



Projektowanie Układów Sterowania

Laboratorium 5

Instrukcja – sterownik PLC, panel operatora, INTECO Część II

SPIS TREŚCI

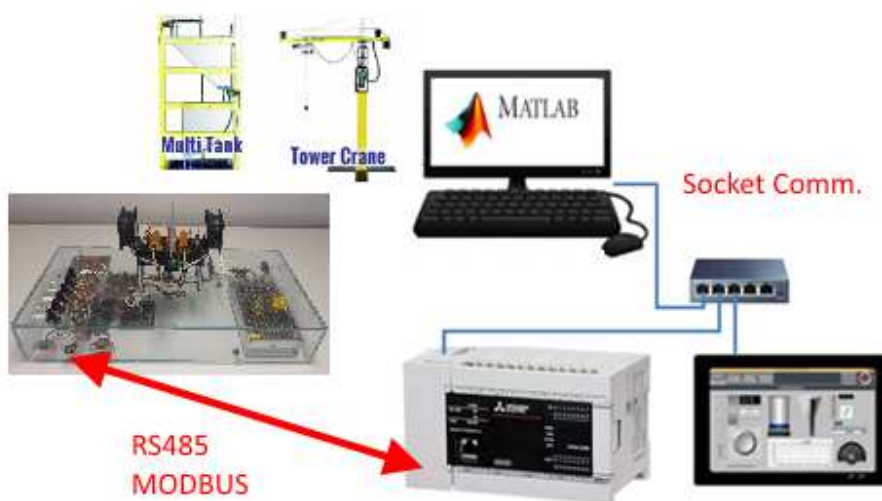
1	CEL ĆWICZENIA	3
2	WYKAZ SPRZĘTU ORAZ OPROGRAMOWANIA.....	3
3	TWORZENIE KODU STERUJĄCEGO STEROWNIKA PLC W ŚRODOWISKU GXWORKS3.....	4
3.1	TWORZENIE NOWEGO PROJEKTU	4
3.2	ELEMENTY JĘZYKA FBD/LD	5
3.3	DEFINICJA ZMIENNYCH	9
3.4	TWORZENIE KODU STERUJĄCEGO.....	10
3.5	KONFIGURACJA STEROWNIKA	12
3.6	PROGRAMOWANIE STEROWNIKA.....	14
3.7	DIAGNOSTYKA, MONITOROWANIE DZIAŁANIA PROGRAMU	16
3.8	PIERWSZY PROGRAM PLC.....	16
3.9	PRZYKŁADOWE IMPLEMENTACJE INTERFEJSÓW SPRZĘTOWYCH.....	18
3.10	PIERWSZY PROGRAM PLC – REGULACJA CIĄGŁA	25
4	DEFINICJA TABLICY TYPU FLOAT	29
5	OPIS KOMUNIKACJI RS485 MODBUS, SOCKET COMMUNICATION.....	30
6	TWORZENIE GRAFIK OPERATORSKICH W ŚRODOWISKU GT DESIGNER 3	35
6.1	PROJEKT DEMO	35
6.2	PANEL MENU – 1	35
6.3	PANEL WEWY – 2.....	36
6.4	PANEL PAMIEC – 3.....	37
6.5	PANEL MANUAL – 4	38
6.6	PANEL PROCES – 5.....	39
6.7	PANEL WYKRES – 6.....	40
6.8	PANEL AUTOMAT – 7.....	40
6.9	PANEL AUTOMAT P – 8.....	41
6.10	PANEL AUTOMAT P – 10	42
6.11	WGRYWANIE PROJEKTU DO PANELA OPERATORA.....	43
7	DOKUMENTACJA.....	44
8	PROJEKT	44

1 Cel ćwiczenia

Zapoznanie się ze sposobem tworzenia oprogramowania dla sterownika programowalnego FX5U firmy Mitsubishi oraz wizualizacji procesu na panelu operatora typu GOT Simple. W ramach ćwiczenia studenci tworzą projekt sterownika w środowisku GxWorks3 oraz wizualizację na panel operatora w środowisku GT Designer 3. Obiektem sterowania będzie stanowisko firmy INTECO w konfiguracji wielowymiarowej. Komunikacja ze stanowiskiem będzie się odbywać przy pomocy dedykowanej płytki konwertującej sygnały. Do odbierania i archiwizacji danych posłuży skrypt napisany w MATLAB na komputerze PC. Komunikacja z MATLAB będzie wykonana przy pomocy Socket Communication.

2 Wykaz sprzętu oraz oprogramowania

Stanowiska laboratoryjne składają się z zestawów dydaktycznych wyposażonych w sterownik PLC MELSEC FX5U firmy Mitsubishi, panel operatorski GOT, komputer stacjonarny oraz stanowisko badawcze.



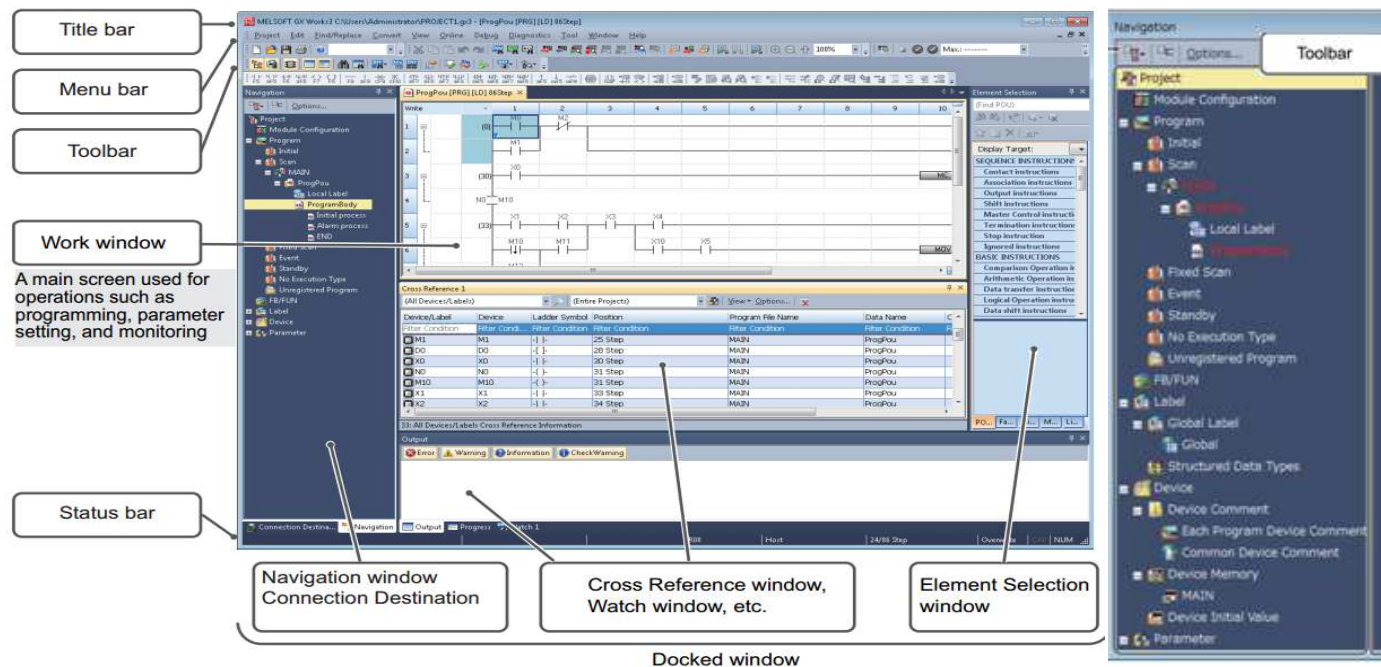
Rysunek 1 Wyposażenie stanowisk laboratoryjnych

W trakcie realizacji ćwiczenia wykorzystane zostanie następujące oprogramowanie:

- **GxWorks3** – oprogramowanie służące do tworzenia aplikacji dla sterownika PLC (*kompilacja, komunikacja ze sterownikiem, debug, alarmowanie*),
- **GT Designer 3** – oprogramowanie służące do tworzenia aplikacji dla panela operatora GOT Simple
- **MATLAB** – odbieranie danych ze sterownika PLC, rysowanie wykresów na komputerze.


3 Tworzenie kodu sterującego sterownika PLC w środowisku GxWorks3

Środowisko GxWorks3



Rysunek 2 Okno główne programu GxWorks3 (z lewej); Pasek nawigacji projektu (z prawej)

3.1 Tworzenie nowego projektu

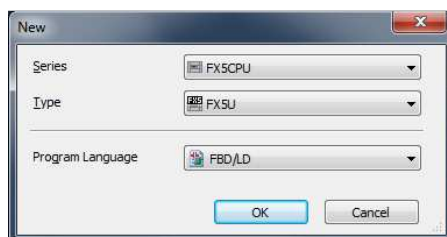
Proszę utworzyć nowy projekt ([Project] → [New]) ()

Proszę zidentyfikować model oraz serię sterownika PLC znajdującego się na stanowisku. Na poniższym rysunku oznaczono czerwonym prostokątem miejsce, w którym znajduje się wymagana informacja.



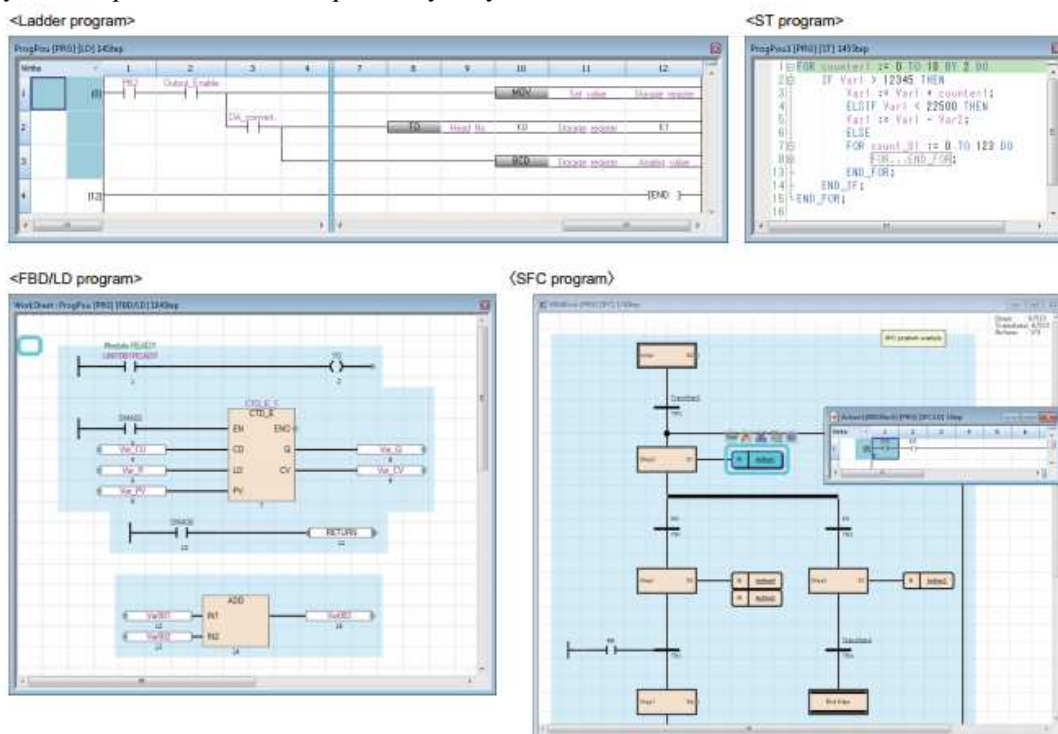
Rysunek 3 Oznaczenie modelu i serii sterownika.

Następnie w programie GxWorks3 proszę wprowadzić dane sterownika oraz wybrać język programowa **FBD/LD** po czym zatwierdzić ustawienia klikając przycisk OK.



Rysunek 4 Parametry wstępne projektu – **NIE POMYLIĆ SERII STEROWNIKA**

Środowisko GxWorks3 wspiera programowanie zgodnie z normą IEC61131-3 (wsparcie dla: FBD/LD, Ladder Diagram, ST i SFC). Przykłady programów we wspomnianych językach zaprezentowano na poniższym rysunku.

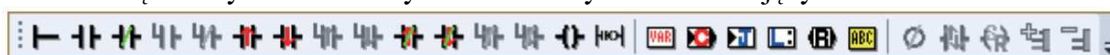


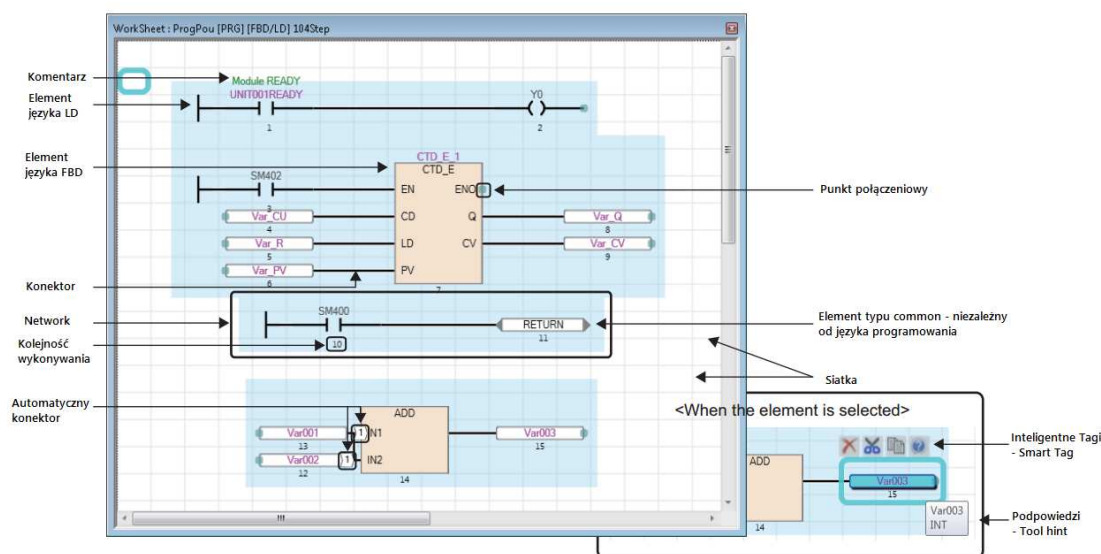
Rysunek 5 Języki programowania wspierane przez aplikację GxWorks3..

Uwaga: Proszę upewnić się, że projekt został zapisany [Project] → [Save] (💾)

3.2 Elementy języka FBD/LD

Pasek narzędziowy zawiera wszystkie elementy strukturalne języka FBD/LD:




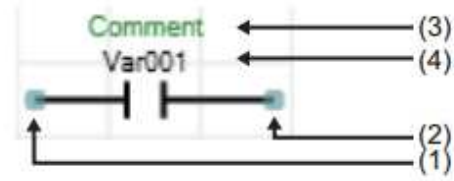
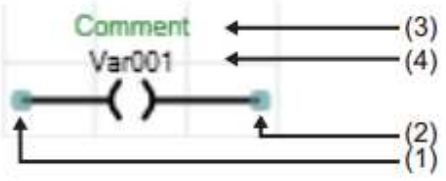


Rysunek 6 Edytor języka FBD/LD.

