu PID. Wartość zadaną równą 30 układ osiągnął po upływie ok. 190 s czyli po dwukrotnie krótszym czasie. Jednocześnie czas przeprowadzenia całej symulacji jest zdecydowanie krótszy niż w przypadku zwykłego algorytmu PID. Przeregulowania są znaczne mniejsze.

W celu polepszenia jakości regulacji można pokusić się o dokładniejsze wyliczenie nastaw oraz zwiększenie liczby punktów, w których dobrano nastawy. Wykorzystano jedynie 29% możliwych do wysłania regulatorowi nastaw.

Przedstawione powyżej wyniki symulacji pokazują przewagę algorytmu rozmytego nad zwykłym PID. Stanowią również potwierdzenie dużych możliwości wykorzystania programu *Symulator*.

13. Współpraca z firmą LAB – EL

Niniejsza praca dyplomowa powstała dzięki uprzejmości firmy LAB – EL, która wypożyczyła na czas pisania pracy mikroprocesorowy regulator przemysłowy LB – 600. Z uwagi na fakt, że urządzenie jest cały czas udoskonalane i dostosowywane do potrzeb indywidualnych klientów istniała potrzeba przeładowywania oprogramowania regulatora.

W czasie spotkań z przedstawicielem firmy i zarazem projektantem urządzenia, p. Wojciechem Szkolnikowskim, zostało rozwiązanych wiele problemów dotyczących uruchomienia poszczególnych funkcji regulatora. Jego cenne uwagi przyczyniły się do powstania tej pracy.

Temat pracy i związane z nim wykorzystanie niektórych właściwości regulatora przyczyniły się do rozbudowania oprogramowania urządzenia.

14. Zakończenie

Wszystkie zadania ujęte w zakresie pracy zostały zrealizowane, jednakże praca nie wyczerpała wszystkich zagadnień związanych z prezentacją regulatora LB – 600. Starano się w niej pokazać jedynie najważniejsze cechy urządzenia, świadczące o jego dużych możliwościach.

Praca miała charakter dydaktyczny. Pozwoliła na poznanie budowy zarówno systemu iFIX, jak i przemysłowego regulatora mikroprocesorowego wykorzystywanego przez największe firmy w kraju, łącznie ze sposobem jego programowania. Realizacja przyjętych na wstępie założeń okazała się trudniejsza niż na wstępie zakładano. Szczególne trudności nastroczało oprogramowanie eksperymentu samostrojenia. Nie lada problemem sprawiło również każde przeładowanie oprogramowania regulatora. Wiązało się to niejednokrotnie ze zmianą adresów zmiennych. Mimo to efekt końcowy jest zgodny z założeniami projektowymi.

Praca umożliwia przesymulowanie zachowania się obiektu o dowolnej dynamice i nieliniowości. Staje się wygodnym narzędziem do przedstawienia zachowania się różnego typu obiektów, od jednoinercyjnych po oscylacyjne, na skoki lub impulsy sterowania.

Na szczególną uwagę zasługuje przedstawiony w pracy algorytm regulacji rozmytej. Nie jest on powszechnie wykorzystywany w rozwiązaniach przemysłowych ale mam nadzieję, że zamieszczone w pracy wyniki przeprowadzonej symulacji przekonują o dużych możliwościach jego stosowania.

Cennym walorem dydaktycznym jest możliwość uruchomienia eksperymentu samostrojenia. Identyfikuje on obiekt, podając jego trzy parametry: wzmocnienie, opóźnienie i stałą czasową.

Program może zostać wykorzystany w celu prezentacji systemu iFIX gronu studentów rozpoczynających kursy z zakresu automatyki i wizualizacji procesów przemysłowych. Obszerny opis konfiguracji systemu może służyć jako instrukcja obsługi dla osób pragnących samodzielnie wykorzystać omawiany system.

Z pewnością praca dyplomowa przyczyniła się do pogłębienia mojej wiedzy z zakresu wizualizacji systemów i urządzeń automatyki a przede wszystkim budowy urządzeń wykorzystanych w przemyśle. Pogłębiła moją wiedzę z zakresu dobierania nastaw regulatora PID.

