



Projektowanie Układów Sterowania

Laboratorium

Instrukcja – sterownik PLC, panel operatora, INTECO

SPIS TREŚCI

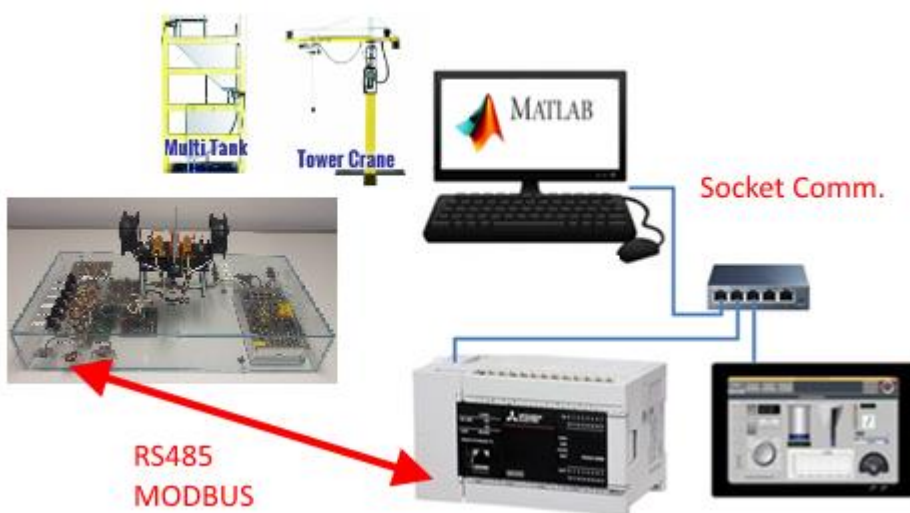
1	CEL ĆWICZENIA	3
2	WYKAZ SPRZĘTU ORAZ OPROGRAMOWANIA	3
3	TWORZENIE KODU STERUJĄCEGO STEROWNIKA PLC W ŚRODOWISKU GXWORKS3	4
3.1	TWORZENIE NOWEGO PROJEKTU	17
3.2	ELEMENTY JĘZYKA FBD/LD	18
3.3	DEFINICJA ZMIENNYCH	21
3.4	TWORZENIE KODU STERUJĄCEGO.....	23
3.5	KONFIGURACJA STEROWNIKA.....	26
3.6	PROGRAMOWANIE STEROWNIKA.....	28
3.7	DIAGNOSTYKA, MONITOROWANIE DZIAŁANIA PROGRAMU	29
3.8	PIERWSZY PROGRAM PLC.....	30
3.9	PRZYKŁADOWE IMPLEMENTACJE INTERFEJSÓW SPRZĘTOWYCH.....	33
3.10	PIERWSZY PROGRAM PLC – REGULACJA CIĄGŁA	45
4	DEFINICJA TABLICY TYPU FLOAT	60
5	OPIS KOMUNIKACJI RS485 MODBUS, SOCKET COMMUNICATION.....	61
6	TWORZENIE GRAFIK OPERATORSKICH W ŚRODOWISKU GT DESIGNER 3.....	66
6.1	PROJEKT DEMO	66
6.2	PANEL MENU – 1	66
6.3	PANEL WEWY – 2.....	67
6.4	PANEL PAMIĘC – 3.....	68
6.5	PANEL MANUAL – 4	69
6.6	PANEL PROCES – 5.....	70
6.7	PANEL WYKRES – 6.....	71
6.8	PANEL AUTOMAT – 7.....	71
6.9	PANEL AUTOMAT P – 8.....	72
6.10	PANEL AUTOMAT P – 10.....	73
6.11	WGRYWANIE PROJEKTU DO PANELA OPERATORA.....	74
7	DOKUMENTACJA.....	75
8	PROJEKT	75

1 Cel ćwiczenia

Zapoznanie się ze sposobem tworzenia oprogramowania dla sterownika programowalnego FX5U firmy Mitsubishi oraz wizualizacji procesu na panelu operatora HMI typu GOT Simple. W ramach ćwiczenia studenci tworzą projekt sterownika w środowisku GxWorks 3 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 Infrastruktura sprzętowa w laboratorium

3.1 Obiekt termiczny

UWAGA: Przed przystąpieniem do pracy ze stanowiskiem konieczne jest odbycie szkolenia oraz zapoznanie się z treścią podrozdziału 3.1.1

3.1.1 Bezpieczeństwo i wymagania

Korzystając ze stanowiska należy pamiętać, co następuje:

- W urządzeniu występuje napięcie sieciowe 120/230V.
- W celu wyłączenia stanowiska należy odłączyć wtyczkę zasilającą 120/230V od sieci.
- Zastosowany wyłącznik główny jest dwubiegunowy, rozłącza zarówno linie L jak i N.
- Urządzenie przeznaczone jest do użytku wewnątrz pomieszczeń. Nie może pracować w warunkach wystąpienia kondensacji pary wodnej.
- Urządzenie wykonano w Klasie I ochronności i wymaga ono podłączenia uziemienia. Uziemienie wykonywane jest przewodem zasilającym.
- Utrzymywać w czystości urządzenie i jego otoczenie.

Czynności zabronione:

- Korzystanie z uszkodzonych przewodów zasilających.
- Pozostawienie bez nadzoru stanowiska będącego pod napięciem 120/230V, bez zabezpieczenia przed dostępem osób niepowołanych.
- Dotykanie wentylatorów z uwagi na możliwe wysokie obroty mogące powodować skaleczenie.
- Dotykanie rezystorów mocy i struktury obiektu w ich okolicy z uwagi na możliwość wystąpienia wysokiej temperatury na ich powierzchni, która może spowodować poparzenie.

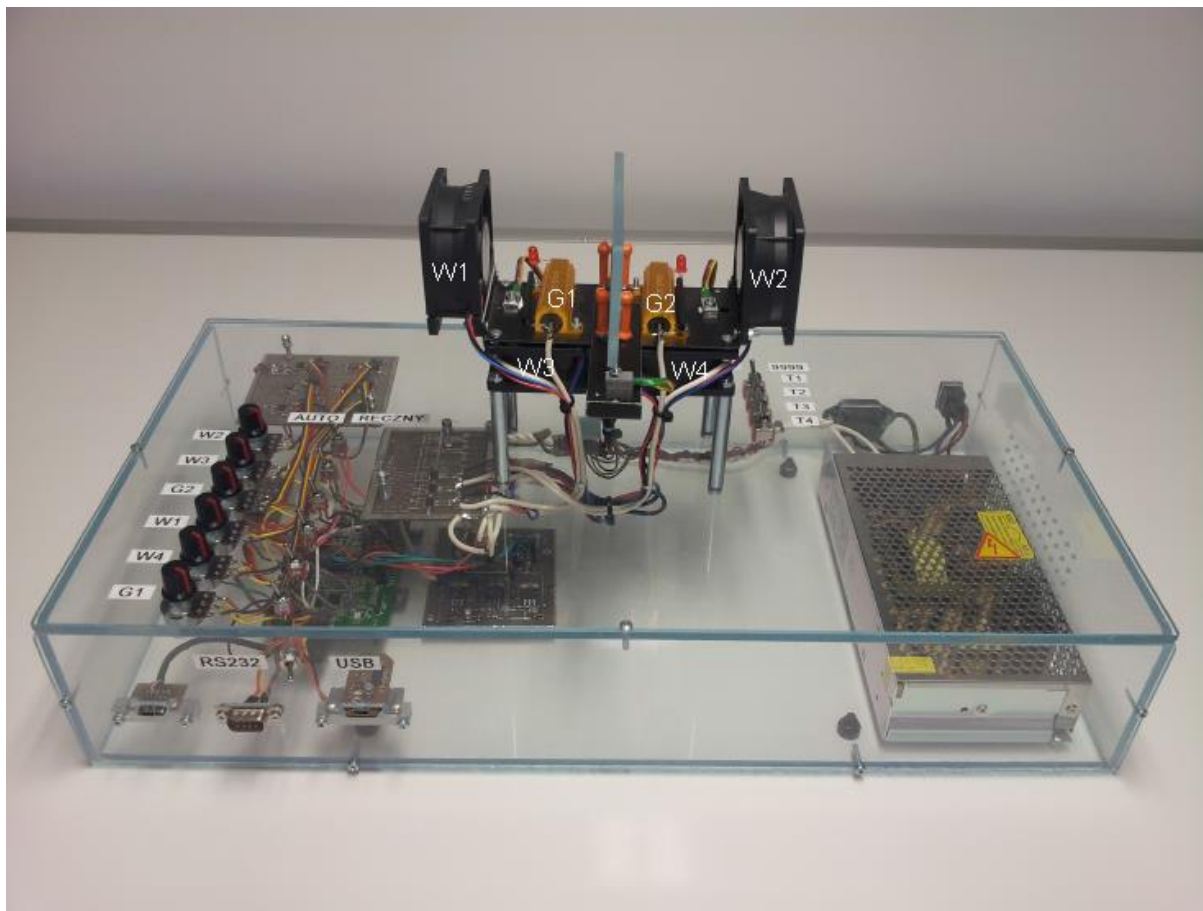
Konieczne czynności przygotowujące przed rozpoczęciem pracy:

- Sprawdzenie stanu wtyczek, gniazd i przewodów sieci 120/230V.
- Usunięcie zbędnych przedmiotów ze stanowiska pracy.
- Upewnienie się, że rozpoczęcie pracy nie spowoduje zagrożeń dla osób na stanowisku pracy oraz innych osób mogących się znaleźć w pobliżu.

Przed przystąpieniem do pracy ze stanowiskiem należy sprawdzić czy kabel zasilający jest włożony do odpowiedniego gniazda na tylnej ścianie obudowy. Jeżeli istnieją jakiegokolwiek obawy, że kabel może być uszkodzony, należy zaprzestać pracy do czasu wymiany kabla. Włączenie zasilania następuje poprzez przełączenie przełącznika w pozycję „I”. Po tej czynności na zasilaczu powinna zapalić się zielona dioda. Mikrokontroler zarządzający posiada 4 LEDy: zieloną, pomarańczową, czerwoną i niebieską. Oznaczają one kolejno: inicjalizację sprzętu, dokonywanie pomiarów, błąd krytyczny, inicjalizację programową. Poprawna inicjalizacja powinna skutkować zgaszeniem wszystkich diod poza pomarańczową, która powinna zapalać się na chwilę co sekundę (oznacza to, że pomiary są okresowo wykonywane).

3.1.2 Charakterystyka obiektu

Obiektem regulacji jest stanowisko laboratoryjne (Rys. 3.1 i Rys. 3.2), którego struktura jest przedstawiona na Rys. 3.3. Jest to obiekt cieplny, w którym jako elementy grzewcze wykorzystano rezystory mocy, w specjalnych obudowach dobrze odprowadzających wytworzone ciepło, oznaczone jako G1 i G2. Do chłodzenia wykorzystano wysokoobrotowe wentylatory (W1, W2, W3, W4). Zastosowano czujniki temperatury z magistralą danych OneWire – oznaczenia T1, T2, T3, T4, T5. Dodatkowo wykorzystano płytę pomiarową służącą do odczytu wartości prądu oraz napięcia – oznaczenia P1, P2.

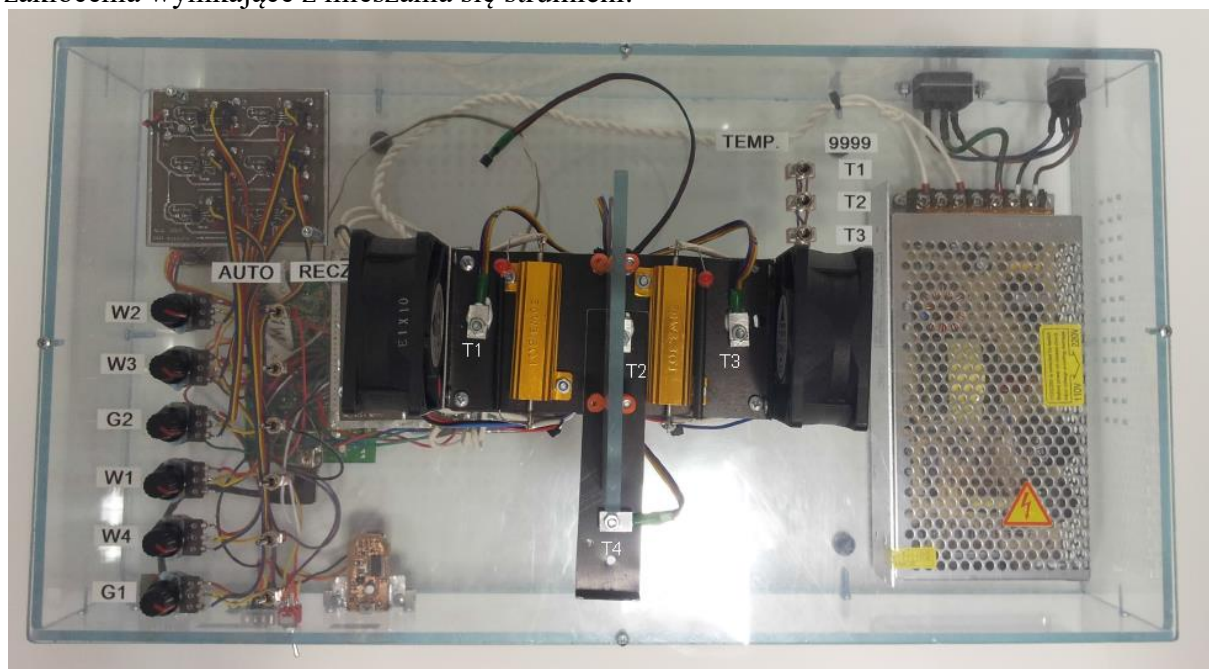


Rys. 3.1: Stanowisko laboratoryjne z zaznaczonymi elementami wykonawczymi: elementami grzewczymi G1 i G2 oraz wentylatorami W1, W2, W3 i W4

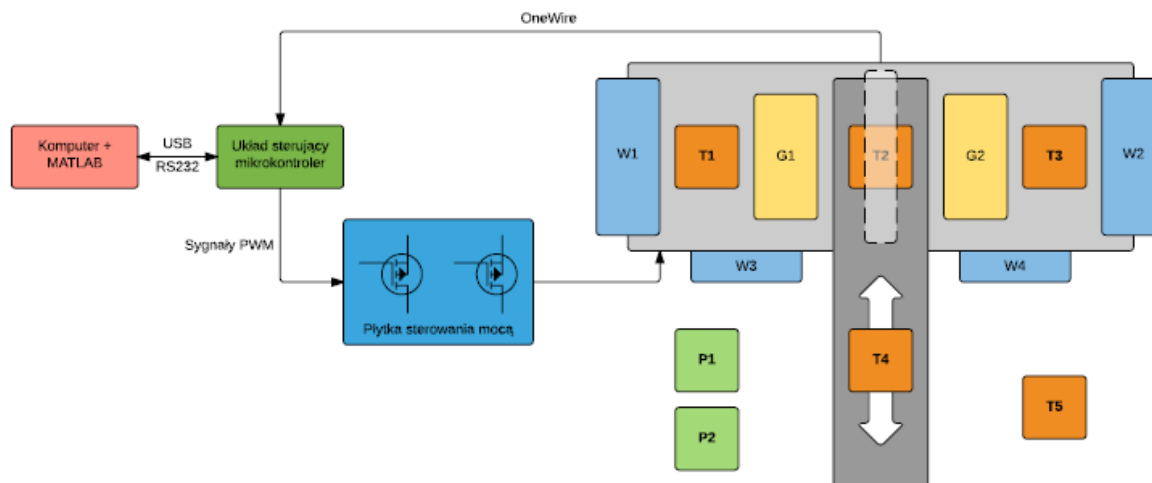
Urządzenie może pracować w trzech trybach komunikacji, poprzez:

- dedykowany protokół komunikacyjny przy użyciu standardu USB (przełącznik na frontowej ścianie musi być ustawiony w pozycji USB);
- protokół MODBUS RTU oraz standard napięciowy RS485 (frontowy przełącznik ustawiony na MODBUS/ANALOG, a przełącznik na tylnej ścianie obudowy ustawiony na MODBUS);
- standard sygnałów analogowych 0-10V podłączając regulator przy użyciu złączy śrubowych (frontowy przełącznik ustawiony w pozycji MODBUS/ANALOG, tylny przełącznik w pozycji ANALOG).