Element	Opis
Komentarz	Komentarz etykiety lub bloku funkcjonalnego. Nie podlega kompilacji.
Element języka LD	Element pochodzący z języka programowania Ladder Diagram
Element języka FBD	Element pochodzący z języka Function Block Diagram (FBD)
Element typu common	Element wbudowany, niezależny od języka programowania
Konektor	Linia łącząca punkty pomiędzy elementami programu. Możliwe jest automatyczne łączenie punktów poprzez zbliżenie bloków.
Network	Pojedyncza sieć zbudowana ze wszystkich elementów podłączonych razem. Program może zawierać maksymalnie 4096 networków.
Kolejność wykonywania (execution order)	Liczba określająca kolejność wykonania danego elementu programu.
Automatyczny konektor	Jeśli konektor nie może być wyświetlony w danym miejscu, wtedy zostaje zastąpiony liczbą.
Punkt podłączeniowy	Terminal (punkt) pozwalający na połączenie bloków/elementów programu poprzez konektor. Punkty powinny być łączone z uwzględnieniem typów danych.
Siatka	Linie siatki arkusza na którym umieszczane są elementy programu
Smart tag	Przyciski wyświetlane nad wybranym

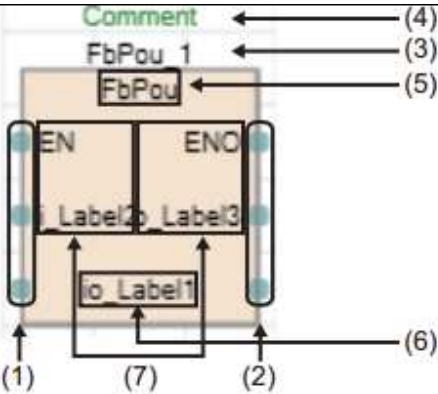
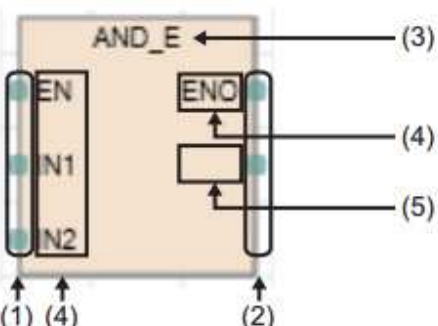
	elementem, pozwalające na wykonanie operacji t.j np. usuwanie lub kopiowanie elementu.
Tool hint	Informacja o elementach programu wyświetlana po najechnaniu kursorem myszki

Elementy języka LD

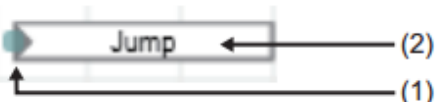

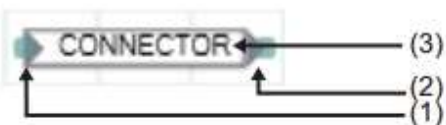
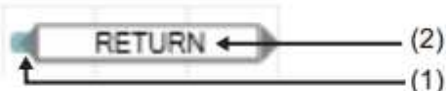
Element	Opis
<p>Lewa szyna zasilająca</p> 	<p>(1) Wyjściowy punkt połączeniowy (2) Lewa szyna zasilająca</p>
<p>Element stykowy</p>  <p>Cewka</p> 	<p>(1) Wejściowy punkt połączeniowy (2) Wyjściowy punkt połączeniowy (3) Komentarz urządzenia/etykiety (4) Urządzenie/etykieta</p>

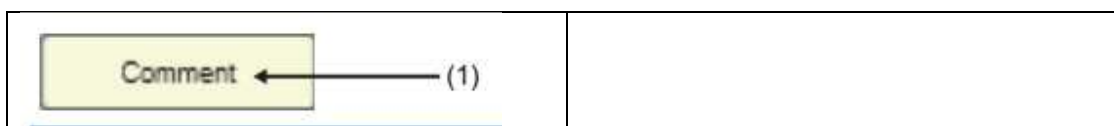
Elementy języka FBD

Element	Opis
<p>Zmienna (lokalna/globalna)</p> 	<p>(1) Wejściowy punkt połączeniowy (2) Wyjściowy punkt połączeniowy (3) Komentarz urządzenia/etykiety (4) Urządzenie/etykieta</p>
<p>Stała</p> 	<p>(1) Wyjściowy punkt połączeniowy (2) Stała wartość</p>
<p>Blok funkcyjny</p>	<p>(1) Wejściowy punkt połączeniowy (2) Wyjściowy punkt połączeniowy</p>

	<p>(3) Nazwa instancji FB (4) Komentarz etykiety (5) Typ danych (6,7) Etykieta wejścia/wyjścia</p> <p>Uwaga: wyjścia z bloku mogą być umieszczone również z lewej strony symbolu bloku funkcyjnego</p>
<p>Funkcja</p> 	<p>(1) Wejściowy punkt połączeniowy (2) Wyjściowy punkt połączeniowy (3) Typ danych (4) Etykieta wejścia/wyjścia</p> <p>Uwaga: wyjścia z bloku mogą być umieszczone również z lewej strony symbolu bloku funkcyjnego</p>

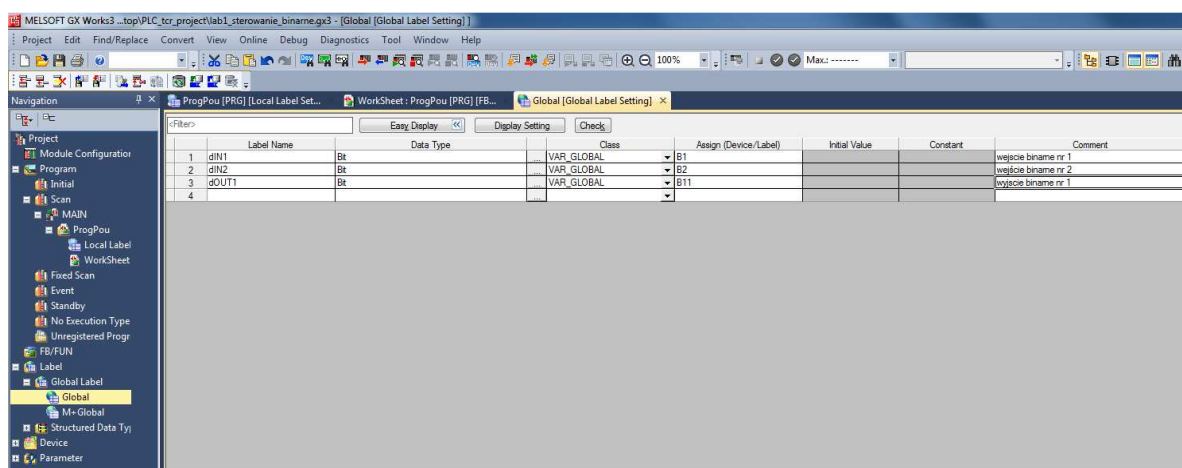
Elementy wspólne (common element)

Element	Opis
<p>Instrukcja skoku (Jump element)</p> 	<p>(1) Wejściowy punkt połączeniowy (2) Etykieta</p>
<p>Etykieta instrukcji skoku</p> 	<p>(1) Wyjściowy punkt połączeniowy</p>
<p>Konektor</p> 	<p>(1) Wejściowy punkt połączeniowy (2) Wyjściowy punkt połączeniowy (3) Komentarz etykiety</p>
<p>Instrukcja return</p> 	<p>(1) Wejściowy punkt połączeniowy (2) Wyjściowy punkt połączeniowy</p>
<p>Blok komentarza</p>	<p>(1) Powierzchnia na której wyświetlona zostanie treść komentarza</p>



3.3 Definicja zmiennych

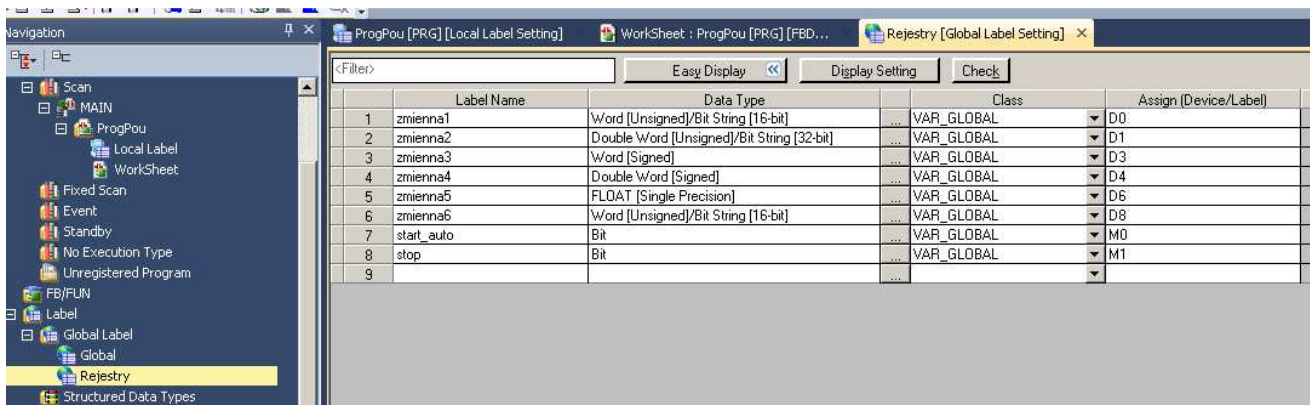
W celu tworzenia czytelnego kodu zalecane jest wykorzystanie zmiennych symbolicznych zamiast adresowania bezpośredniego. W tym celu należy wykorzystać „Labeli”, które mogą mieć zakres lokalny (widoczne tylko w danym komponencie) lub globalny (widoczne w całym systemie i propagowane po sieci – to rozwiązanie jest zalecane z uwagi na możliwość przypisania fizycznych urządzeń, które później będą używane w panelu operatora). Definicja „Labeli” możliwa jest poprzez wypełnienie tabeli (patrz rysunek poniżej) lub poprzez bezpośrednie definiowanie w trakcie tworzenia kodu sterującego - wpisanie nazwy nowej zmiennej symbolicznej w miejscu jej użycia powoduje otwarcie okna dialogowego, gdzie możliwa jest konfiguracja zmiennej.



Rysunek 7 Deklaracja lokalnych/globalnych „Labeli”.

Możliwe jest również tworzenie grup zmiennych tworząc pomocnicze kontenery (np. Wejścia, Wyjścia, Sygnały_analogowe, Sygnały_dyskretne itp.). Aby to zrobić należy w oknie Navigation przejść do zakładki Label->Global Label, następnie kliknąć prawym przyciskiem myszy i wybrać opcję Add New Data.

W trakcie laboratorium przydatne będą urządzenia typu Bit, Rejestr. Zakres urządzeń typu Bit zawiera się w zakresie od M0 do M7680 numerowane co jeden. Zakres urządzeń typu rejestr (16 bit) zawiera się w zakresie D0 do D7999 numerowane co jeden. Proszę zwrócić szczególną uwagę, że zmienne typu Double lub Float zajmują dwa rejestry D. Rysunek 7a przedstawia przykładową konfigurację. W programie PLC można używać „Labeli”, ale do elementów w panelu operatora należy używać bezpośrednich adresów pamięci zadeklarowanych w kolumnie Assign.



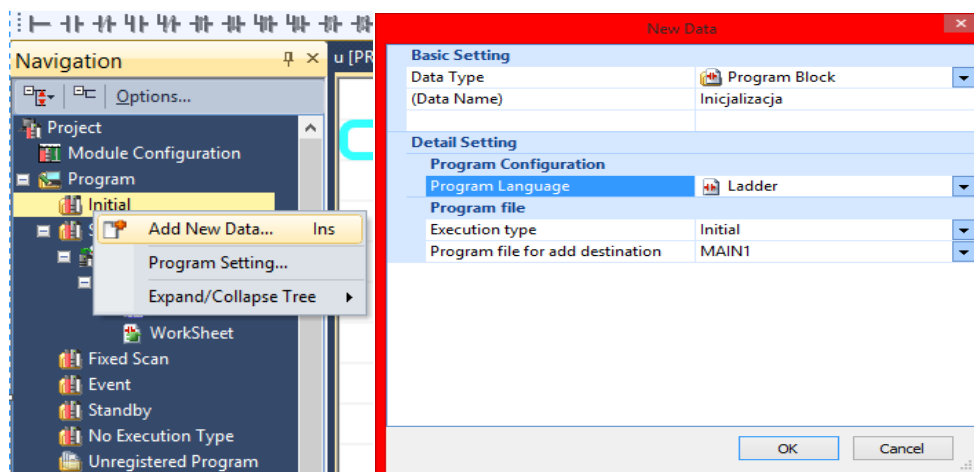
Rysunek 8a Deklaracja globalnych „Labeli”.

3.4 Tworzenie kodu sterującego

W zależności od sposobu wykonywania programu sterującego należy określić jego lokalizację w drzewie projektu. Wspierane są następujące sekcje:

- Initial - instrukcje wykonywane tylko w pierwszym cyklu sterownika,
- Scan - główny skan procesora, czas cyklu zależny obciążenia,
- Fixed scan - skan z narzuconym okresem wykonania (czas konfigurowalny),
- Event - obsługa zdarzeń,
- No execution Type - magazyn kodu, który nie jest wykonywany.

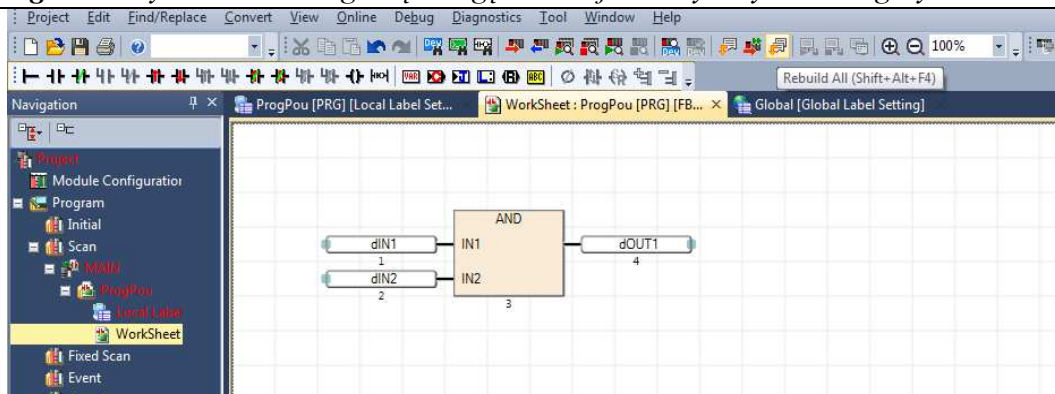
W każdej z sekcji można stworzyć kilka podprogramów klikając w sekcję prawym przyciskiem myszy a następnie wybierając z menu opcję „Add New Data” (Patrz poniżej).



Rysunek 9 Dodawanie nowych podprogramów

Wstawianie elementów języka FBD na arkusz roboczy może odbywać się dzięki technice „drag and drop” z biblioteki, lub poprzez bezpośrednie wpisywanie z klawiatury nazwy bloku funkcyjnego. W trakcie wpisywania kolejnych znaków odpowiedzi o dostępnych blokach są wyświetlane pod tworzącym blokiem.

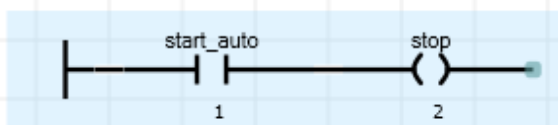
Uwaga: Należy zwrócić szczególną uwagę na kolejność wykonywania algorytmów.



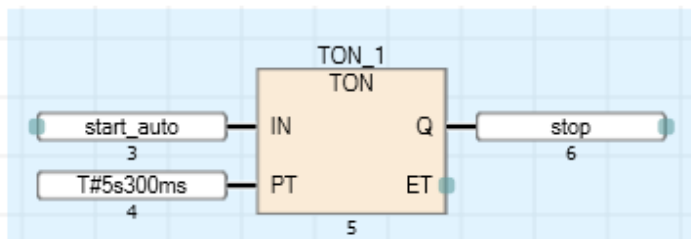
Rysunek 10 Przykład tworzenia kodu sterującego.

W czasie laboratorium najbardziej użyteczne będą następujące instrukcje:

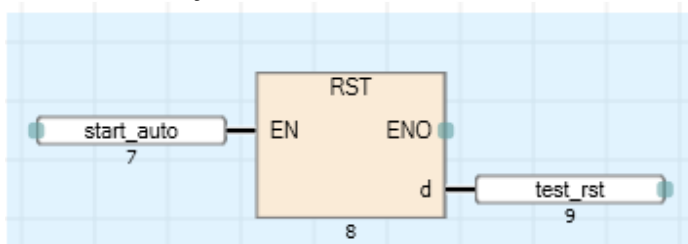
1. Styk normalnie otwarty
2. Cewka wyjściowa (należy pamiętać, że w programie powinna być tylko jeden raz do jednej zmiennej)



3. Opóźnienie załączenia TON, opóźnienie wyłączenia TOF, impuls o zadanym czasie TP



4. Instrukcja ustawienia SET, kasowania RST



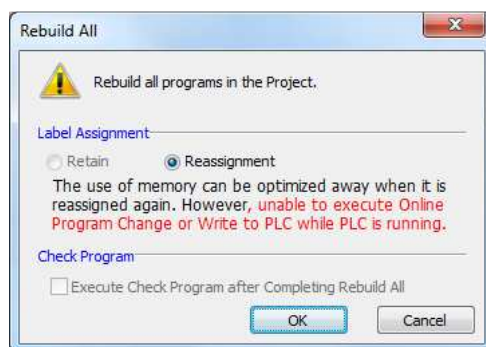
Należy unikać stosowania nazw zmiennych, które mogą być nazwami własnymi zastrzeżonymi w programie np. STOP, ST. Mogą one oznaczać nazwy instrukcji lub nazwy urządzeń fizycznych.

Kompilacja kodu



- 1 – Kompilacja (po małych modyfikacjach)
- 2 – Rekompilacja (po zmianach konfiguracyjnych)

Rekompilacja wymaga potwierdzenia komunikatu:

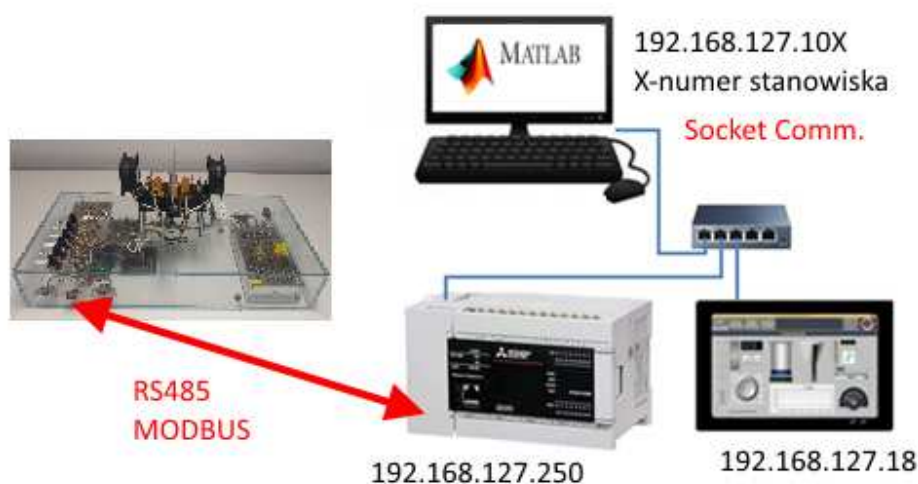


Rysunek 11 Rekompilacja – okno dialogowe

Po rekompilacji projektu nie możliwe będzie ładowanie sterownika w trybie Online. Z tego powodu drobne zmiany w programie należy zatwierdzać bezpośrednio zapisując projekt i wywołując komendę „Online Program Change”.