15. Literatura

1. Z. Komor, A. Łobzowski, w. Szkolnikowski,
"Technika regulacji, Regulator LB – 600 "
2. Instrukcja eksploatacyjna i tablice konfiguracyjne regulatora LB-600
3. A. Markowski, J.Kostro, A. Lewandowski,
"Automatyka w pytaniach i odpowiedziach"
4. skrypt autorstwa firmy AB-MICRO,
"Wprowadzenie do systemu oprogramowania iFIX firmy GE Fanuc Intellution"
5. <http://support.microsoft.com>
6. <http://www.magictree.com/vbcourse>
7. <http://vb4all.canpol.pl/teoria/iso2/>

16. Załączniki

16.1 Przykładowe skrypty w Visual Basic'u

```
Private Sub CommandButton16_Click()           'zatwierdzenie danych o dynamicce  
obiektu  
    inz2.Group32.Visible = False  
    ReDim u(inz2.opoznienie.CurrentValue * (1000 /  
            inz2.timer.Interval) + 1)  
    opoznienie_true = True  
    If nielin_true = True Then  
        inz2.start.Enabled = True  
        inz2.skok.Enabled = True  
        inz2.ster_reczne.Enabled = True  
    Else  
        inz2.start.Enabled = False  
        inz2.skok.Enabled = False  
        inz2.ster_reczne.Enabled = False  
        inz2.info.Caption = "Wczytaj nieliniowosc"  
    End If  
    inz2.info.Caption = "Wczytaj nastawy"  
End Sub
```

```
Private Sub help_Click()                     'uruchomienie pliku pomocy  
    Dim help As New Word.Application  
    help.Documents.Open ("C:\help.rtf")  
    help.ActiveWindow.ActivePane.View.Zoom.Percentage = 100  
    help.ActiveWindow.View.Type = wdPrintView  
    help.Visible = True  
End Sub
```

16.2 Lista zmiennych użytych w programie *Symulator*

Lista odczytywanych zmiennych z regulatora			
nazwa zmiennej	typ	adres w systemie iFIX	funkcja
AKT_7	Unsigned	44987	aktywność bloku wartstwy 7
AKT_DI	Unsigned	42604	aktywność bloku START fuzzy i samostroj
AKT_DI_PAUSE	Unsigned	42624	
AKT_DI_STOP	Unsigned	42614	aktywność bloku STOP fuzzy i samostroje
AUTO_KP	Float	44917	Kp przy regulacji PID
AUTO_TD	Float	44921	Td przy regulacji PID
AUTO_TI	Float	44919	Ti przy regulacji PID
CV_MAX	Float	44909	max wartość sterowania
CV_MIN	Float	44907	min wartość sterowania
F_1_1	Float	47101	parametry tablicy do fuzzy zestaw 1, SP
F_1_2	Float	47103	parametry tablicy do fuzzy zestaw 1, Kp
F_1_3	Float	47105	parametry tablicy do fuzzy zestaw 1, Ti
F_1_4	Float	47107	parametry tablicy do fuzzy zestaw 1, Td
F_2_1	Float	47111	
F_2_2	Float	47113	...
F_2_3	Float	47115	...
F_2_4	Float	47117	...
F_3_1	Float	47121	
F_3_2	Float	47123	
F_3_3	Float	47125	
F_3_4	Float	47127	
F_4_1	Float	47131	
F_4_2	Float	47133	
F_4_3	Float	47135	
F_4_4	Float	47137	
F_5_1	Float	47141	parametry tablicy do fuzzy zestaw 5
F_5_2	Float	47143	
F_5_3	Float	47145	...
F_5_4	Float	47147	...
F_6_1	Float	47151	...
F_6_2	Float	47163	
F_6_3	Float	47155	
F_6_4	Float	47157	
F_7_1	Float	47161	