Użytkownik ponadto ma możliwość zmiany charakteru obiektu, w tym celu zamontowana została fizyczna przegroda. Wykorzystana jest ona do oddzielenia dwóch strumieni powietrza wentylatora lewego i prawego, dzięki czemu można zredukować zakłócenia wynikające z mieszania się strumieni.



Rys. 3.2: Stanowisko laboratoryjne z zaznaczonymi czujnikami temperatury T1, T2, T3 i T4



Rys. 3.3: Schemat stanowiska [1]

3.1.3 Konfiguracja procesu

Omawiany obiekt należy do klasy obiektów wielowymiarowych, możliwe jest oddziaływanie 6 sygnałami wejściowymi (traktowanymi jako sygnały sterujące lub zakłócające) na 6 sygnałów wyjściowych (traktowanych jako sygnały regulowane lub

procesowe) (tabl. 1), co pozwala na prowadzenie prac na stanowisku w wielu różnych konfiguracjach.

sygnały wejściowe	Sygnały wyjściowe
intensywność grzania grzałki G1	temperatura T1
intensywność grzania grzałki G2	temperatura T2
wydatek wentylatora W1	temperatura T3
wydatek wentylatora W2	temperatura T4
wydatek wentylatora W3	prąd C
wydatek wentylatora W4	napięcie V

Istnieje bardzo wiele możliwości konfiguracji procesu, w najprostszej wersji rozważa się obiekt o jednym sygnale wejściowym (np. W1) i jednym wyjściowym (np. T1), przy stałej pracy grzałki (np. G1). Znacznie bardziej elastyczny jest obiekt uwzględniający dwa sygnały wejściowe (np. G1 i W1, gdzie zmiany obciążenia wentylatora traktowane mogą być jako zakłócenie) oraz jeden sygnał wyjściowy (np. T1). Możliwe jest dalsze zwiększanie poziomu skomplikowania obiektu poprzez dodawanie kolejnych wejść i wyjść zyskując jednocześnie większą kontrolę nad rozkładem temperatur w punktach T1, do T5.

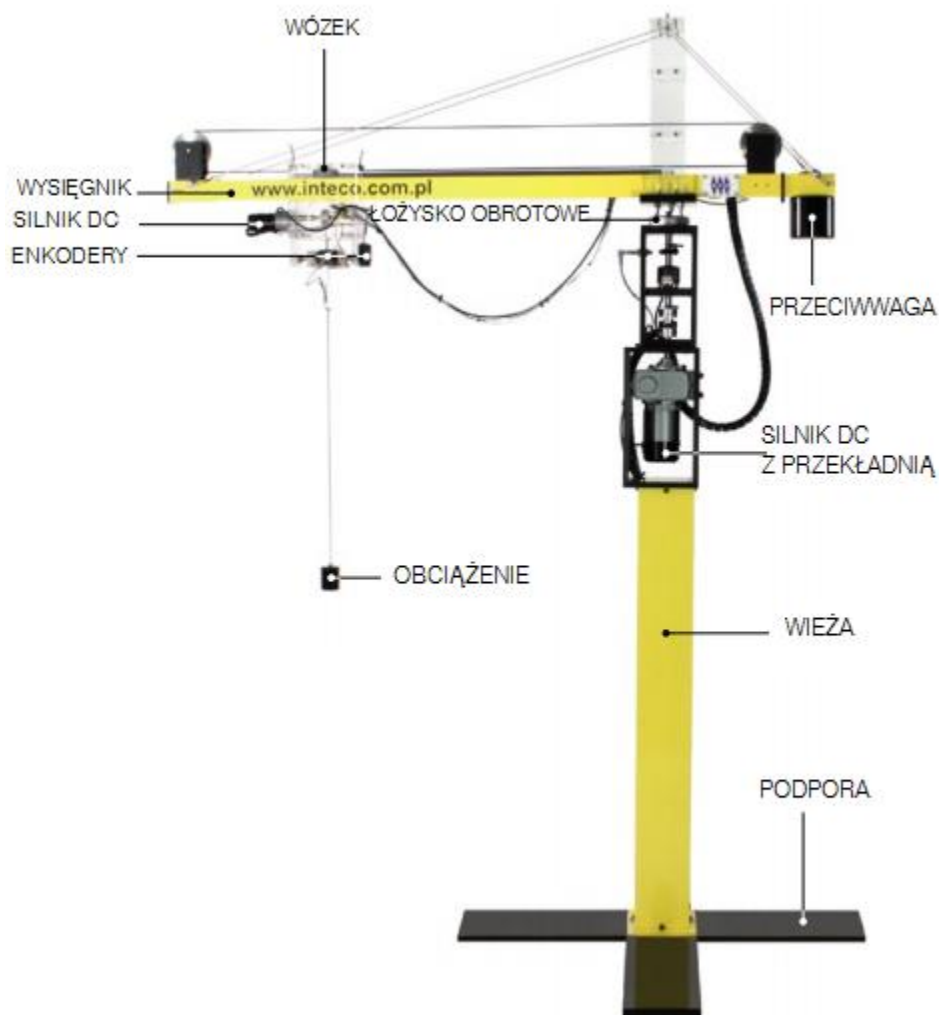
Wartymi uwagi są następujące kombinacje:

1. wejścia (W1, W3), 1 wyjście (T1) – regulacja temperatury T1 stale grzejącego się elementu (symulowane przez stale włączoną grzałkę G1),
2. wejścia (G1, W1), 1 wyjście (T1) – regulacja temperatury T1 przy użyciu grzałki i wentylatora z uwzględnieniem zakłócenia w postaci zmieniającej się temperatury otoczenia T5,
3. wejścia (G1, W1), 2 wyjścia (T1, P1) – regulacja temperatury T1 przy użyciu grzałki i wentylatora z jednoczesną minimalizacją pobieranego prądu (a co za tym idzie mocy)
4. wejścia (G1, G2, W1, W2), 1 wyjście (T2) – regulacja temperatury T2 przy użyciu redundantnych elementów wykonawczych, które mogą pracować wymiennie,
5. wejścia (G1, G2, W1, W2), 2 wyjścia (T1, T3) – regulacja temperatur T1 i T3 z uwzględnieniem słabych zakłóceń (przegroda zamontowana) lub mocnych zakłóceń (przegroda zdjęta), głównie wynikających ze zderzających się strumieni powietrza wytwarzanych przez W1 i W2,
6. wejścia (G1, G2, W3, W4), 2 wyjścia (T2, T4) – regulacja temperatur T2 i T4, z czego zmiana temperatury odczytanej przez T4 następuje z wyraźnym opóźnieniem w stosunku do T2.

Dodatkowo każda z omawianych konfiguracji może być prowadzona w warunkach optymalizacji, tzn. w sposób zapewniający minimalizację sygnałów C i V (prądu i napięcia).

Podczas wykonywania ćwiczenia każdy zespół laboratoryjny otrzyma opis konfiguracji od prowadzącego zajęcia.

3.2 INTECO – Dźwig (TCRANE)

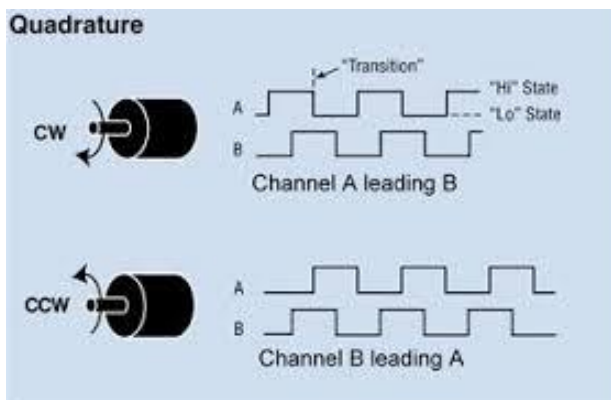


Rys. 3.4: Stanowisko laboratoryjne T-Crane

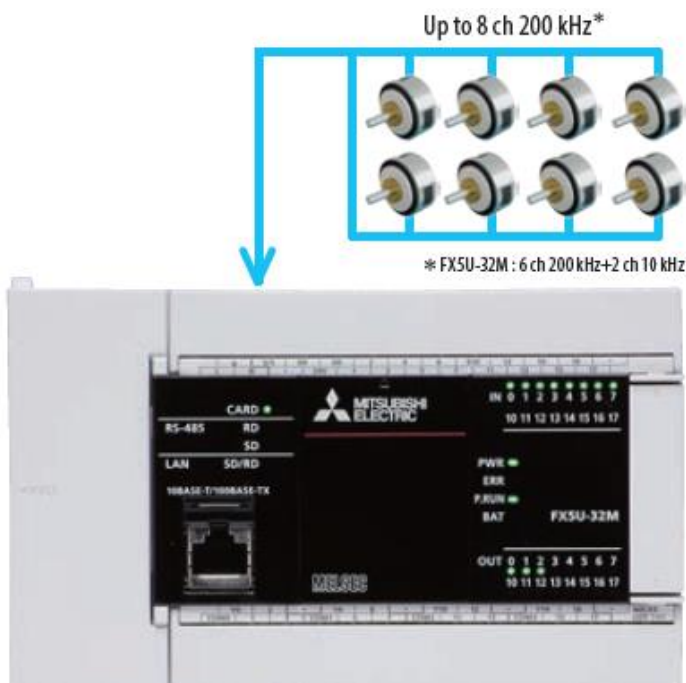
Stanowisko laboratoryjne T-Crane posiada 5 enkoderów inkrementalnych. Trzy z nich mierzą położenie elementów napędzanych przez silniki. Dwa z nich znajdują się na karetkie dźwigu i przedstawiają aktualne wychylenie obciążenia od pionu.

Enkoder (przetwornik położenia) służy do pomiaru położenia. W powyższej wersji mamy do czynienia z przetwornikiem obrotowym. Zatem możemy dzięki niemu określić położenie kątowe wokół osi. Jeżeli podłączymy go do liniowego układu przeniesienia napędu możemy określić położenie liniowe wyrażane w odległości.

UWAGA: Pamiętajmy, że do określenia kierunku potrzebujemy dwóch sygnałów (tzw. fazy A i B). W sterowniku wykorzystujemy dwa wejścia do zliczania impulsów z fazy A i B. Omawianą sytuację przedstawia rysunek 3.5. Wykrywanie kierunku jest wykonywane automatycznie w sterowniku. Przy pomocy mechanizmu sprzętowych liczników możemy w dowolnym momencie odczytać aktualne położenie enkodera. Rysunek 3.6 przedstawia możliwość podłączenia 8 enkoderów inkrementalnych do sterownika FX5. W pamięci sterownika pozycja będzie przedstawiona w odpowiednim REJESTRZE 32 bitowym. Reprezentacja liczby może być UINT lub INT(U2).

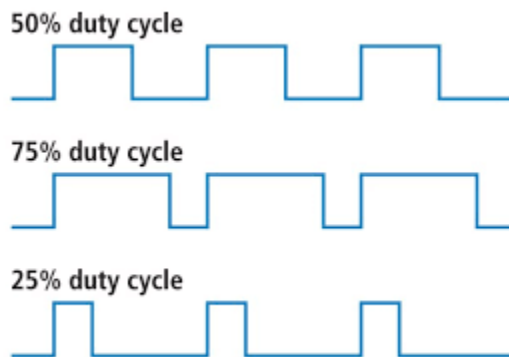


Rys. 3.5: Sygnał wyjściowy (kwadraturowy) enkodera inkrementalnego



Rys. 3.6: Wykorzystanie sygnałów kwadraturowych w sterowniku PLC

Regulacja pozycji - sterowanie silnikiem w naszym przypadku będzie sterowaniem jego prędkości obrotową. Silnik porusza elementem wykonawczym a jego położenie jest określone przez enkoder. Zmiana prędkości będzie proporcjonalna do wypełnienia sygnału PWM podanego na silnik. Sygnał PWM generowany będzie na wyjściu sterownika PLC. Rysunek 3.7 przedstawia przykładowe wypełnienia sygnału PWM.

**Rys. 3.7: Prezentacja wypełnienia sygnału**

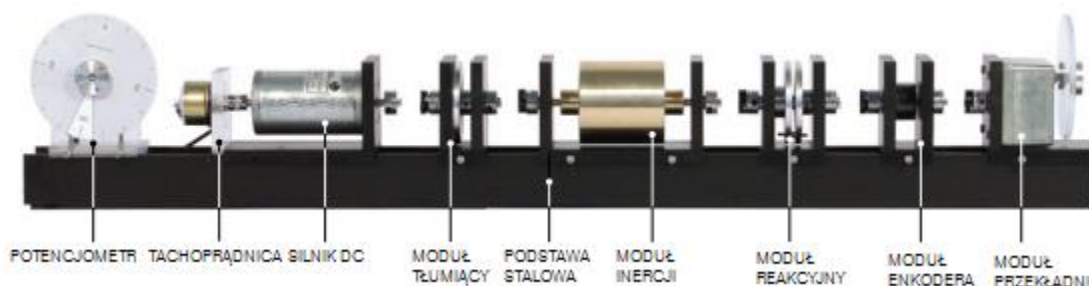
Rozdzielczość fizyczna enkoderów inkrementalnych użytych w osiach X i Z wynosi 1024 impulsy/obróć, natomiast w osi Th wynosi 512 impulsów/obróć.

Poniższa tabela przedstawia podłączenie wejść i wyjść fizycznych sterownika PLC FX5. Na jej podstawie należy utworzyć odpowiednie grupy etykiet.