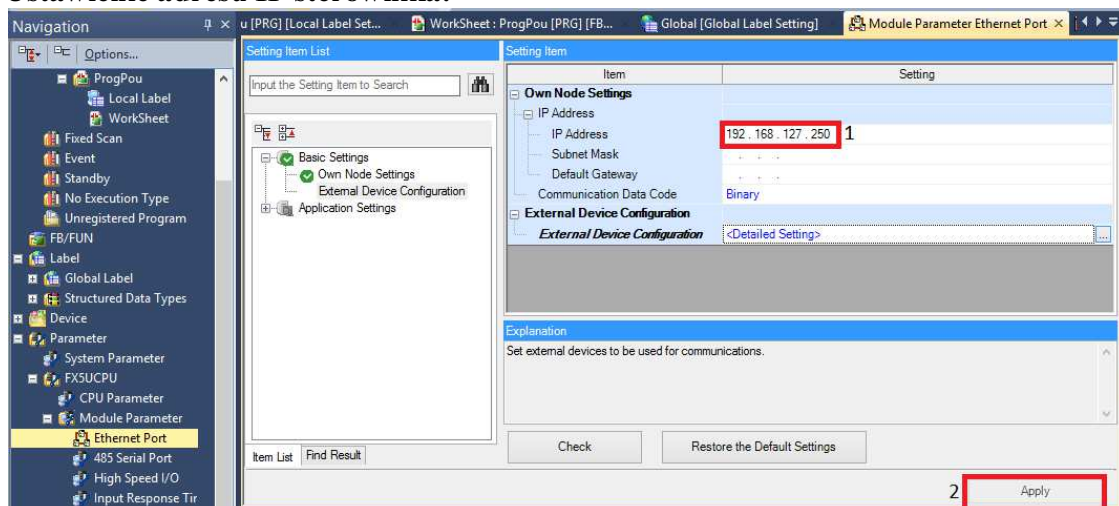
3.5 Konfiguracja sterownika

W celu umożliwienia komunikacji sterownika z panelem GOT oraz komputerem (MATLAB) należy skonfigurować jego ustawienia sieciowe. Poniższe instrukcję przeprowadzają przez wymagane operacje.



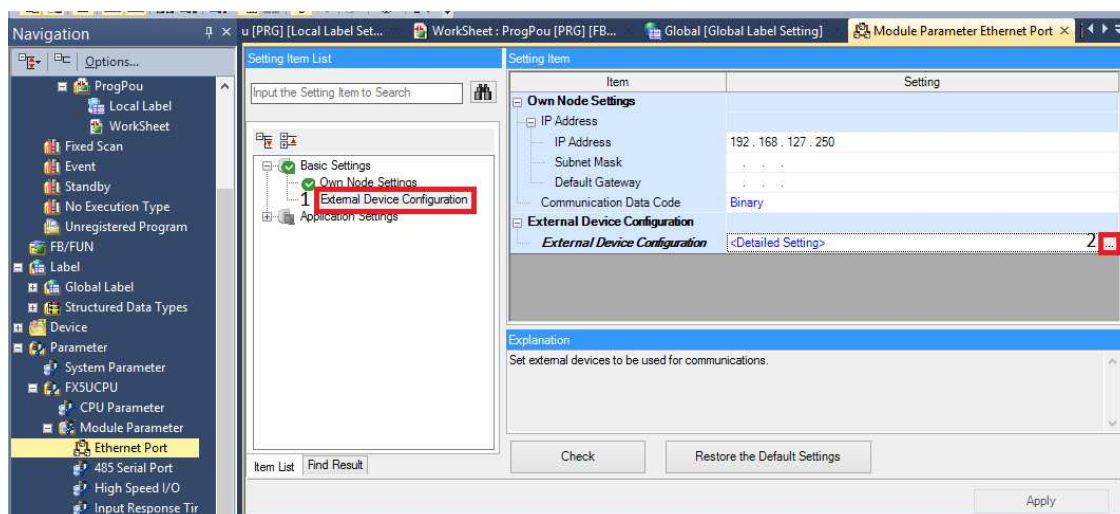
Rysunek 12 Adresacja urządzeń w sieci lokalnej stanowiska

Ustawienie adresu IP sterownika:

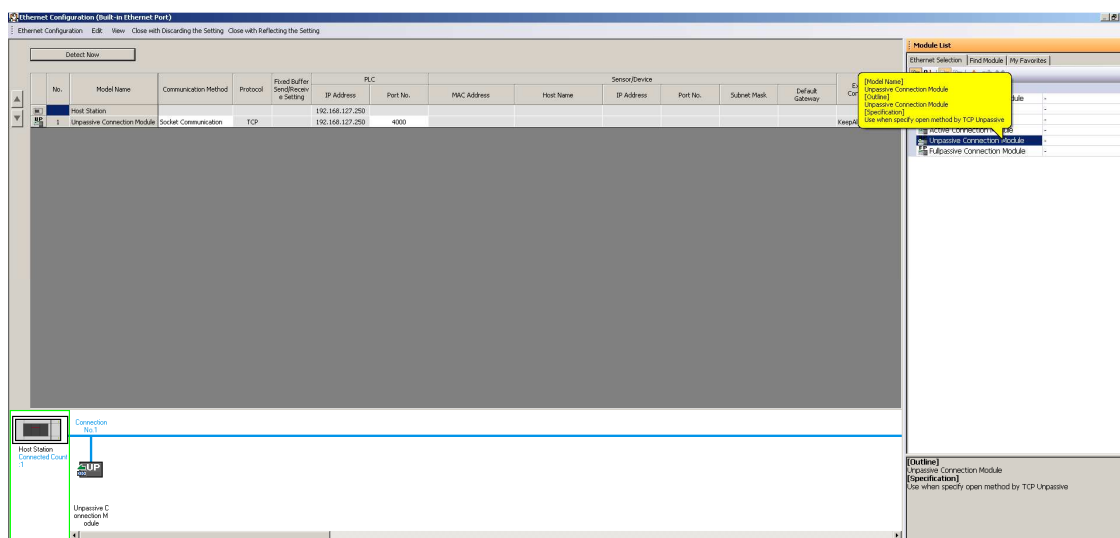


Rysunek 13 Ustawienie adresu IP portu Ethernet

Ustawienie komunikacji z komputerem:



Rysunek 14 Wywołania okna konfiguracji zewnętrznej komunikacji



Rysunek 15 Dodanie komunikacji Unpassive (port 4000)

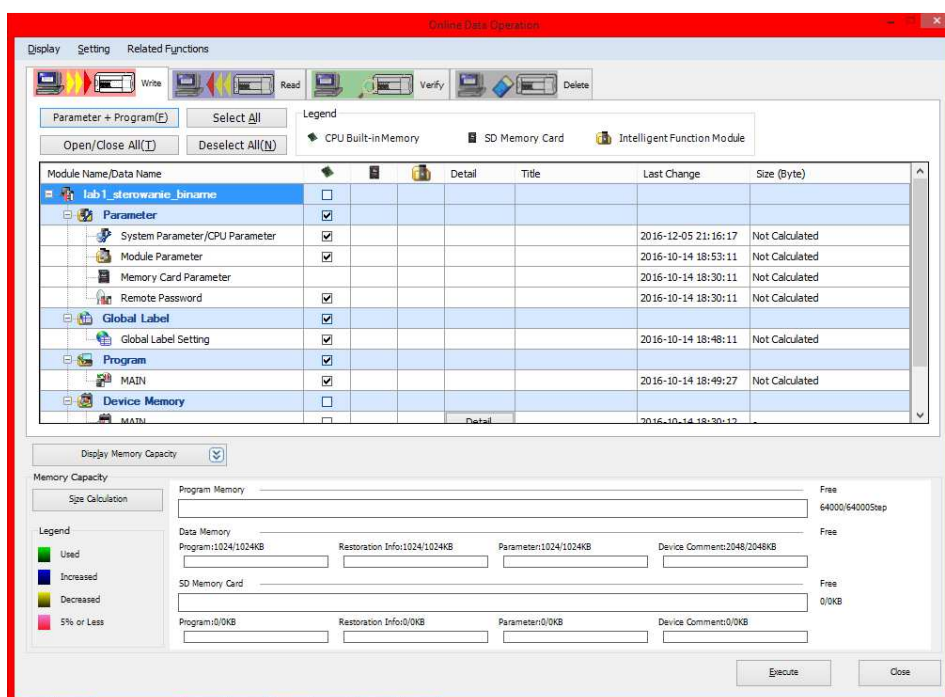
3.6 Programowanie sterownika

Zmiany konfiguracyjne wymagają pełnego ładowania sterownika z ręcznym restartem. W tym celu należy skompilować projekt i wybrać opcję „Write to PLC” (opcja 1 z poniższego rysunku). Do wprowadzenia szybkich zmian na sterowniku (np. modyfikacja logiki kontrolera) bez restartu kontrolera należy wykorzystać opcję „Online Program Change” (opcja 2 z poniższego rysunku). Operacja ta nie może być poprzedzona kompilacją, gdyż ta odbywa się automatycznie przed aktualizacją programu sterującego.



Rysunek 16 Operacja ładowania sterownika.

Wybór opcji „Write to PLC” przekierowuje do okna „Online Data Operation”, gdzie można przeprowadzić operacje: zapisu, odczytu, weryfikacji oraz czyszczenia pamięci kontrolera.



Rysunek 17 Okno Online Data Operation – Zapis/Odczyt/Veryfikacja/Czyszczenie sterownika.

Uwaga 1: Przed operacją ładowania kontrolera należy upewnić się że projekt nie zawiera błędów

Uwaga 2: Jeżeli przy próbie wgrywania programu do sterownika otrzymamy komunikat błędu „Inconsistency.....” należy wówczas przejść do zakładki Delete, wybrać wszystkie elementy przez Select All i wcisnąć Execute (nastąpi usunięcie starych parametrów i programów ze sterownika). Następnie należy powrócić do zakładki Write i przez Select All a następnie Execute wgnać nowy program i parametry do sterownika.

Uwaga 3: Po wykonaniu operacji wgrania parametrów i programu należy wykonać sprzętowy RESET sterownika PLC. Wykonuje się to przez otwarcie pokrywki po lewej stronie sterownika, przełączenie dźwigi z pozycji RUN do RESET, przytrzymanie dźwigi do momentu pojawienia się diody ERR na sterowniku a następnie powrót do

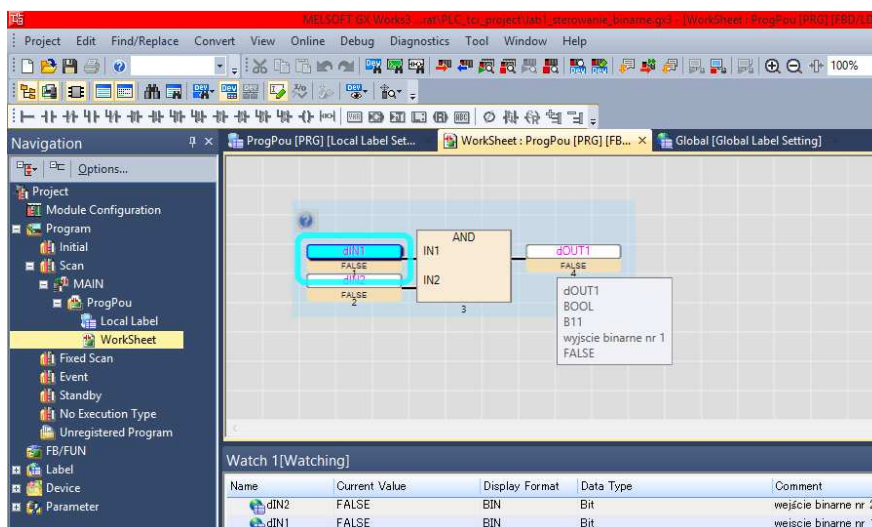
pozycji RUN. W tym momencie sterownik został zresetowany i można kontynuować pracę.

3.7 Diagnostyka, monitorowanie działania programu

Po załadowaniu kontrolera możliwy jest podgląd wykonywania programu za pomocą opcji „Start Monitoring”.

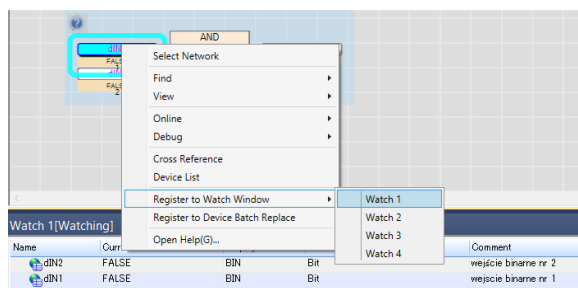


Rysunek 18 Uruchamianie Monitora.



Rysunek 19 Podgląd wykonywania programu

W celu zmiany/wyświetlania wartości zmiennych należy dodać je do podglądu za pomocą mechanizmu Watch'a. Należy najechać kursorem na nazwę zmiennej a następnie prawym przyciskiem myszy wybrać otworzyć menu i wybrać Register to Watch Window -> Watch 1. Z poziomu okienka Watch można zmieniać wartości zmiennych w celu testowania działania programu.



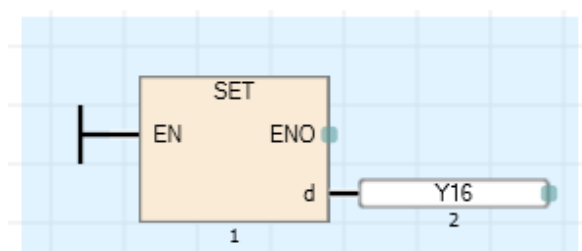
Rysunek 20 Uruchomienie Watch'a.

3.8 Pierwszy program PLC

Pierwszy program wgrywany do sterownika powinien obejmować:

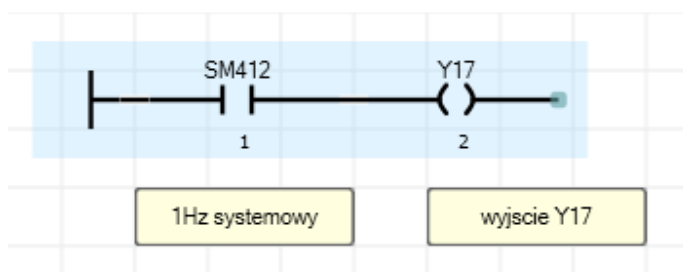
1. ustawienie adresu IP sterownika PLC na 192.168.127.250
2. konfigurację komunikacji dla MATLAB poprzez dodanie Unpassive TCP connection i ustawienie portu 4000
3. dodanie programu w sekcji INIT – inicjalizacja zmiennych dla symulacji procesów i regulatorów
4. dodanie programu w sekcji SCAN – operacje wykonywane cyklicznie ze skanem procesora
5. dodanie programu FIXED SCAN – operacje wykonywane cyklicznie ze skanem 1000ms (czas dyskretyzacji procesów regulacji i regulatorów)

Program przykładowy w sekcji INIT:



Po uruchomieniu sterownika lub jego resecie po wgraniu programu powinno aktywować się wyjście Y16, co można zaobserwować na zielonych diodach na sterowniku.

Program przykładowy w sekcji SCAN:



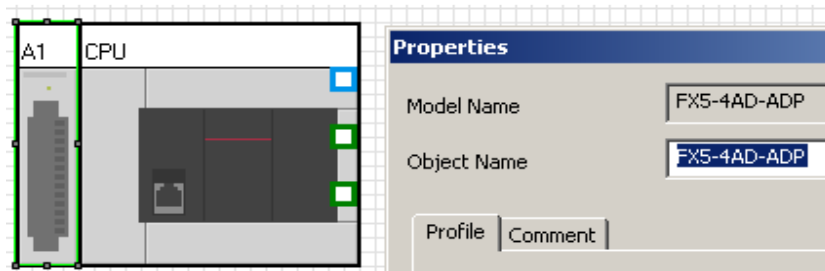
Program powinien mrugać wyjściem Y17 zgodnie z zegarem wewnętrznym 1Hz. Później ten program posłuży do stworzenia pierwszego powiązania panela operatora ze sterownikiem PLC.

Po utworzeniu programu wstępnego należy go skompilować opcją Rebuild All a następnie wgrać do sterownika. Po wykonaniu restartu sterownika wyjście Y16 powinno się zapalić a wyjście Y17 powinno cyklicznie się zmieniać.

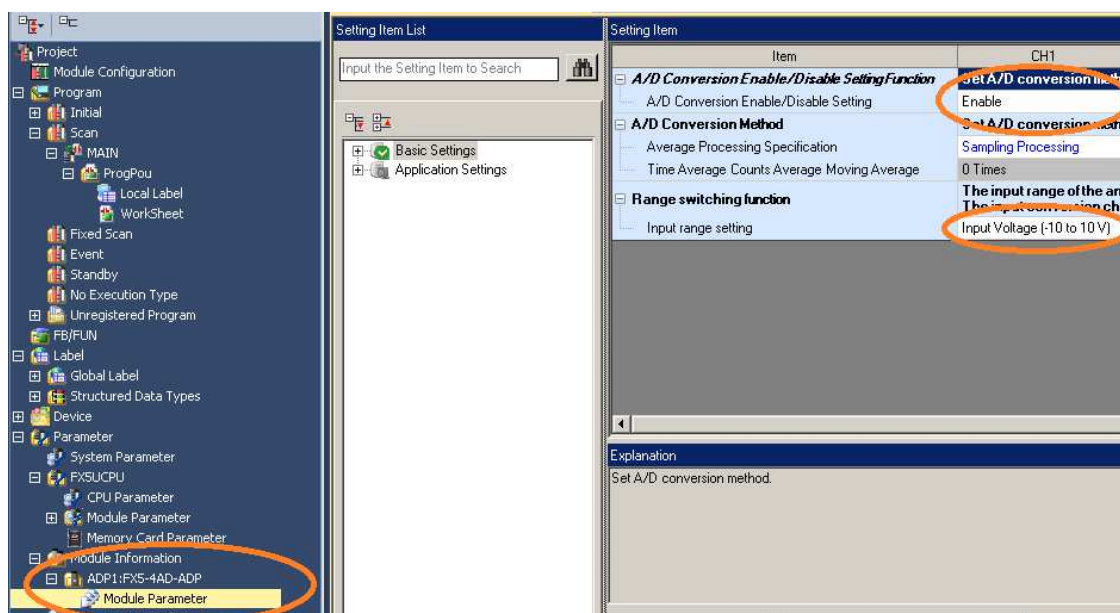
3.9 Przykładowe implementacje interfejsów sprzętowych

Wejścia analogowe

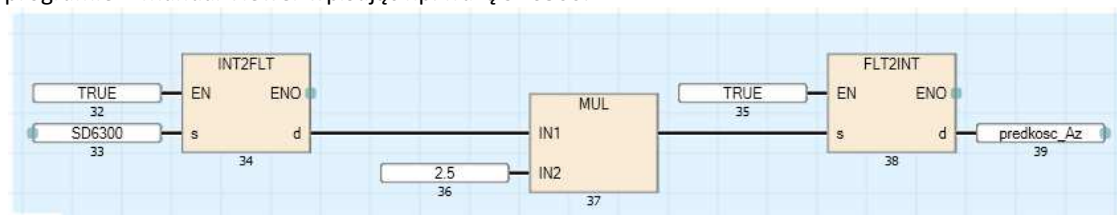
Wejście analogowe znajduje się na dodatkowym module dołączonym po lewej stronie głównego sterownika PLC. Konfigurację sprzętową ustawia się w zakładce Module Configuration.