F_7_2	Float	47163	
F_7_3	Float	47165	
F_7_4	Float	47167	
F_8_1	Float	47171	
F_8_2	Float	47173	
F_8_3	Float	47175	
F_8_4	Float	47177	
F_9_1	Float	47181	
F_9_2	Float	47183	
F_9_3	Float	47185	
F_9_4	Float	47187	
F_10_1	Float	47191	parametry tablicy do fuzzy zestaw 10
F_10_2	Float	47193	
F_10_3	Float	47195	...
F_10_4	Float	47197	...
F_11_1	Float	47201	...
F_11_2	Float	47203	
F_11_3	Float	47205	
F_11_4	Float	47207	
F_12_1	Float	47211	
F_12_2	Float	47213	
F_12_3	Float	47215	
F_12_4	Float	47217	
F_13_1	Float	47221	
F_13_2	Float	47223	
F_13_3	Float	47225	
F_13_4	Float	47227	
F_14_1	Float	47231	
F_14_2	Float	47233	
F_14_3	Float	47235	
F_14_4	Float	47237	
F_15_1	Float	47241	parametry tablicy do fuzzy zestaw 15
F_15_2	Float	47243	
F_15_3	Float	47245	...
F_15_4	Float	47247	...
F_16_1	Float	47251	...
F_16_2	Float	47253	
F_16_3	Float	47255	
F_16_4	Float	47257	
F_17_1	Float	47261	
F_17_2	Float	47263	
F_17_3	Float	47265	

F_17_4	Float	47267	
F_18_1	BWLong	47271	
F_18_2	Float	47273	
F_18_3	Float	47275	
F_18_4	Float	47277	
F_19_1	Float	47281	
F_19_2	Float	47283	
F_19_3	Float	47285	
F_19_4	Float	47287	
F_20_1	Float	47291	parametry tablicy do fuzzy zestaw 20
F_20_2	Float	47293	
F_20_3	Float	47295	...
F_20_4	Float	47297	...
F_21_1	Float	47301	...
F_21_2	Float	47303	
F_21_3	Float	47305	
F_21_4	Float	47307	
F_22_1	Float	47311	
F_22_2	Float	47313	
F_22_3	Float	47315	
F_22_4	Float	47317	
F_23_1	Float	47321	
F_23_2	Float	47323	
F_23_3	Float	47325	
F_23_4	Float	47327	
F_24_1	Float	47331	
F_24_2	Float	47333	
F_24_3	Float	47335	
F_24_4	Float	47337	
F_25_1	Float	47341	parametry tablicy do fuzzy zestaw 25
F_25_2	Float	47343	
F_25_3	Float	47347	...
F_25_4	Float	47347	...
F_26_1	Float	47351	...
F_26_2	Float	47353	
F_26_3	Float	47355	
F_26_4	Float	47357	
F_27_1	Float	47361	
F_27_2	Float	47363	
F_27_3	Float	47365	
F_27_4	Float	47367	
F_28_1	Float	47371	