	WEJŚCIA CYFROWE
X0	EncA1_X Enkoder inkrementalny, fala A, oś X
X1	EncB1_X Enkoder inkrementalny, fala B, oś X
X2	EncA2_Y Enkoder inkrementalny, fala A, oś Y
X3	EncB2_Y Enkoder inkrementalny, fala B, oś Y
X4	EncA4_AX Enkoder inkrementalny, fala A, kąt AX
X5	EncB4_AX Enkoder inkrementalny, fala B, kąt AX
X6	EncA5_AY Enkoder inkrementalny, fala A, kąt AY
X7	EncB5_AY Enkoder inkrementalny, fala B, kąt AY
X10	EncA3_Z Enkoder inkrementalny, fala A, oś Z
X11	EncB3_Z Enkoder inkrementalny, fala B, oś Z
X12	Switch1_Z Wyłącznik krańcowy, oś Z
X13	Switch3_X Wyłącznik krańcowy, oś X
X14	Switch2_Y Wyłącznik krańcowy, oś Y
X15	Therm_Z Flaga limitu temperatury, oś Z
X16	Therm_Y Flaga limitu temperatury, oś Y
X17	Therm_X Flaga limitu temperatury, oś X
	WYJŚCIA CYFROWE
Y1	PWM_Z Sygnał sterujący typu PWM dla silnika DC, oś Z. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y3	Brake_Z Sygnał zatrzymujący pracę silnika DC, oś Z
Y4	Dir_Z Sygnał zmiany kierunku obrotów silnika DC, oś Z
Y2	PWM_Y Sygnał sterujący typu PWM dla silnika DC, oś Y. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y5	Brake_Y Sygnał zatrzymujący pracę silnika DC, oś Y
Y6	Dir_Y Sygnał zmiany kierunku obrotów silnika DC, oś Y

Y0	PWM_X Sygnał sterujący typu PWM dla silnika DC, oś X. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y7	Brake X Sygnał zatrzymujący pracę silnika DC, oś X
Y10	Dir X Sygnał zmiany kierunku obrotów silnika DC, oś X

3.3 INTECO – Serwomechanizm (SERVO)



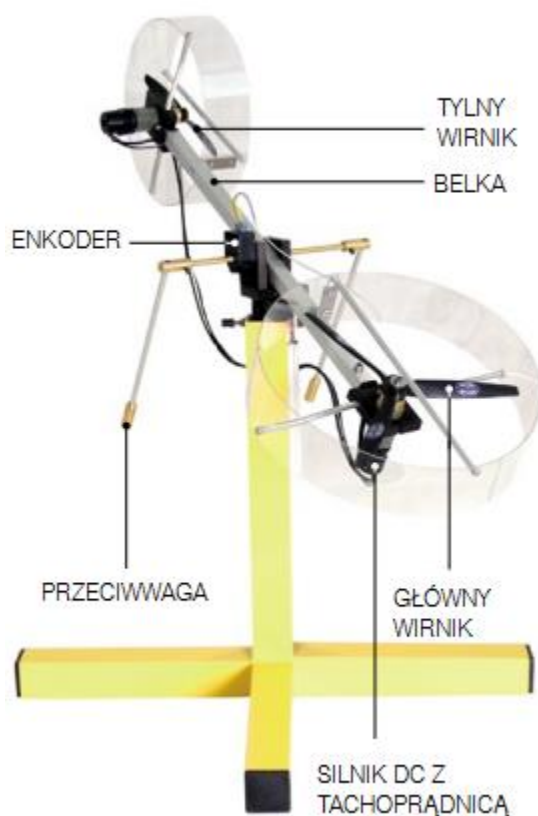
Rys. 3.8: Stanowisko laboratoryjne Modular Servo

Rozdzielczość fizyczna użytego enkodera inkrementalnego wynosi 1024 impulsy/obrot. Zakres wartości napięcia pojawiający się na wyprowadzeniach „Potentiometer” i „Tacho” wynosi ± 10 V. Sygnały analogowe przekazane są do sterownika przez moduł wejść analogowych FX5-4ADP. Możliwości regulacji w tym przypadku mogą obejmować pomiar prędkości obrotowej przy pomocy tachoprądnicy (generator – napięcie stałe na wyjściu rośnie proporcjonalnie do prędkości obrotowej - analogia dynamo w rowerze) i sterowanie prędkością obrotową silnika przez zadawanie wypełnienia sygnału PWM. Druga opcja sterowania to pomiar pozycji z enkodera inkrementalnego i sterowanie położenia silnika.

	WEJŚCIA CYFROWE
AIN1	Potentiometer Analogowy sygnał pomiarowy z potencjometru (zadajnik położenia, prędkości kątowej itp.).
AIN2	Tacho Analogowy sygnał pomiarowy prędkości obrotowej z tachoprądnicy.
X2	Therm Flaga limitu temperatury.
X1	EncB1 Enkoder inkrementalny, fala B, oś pozioma.
X0	EncA1 Enkoder inkrementalny, fala A, oś pozioma.
	WYJŚCIA CYFROWE

Y0	PWM Sygnał sterujący typu PWM dla silnika DC. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y1	Brake Sygnał zatrzymujący pracę silnika DC.
Y2	Dir0 Sygnał zmiany kierunku obrotów silnika DC.

3.4 INTECO – Helikopter (TRAS)



Rys. 3.9: Stanowisko laboratoryjne TRAS

Rozdzielczość fizyczna wszystkich użytych enkoderów inkrementalnych wynosi 1024 impulsy/obrot. Zakres wartości napięcia pojawiający się na wyprowadzeniach „Tacho_Pt” i „Tacho_Az” wynosi ± 10 V. Sygnały analogowe przekazane są do

sterownika przez moduł wejść analogowych FX5-4ADP. Możliwości regulacji w tym przypadku obejmują pomiary położenia dwóch osi w przestrzeni i odpowiednie sterowanie silnikami przez zmianę wypełnienia PWM.

	WEJŚCIA CYFROWE
AIN1	Tacho+B1_Pt Analogowy sygnał pomiarowy z tachoprądnicy, śmigło - oś pionowa (Pitch). Współczynnik przetwarzania tachoprądnicy wynosi 0.5 [V]/1000 [obr/min] przy założeniu, że napięcie mierzone jest bezpośrednio na tachoprądnicy.
AIN2	Tacho_Az Analogowy sygnał pomiarowy z tachoprądnicy, śmigło - oś pozioma (Azimuth) . Współczynnik przetwarzania tachoprądnicy wynosi 0.5 [V]/1000 [obr/min] przy założeniu, że napięcie mierzone jest bezpośrednio na tachoprądnicy.
X0	EncA2_Pt Enkoder inkrementalny, fala A, oś pionowa
X2	EncB1_Az Enkoder inkrementalny, fala B, oś pozioma
X1	EncB2_Pt Enkoder inkrementalny, fala B, oś pionowa
X3	EncA1_Az Enkoder inkrementalny, fala A, oś pozioma
X4	Therm1_Az Flaga limitu temperatury, oś pozioma
X5	Therm0_Pt Flaga limitu temperatury, oś pionowa
	WYJŚCIA CYFROWE
Y2	Brake1_Az Sygnał zatrzymujący pracę silnika DC, oś pozioma
Y3	Dir1_Az Sygnał zmiany kierunku obrotów silnika DC, oś pozioma
Y0	PWM1_Az Sygnał sterujący typu PWM dla silnika DC, oś pozioma. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y1	PWM0_Pt Sygnał sterujący typu PWM dla silnika DC, oś pionowa. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y4	Brake0_Pt Sygnał zatrzymujący pracę silnika DC, oś pionowa. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y5	Dir0_Pt Sygnał zmiany kierunku obrotów silnika DC, oś pionowa

3.5 INTECO - Zbiorniki wodne (TANKS)



Rys. 3.10: Stanowisko laboratoryjne 3D-Tanks

Zgodnie z zaleceniami producenta zaworów proporcjonalnych, zalecana częstotliwość sygnału sterującego typu PWM (wyprowadzenia „Valve1”, „Valve2” i „Valve3”) powinna wynosić między 5 a 15 kHz. Do budowy czujnika poziomu cieczy w systemie 3-Tanks (wyjścia „Level1”, „Level2” i „Level3”) wykorzystano czujnik ciśnienia wraz z półprzewodnikowym układem do konwersji U/f (napięcie na częstotliwość). Zakres zmian częstotliwości to przedział od 80 kHz dla wartości poziomu cieczy 0.0 cm do około 180 kHz dla wartości poziomu cieczy 25 cm. Warto zweryfikować czy zakres ten jest prawidłowy.

Stanowisko pozwala na regulację poziomu wody w trzech zbiornikach, gdzie każdy jest wyposażony w jeden czujnik poziomu i jeden zawór. Każdy zawór może być sterowany wypełnieniem sygnału PWM. Dodatkowo pompa, który dostarcza wodę do górnego zbiornika jest sterowana wypełnieniem sygnału PWM.

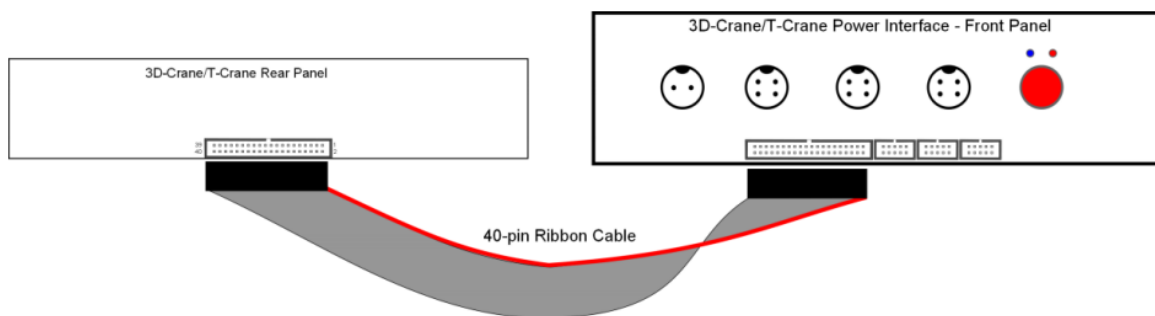
	WEJŚCIA CYFROWE
X0	Level1 Cyfrowy sygnał pomiarowy z czujnika poziomu cieczy, typu częstotliwościowego, zbiornik górny.

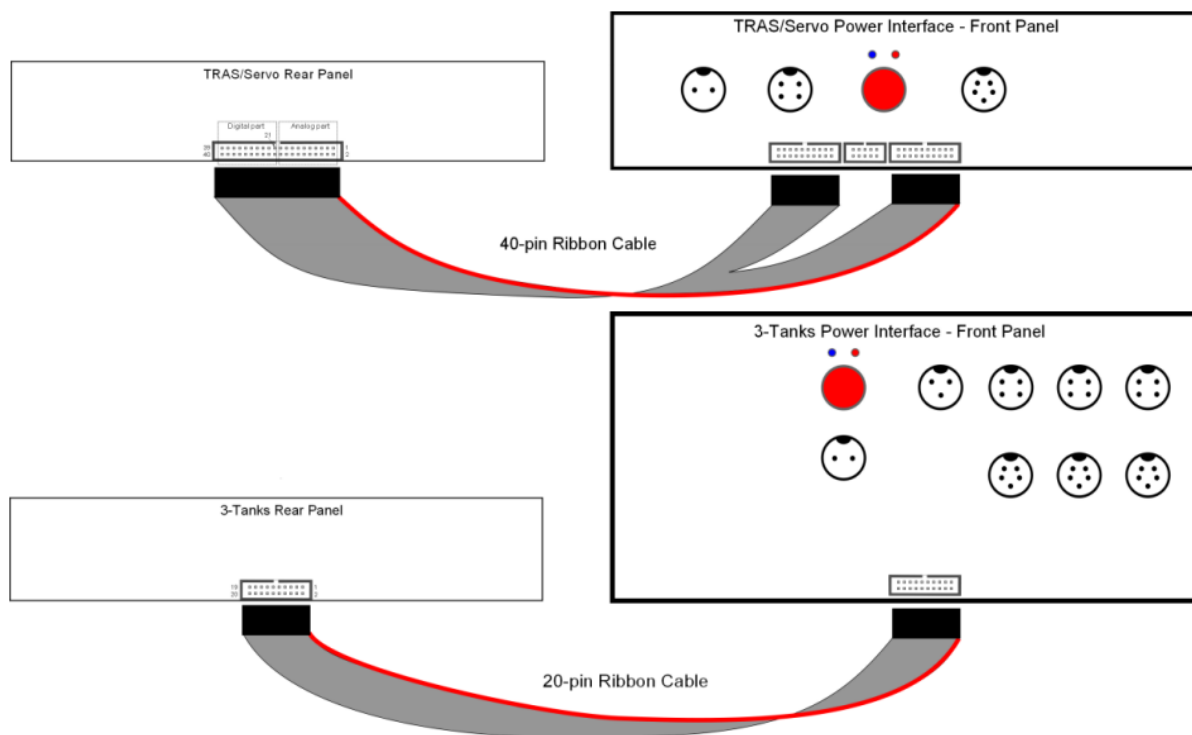
X1	Level2 Cyfrowy sygnał pomiarowy z czujnika poziomu cieczy, typu częstotliwościowego, zbiornik środkowy.
X2	Level3 Cyfrowy sygnał pomiarowy z czujnika poziomu cieczy, typu częstotliwościowego, zbiornik dolny.
WYJŚCIA CYFROWE	
Y0	Pump Sygnał sterujący typu PWM dla pompy wody. Zalecana częstotliwość tego sygnału mieści się w przedziale (5-15) kHz.
Y1	Valve1 Sygnał sterujący typu PWM dla zaworu proporcjonalnego, zbiornik górny.
Y2	Valve2 Sygnał sterujący typu PWM dla zaworu proporcjonalnego, zbiornik środkowy.
Y3	Valve3 Sygnał sterujący typu PWM dla zaworu proporcjonalnego, zbiornik dolny.

3.6 Podłączenie zestawów INTECO

Każdy z interfejsów PLC musi zostać prawidłowo podłączony do interfejsu mocy wybranego zestawu firmy INTECO. Połączenie realizowane jest poprzez płaskie taśmy 20- lub 40- przewodowe z wykorzystaniem złącz typu IDC znajdujących się na tylnych panelach interfejsów PLC oraz panelach frontowych poszczególnych sterowników mocy (szczegółowy opis można znaleźć w dokumentacji technicznej dotyczącej danego zestawu).

Przy realizacji połączenia należy zwracać uwagę na oznaczenia poszczególnych gniazd IDC oraz dostarczonych płaskich kabli. Szczególnie dotyczy to przypadków, w których dokonuje się połączenia zarówno sygnałów cyfrowych jak i analogowych. Pomyłka w połączeniu może doprowadzić do uszkodzenia interfejsu PLC lub kanałów I/O sterownika mocy. Na poniższych rysunkach przedstawiono wygląd tylnych paneli interfejsów PLC dla poszczególnych zestawów. Szczególną uwagę należy zwrócić na ostatni rysunek, który przedstawia tylny panel dla zestawu TRAS i SERVO. Złącze 40-końcówkowe jest podzielone na dwie części. Pierwsza z nich (wyprowadzenia 1-20) stanowi interfejs analogowy, natomiast druga (wyprowadzenia 21-40) interfejs cyfrowy. Przy łączeniu należy zwrócić uwagę na czerwony marker płaskiej taśmy łączeniowej, który oznacza wyprowadzenie nr 1. Jest to szczególnie istotne dla systemów TRAS i SERVO, gdzie płaska 40-kablowa taśma rozdziela się na dwie 20-kablowe części (analogową i cyfrową).