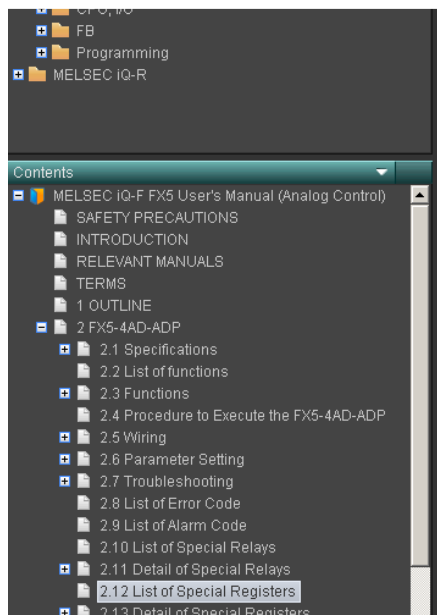
Następnie należy ustawić odpowiednie parametry danego wejścia. Przykładowa konfiguracja została przedstawiona poniżej. Należy włączyć odpowiedni kanał przetwarzania analogowo cyfrowego i prawidłowy zakres napięcia wejściowego -10V do +10V.



Przykład programu PLC do odczytu wartości cyfrowej z przetwornika C/A został przedstawiony poniżej. Rejestr specjalny sterownika PLC SD6300 jest przypisany do kanału pierwszego modułu pierwszego po lewej stronie sterownika PLC. Dokładniejsze opisy rejestrów można odszukać w dokumentacji lub w programie E-Manual Viewer wpisując np. frazę SD6300.



Wyciąg z programu E-Manual Viewer.



The special registers list for the 1st FX5-4AD-ADP module is shown below.

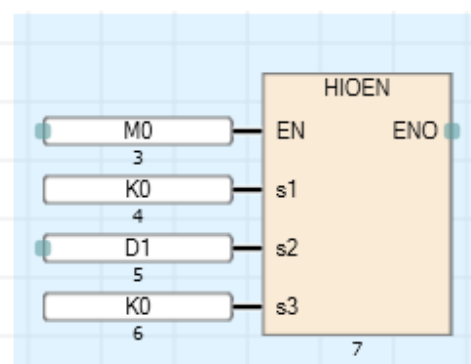
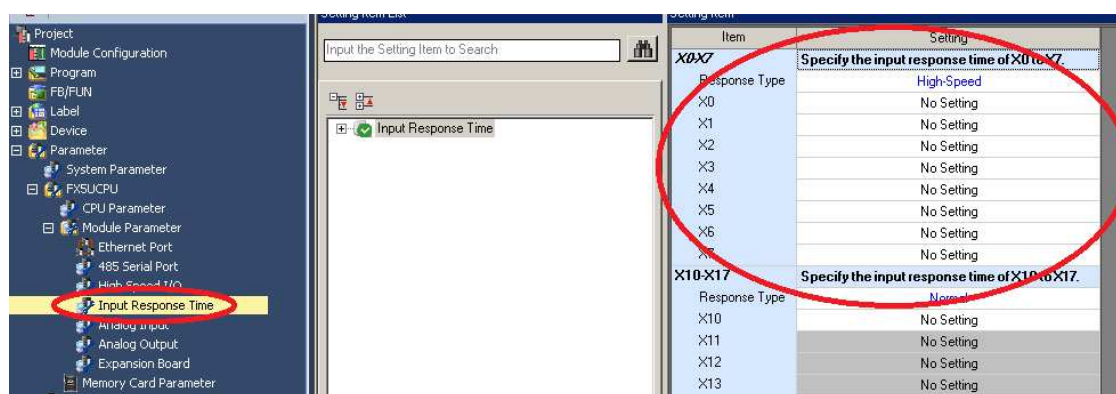
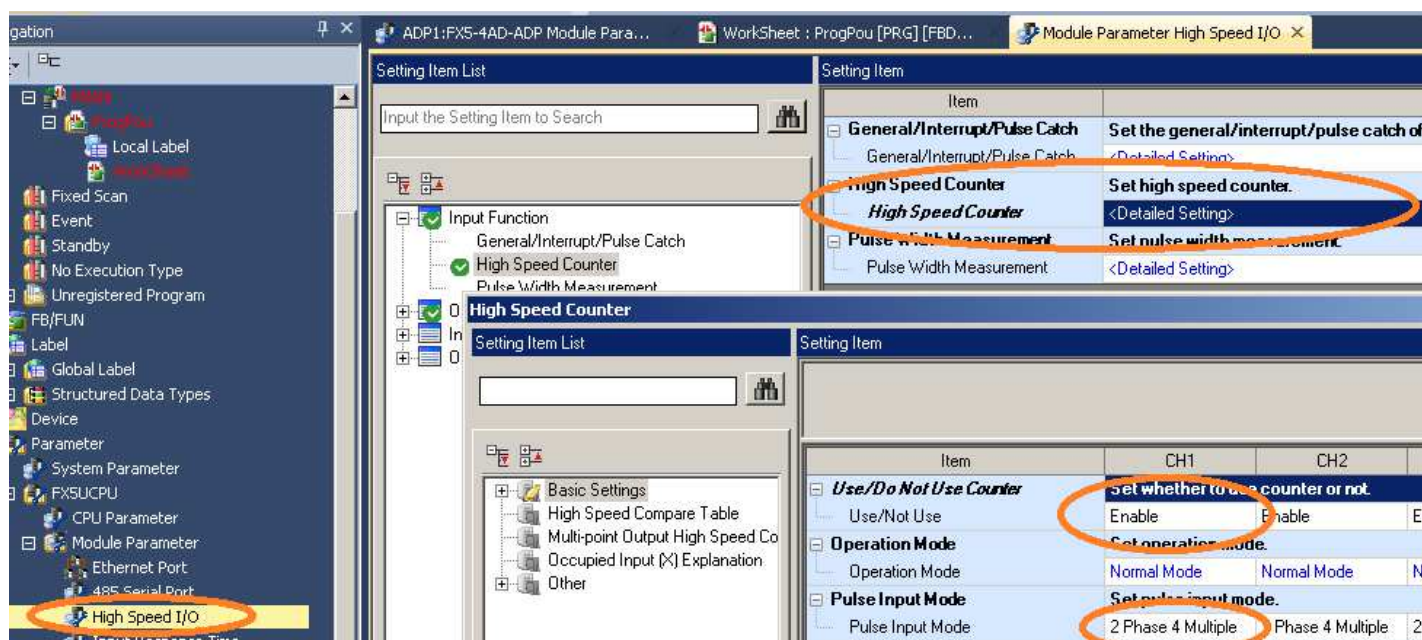
Special registers				Name	Reference
CH1	CH2	CH3	CH4		
SD6300	SD6340	SD6380	SD6420	Digital output value	Digital output value
SD6301	SD6341	SD6381	SD6421	Digital operation value	Digital operation value
SD6302	SD6342	SD6382	SD6422	Analog input value monitor	Analog input value monitor
SD6303	SD6343	SD6383	SD6423	Average processing specify	Average processing specify
SD6304	SD6344	SD6384	SD6424	Time Average/Count Average/Moving Average setting	Time Average/Count Average/Moving Average setting
SD6305	SD6345	SD6385	SD6425	Input range setting	Input range setting
SD6306	SD6346	SD6386	SD6426	Maximum value	Maximum value
SD6307	SD6347	SD6387	SD6427	Minimum value	Minimum value
SD6308	SD6348	SD6388	SD6428	Scaling upper limit value	Scaling upper limit value
SD6309	SD6349	SD6389	SD6429	Scaling lower limit value	Scaling lower limit value
SD6310	SD6350	SD6390	SD6430	Conversion value shift amount	Conversion value shift amount
SD6311	SD6351	SD6391	SD6431	Process alarm upper upper limit value	Process alarm upper upper limit value
SD6312	SD6352	SD6392	SD6432	Process alarm upper lower limit value	Process alarm upper lower limit value
SD6313	SD6353	SD6393	SD6433	Process alarm lower upper limit value	Process alarm lower upper limit value
SD6314	SD6354	SD6394	SD6434	Process alarm lower lower limit value	Process alarm lower lower limit value
SD6315	SD6355	SD6395	SD6435	Rate alarm upper limit value	Rate alarm upper limit value
SD6316	SD6356	SD6396	SD6436	Rate alarm lower limit value	Rate alarm lower limit value
SD6317	SD6357	SD6397	SD6437	Rate alarm warning detection period setting	Rate alarm warning detection period setting
SD6322	SD6362	SD6402	SD6442	Convergence detection upper limit value	Convergence detection upper limit value

Wejście licznikowe – pomiar pozycji z enkodera

Wejście licznikowe jest fizycznym wejściem cyfrowym sterownika. Aby wejście pracowało jako szybki licznik należy przeprowadzić konfigurację parametrów przedstawioną poniżej. Enkodery podłączone przy pomocy fazy A i B muszą zostać ustawione jako 2 phase. Jeżeli podłączona jest tylko jedna faza to wykorzystuje się opcję 1 phase.

Istotnym parametrem jest wyłączenie filtrów wejściowych, aby uzyskać prawidłowe pomiary.

Do aktywacji zliczania należy w programie PLC umieścić odpowiednio skonfigurowaną instrukcję HIOEN. Wejście EN podłączone zostało do bitu aktywującego M0. Argument s1 wybiera funkcję licznika szybkiego. Argument s2 wybiera bitowo, które kanały mają zostać uruchomiono: przykładowo, jeżeli chcemy użyć tylko kanału pierwszego to do rejestru D0 należy wpisać wartość 1; jeżeli chcemy użyć dwóch pierwszych kanałów to należy wpisać $2+1 = 3$. Argument s3 służy do deaktywacji kanałów licznika. Argument ten może pozostać z wartością zerową.

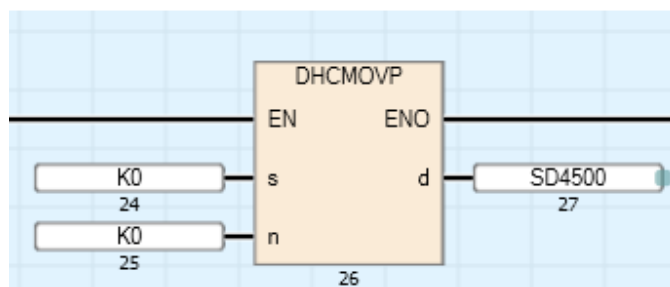


Aby odczytywać wartość aktualną licznika należy odszukać w dokumentacji, który rejestr specjalny jest powiązany z kanałem pierwszym – w naszym przypadku będzie to SD4500. Alternatywna wersja uruchomienia instrukcji HIOEN w języku ST:

HIOEN(TRUE, 0, 31, 0); //liczniki szybkie aktywacja

<div>Category</div> <ul style="list-style-type: none"> Engineering Software MELSEC IQ-F <ul style="list-style-type: none"> Analog input/output <ul style="list-style-type: none"> FX5U FX5UC CPU, I/O FB Programming MELSEC IQ-R <div>Contents</div> <ul style="list-style-type: none"> 16 DEVICE/LABEL ACCESS SERVICE PROC... 17 RAS FUNCTIONS 18 SECURITY FUNCTIONS 19 HIGH-SPEED INPUT/OUTPUT FUNCTION <ul style="list-style-type: none"> 19.1 High-speed Counter Function <ul style="list-style-type: none"> High-speed counter function overview High-speed counter function execution p... High-speed counter specifications Assignment for high-speed counters High-speed counter parameters High-speed counter (normal mode) 	SD4500	High-speed counter current value (CH1)	-2147483648 to +2147483647	0	R/W
	SD4501				
	SD4502	High-speed counter maximum value (CH1)	-2147483648 to +2147483647	-2147483648	R/W
	SD4503				
	SD4504	High-speed counter minimum value (CH1)	-2147483648 to +2147483647	2147483647	R/W
	SD4505				
	SD4506	High-speed counter pulse density (CH1)	0 to 2147483647	0	R/W
	SD4507				
	SD4508	High-speed counter rotational speed (CH1)	0 to 2147483647	0	R/W
	SD4509				
	SD4510	High-speed counter <u>preset</u> control switch (CH1)	0: Rising edge 1: Falling edge 2: Both edges 3: Constant when ON	Parameter set value	R/W
	SD4511	Not used	—	—	—
	SD4512	High-speed counter preset value (CH1)	-2147483648 to +2147483647	Parameter set value	R/W
	SD4513				
	SD4514	High-speed counter ring length (CH1)	2 to 2147483647	Parameter set value	R/W
	SD4515				
	SD4516	High-speed counter measurement unit time (CH1)	1 to 2147483647	Parameter set value	R/W
	SD4517				
	SD4518	High-speed counter number of pulses per rotation (CH1)	1 to 2147483647	Parameter set value	R/W
	SD4519				
	SD4520 to SD4529	Not used	—	—	—

Poniższy przykład programu pokazuje w jaki sposób można wykonać referencję (bazowanie) liczniki – procedura wpisuje zadaną wartość do rejestru licznika – w omawianym przypadku będzie to wpisanie wartości 0 do rejestru kanału 1 licznika SD4500.



Alternatywna forma zapisu w języku ST:

```
DHCMOVP (TRUE, 0, 0, SD4620);
```

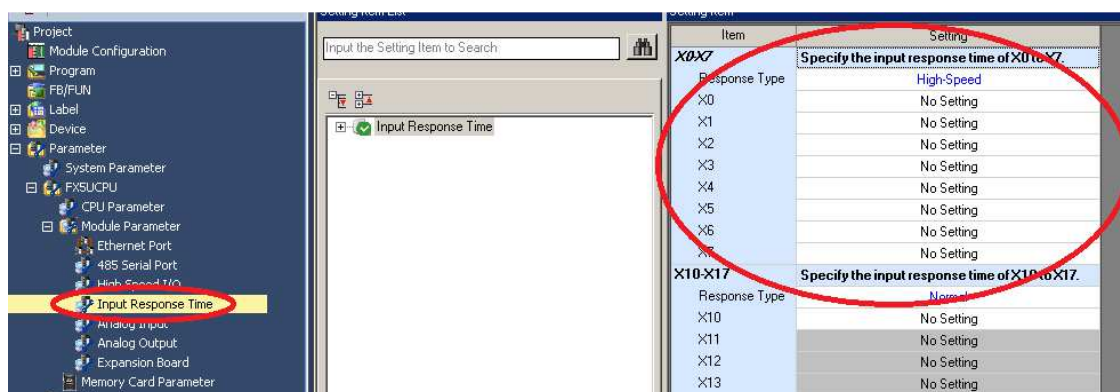
Wejście licznikowe – pomiar częstotliwości

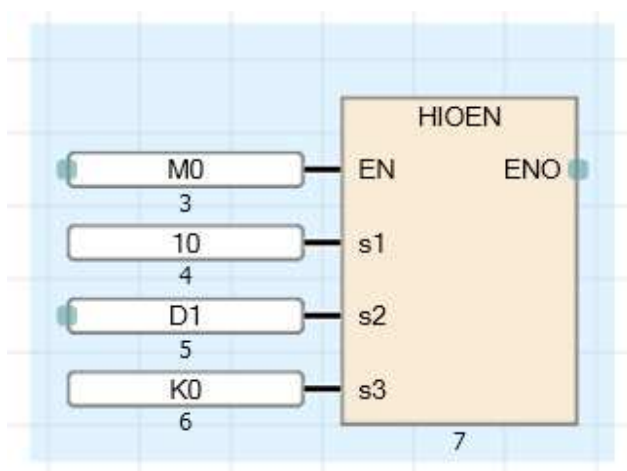
Konfiguracja wejścia licznikowe w trybie pomiaru częstotliwości jest analogiczne do powyższego ustawienia w trybie pomiaru licznikowego. Poniżej przedstawiono odpowiednie ustawienie parametrów. Istotnym parametrem jest wyłączenie filtrów wejściowych, aby uzyskać prawidłowe pomiary.

Item	CH1	CH2
Use/Do Not Use Counter	Set whether to use counter.	
Use/Not Use	Enable	Enable
Operation Mode	Set operation mode.	
Operation Mode	Pulse Density Measurement Mode	Pulse Density Measurem
Pulse Input Mode	Set pulse input mode.	
Pulse Input Mode	1-Phase 1 Input (S/W Up/Down Switch)	1-Phase 1 Input (S/W Up
Preset Input	Set preset input.	
Preset Input Enable/Disable	Disable	Disable
Input logic	Positive Logic	Positive Logic
Input Comparison Enable/Disable	Disable	Disable
Control Switch	Rising	Rising
Preset Value		
Preset Value	0	0
Enable Input	Set enable input.	
Enable Input Enable/Disable	Disable	Disable
Input logic	Positive Logic	Positive Logic
Ring Length Setting	Set ring length.	
Ring Length Enable/Disable	Disable	Disable
Ring Length		
Measurement Unit Time	Set the measurement unit time (ms) for the pulse density measurement mode and rotation speed measurement mode.	
Measurement Unit Time	100	100
No. of Pulse per Rotation	Set the number of pulses per rotation when using the rotation	
No. of Pulse per Rotation	1000	1000

Ważne jest ustawienie czasu pomiaru – to ustawienie jest związane z rozdzielczością pomiaru częstotliwości.