F_28_2	Float	47373	
F_28_3	Float	47375	
F_28_4	Float	47377	
F_29_1	Float	47381	
F_29_2	Float	47383	
F_29_3	Float	47385	
F_29_4	Float	47387	
F_30_1	Float	47391	parametry tablicy do fuzzy zestaw 30
F_30_2	Float	47393	
F_30_3	Float	47395	...
F_30_4	Float	47397	...
F_31_1	Float	47401	...
F_31_2	Float	47403	
F_31_3	Float	47405	
F_31_4	Float	47407	
F_32_1	Float	47411	
F_32_2	Float	47413	
F_32_3	Float	47415	
F_32_4	Float	47417	
F_33_1	Float	47421	
F_33_2	Float	47423	
F_33_3	Float	47425	
F_33_4	Float	47427	
F_34_1	Float	47431	
F_34_2	Float	47433	
F_34_3	Float	47435	
F_34_4	Float	47437	
F_35_1	Float	47441	parametry tablicy do fuzzy zestaw 35
F_35_2	Float	47443	
F_35_3	Float	47445	
F_35_4	Float	47447	
JEDEN2	Float	42914	współczynnik 3114 wyjście układu regulacji
LOGIKA_DI_PAUSE	Unsigned	42622	
LOGIKA_DI_STOP	Unsigned	42612	logika bloku STOP fuzzy i samostrojzenia
LOG_DI	Unsigned	42602	logika bloku START fuzzy i samostrojzeni
NUMER_TABL	Unsigned	44984	numer aktualnie pobieranego zestawu fuzz
ODCHYLKA	Float	44985	uchyb regulacji w %
SAMO_CZAS_STAB	Unsigned	44966	samostrojzenie czas stabilizacji
SAMO_KRYT_REG	Unsigned	44965	samostrojzenie kryterium regulacji
SAMO_SKOK_CV	Float	44968	samostrojzenie skok CV na początku ekspe
SAMO_T	Float	44963	max wartość sygnału odpowiedzi na wymusz
SAMO_ZMIANY_PV	Unsigned	44967	dopuszczalne zmiany wartości PV

SP_TOR	Unsigned	44906	tor pochodzenia SP
SP_WAR	Unsigned	44905	warstwa pochodzenia SP
STER	Unsigned	44988	sterowaniewejście dla programu Model
TRYB_PRACY_7157	Unsigned	44979	tryb pracy regulatora
WAR_ZAD	Float	44929	wartość zadana dla regulacji PID
WAR_ZAD_FUZZY	Float	42974	współczynnik 3314, SP dla fuzzy

Lista zmiennych wewnątrz programu *Symulator*

nazwa zmiennej	typ	funkcja
war_zad		wartość zadana dla układu regulacji
wyjście		wartość wyjścia układu
sterowanie		wartość sterowania
opoznienie		opóźnienie obiektu
numer_tab		numer aktualnie pobieranego zestawu przy regulacji rozmytej
K		wzmocnienie obiektu
T1		stała czasowa 1
T2		stała czasowa 2
Tp		okres próbkowania obiektu

Lista zmiennych w kodzie źródłowym języka Visual Basic

nazwa zmiennej	typ	funkcja
v	Double	zmienna pomocnicza przy modelu dynamiki
vp	Double	zmienna pomocnicza przy modelu dynamiki
mp	Double	zmienna pomocnicza przy modelu dynamiki, przeszłe sterowanie
w	Double	zmienna pomocnicza przy modelu dynamiki
wp	Double	zmienna pomocnicza przy modelu dynamiki
tmp	Double	zmienna pomocnicza
sterowanie2	Double	wartość sterowania
u()	Double	przesuwana tablica na przeszłe wartości sterowania
takt	Long	indeks, przydatny przy wyświetlaniu czasu
takt2	Long	indeks, przydatny przy kasowaniu upłyniętego czasu
temp	Double	zmienna pomocnicza
g1	Double	zmienna pomocnicza przy modelu dynamiki
g2	Double	zmienna pomocnicza przy modelu dynamiki
A	Double	zmienna pomocnicza przy modelu dynamiki

B	Double	zmienna pomocnicza przy modelu dynamiki
nielin(35)	Double	tablica na wartości nieliniowości
CV_min	Double	minimalna wartość sterowania
CV_max	Double	maksymalna wartość sterowania
wynik	Long	wynik przeprowadzonego testu szybkości obliczeń
otwarty	Boolean	czy układ regulacji jest otwarty
fuzzy	Boolean	czy jest uruchomiona regulacja rozmyta
auto	Boolean	czy jest uruchomiona regulacja automatyczna
jeden_raz	Boolean	wypełnienie tablicy TLKO jeden raz, na początku programu
zatrask	Double	zatraskiwana wartość wyjścia przy odp. skokowej
punkt_pracy	Double	wartość sterowania odpowiadająca jednemu zestawowi nastaw
temp_nastawy(36,4)	Double	zmienna pomocnicza na zestaw 35 nastaw PID
exa	Double	zmienna pomocnicza
wspolczynnik	Double	"szerokość sterowania" przy nieliniowości
za_NL	Double	wartość sterowania po przejściu przez nielin. obiektu
ind	Double	indeks tablicy <i>nielin()</i> , rzeczywisty
ind_int	Integer	indeks tablicy <i>nielin()</i> , całkowity
nielin_true	Boolean	czy poprawnie wczytano dane o nieliniowości obiektu
nastawy_true	Boolean	czy poprawnie wczytano dane o nastawach PID regulatora
samo_true	Boolean	czy uruchomiono samostroenie
manual_komp	Boolean	czy uruchomiono sterowanie ręczne w programie
ster_skok	Double	wartość sterowania, przydatna do określenia etapu samostrojenia
ster_skok0	Double	wartość sterowania, przydatna do określenia etapu samostrojenia