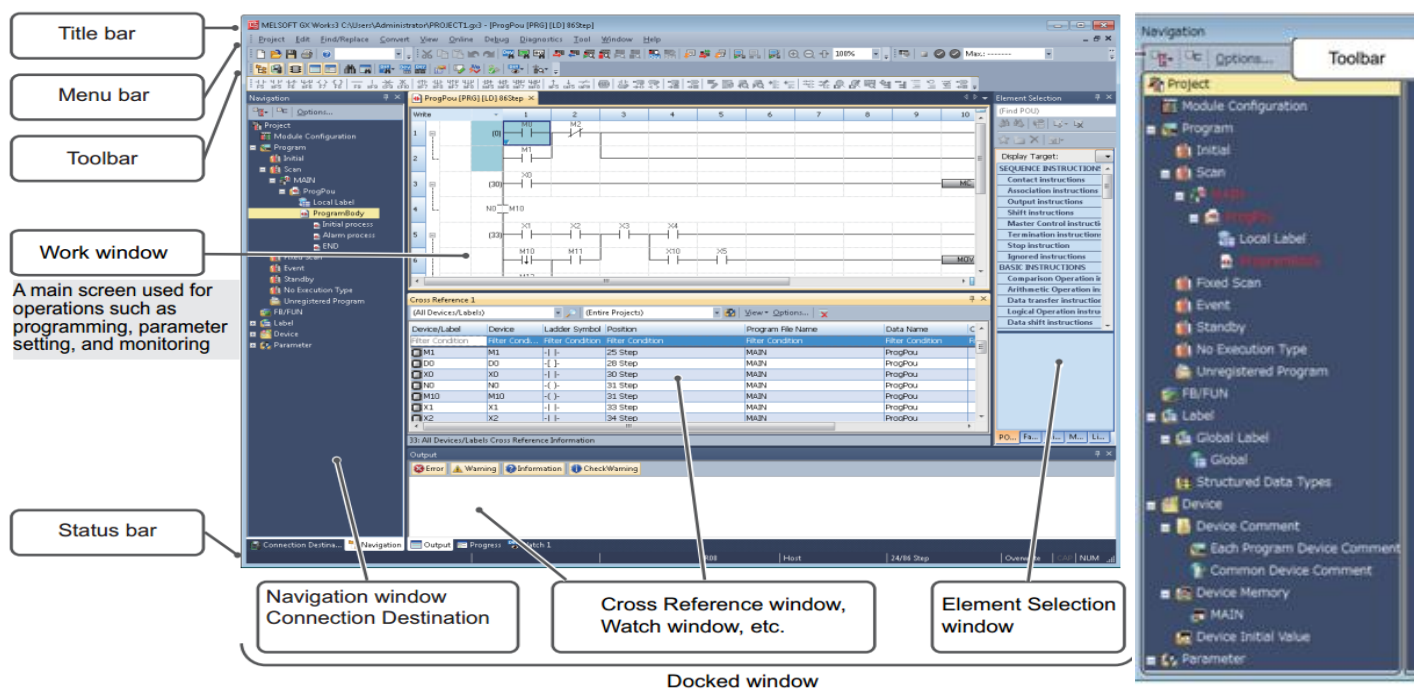


Rys. 3.11: Sposób podłączenia interfejsów dla obiektów laboratoryjnych

4

5 Tworzenie kodu sterującego sterownika PLC w środowisku GxWorks3

Środowisko GxWorks3



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

5.1 Tworzenie nowego projektu

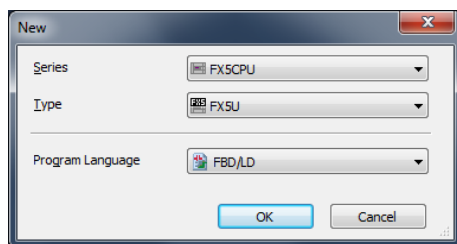
Proszę utworzyć nowy projekt ([Project] → [New] ())

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



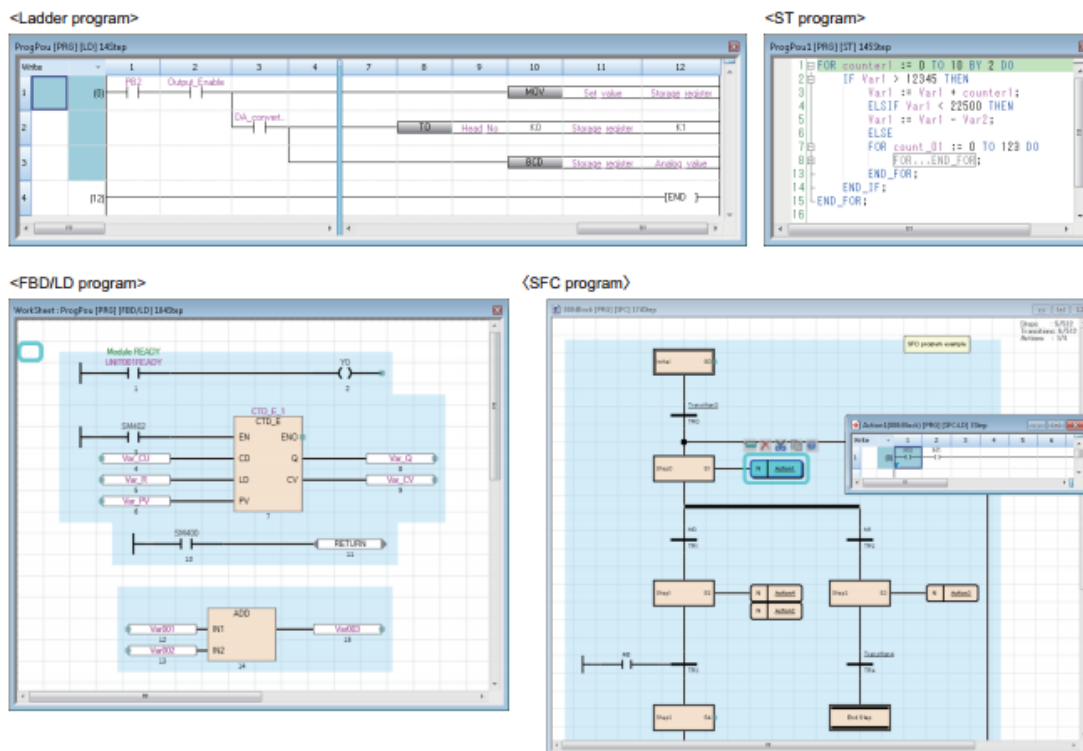
***Rysunek 3** Oznaczenie modelu i serii sterownika.*

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



***Rysunek 4** Parametry wstępnego 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.

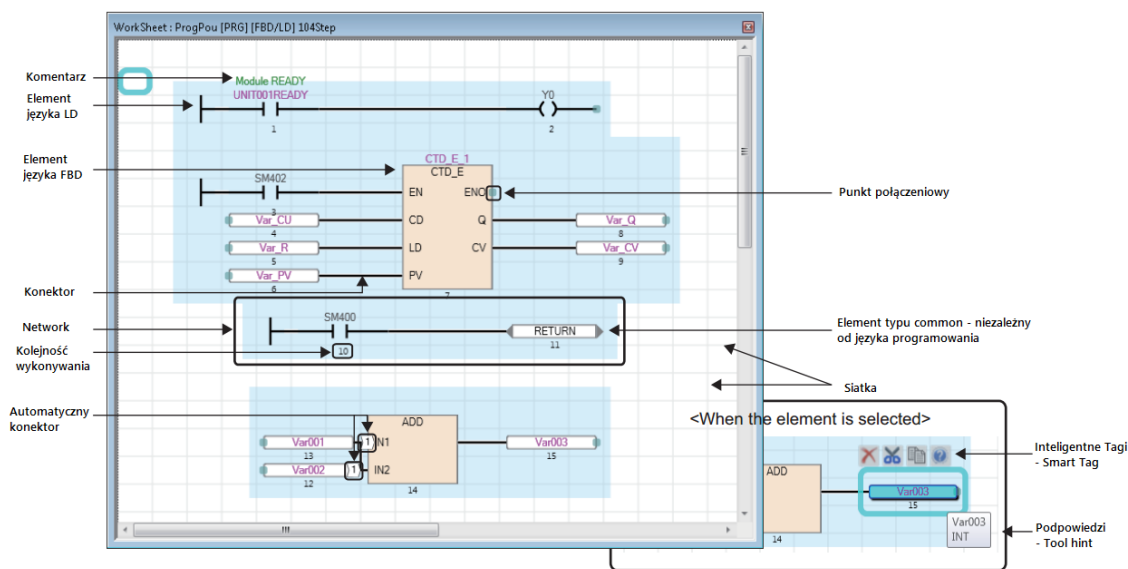
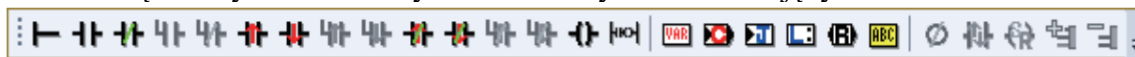


Rysunek 5 Języki programowania wspierane przez aplikację GxWorks3..

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

5.2 Elementy języka FBD/LD

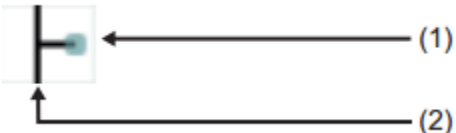
Pasek narzędziowy zawiera wszystkie elementy strukturalne języka FBD/LD:

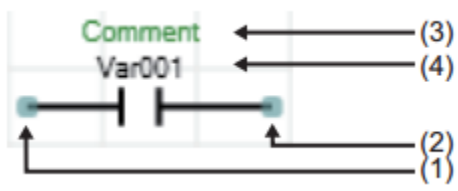
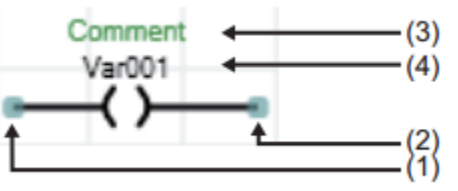


Rysunek 6 Edytor języka FBD/LD.

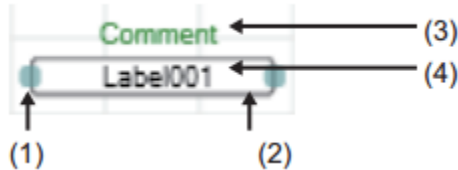

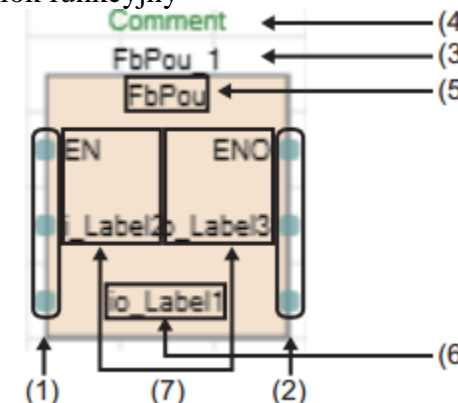
Element	Opis
Komentarz	Komentarz etykiety lub bloku funkcjonalnego. Nie podlega kompilacji.
Element języka LD	Element pochodzący z języka programowania Ladder Diagram
Element języka FBD	Element pochodzący z języka Function Block Diagram (FBD)
Element typu common	Element wbudowany, niezależny od języka 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 najechniu kursorem myszki

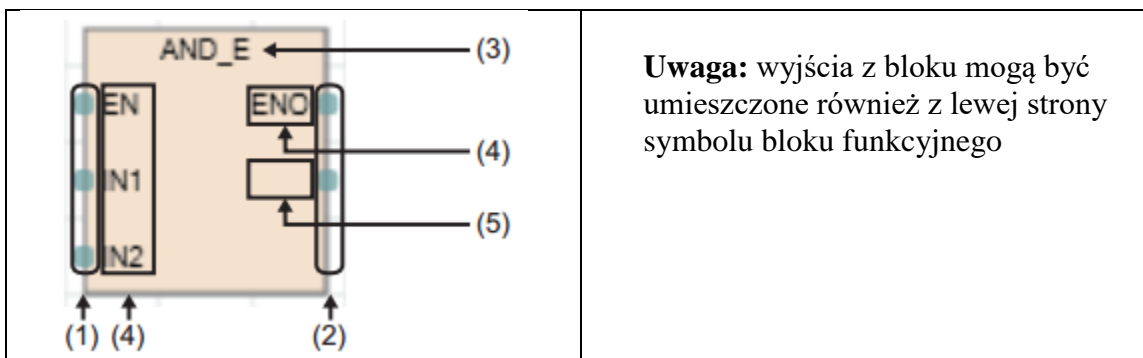
Elementy języka LD

Element	Opis
Lewa szyna zasilająca 	(1) Wyjściowy punkt połączeniowy (2) Lewa szyna zasilająca

<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 (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>



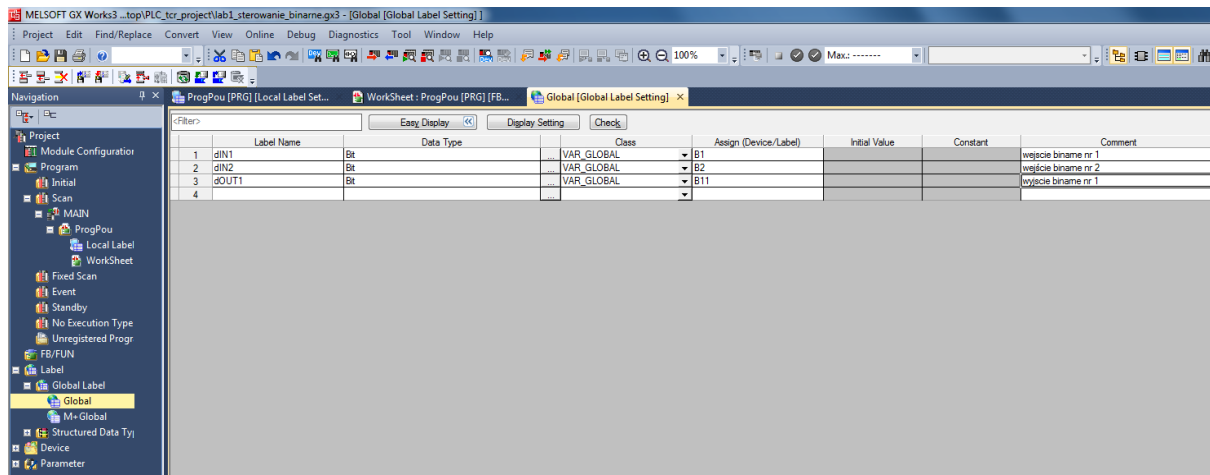
Elementy wspólne (common element)

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

5.3 Definicja zmiennych

W celu tworzenia czytelnego kodu zalecane jest wykorzystanie zmiennych symbolicznych zamiast adresowania bezpośredniego pamięci sterownika. W tym celu należy wykorzystać mechanizm etykiet (label), które mogą mieć zakres lokalny (widoczne tylko w danym komponencie lub podprogramie) lub globalny (widoczne w całym systemie i propagowane po sieci – to rozwiązanie jest zalecane z uwagi na możliwość przypisania fizycznych urządzeń, które później będą skanowane w systemie SCADA, przez fizyczne urządzenia rozumiemy tutaj odpowiedniego adresy pamięci sterownika PLC). Definicja etykiet 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

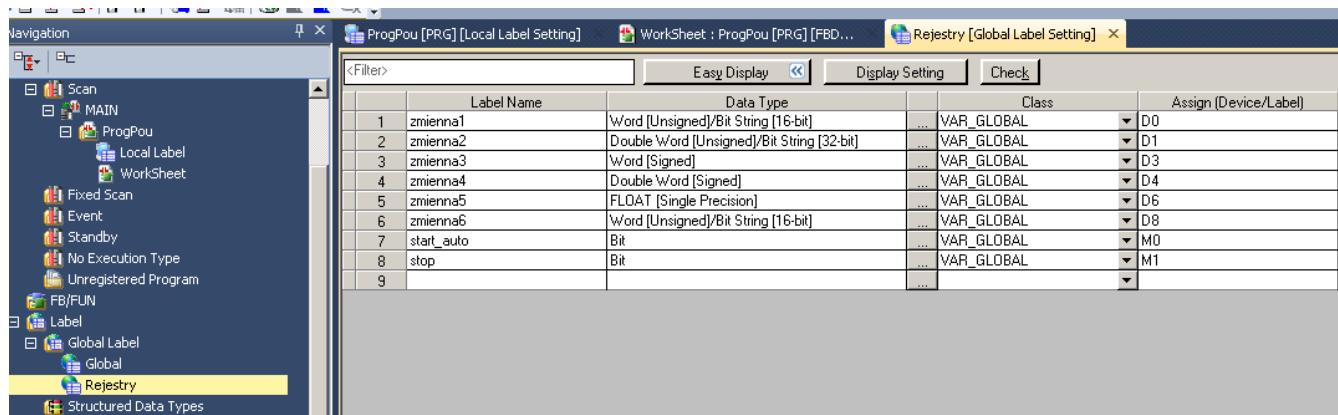
powoduje otwarcie okna dialogowego, gdzie możliwa jest konfiguracja zmiennej. Jeżeli posługujemy się językiem skryptowym ST po wpisaniu tekstu nowej nazwy klikamy na niej prawym przyciskiem i wybieramy opcję „Register Label”. Jeżeli etykieta nie ma koloru „magenta” znaczy to, że nie została zadeklarowana bądź wpisano złą jej nazwę w kodzie.