W programie sterownika należy umieścić instrukcję aktywacji pomiaru. Podobnie jak poprzednio wykorzystujemy funkcję HIOEN. Tym razem w argumencie s1 podajemy 10 – funkcja pomiaru częstotliwości.



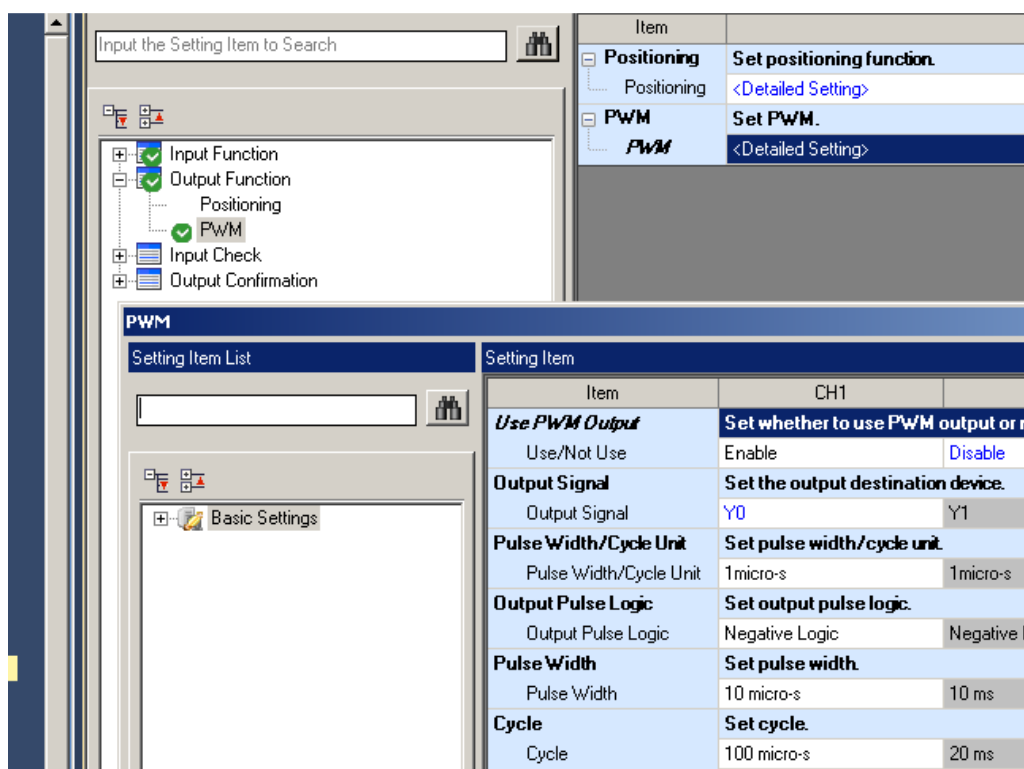


Wartość mierzonej częstotliwości będzie w rejestrze SD4506 (i SD4507 – należy uważać, które wartości są zwracane jako 16 bit czy 32 bit).

SD4506	High-speed counter pulse density (CH1)	0 to 2147483647	0	R/W
SD4507				
SD4508	High-speed counter rotational speed (CH1)	0 to 2147483647	0	R/W
SD4509				
SD4510	High-speed counter preset control switch (CH1)	0: Rising edge 1: Falling edge 2: Both edges 3: Constant when ON	Parameter set value	R/W

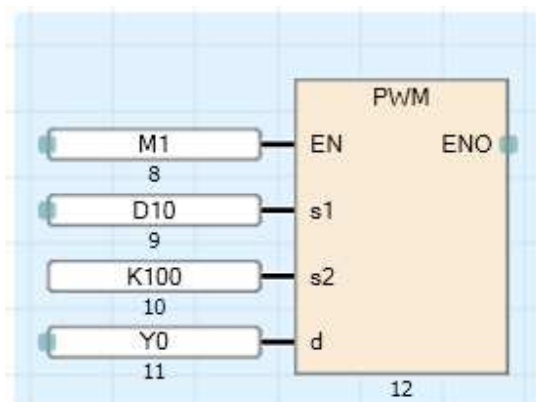
Wyjście PWM

Ustawienie wyjścia PWM zaczynamy od parametrów.



Powyższe ustawienie dotyczy wyjścia Y0. Okres sygnału PWM wynosi 100us – 10kHz. Przy takim ustawieniu możliwe wartości do sterowania wypełnieniem znajdują się od 1 do 100. Wpisanie wartości 0 do rejestru sterowania wypełnieniem wprowadzi sterownik w błąd. Istotne jest również ustawienie prawidłowej logiki wyjściowej, aby później zwiększanie wypełnienia powodowało np. zwiększenie prędkości obrotowej silnika – będzie to naturalne podejście do dalszej regulacji.

W programie PLC należy dodać instrukcję PWM. Wejście EN aktywuje wyjście PWM. Rejestr D0(arg. S1) odpowiada za wartość wypełnienia. Argument s2 odpowiada za okres sygnału PWM. Argument d powinien mieć przypisane właściwe wyjście sterownika.



Alternatywna forma zapisu w języku ST:

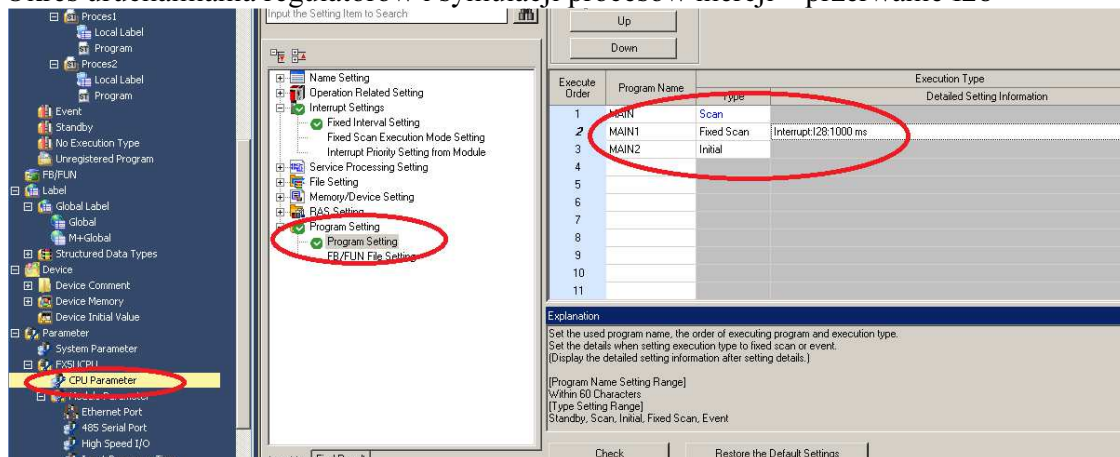
PWM(TRUE, Y_PWM, K200, Y2) ;

3.10 Pierwszy program PLC – regulacja ciągła

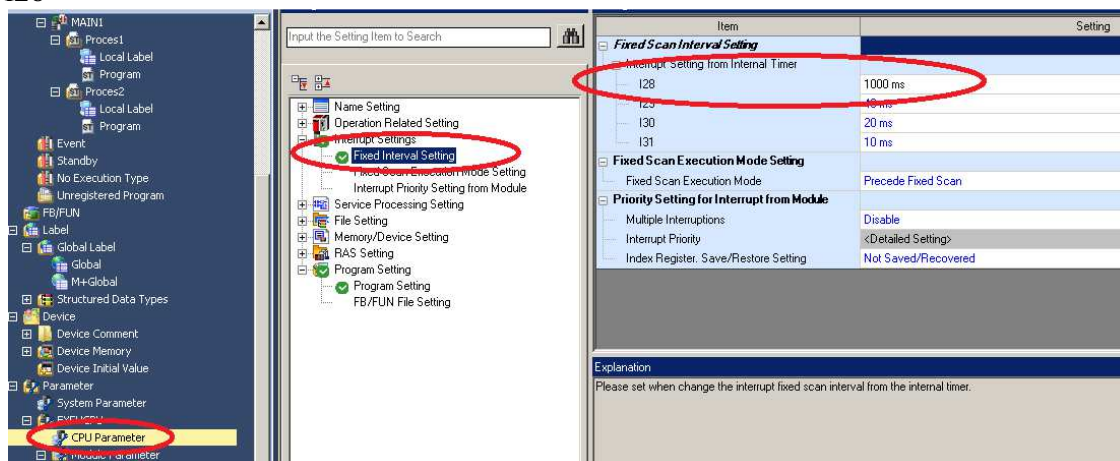
W tej części zostanie dokładnie omówiona część programowa sterownika PLC, która pozwoli na realizację regulatora (PID wbudowany, PID z równania różnicowego). Na laboratorium należy uzupełnić podany przykładowy program.

Poniżej przedstawiono najważniejsze punkty programów.

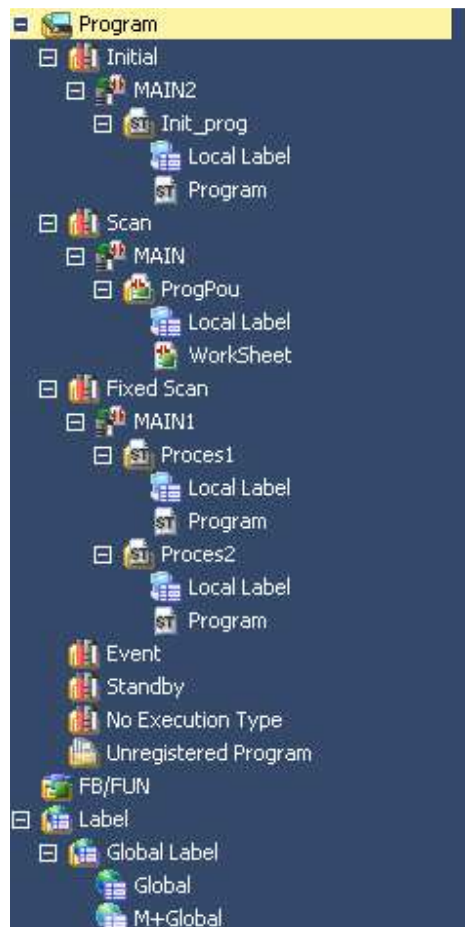
Okres uruchamiania regulatorów i symulacji procesów inercji – przerwanie I28



Okres uruchamiania regulatorów i symulacji procesów inercji – edycja okresu przerwania I28



Struktura przygotowanej części programów



Program Init_prog – kod źródłowy w języku ST

//Okres probkowania 1s=1000ms - parametr w programie FIXED SCAN
EMOV(TRUE, 1.0, okres_probkowania);

// Ustawienie wartosci początkowych procesu 1
EMOV(TRUE, 5.0, stala_czasowa1);
EMOV(TRUE, 10.0, K_p_proces1);

// Ustawienie wartosci początkowych procesu 2
EMOV(TRUE, 9.0, stala_czasowa2);
EMOV(TRUE, 3.0, K_p_proces2);

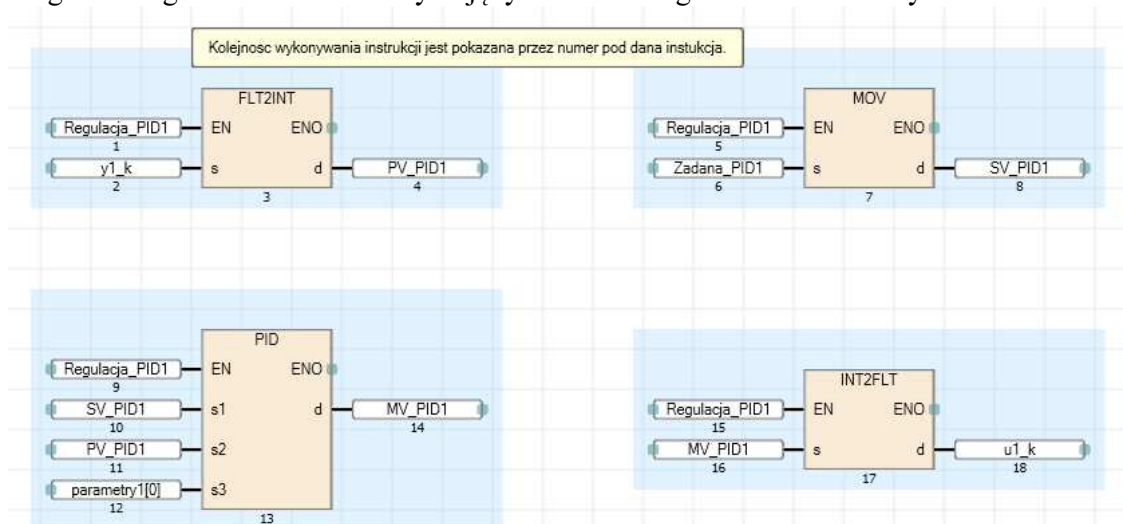
//Parametry regulatora wbudowanego PID
parametryl[0] := K1000; //okres regulacji w milisekundach
parametryl[3] := K1; //wzmocnienie regulatora P
parametryl[4] := K0; //TI = 0 oznacza nieskonczony czas calkowania - inaczej mowiac
calkowanie wylaczone
parametryl[5] := K0; //KD = 0 oznacza zerowe wzmocnienie rozniczkiowania
parametryl[6] := K0; //TD = 0 oznacza wylaczone rozniczkiowanie


```
parametry1[22] := 100; //gorny limit wartosci wyjsciowej z regulatora - zapobiega
rowniez efektowi wind-up
parametry1[23] := 0; //dolny limit wartosci wyjsciowej z regulatora - -||-
SET(TRUE, parametry1[1].5); //aktywacja limitow na wyjsciu regulatora
```

//Parametry regulatora z dyskretnego rownania roznicowego

```
K_PID3 := 1.0;
TI_PID3 := 99999.0;
TD_PID3 := 0.00001;
E0_PID3 := 0.0;
E1_PID3 := 0.0;
E2_PID3 := 0.0;
R0_PID3 := 0.0;
R1_PID3 := 0.0;
R2_PID3 := 0.0;
U_PID3 := 0.0;
Zadana_PID3 := 0;
```

Program ProgPou – kod źródłowy w języku FBD – regulator wbudowany PID



Definicje zmiennych globalnych – deklaracje parametrów regulatorów, procesów, zmiennych pomocniczych – należy pamiętać o prawidłowym przydzielaniu fizycznych rejestrów, aby później było możliwe odczytywanie zmiennych na panelu operatora.

	Label Name	Data Type	Class	Assign (Device/Label)
1	SV_PID1	Word [Signed]	VAR_GLOBAL	D2000
2	PV_PID1	Word [Signed]	VAR_GLOBAL	D2001
3	MV_PID1	Word [Signed]	VAR_GLOBAL	D2002
4	parametry1	Word [Unsigned]/Bit String (16-bit)(0..29)	VAR_GLOBAL	D2010
5	Regulacja_PID1	Bit	VAR_GLOBAL	M0
6	Zadana_PID1	Word [Signed]	VAR_GLOBAL	D2050
7				
8				
9	okres_probkowania	FLOAT [Single Precision]	VAR_GLOBAL	D1000
10				
11	stala_czasowa1	FLOAT [Single Precision]	VAR_GLOBAL	D1002
12	Alfa1	FLOAT [Single Precision]	VAR_GLOBAL	D1004
13	K_p_proces1	FLOAT [Single Precision]	VAR_GLOBAL	D1006
14	u1_k	FLOAT [Single Precision]	VAR_GLOBAL	D1008
15	u1_k_1	FLOAT [Single Precision]	VAR_GLOBAL	D1010
16	y1_k	FLOAT [Single Precision]	VAR_GLOBAL	D1012
17	y1_k_1	FLOAT [Single Precision]	VAR_GLOBAL	D1014
18	A_p1	FLOAT [Single Precision]	VAR_GLOBAL	D1016
19	stala_czasowa2	FLOAT [Single Precision]	VAR_GLOBAL	D1022
20	Alfa2	FLOAT [Single Precision]	VAR_GLOBAL	D1024
21	K_p_proces2	FLOAT [Single Precision]	VAR_GLOBAL	D1026
22	u2_k	FLOAT [Single Precision]	VAR_GLOBAL	D1028
23	u2_k_1	FLOAT [Single Precision]	VAR_GLOBAL	D1030
24	y2_k	FLOAT [Single Precision]	VAR_GLOBAL	D1032
25	y2_k_1	FLOAT [Single Precision]	VAR_GLOBAL	D1034
26	A_p2	FLOAT [Single Precision]	VAR_GLOBAL	D1036
27				

Program PID3 – kod źródłowy w języku ST – regulator PID opisany równaniem różnicowym – **do dokończenia**

//Regulator PID na podstawie rownania roznicowego

```
SV_PID3 := Zadana_PID3;
PV_PID3 := REAL_TO_INT(y3_k);
```

//Wylczenie parametrow

```
R0_PID3 := 1.0; //r0 = K*( 1+(Tp/(2*Ti))+Td/Tp );
R1_PID3 := 1.0; //r1 = K*( (Tp/(2*Ti))-(2*Td/Tp)-1 );
R2_PID3 := 1.0; //K*Td/Tp;
```

//Wylczenie uchybu regulacji i przesuniecie historii

```
E2_PID3 := E1_PID3;
E1_PID3 := E0_PID3;
E0_PID3 := SV_PID3 - PV_PID3;
```

//Obliczenie sterowania

```
U_PID3 := R2_PID3*E2_PID3 + R1_PID3*E1_PID3 + R0_PID3*E0_PID3 + U_PID3;
//u = R2*E2 + R1*E1 + R0*E0 + u;
```

```
IF (U_PID3 > 100.0) THEN
    U_PID3 := 100.0;
END_IF;
```

```
IF (U_PID3 < 0.0) THEN
    U_PID3 := 0.0;
END_IF;
```

Alternatywnie łatwiejsza implementacja regulatora PID w języku ST:

Program w grupie SCAN:

```
PID(Reguluj, SV_PID, PV_PID, parametry, MV_PID);
```

```
MOV(Reguluj, MV_PID, D114); //wysterowanie grzałki 1
D110 := 400;                //wysterowanie wentylatora 1
```

```
MOV(Reguluj, D100, PV_PID);
```

Definicje zmiennych globalnych:

	Label Name	Data Type		Class	Assign (Device/Label)
1	SV_PID	Word [Signed]	...	VAR_GLOBAL	D4500
2	PV_PID	Word [Signed]	...	VAR_GLOBAL	D4501
3	parametry	Word [Unsigned]/Bit String [16-bit]	...	VAR_GLOBAL	D4000
4	MV_PID	Word [Signed]	...	VAR_GLOBAL	D4502
5	Reguluj	Bit	...	VAR_GLOBAL	
6			...		