16.3 Spis rysunków

Rys. 1.1	Opracowany układ regulacji.....	6
Rys. 2.1	Odpowiedź skokowa części dynamicznej obiektu.....	7
Rys. 2.2	Przykładowa nieliniowość obiektu.....	8
Rys. 3.1	Przykładowy funktor.....	9
Rys. 3.2	Zmienne z funkтора 31.....	10
Rys. 4.1	Struktura funkcjonalna zaprogramowanego regulatora LB – 600.....	11
Rys. 6.1	Schemat układ regulacji rozmytej z wykorzystaniem funktoró.....	15
Rys. 6.2	Schemat blokowy podstawowego układu regulacji.....	17
Rys. 6.3a	Schemat tablicy wykorzystanej do regulacji rozmytej.....	18
Rys. 6.3b	Przykładowa nieliniowa zależność.....	18
Rys. 8.1	Podział na funkcje przynależności i zbiory rozmyte.....	20
Rys. 8.2	Plik konfiguracyjny programu iFIX.....	21
Rys. 9.1	Konfiguracja węzła SCADA: wybór bazy danych i drajwera.....	22
Rys. 9.2	Konfiguracja drajwera MB1.....	23
Rys. 9.3	Funkcje dostępne dla definicji zmiennych i bloków.....	23
Rys. 9.4	Konfiguracja rejestru analogowego.....	24
Rys. 9.5	Okno kontrolne systemu iFIX.....	24
Rys. 9.6	Arkusz przykładowej bazy danych.....	25
Rys. 10.1	Ekran główny programu Symulator.....	27
Rys. 10.2	Okno wprowadzania danych dynamiki obiektu.....	28
Rys. 10.3	Dane do eksperymentu samostrojzenia.....	29
Rys. 10.4	Program oczekuje na wciśnięcie przysku Testuj.....	31
Rys. 10.5	Ustawienie parametrów dynamiki.....	31
Rys. 10.6	Program informuje o pracy automatycznej.....	32
Rys. 10.7	Nastawy używane w regulacji automatycznej.....	32
Rys. 10.8	Suwak zadawanie wartości sterowania.....	32
Rys. 10.9	Numer aktualnie pobieranych nastaw.....	33
Rys. 10.10a	Ekran samostrojzenia – ustalanie równowagi.....	34
Rys. 10.10b	Ekran samostrojzenia – właściwy eksperyment.....	34
Rys. 11.1	Opis mechanizmu opóźnienia – wstawienie wartości początkowej.....	36
Rys. 11.2	Schemat działania mechanizmu opóźnienia.....	36
Rys. 12.1	Wyniki symulacji zwykłego regulatora PID.....	43
Rys. 12.1	Wyniki symulacji algorytmem rozmytym.....	43

16.4 Zdjęcie stanowiska pracy



16.5 Oprogramowanie na płycie CD-ROM

Na dołączonej płycie znajduje się:

- program *Symulator.grf*,
- pliki konfiguracyjne drajwera MB1:
TABLICA.mb1,
ZMIENNE.mb1,
- plik pomocy *Help.doc*,
- wyżej przedstawioną pracę w formie elektronicznej (format *pdf*)