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. Przy pisaniu programu PLC można używać etykiet, ale do skanowania zmiennych w systemie MAPS należy używać bezpośrednich adresów pamięci zadeklarowanych w kolumnie Assign. Przykładowo, jeżeli będziemy chcieli wyświetlić wartość „zmienna6” w systemie MAPS trzeba zeskanować rejestr D8 sterownika PLC.



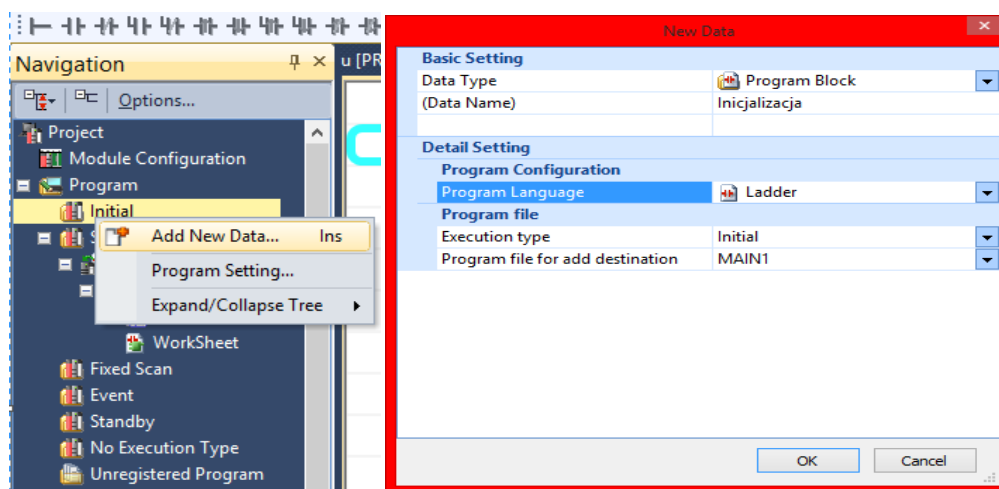
Rysunek 8a Deklaracja globalnych „Labeli”.

5.4 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.

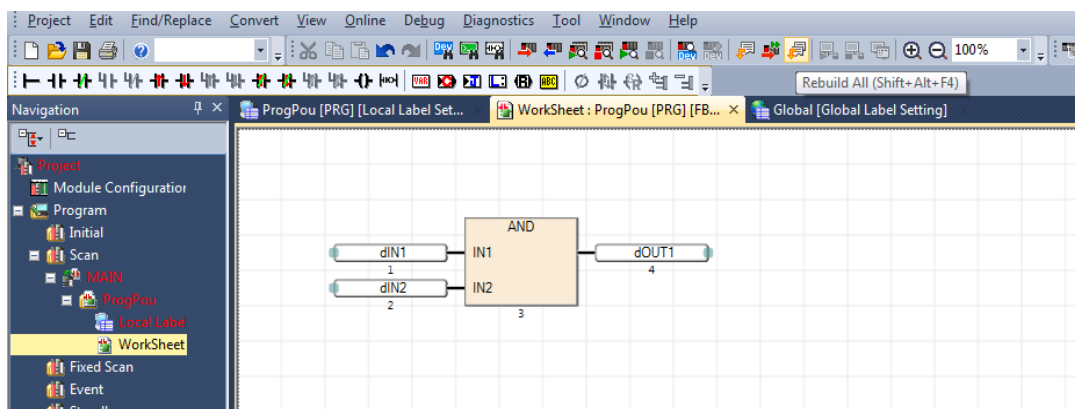
W każdej z sekcji można stworzyć kilka podprogramów klikając w sekcję prawym przyciskiem myszy a następnie wybierając z menu opcję „Add New Data” (Patrz poniżej).



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 podpowiedzi o dostępnych blokach są wyświetlane pod tworzoną blokiem.

Uwaga: Należy zwrócić szczególną uwagę na kolejność wykonywania algorytmów. Kolejność wykonywania bloków jest zaznaczona małymi cyferkami pod każdym blokiem. Ułożenie bloków na karcie edycji może zmienić kolejność wykonania poszczególnych instrukcji, co bezpośrednio ma wpływ na działania.

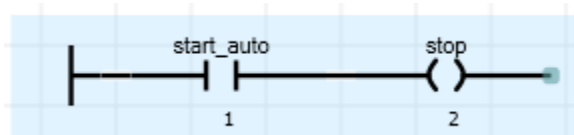


Rysunek 10 Przykład tworzenia kodu sterującego.

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)

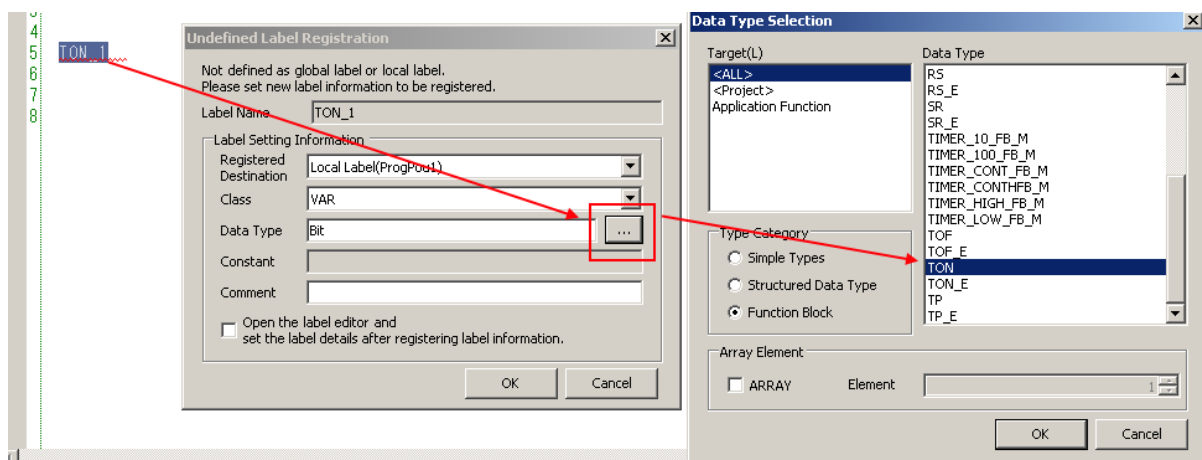
FBD:



ST:

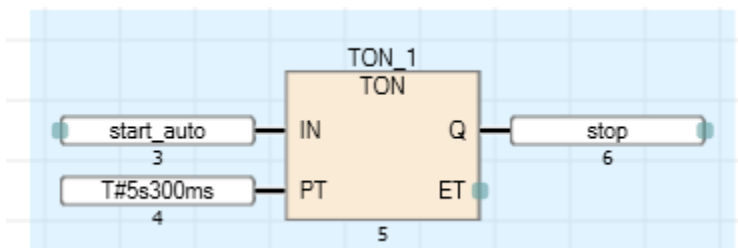
OUT(start_auto, stop);

3. Opóźnienie załączenia TON, opóźnienie wyłączenia TOF, impuls o zadanym czasie TP. Wszystkie te funkcje potrzebują deklaracji instancji w zmiennych lokalnych. Wprowadzenie nowej instancji można wykonać wpisując jej nazwę, następnie klikamy prawym przyciskiem myszy na wprowadzonej nazwie, wybieramy „Register Label” i wybieramy kolejne opcje jak na rysunku poniżej.



Rys. 5.1: Dodanie instrukcji TON

FBD:



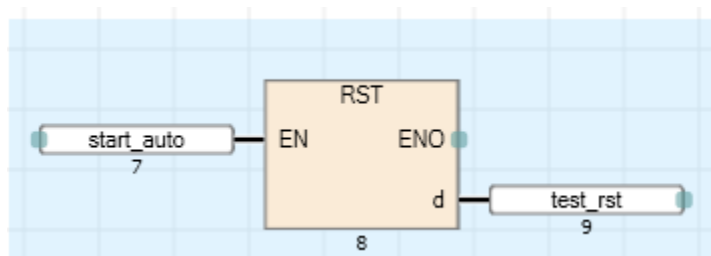
Rys. 5.2: Użycie instrukcji TON

ST:

```
TON_1(IN:= start_auto , PT:= T#5s300ms , Q=> stop );
```

4. Instrukcja ustawienia SET, kasowania RST

FBD:



Rys. 5.3: Użycie instrukcji SET/RST

ST:

```
SET(start_auto, test_set);  
RST(start_auto, test_rst);
```

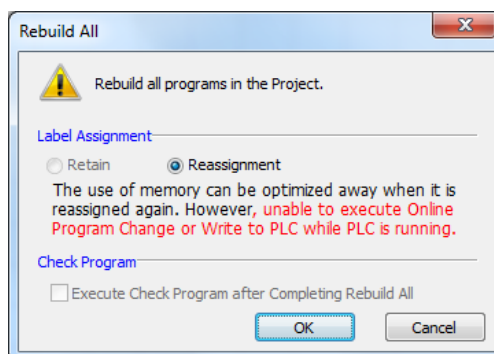
Należy unikać stosowania nazw zmiennych, które mogą być nazwami własnymi zastrzeżonymi w programie np. STOP, ST. Mogą one oznaczać nazwy instrukcji lub nazwy urządzeń fizycznych.

Kompilacja kodu



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

Rekompilacja wymaga potwierdzenia komunikatu:

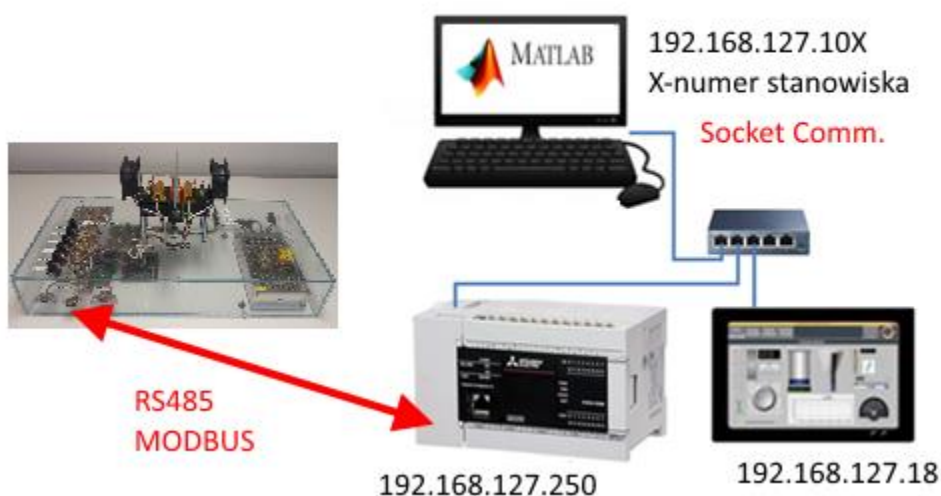


Rysunek 11 Rekompilacja – okno dialogowe

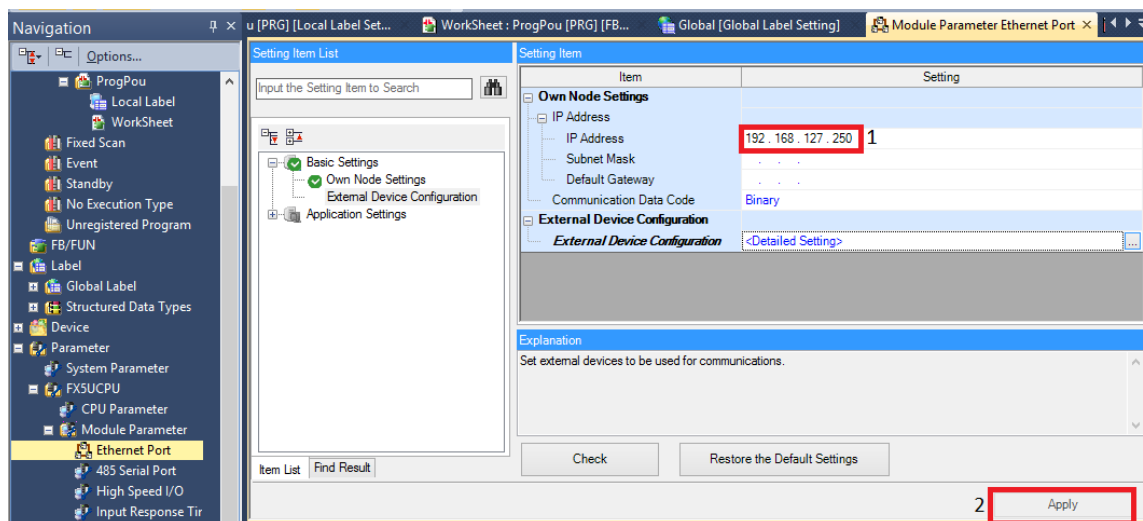
Po rekompilacji projektu nie będzie możliwe ładowanie sterownika w trybie Online. Z tego powodu drobne zmiany w programie należy zatwierdzać bezpośrednio zapisując projekt i wywołując komendę „Online Program Change” – przycisk pomiędzy opcjami 1 i 2.

5.5 Konfiguracja sterownika

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

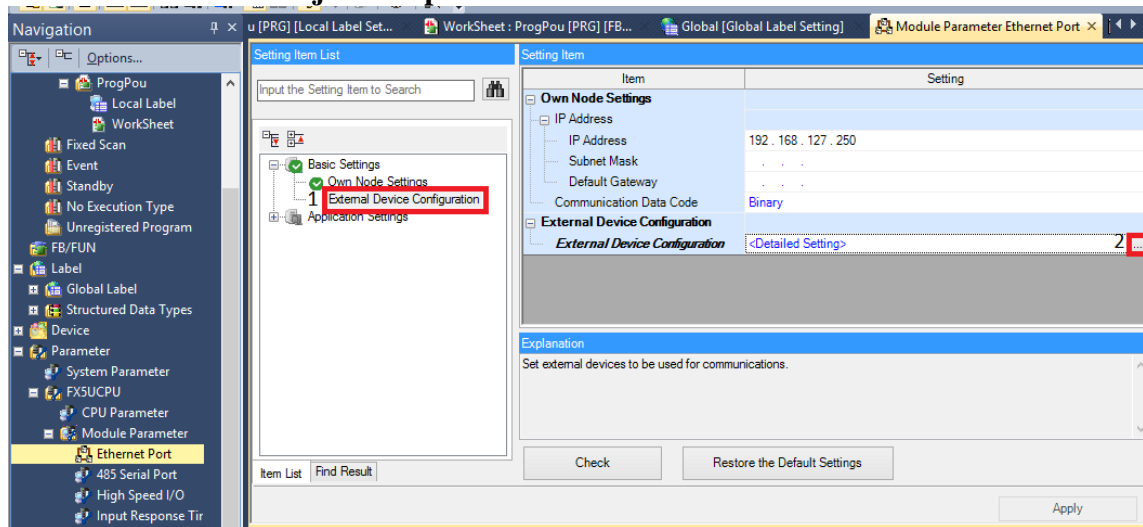


Rysunek 12 Adresacja urządzeń w sieci lokalnej stanowiska
Ustawienie adresu IP sterownika:

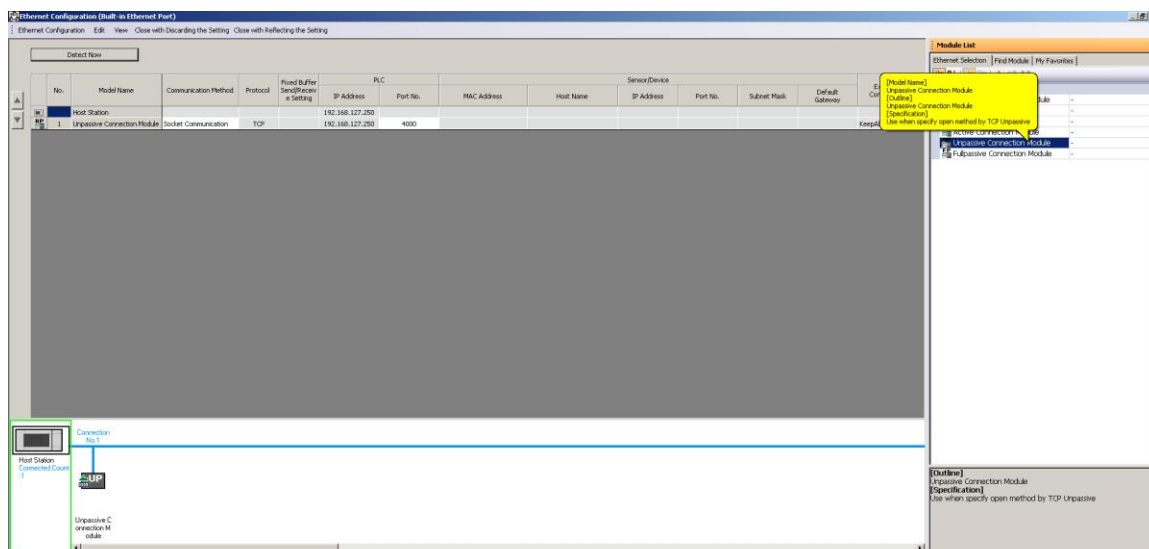


Rysunek 13 Ustawienie adresu IP portu Ethernet

Ustawienie komunikacji z komputerem:



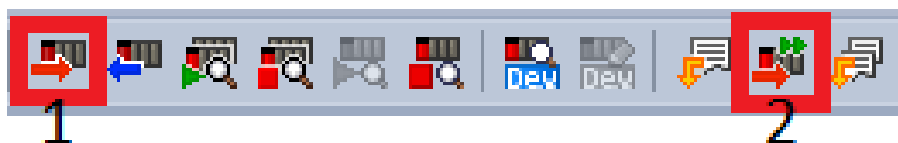
Rysunek 14 Wywołania okna konfiguracji zewnętrznej komunikacji



Rysunek 15 Dodanie komunikacji Unpassive (port 4000)

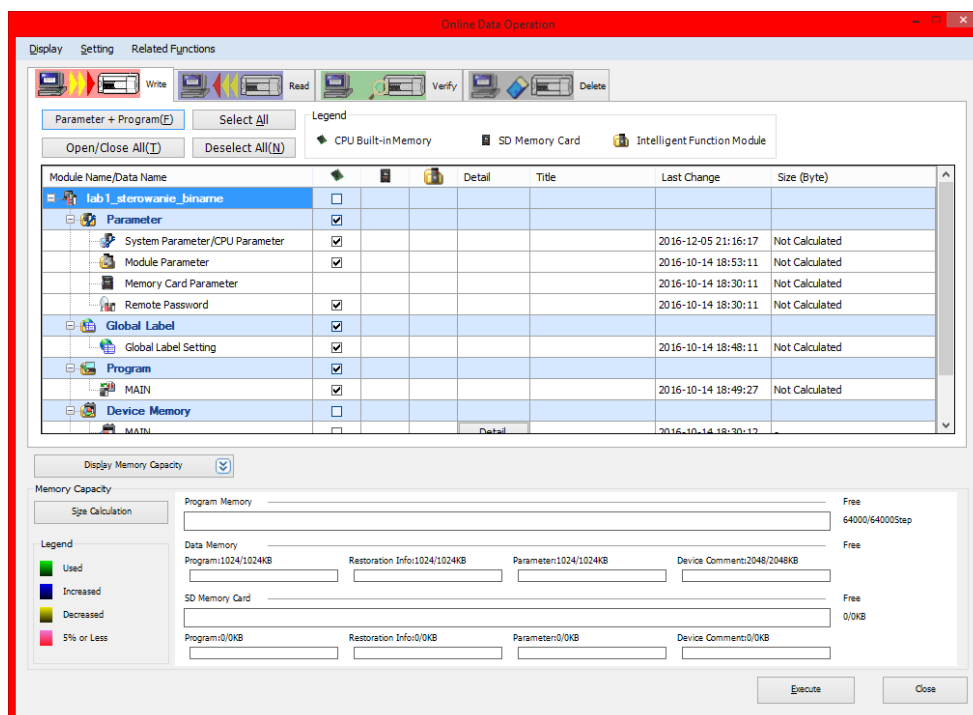
5.6 Programowanie sterownika

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/Verifyfikacja/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 wgrać nowy program i parametry do sterownika.

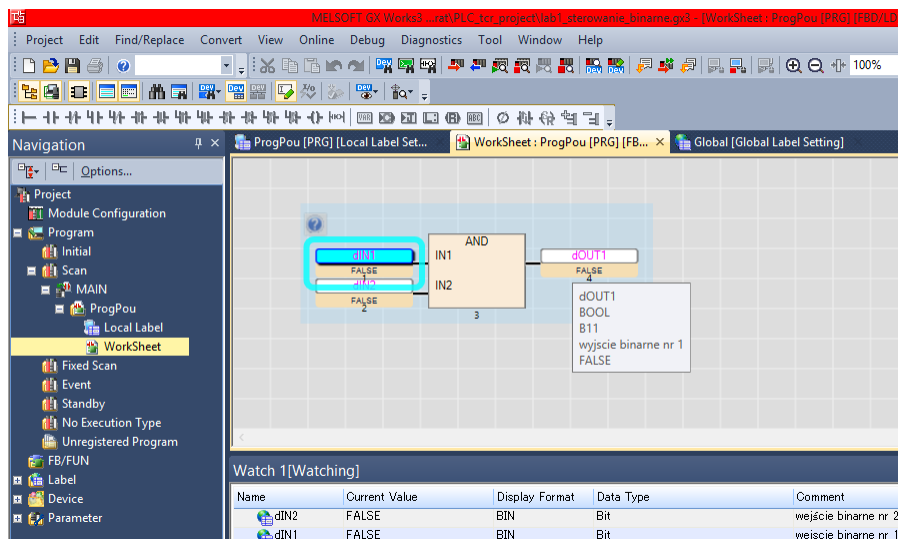
Uwaga 3: Po wykonaniu operacji wgrania parametrów i programu należy wykonać sprzętowy RESET sterownika PLC. Wykonuje się to przez otwarcie pokrywki po lewej stronie sterownika, przełączenie dźwigienki z pozycji RUN do RESET, przytrzymanie dźwigienki do momentu pojawienia się diody ERR na sterowniku a następnie powrót do pozycji RUN. W tym momencie sterownik został zresetowany i można kontynuować pracę.

5.7 Diagnostyka, monitorowanie działania programu

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

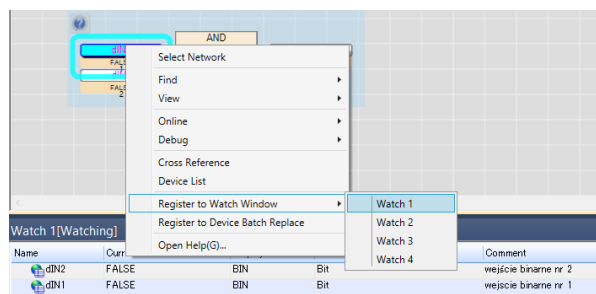


Rysunek 18 Uruchamianie Monitora.



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]. Można również w kolejnych wierszach wpisywać bezpośrednio nazwy zmiennych lub rejestry pamięci sterownika (np. D100, M22, X10, Y2). Z poziomu okienka Watch można zmieniać wartości zmiennych w celu testowania działania programu. Oczywiście w czasie pracy sterownika niektóre zmienne mogą być natychmiast nadpisywane przez program. Dodatkowo warto zwrócić uwagę na kolumnę typu danych. W szczególności jest to istotne, kiedy operujemy na 32 bitowych zmiennych, aby wybrać typ np. Double Word.



Rysunek 20 Uruchomienie Watch'a.

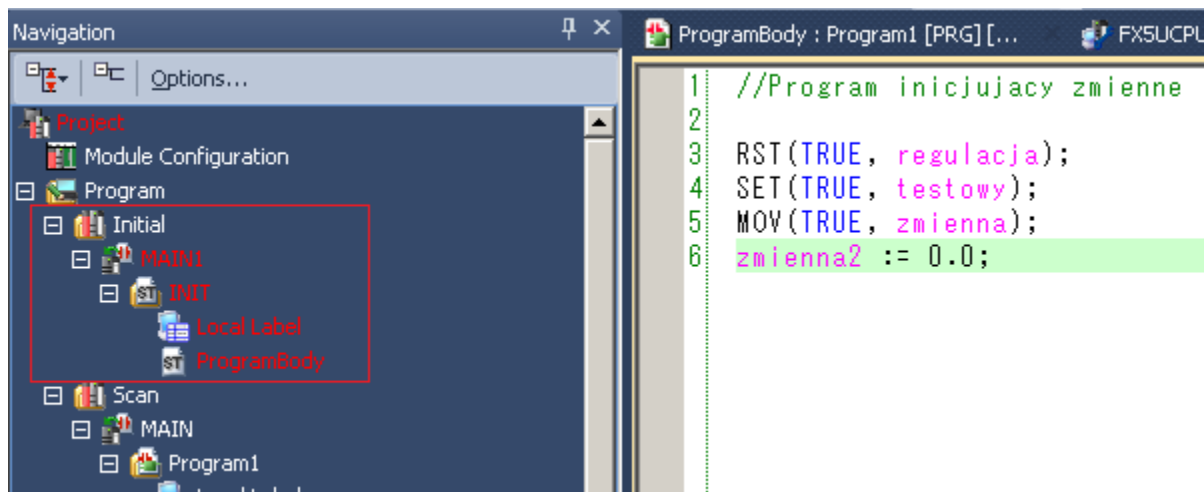
5.8 Pierwszy program PLC

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

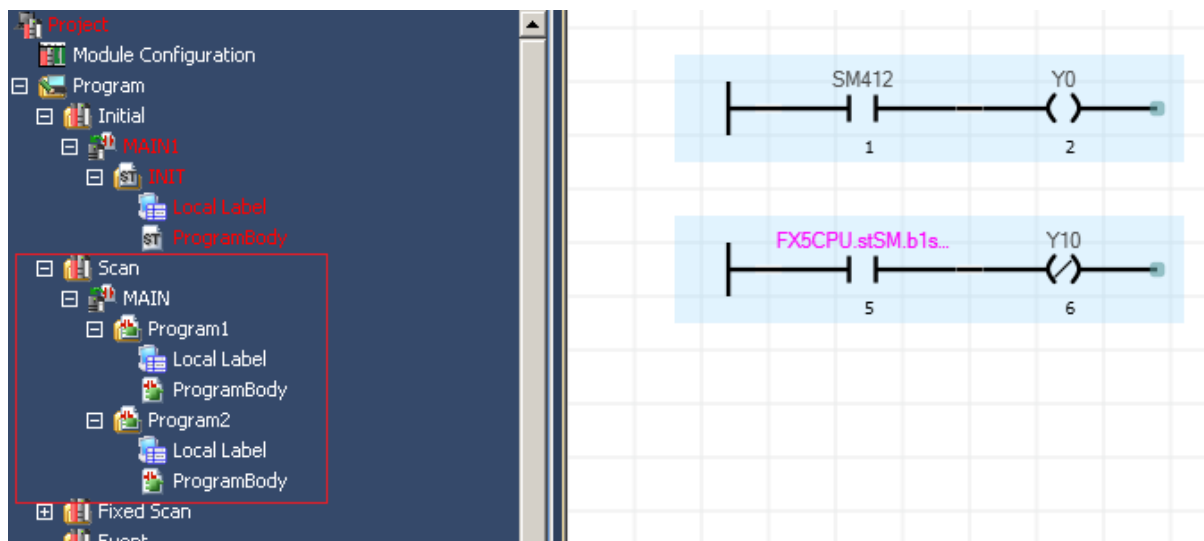
3. dodanie programu w sekcji INIT – inicjalizacja zmiennych dla symulacji procesów i regulatorów
4. dodanie programu w sekcji SCAN – operacje wykonywane cyklicznie ze skanem procesora
5. dodanie programu FIXED SCAN – operacje wykonywane cyklicznie ze skanem 1000ms (czas dyskretyzacji procesów regulacji i regulatorów)

Dodanie programu w sekcji INIT – inicjalizacja zmiennych dla symulacji procesów i regulatorów



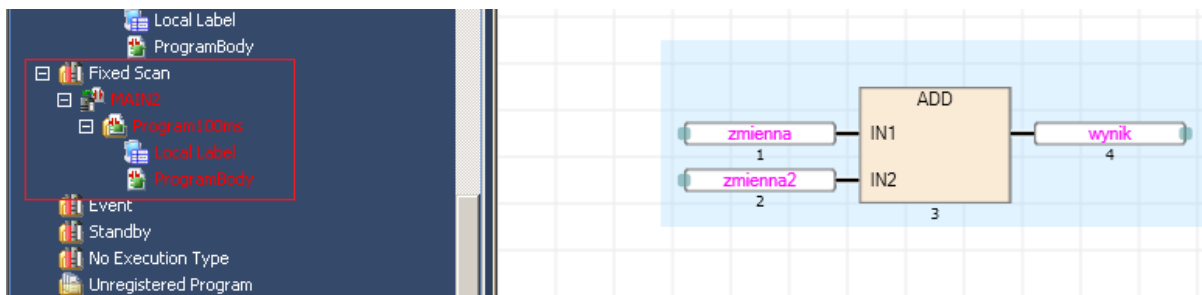
Rys. 5.4: Dodanie programu do sekcji Initial

Dodanie programu w sekcji SCAN – operacje wykonywane cyklicznie ze skanem procesora. W tej grupie może znajdować się kilka programów, które są odpowiedzialne za różne procesy. Należy pamiętać, że programy w czasie jednego skanu PLC są wykonywane jeden po drugim.

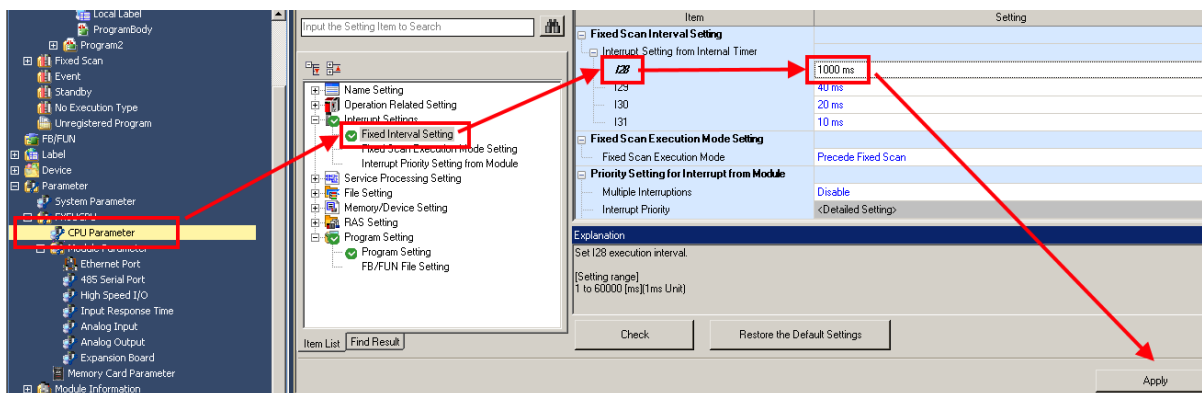


Rys. 5.5: Dodanie programu do sekcji Scan

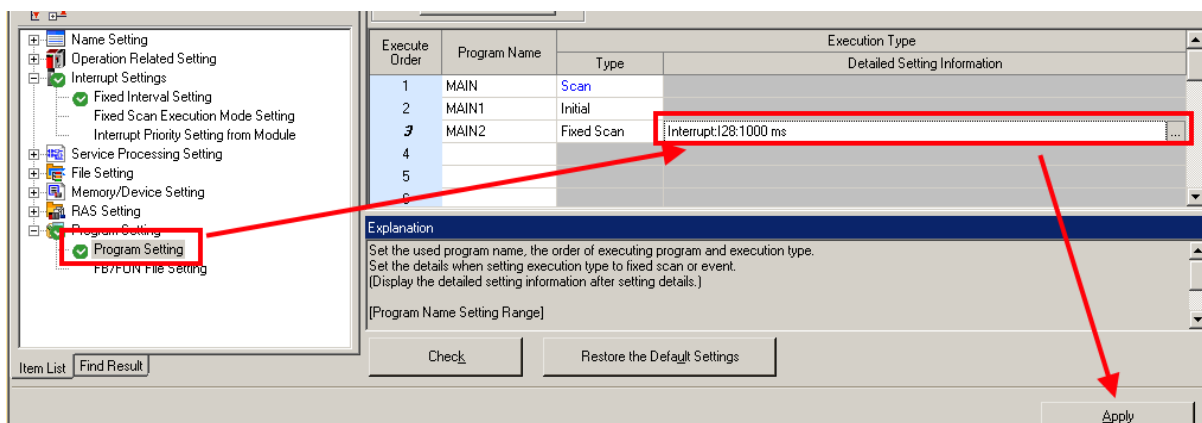
Dodanie programu FIXED SCAN – operacje wykonywane cyklicznie ze skanem 1000ms (czas dyskretyzacji procesów regulacji i regulatorów). Okres próbkowania programu z tej grupy jest ustawiony w parametrach sterownika, co zostało przedstawione poniżej.