Inicjalizacja regulatora:

```
//Ustawienia PID
D4000 := 100; //okres regulacji 100ms
D4022 := 1000; //limit gorny wartosci wyjsciowej
D4023 := 0;    //limit dolny wartosci wyjsciowej
SET(TRUE, D4001.5); //aktywacja limitow wyjsciowych – od razu antiwindup
D4003 := 10; //wzmocnienie regulatora
D4004 := 20; //stala czasowa calkowania regulatora
SET(TRUE, D4001.0); //aktywacja trybu grzania – znak petli sprzezenia
zwrotnego
```

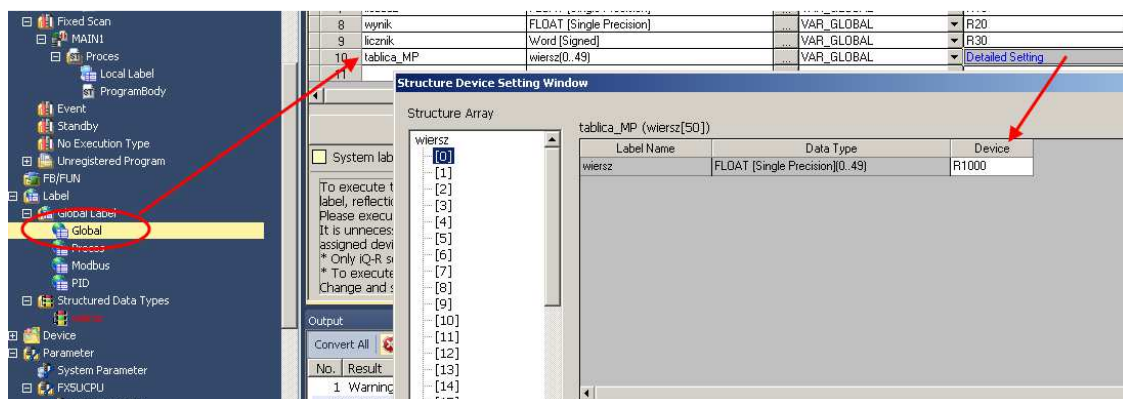
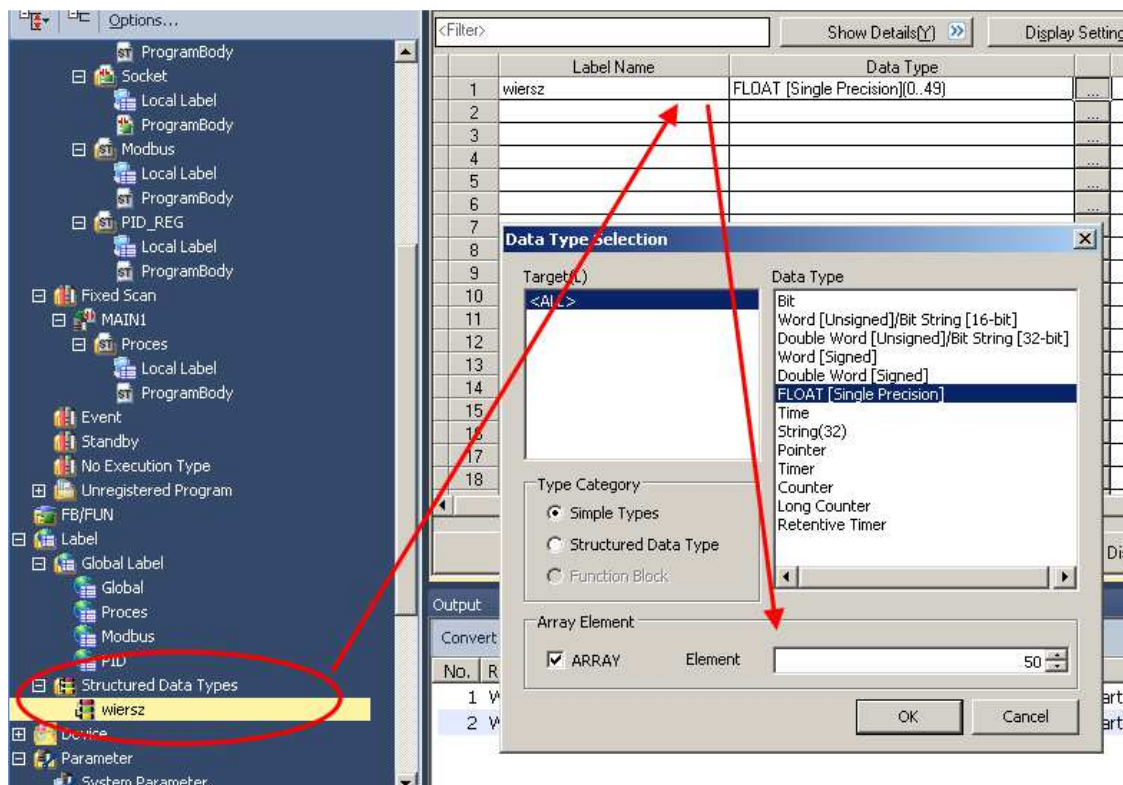
4 Definicja tablicy typu float

W pierwszej kolejności definiujemy strukturę, w której dodajemy jeden element typu wektor float o zadanej długości. Następnie dodajemy zmienną w wybranej grupie Labeli. Zmienna będzie typu stworzonej przed chwilą struktury. Zmienna powinna być zadeklarowana, jako wektor, dzięki czemu uzyskamy wektor wektorów – czyli tablicę dwuwymiarową. Przykład zapisu stałej wartości do tablicy:

```
tablica_MP[0].wiersz[0] := 1.321;
```

Użycie tablicy w obliczeniach:

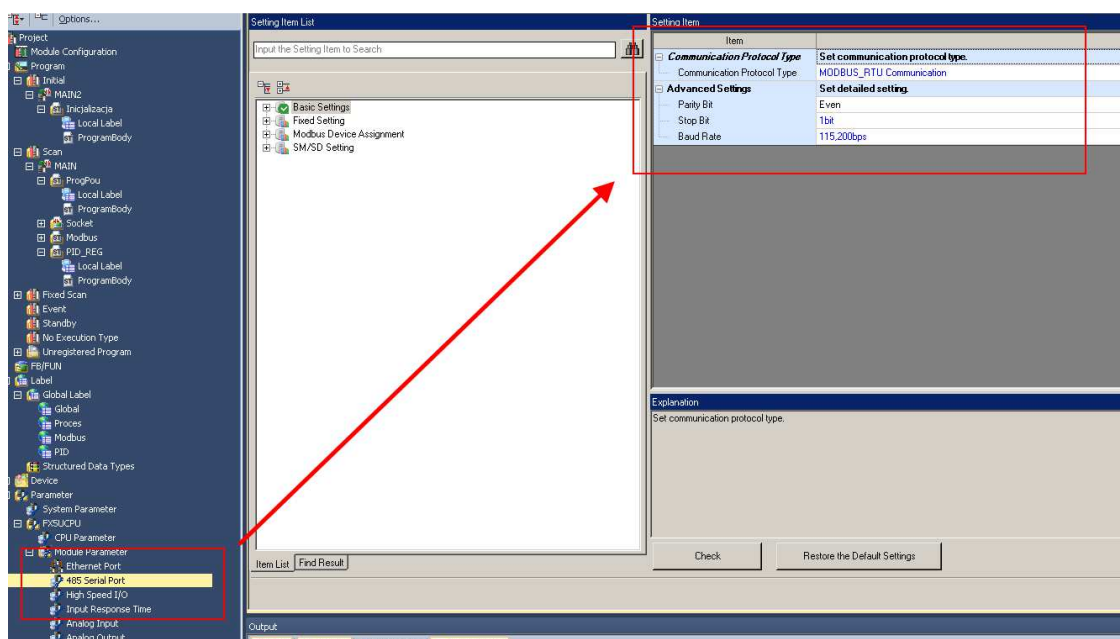
```
FOR licznik := 0 TO 49 BY 1 DO
    wynik := liczba1 * liczba2;
    wynik := tablica_MP[licznik].wiersz[licznik] * INT_TO_REAL(licznik);
END_FOR;
```



5 Opis komunikacji RS485 MODBUS, Socket Communication

MODBUS:

Parametry komunikacji



Deklaracja zmiennych

<Filter>		Easy Display	Display Setting	Check		
	Label Name	Data Type		Class	Assign (Dev	
1	Slave_adres	Word [Signed]	...	VAR_GLOBAL		
2	Function_code	Word [Signed]	...	VAR_GLOBAL		
3	Modbus_adres	Word [Signed]	...	VAR_GLOBAL		
4	Device_count	Word [Signed]	...	VAR_GLOBAL		
5	Pomiar_MODBUS	Bit	...	VAR_GLOBAL		
6	Zapis_MODBUS	Bit	...	VAR_GLOBAL		
7	Pozwolenie_pomiar_MODBUS	Bit	...	VAR_GLOBAL		
8	Pozwolenie_zapis_MODBUS	Bit	...	VAR_GLOBAL		
9			...			

Inicjalizacja

```
//Inicjalizacja MODBUS
```

```
Pomiar_MODBUS := 0;
```

```
Zapis_MODBUS := 0;
```

```
MOV(TRUE, K11, Slave_adres);
```

```
MOV(TRUE, K4, Function_code); //4-pomiar, 3-sterowanie
```

```
MOV(TRUE, K0, Modbus_adres); //zaczynamy liczyc od 0
```

```
MOV(TRUE, K7, Device_count); //7 pomiarow, 6 sterowan
```

```
//Ustawienie poczatkowe wyjsc procesu
```

```
ZRST(TRUE, D110, D120);
```

Komunikacja

```

SET(Pozwolenie_pomiar_MODBUS AND LDP(TRUE, SM413), Pomiar_MODBUS);
IF(Pomiar_MODBUS) THEN
    Function_code := 4;
    Device_count := 7;
    ADPRW( Pomiar_MODBUS AND NOT Zapis_MODBUS, Slave_adres , Function_code ,
Modbus_adres, Device_count , D100, M100);
    IF(M101) THEN
        RST(TRUE, Pomiar_MODBUS);
        RST(TRUE, M101);
        RST(TRUE, M100);
    END_IF;
END_IF;

```

```

SET(Pozwolenie_zapis_MODBUS AND LDF(TRUE, SM413), Zapis_MODBUS);
IF(Zapis_MODBUS) THEN
    Function_code := 16;
    Device_count := 6;
    ADPRW( Zapis_MODBUS AND NOT Pomiar_MODBUS, Slave_adres , Function_code ,
Modbus_adres, Device_count , D110, M110);
    IF(M111) THEN
        RST(TRUE, Zapis_MODBUS);
        RST(TRUE, M111);
        RST(TRUE, M110);
    END_IF;
END_IF;

```

```

IF Zapis_MODBUS AND Pomiar_MODBUS THEN
    RST(TRUE, Zapis_MODBUS);
    RST(TRUE, Pomiar_MODBUS);
END_IF;

```

Socket Communication:

Parametry komunikacji -> patrz rozdział 3.5

Deklaracja zmiennych

Lokalne

		Label Name	Data Type	
1		wyslałem	Bit	...
2		nie_wyslałem	Bit	...
3		zmienna_int	Word [Signed]	...
4		auto_send	Bit	...
5				...

Globalne

		Label Name	Data Type	Class	Assign (Device)
1		send_string	String(32)	VAR_GLOBAL	D301
2		temp_string	String(32)	VAR_GLOBAL	D2020
3		tekst_temp	String(32)	VAR_GLOBAL	
4		dlugosc_tekstu	Word [Signed]	VAR_GLOBAL	
5					

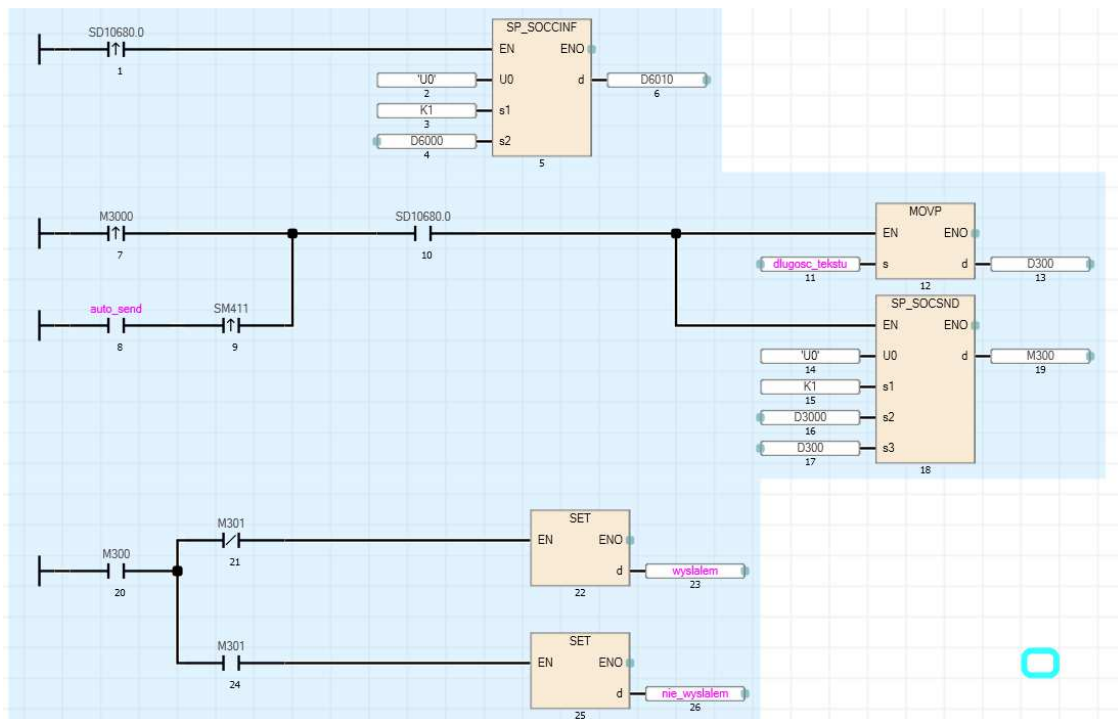
Przygotowanie ramki

```
//Generacja tekstu do wyslania przez socket communication
tekst_temp := 'U=';
tekst_temp := CONCAT(tekst_temp, INT_TO_STRING(REAL_TO_INT(u_k)));
tekst_temp := CONCAT(tekst_temp, ';Y=');
tekst_temp := CONCAT(tekst_temp, INT_TO_STRING(REAL_TO_INT(y_k)));
tekst_temp := CONCAT(tekst_temp, ';$L');

send_string := tekst_temp;

//Dlugosc tekstu
dlugosc_tekstu := LEN(send_string);
```

Komunikacja



Skrypt MATLAB 2017

```
delete(instrfindall)
```

```
pause(2);
```

```
close all;  
clear all;
```

```
t = tcpip('192.168.127.250',4000, 'NetworkRole', 'client');
```

```
t.OutputBufferSize = 3000;  
t.InputBufferSize = 3000;
```

```
fopen(t);  
fprintf('Fopen zadzialal');  
iterator = 1;  
data = zeros(2,2);  
figure(1);  
while(1)  
    if (t.BytesAvailable ~= 0)  
        temp = fscanf(t);  
        temp  
        eval(temp);  
        data(1,iterator) = U;  
        data(2,iterator) = Y;  
        fprintf('Fscanf zadzialal');  
        iterator=iterator + 1;
```

```

        plot(1:length(data(2,:)), data(2,:));
        hold on;
        grid on;
        plot(1:length(data(1,:)), data(1,:));
        hold off;
    end
    pause(0.05);
end

fclose(t);
delete(t);
clear t;

```

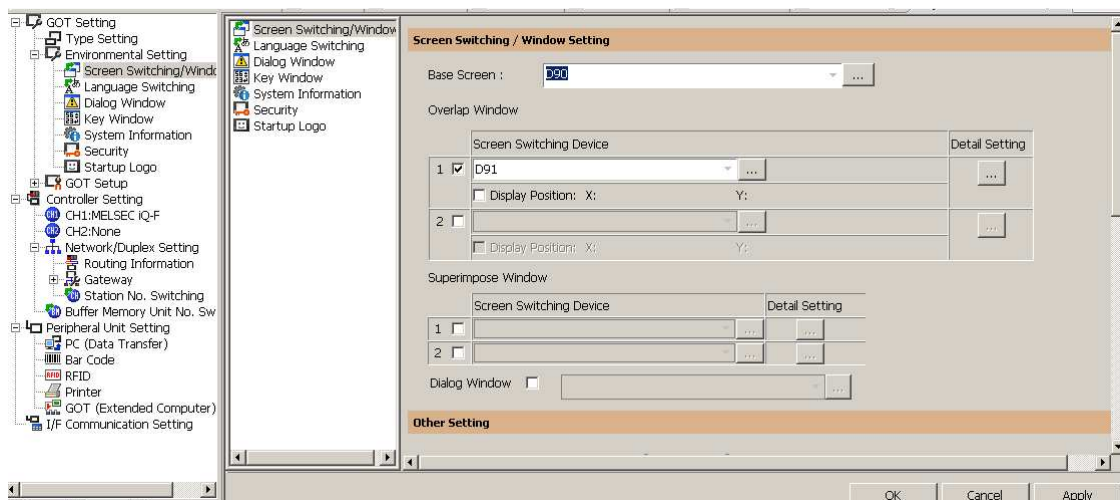
6 Tworzenie grafik operatorskich w środowisku GT Designer 3

6.1 Projekt demo

W ramach tego ćwiczenia studenci otrzymują przykładowy projekt na panel operatorski GOT Simple. Na bazie tego projektu należy przygotować aplikację zgodnie z opisem podanym w instrukcji do ćwiczenia. Projekt zawiera przykładowe ekrany z podstawowymi operacjami dostępnymi na panelach operatora. Należy zapoznać się z funkcjami i wykorzystać je w dalszej pracy do realizacji finalnej aplikacji. W następnych rozdziałach opisane zostaną poszczególne panele. Finalny projekt powinien opierać się na zaproponowanej strukturze. Jednak ocena końcowa będzie silnie zależała od wprowadzonych modyfikacji w panelu i użyciu nowatorskich pomysłów.

6.2 Panel MENU – 1

Po uruchomieniu panelu operatora lub po wgraniu projektu pierwszy ekran, który się zgłosi przedstawiono na poniższym rysunku. Po prawej stronie znajdują się przyciski do przechodzenia między innymi ekranami. W górnej części mamy pola godziny i daty pobieranej ze sterownika PLC. Po lewej stronie znajduje się lampka podłączona do bitu PLC SM412 (1Hz). Zgodnie z komentarzem na zielono wymagany jest adres sterownika PLC 192.168.127.250. W środkowej części można zobaczyć pole wyświetlające stały ciąg znaków w zależności od rejestru panela operatora GD1000 – rejestr ten jest ustawiany w polu wejściowym numerycznym znajdującym się na prawo od wyświetlanego tekstu. Na samym dole znajduje się przycisk, który pozwala na wyświetlenie okienka typu „popup”. Wyświetlenie tego okienka odbywa się przez wpisanie odpowiedniej wartości – numeru ekranu – do rejestru sterownika PLC. Połączenie między rejestrem sterownika a opisywaną funkcją znajduje się na kolejnym rysunku. Zgodnie z rysunkiem rejestr ekranów głównych to D90, rejestr ekranów typu „popup” to D91. W rejestrze D90 zawsze znajduje się numer aktualnie wyświetlanego ekranu głównego. Możliwa jest też jego zmiana z poziomu PLC. Natomiast wpisana wartość do rejestru D91 powoduje wyświetlenie odpowiedniego numeru ekranu „popup”. Jeżeli w rejestrze D91 znajduje się wartość 0 to żaden ekran „popup” nie jest wyświetlany.



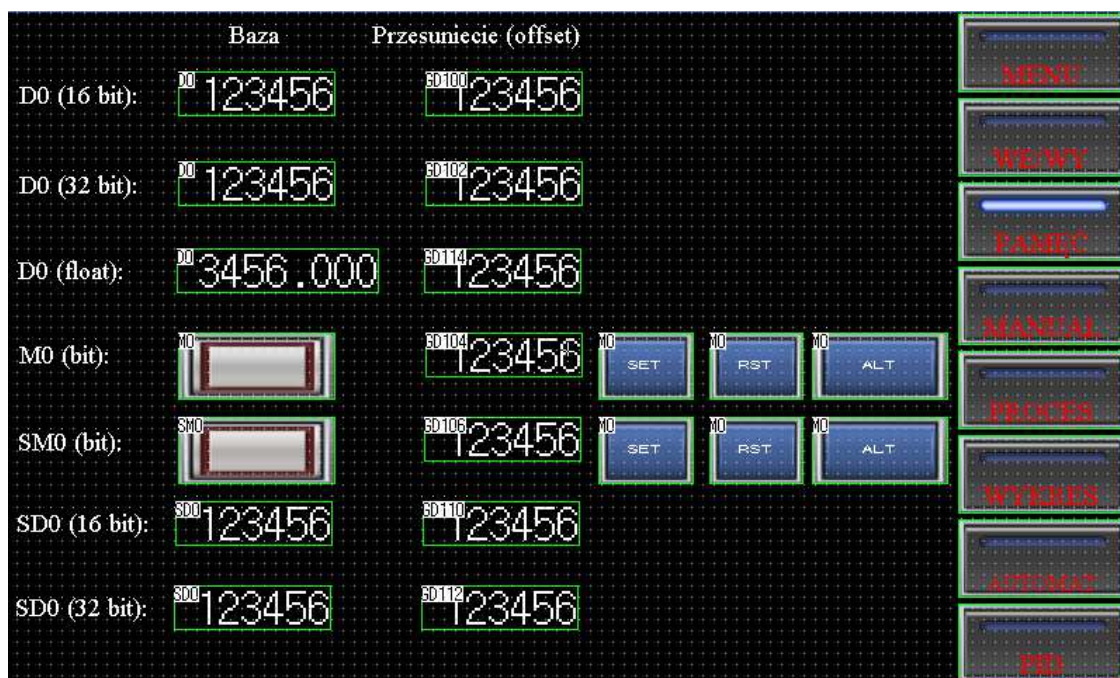
6.3 Panel WEWY – 2

Na panelu drugim można obserwować aktualny stan wejść i wyjść sterownika. W przypadku wejść są to tylko lampki sygnalizujące stan. W przypadku wyjść są to przyciski, które można wcisnąć i zmieniać tym samym stan wyjścia w trybie „alternate” – zawsze zmiana na stan przeciwny.



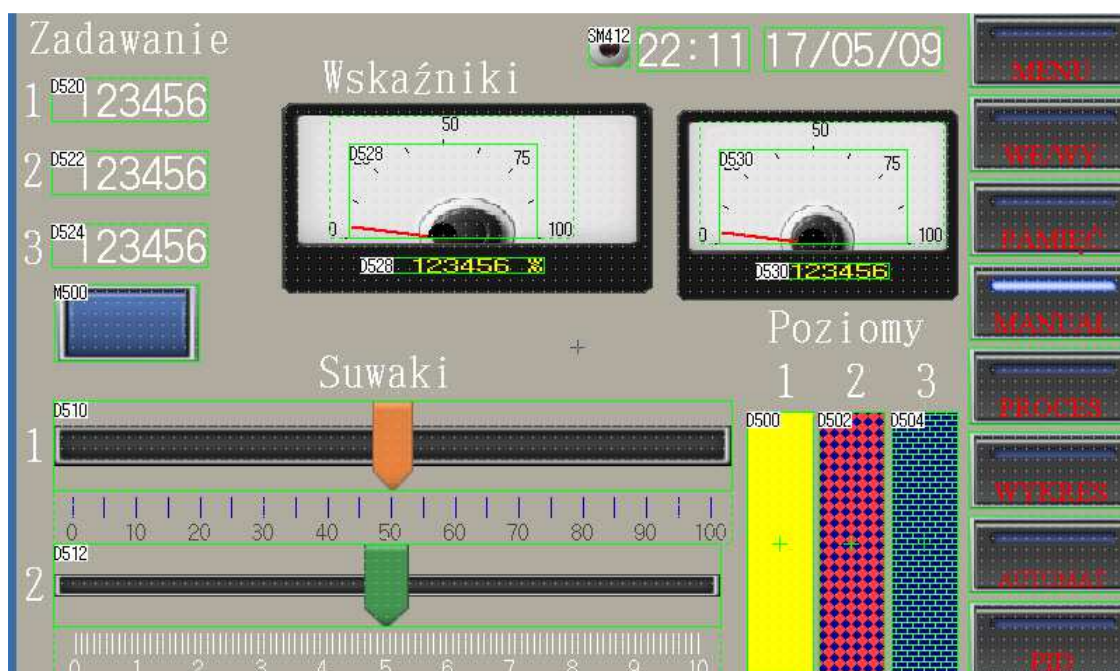
6.4 Panel PAMIEC – 3

W panelu trzecim można obserwować i modyfikować pamięć sterownika PLC. W kolumnie baza widoczna jest wartość rejestru. Obserwowany rejestr jest zawsze określony jako np. D(0 + offset). Offset jest rodzajem przesunięcia numeru rejestru – wskaźnik w tablicy – w rozważanym przypadku do offsetu użyto rejestrów panela operatora zaczynając od GD100. W przypadku bitów pamięci jak np. M0 możliwe operacje do wykonania to SET, RESET i ALTERNATE.



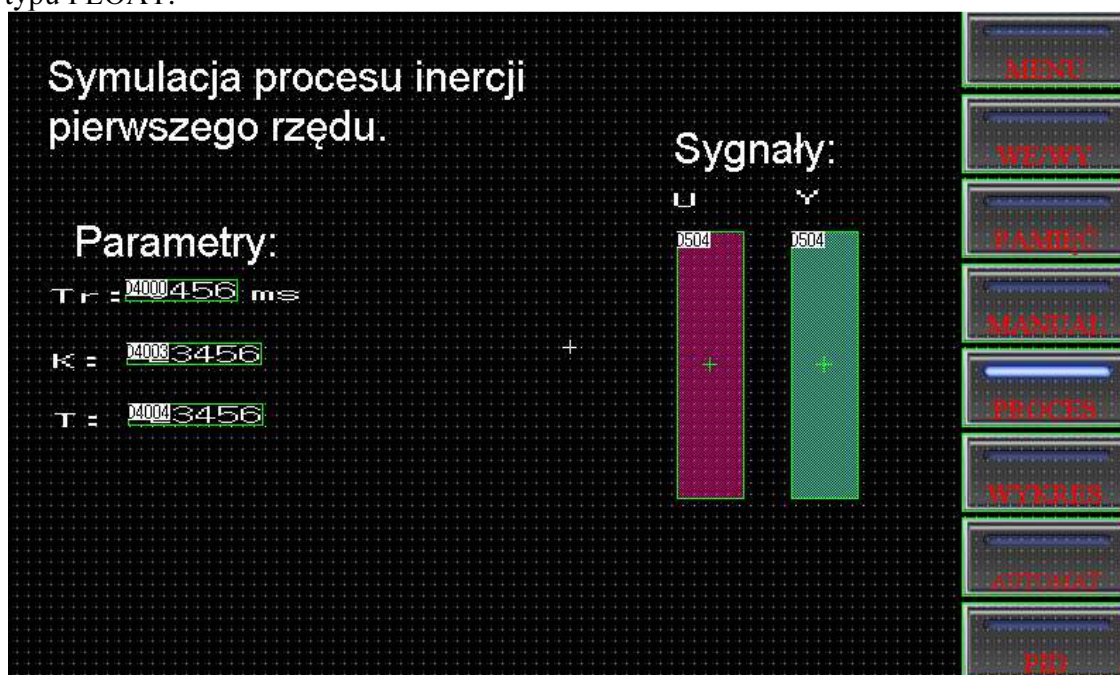
6.5 Panel MANUAL – 4

Na czwartym panelu znajdują się pola wpisu wartości, wskaźniki wychyłowe, poziomy oraz suwaki do zadawania wartości. Wszystkie elementy pobierają lub zapisują dane bezpośrednio w rejestrach typu D w PLC. Dla przypomnienia rejestry te są domyślnie 16 bitowe. Należy zwrócić szczególną uwagę przy ustawieniu parametrów danego elementu na panelu odnośnie liczby bitów zmiennej. Jeżeli oczywiście w PLC używamy zmiennej 32 bitowej należy tak samo to ustawić w parametrach elementu na panelu.



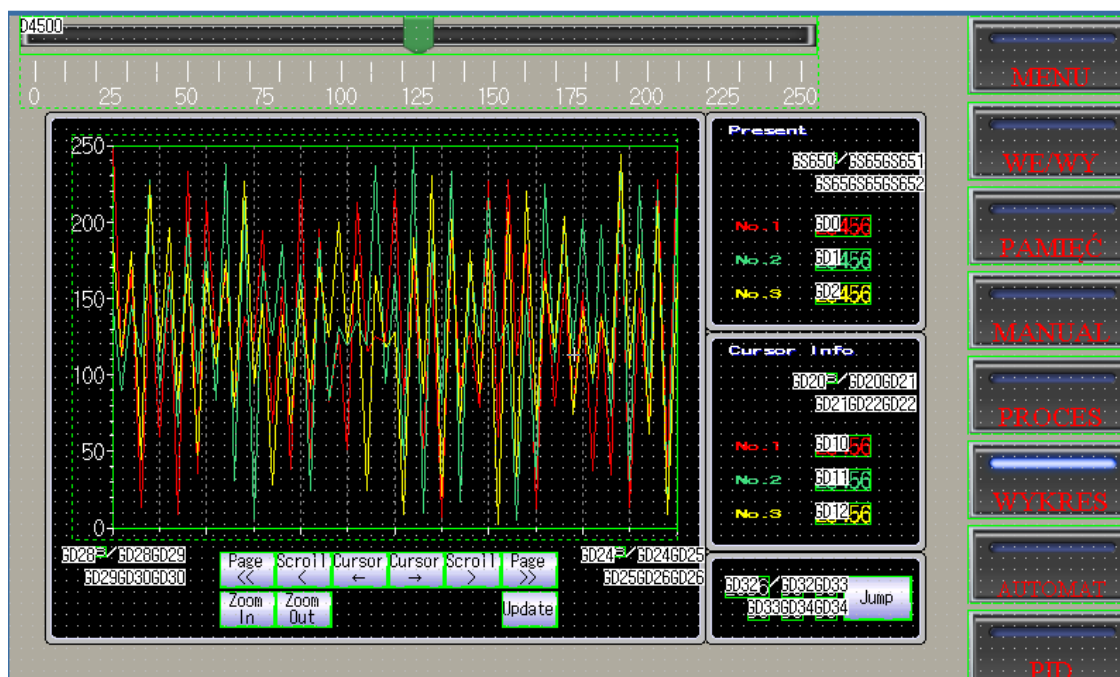
6.6 Panel PROCES – 5

Na panelu piątym przedstawiono parametry symulowanego procesu inercji pierwszego rzędu oraz aktualne wartości sygnału wejściowego U i wyjściowego Y. Symulacja procesu może zostać zrealizowana w sterowniku PLC przy użyciu języka ST i zmiennych typu FLOAT.



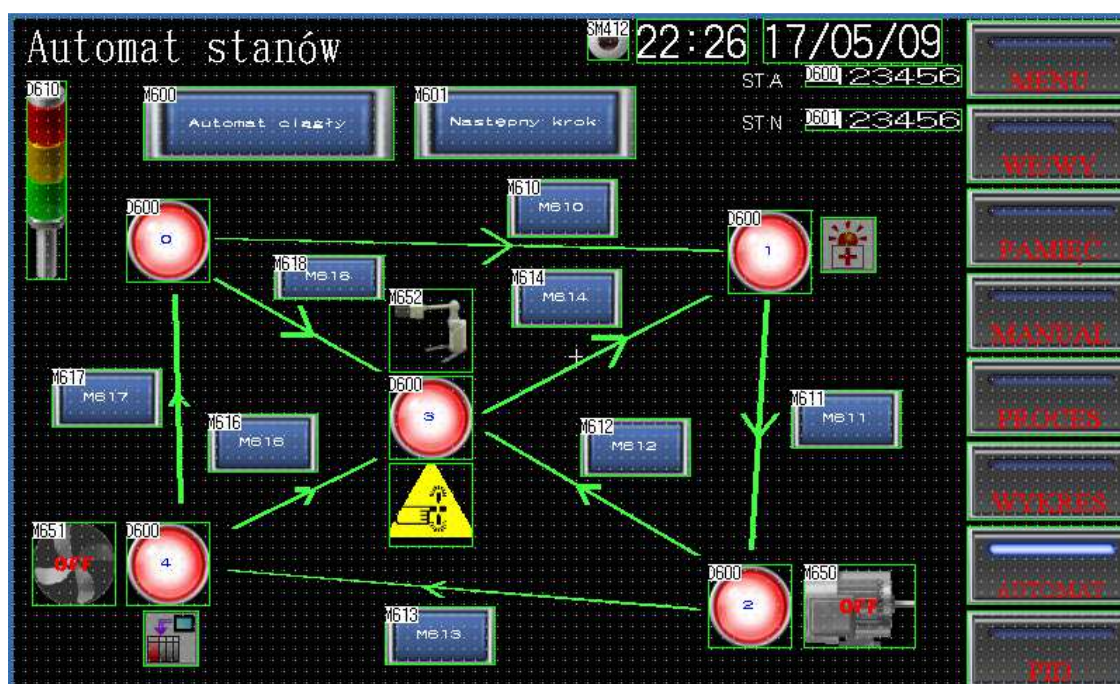
6.7 Panel WYKRES – 6

Na szóstym panelu przedstawiono możliwości rysowania wykresów. Rozwiązanie przedstawia rysowanie trzech pisaków o różnych kolorach. Możliwe jest obserwowanie aktualnej wartości, przejścia do historii, wstrzymanie i wznowienie rysowania. Na samej górze zamieszczono suwak do zmiany na przykład wartości zadanej.