Rys. 5.6: Dodanie programu do sekcji Fixed Scan



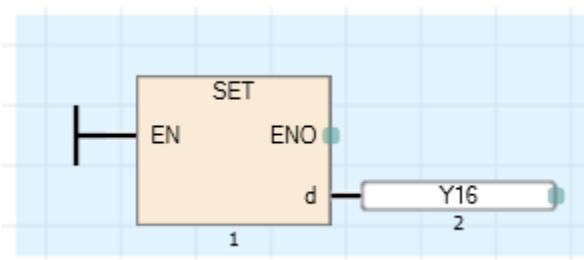
Rys. 5.7: Ustawienia Timera dla programu z grupy Fixed Scan



Rys. 5.8: Ustawienia okresu interwencji dla programu z grupy Fixed Scan

Program przykładowy w sekcji INIT:

FBD:



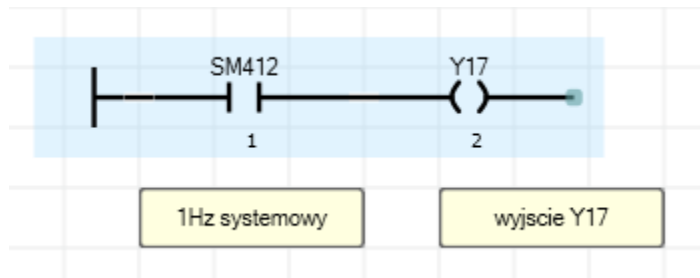
ST:

```
SET(TRUE, Y16);
```

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

Program przykładowy w sekcji SCAN:

FBD:



ST:

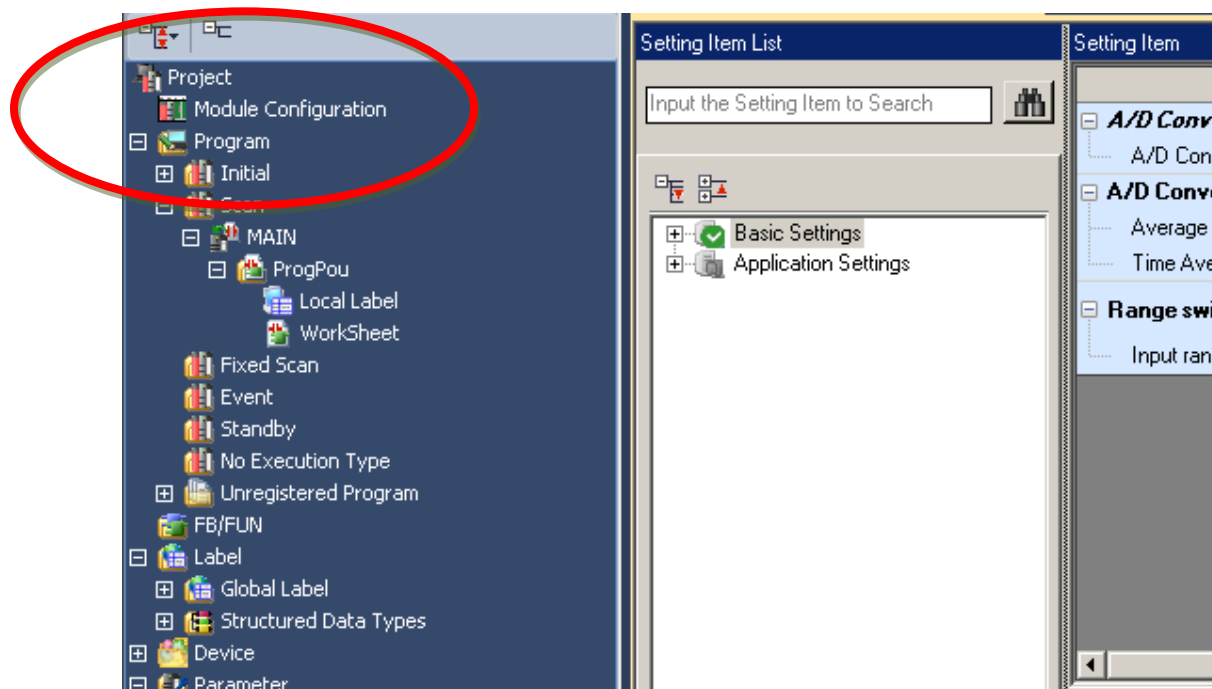
```
OUT(SM412, Y17);
```

Program powinien mrugać wyjściem Y17 zgodnie z zegarem wewnętrznym 1Hz.

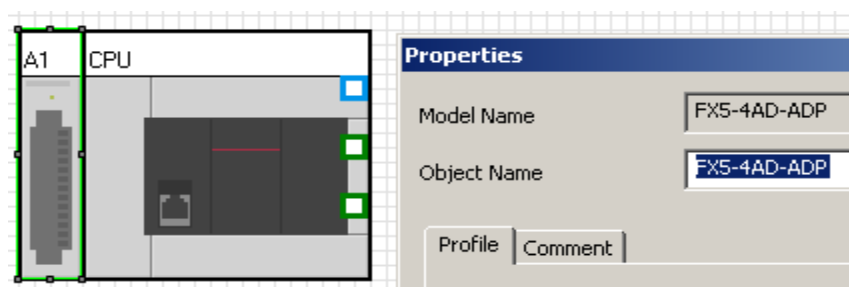
Po utworzeniu programu wstępnego należy go skompilować opcją Rebuild All a następnie wgrać do sterownika. Po wykonaniu restartu sterownika wyjście Y16 powinno się zapalić a wyjście Y17 powinno cyklicznie się zmieniać.

5.9 Przykładowe implementacje interfejsów sprzętowych

Konfigurację modułu analogowe rozpoczynamy od dodania go w sekcji [Project] → [Module Configuration]. Po otwarciu się karty należy z prawej strony z okienka Element Selection przeciągnąć moduł na jego właściwe miejsce. Okienko Element Selection można aktywować z górnego menu [View] → [Docking Window] → [Element Selection]. Po dodaniu modułu należy kliknąć prawym przyciskiem na jednostce CPU (okienko Module Configuration) i wybrać [Parameter] → [Fix]. Po tej operacji moduł jest już prawidłowo dodany do projektu a jego właściwości można modyfikować z sekcji [Navigation] → [Project] → [Parameter] → [Module Information].

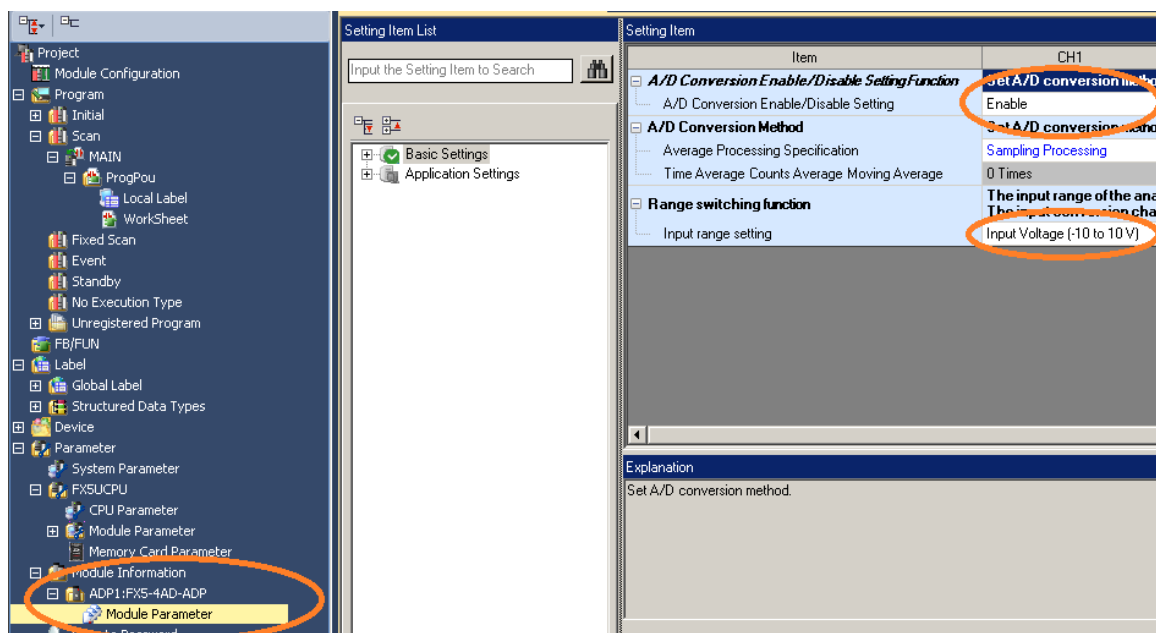


Rys. 5.9: Dodanie zewnętrznego modułu Analogowego - Module Configuration



Rys. 5.10: Dodanie zewnętrznego modułu Analogowego FX5-4AD-ADP

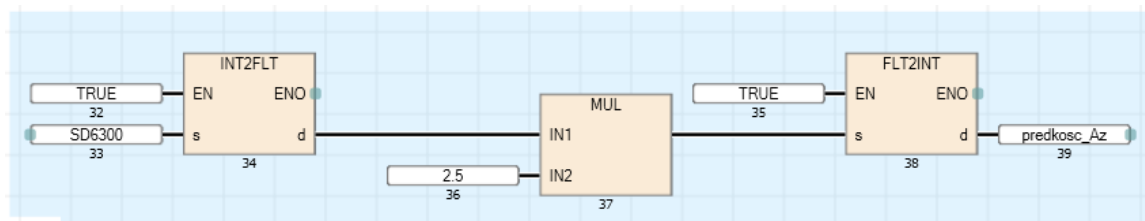
Następnie należy ustawić odpowiednie parametry danego wejścia. Prawidłowa konfiguracja dla kanału pierwszego została przedstawiona poniżej. Należy włączyć odpowiedni kanał przetwarzania analogowo cyfrowego i właściwy zakres napięcia wejściowego -10V do +10V.



Rys. 5.11: Konfiguracja zewnętrznego modułu Analogowego FX5-4AD-ADP

Przykład programu PLC do odczytu wartości cyfrowej z przetwornika C/A został przedstawiony poniżej. Rejestr specjalny sterownika PLC o nazwie 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.

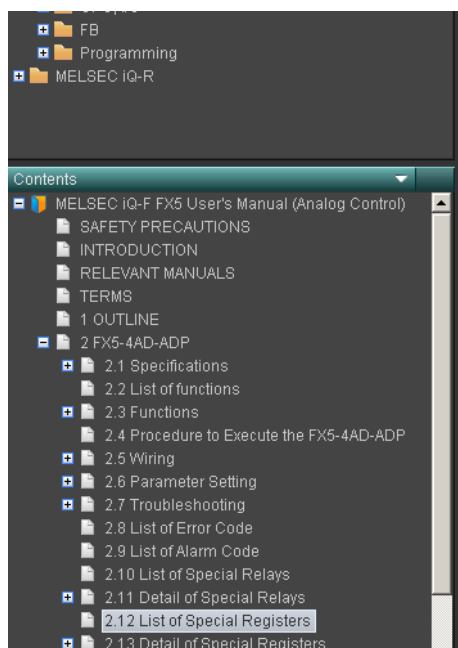
FBD:



ST:

```
predkosc_Az := REAL_TO_INT ( 2.5 * INT_TO_REAL (SD6300) );
```

Wyciąg z programu E-Manual Viewer.



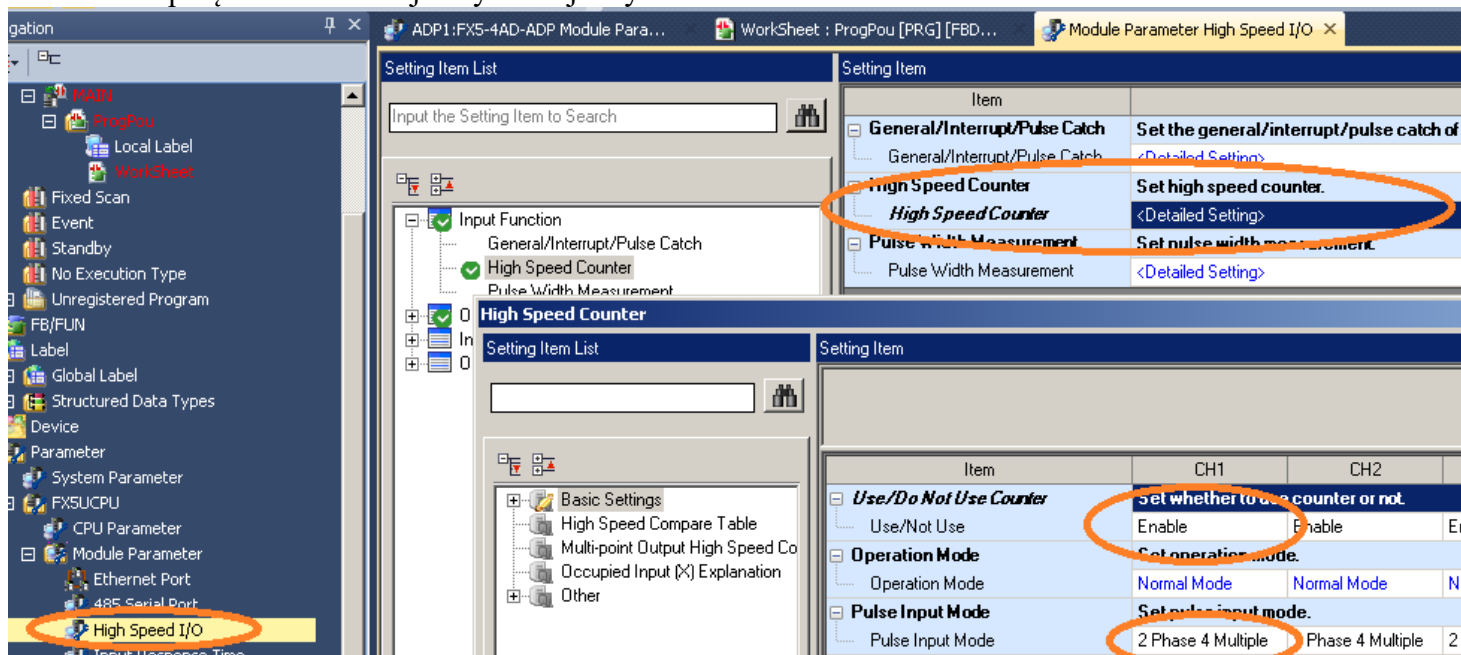
The special registers list for the 1st FX5-4AD-ADP module is shown below.