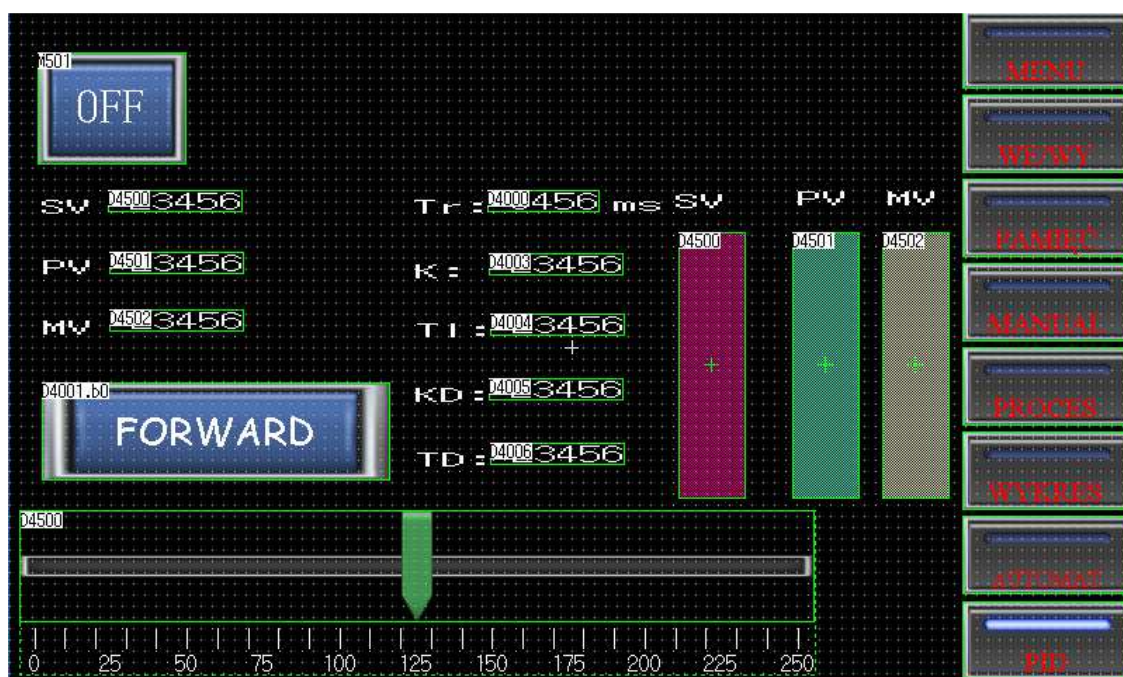
6.8 Panel AUTOMAT – 7

Na panelu siódmym przedstawiono graficzną reprezentację automatu stanów zrealizowane w języku ST w sterowniku PLC. Automat ten może zostać użyty do opracowania głównego automatu stanów, który będzie między innymi mógł wybierać odpowiedni algorytm pracy, przechodzić między pracą auto i ręczną.

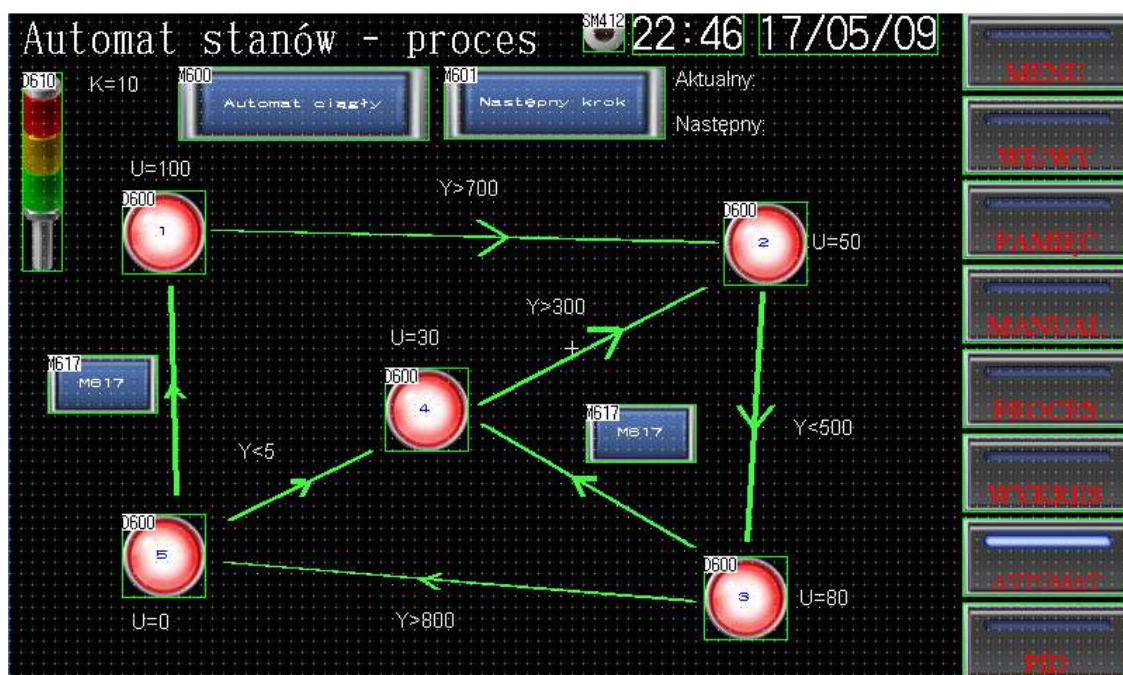


6.9 Panel AUTOMAT P – 8

Na panelu ósmym przedstawiono parametry oraz odpowiednie wartości wejściowe/wyjściowe dla regulatora PID. Znajduję się tutaj wartości zadana SV, mierzona procesu PV, sterowanie MV. Wartości wyświetlają się w postaci numerycznej oraz w postaci wykresów słupkowych. Wartość zadaną SV można zmieniać przy pomocy suwaka. W środkowej części ekranu znajdują się podstawowe parametry regulatora. Aby załączyć regulator należy wcisnąć przycisk OFF na samej górze, jest on połączony z bitem w PLC M501. Ten z kolei powinien być odpowiednio zaprogramowany, aby uruchomić blok PID w sterowniku PLC.



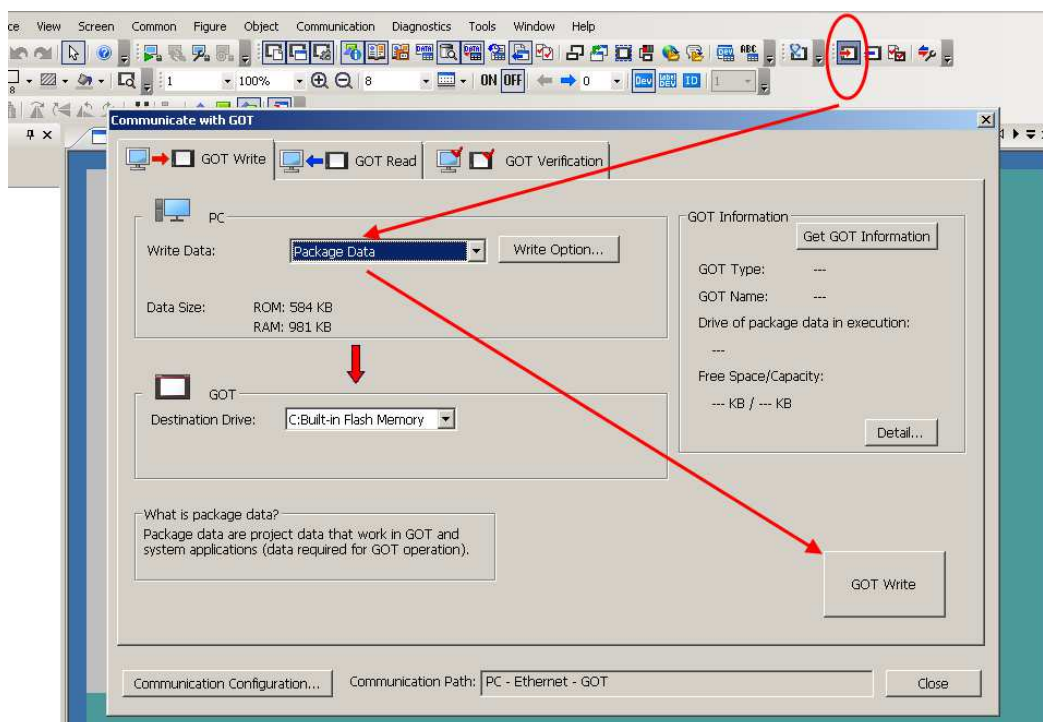
6.10 Panel AUTOMAT P – 10



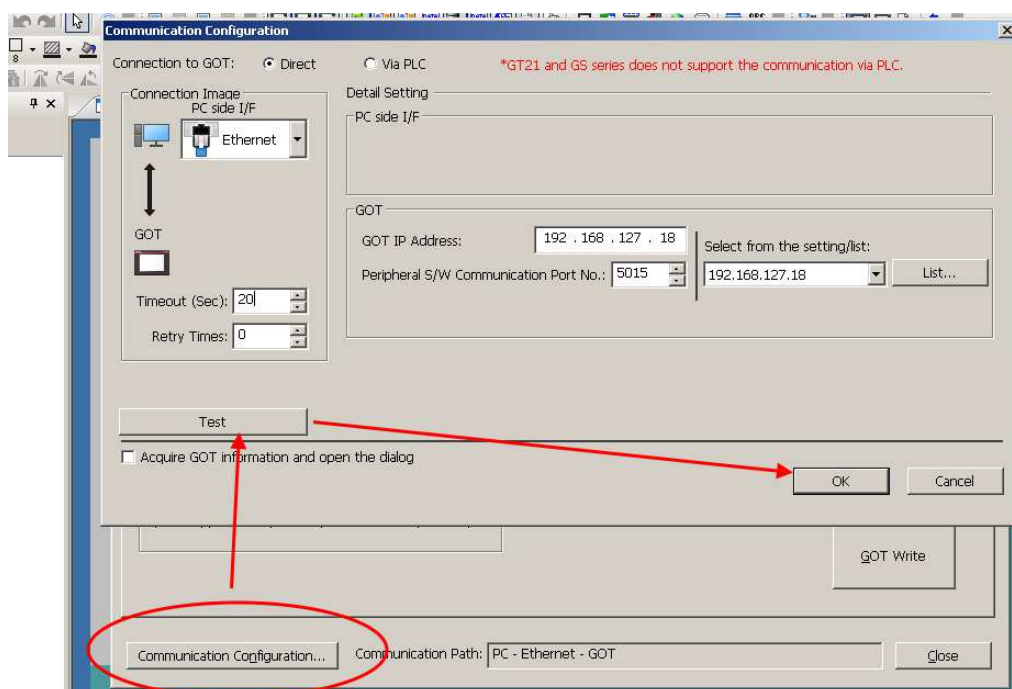
Na panelu dziesiątym przedstawiono graficzną reprezentację automatu stanów zrealizowane w języku ST w sterowniku PLC. Automat ten powinien zostać użyty do zmiany wartości zadanych w algorytmach regulacji w celu całkowitej automatyzacji pomiarów. Automat ten powinien rozpoczynać działanie tylko wtedy, gdy wybrany jest tryb pracy automatyczny.

6.11 Wgrywanie projektu do panela operatora

Przy edycji projektu na panel operatora nie jest wymagana żadna kompilacja. Po wprowadzeniu zmian w projekcie można od razu wgrać zmiany na panel. Operację wgrywania wykonuje się jak przedstawiono na rysunku poniżej.



Gdy procedura wgrywania jest niemożliwa z uwagi na brak komunikacji należy sprawdzić to zgodnie ze schematem z poniższego rysunku.



Gdy komunikacja jest niemożliwa i test nie wykonuje się poprawnie należy spróbować wykonać ping w kierunku adresu panela z poziomu konsoli cmd w Windows.

7 Dokumentacja

Dokumentacja do pobrania z internetu

<http://app.mitsubishielectric.com/app/fa/download/search.do?kisyu=/plcf&mode=manual>

GT Designer 3: Help >> GT Designer 3 Help >> E-Manual Viewer

GX Works 3: Help >> GX Works 3 Help >> E-Manual Viewer

8 Projekt

UWAGA: URUCHOMIENIE STANOWISK ROZPOCZYNAMY OD SPRAWDZENIA POŁĄCZENIA MIĘDZY STANOWISKIEM (PUDEŁKO Z ELEKTRONIKĄ FIRMY INTECO) A STEROWNIKIEM PLC (ZESTAW PLC Z PŁYTĄ KONWERTUJĄCĄ SYGNAŁY) – SZARE TAŚMY KOMPUTEROWE. NASTĘPNIE WCISKAMY CZERWONY PRZYCIŚK „GRZYB” AWARYJNY. WŁĄCZAMY ZASILANIE ZESTAWU PLC, NASTĘPNIE WŁĄCZAMY ZASILANIE STANOWISKA (PRZELĄCZNIK Z TYŁU OBUDOWY PUDEŁKA Z ELEKTRONIKĄ). JEŻELI PLC JEST WŁĄCZONE MOŻNA WYCIĄGNĄĆ PRZYCIŚK „GRZYB” AWARYJNY I WCISNĄĆ NA STANOWISKU CZERWONY PRZYCIŚK WŁĄCZAJĄCY ZASILANIE W STANOWISKU. OD TEGO MOMENTU NALEŻY ZWRÓCIĆ SZCZEGÓLNA UWAGĘ NA PRACĘ W LABORATORIUM. NALEŻY ZACHOWAĆ OSTROŻNOŚĆ, ABY NIE USZKODZIĆ ZESTAWÓW A NAJWAŻNIEJSZE NIE ZROBIĆ KOMUŚ LUB SOBIE KRZYWDY.

NA KAŻDYM ZESTAWIE PLC ZAINSTALOWANY JEST PANEL OPERATORSKI, KTÓRY POZWALA OBSERWOWAĆ STAN WEJŚĆ A TAKŻE STEROWAĆ WYJŚCIA. WYJŚCIA STEROWANE SĄ W TRYBIE PRZELĄCZANIA (TOGGLE, ALTERNATE).

Cel projektu:

Celem projektu jest zaprojektowanie oraz implementacja układu regulacji automatycznej dla wybranego stanowiska laboratoryjnego z wykorzystaniem systemu Mitsubishi PLC + panel GOT.

Etap 1:

Wymagania ogólne:

1. Kompletny projekt dla sterownika przemysłowego PLC:
 - a. Konfiguracja modułów peryferyjnych t.j:
 - i. Moduł komunikacyjny Ethernet,
 - ii. Moduł analogowy,
 - iii. Moduł szybkich liczników HIOEN.
 - b. Właściwy podział na zmienne lokalne i globalne wraz z określeniem adresacji pamięci sterownika
 - c. Obsługa pomiarów z obiektu – konwersja odczytów do wielkości fizycznych (skalowanie – instrukcja SCL)
 - d. Obsługa sterowań (np. PWM)
 - e. Obsługa dodatkowych sygnałów (ciągłych, binarnych), umożliwiających sterowanie obiektem.
 - f. Realizacja sterowania w otwartej/zamkniętej pętli regulacji
 - i. Określenie prawidłowego zakresu wartości zadanej
 - ii. Wstępne dobranie nastaw regulatorów, metodą inżynierską
 - g. Implementacja zabezpieczeń zapewniających bezpieczną pracę obiektu:
 - i. Obsługa krańcówek (jeśli występują – szczególnie dźwig)
 - ii. Bezpieczna szybkość poruszania się elementów ruchomych
 - h. Implementacja możliwości pracy w trybie ręcznym i automatycznym

Uwaga. Nie wymaga się wyboru jednego języka programowania, można wręcz wykorzystać zalety każdego z nich zależnie od potrzeb. Zalecany jest jednak język FBD/LD oraz ST.

Każde stanowisko wyposażone jest w wyłącznik bezpieczeństwa, z którego należy korzystać zawsze, gdy istnieje ryzyko uszkodzenia sprzętu lub istnieje zagrożenie zdrowia osoby znajdującej się w przestrzeni roboczej obiektu.

Wymagania dla poszczególnych stanowisk:

1. TRAS



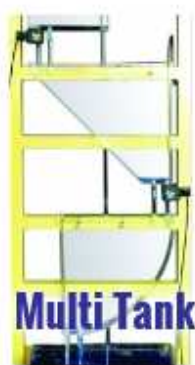
- Realizacja sterowania w zamkniętej pętli umożliwiającą utrzymywanie zadanego kąta obrotu i poziomu helikoptera
(Zadajnik wartości zadanej – programowy (PLC, GOT))

2. SERVO



- Realizacja sterowania w zamkniętej pętli
(Zadajnik wartości zadanej – pokrętło ręczne);
tryb pozycjonowania; tryb prędkościowy ->
użycie pomiaru prędkości z tachoprądnicy,
użycie pomiaru pozycji z enkodera

3. MULTI TANK



- Sterowanie poziomem cieczy w trzech zbiornikach
(Zadajnik wartości zadanej – programowy (PLC, GOT))

4. TOWER CRANE



GOT)

- Ograniczenie sterowania PWM
 - Obsługa krańcówek,
 - Obrót, przesunięcie wózka, wysokość
- bloczka –
w zamkniętej pętli
- Procedura bazowania,
 - Procedura centrowania,
 - Sterowanie z kompensacją
- (Zadajniki wartości zadanej – programowe (PLC,

Etap 2:

1. Kompletną wizualizację procesu na panelu operatora GOT:
 - a. Wizualizacja obiektu
 - b. Panel operatorski do sterowania ręcznego/automatycznego
 - c. Panel z nastawami regulatorów
 - d. Panel z wykreślaniem wartości zmiennych określających jakość regulacji: SP, PV, MV.

Etap 3:

Sprawozdanie dokumentujące stan aplikacji po ukończeniu ostatnich zajęć projektowych.

UWAGA: Opis wejść/wyjść do poszczególnych obiektów został podany w pliku Excel.