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

Rys. 5.12: Adres wartości z modułu analogowego w oknie E-Manual Viewer

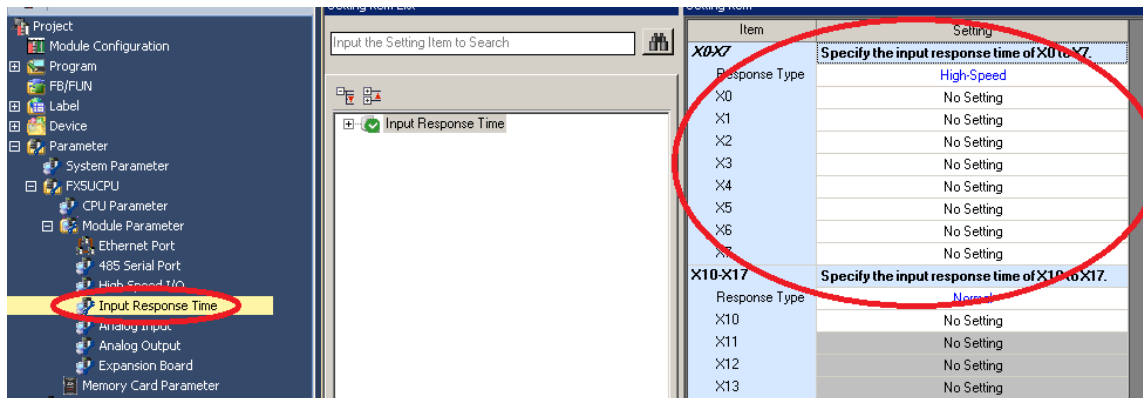
5.9.1 Wejście licznikowe – pomiar pozycji z enkodera inkrementalnego

Wejście licznikowe jest fizycznym wejściem cyfrowym sterownika. Aby dane wejście pracowało jako szybki licznik należy przeprowadzić konfigurację parametrów przedstawioną poniżej. Enkodery inkrementalne 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”. Oczywiście w pierwszym przypadku możliwe jest sprzętowe zliczanie pozycji w obie strony, natomiast w drugim przypadku zliczanie sprzętowe możliwe jest tylko w jednym kierunku.

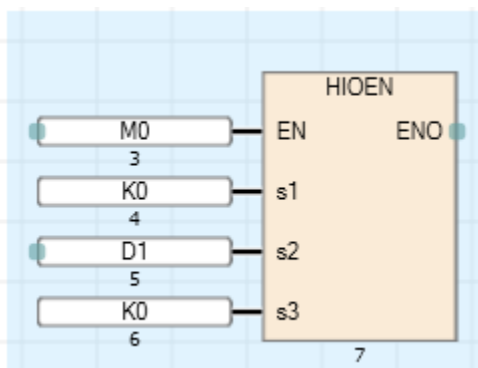


Rys. 5.13: Konfiguracja modułu High Speed I/O

Istotnym parametrem jest wyłączenie filtrów wejściowych, aby uzyskać prawidłowe pomiary. Wynika to z szybkości wysyłanych impulsów przez enkoder. Przy zbyt dużych częstotliwościach filtry mogą wyciąć prawidłowe informacje o pozycji.

**Rys. 5.14: Wyłączenie filtrów wejściowych**

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.

FBD:**ST:**

```

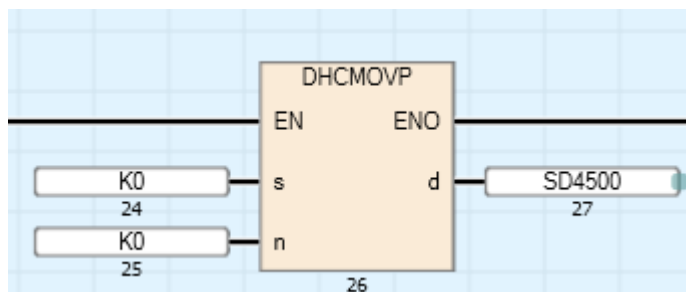
MOV(TRUE, K1, D0);
HIOEN( M0 , K0, D0 , K0 );
  
```

Aby odczytywać wartość aktualną licznika należy odszukać w dokumentacji, który rejestr specjalny jest powiązany z kanałem pierwszym – w naszym przypadku będzie to SD4500.

<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) 	<table> <tr><td>SD4500</td><td>High-speed counter current value (CH1)</td></tr> <tr><td>SD4501</td><td></td></tr> <tr><td>SD4502</td><td>High-speed counter maximum value (CH1)</td></tr> <tr><td>SD4503</td><td></td></tr> <tr><td>SD4504</td><td>High-speed counter minimum value (CH1)</td></tr> <tr><td>SD4505</td><td></td></tr> <tr><td>SD4506</td><td>High-speed counter pulse density (CH1)</td></tr> <tr><td>SD4507</td><td></td></tr> <tr><td>SD4508</td><td>High-speed counter rotational speed (CH1)</td></tr> <tr><td>SD4509</td><td></td></tr> <tr><td>SD4510</td><td>High-speed counter preset control switch (CH1)</td></tr> <tr><td>SD4511</td><td>Not used</td></tr> <tr><td>SD4512</td><td>High-speed counter preset value (CH1)</td></tr> <tr><td>SD4513</td><td></td></tr> <tr><td>SD4514</td><td>High-speed counter ring length (CH1)</td></tr> <tr><td>SD4515</td><td></td></tr> <tr><td>SD4516</td><td>High-speed counter measurement unit time (CH1)</td></tr> <tr><td>SD4517</td><td></td></tr> <tr><td>SD4518</td><td>High-speed counter number of pulses per rotation (CH1)</td></tr> <tr><td>SD4519</td><td></td></tr> <tr><td>SD4520 to SD4529</td><td>Not used</td></tr> </table>	SD4500	High-speed counter current value (CH1)	SD4501		SD4502	High-speed counter maximum value (CH1)	SD4503		SD4504	High-speed counter minimum value (CH1)	SD4505		SD4506	High-speed counter pulse density (CH1)	SD4507		SD4508	High-speed counter rotational speed (CH1)	SD4509		SD4510	High-speed counter preset control switch (CH1)	SD4511	Not used	SD4512	High-speed counter preset value (CH1)	SD4513		SD4514	High-speed counter ring length (CH1)	SD4515		SD4516	High-speed counter measurement unit time (CH1)	SD4517		SD4518	High-speed counter number of pulses per rotation (CH1)	SD4519		SD4520 to SD4529	Not used
SD4500	High-speed counter current value (CH1)																																										
SD4501																																											
SD4502	High-speed counter maximum value (CH1)																																										
SD4503																																											
SD4504	High-speed counter minimum value (CH1)																																										
SD4505																																											
SD4506	High-speed counter pulse density (CH1)																																										
SD4507																																											
SD4508	High-speed counter rotational speed (CH1)																																										
SD4509																																											
SD4510	High-speed counter preset control switch (CH1)																																										
SD4511	Not used																																										
SD4512	High-speed counter preset value (CH1)																																										
SD4513																																											
SD4514	High-speed counter ring length (CH1)																																										
SD4515																																											
SD4516	High-speed counter measurement unit time (CH1)																																										
SD4517																																											
SD4518	High-speed counter number of pulses per rotation (CH1)																																										
SD4519																																											
SD4520 to SD4529	Not used																																										

Rys. 5.15: Adres wartości z modułu szybkich liczników w oknie E-Manual Viewer

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. Procedurę tą należy wykonać po ustawieniu elementu wykonawczego w miejscu bazowym. W przypadku zestawu TCRANE należy wykorzystać krańcówki sprzętowe do wykonania bazowania. W tym celu należy zachować szczególną ostrożność w pisaniu aplikacji, ponieważ element jeżdżący powinien bezzwłocznie się zatrzymać po najechaniu na taką krańcówkę i dopiero później powinna być wykonana procedura zerowania licznika.

FBD:**ST:**

```
DHCMOVP(M55, 0, 0, SD4500);
```

5.9.2 Wejście licznikowe – pomiar częstotliwości

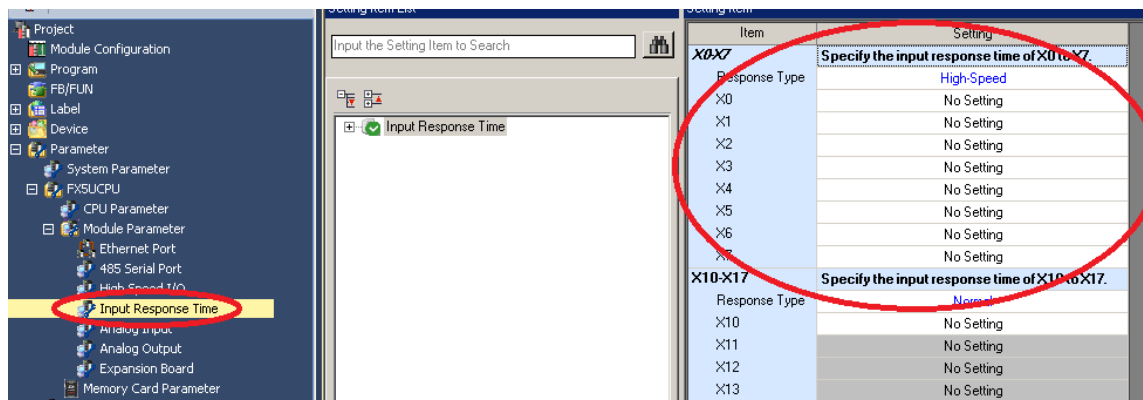
Konfiguracja wejścia licznikowego w trybie pomiaru częstotliwości jest analogiczna do powyższego ustawienia w trybie pomiaru pozycji. Poniżej przedstawiono odpowiednie ustawienie parametrów. Istotnym parametrem jest wyłączenie filtrów wejściowych, aby uzyskać prawidłowe pomiary.

Item	CH1	CH2
Use/Do Not Use Counter	Set whether to use counter or not.	
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

Rys. 5.16: Konfiguracja modułu szybkich liczników do pomiaru sygnału częstotliwościowego

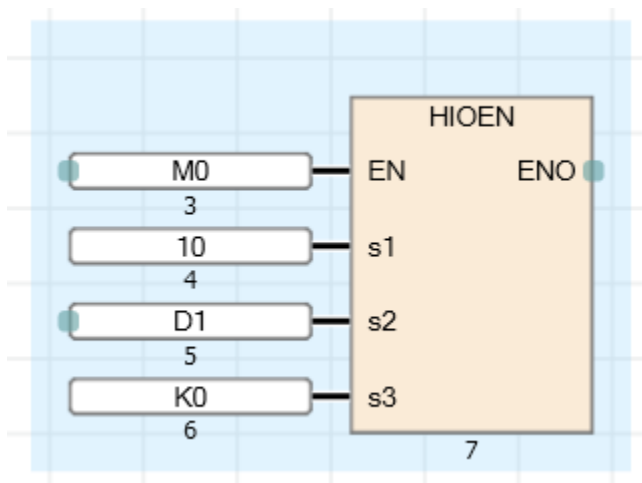
Ważne jest ustawienie czasu pomiaru – to ustawienie jest związane z rozdzielczością pomiaru częstotliwości.

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



Rys. 5.17: Wyłączenie filtrów wejściowych

FBD:



ST:

DHCMOVP(M55, 0, 0, SD4500);

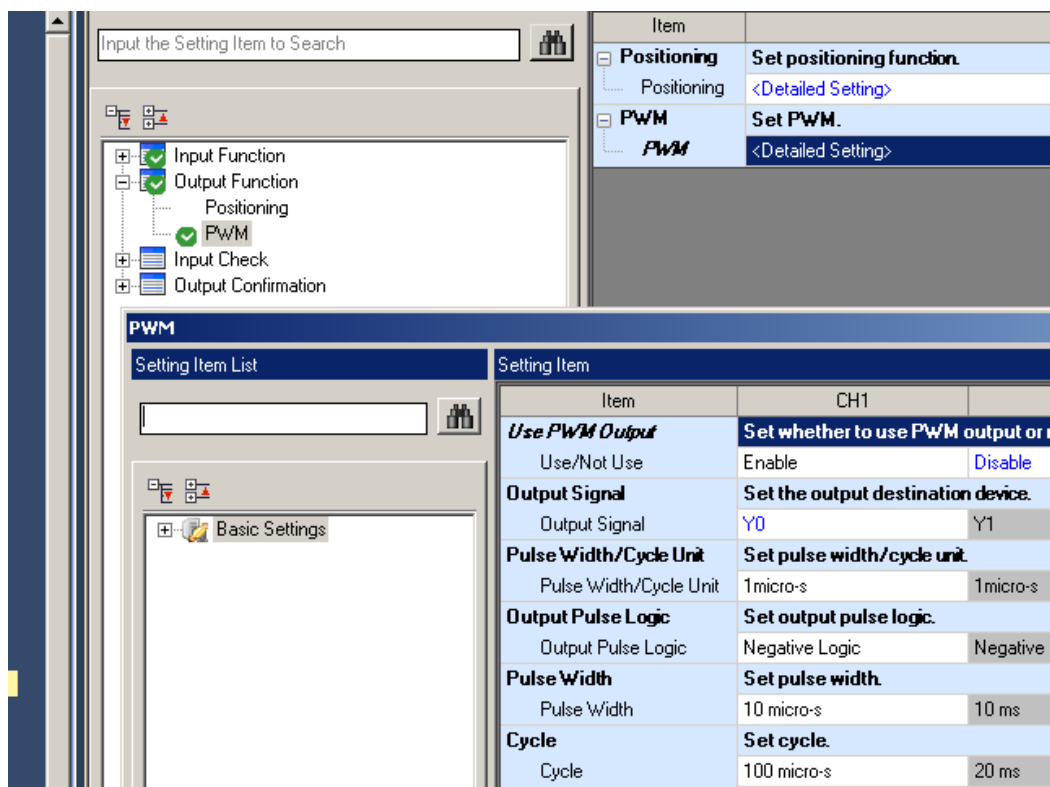
Wartość mierzonej częstotliwości będzie w rejestrze SD4506 (i SD4507 – należy uważać, które wartości są zwracane jako 16 bit czy 32 bit).

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

Rys. 5.18: Adres wartości z modułu szybkich liczników - pomiar częstotliwości w oknie E-Manual Viewer

5.9.3 Wyjście cyfrowe PWM

Ustawienie wyjścia PWM zaczynamy od parametrów.

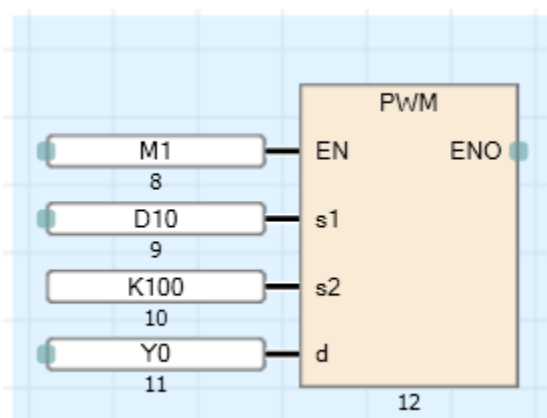


Rys. 5.19: Ustawienie sygnału PWM na wyjściu Y0

Powyższe ustawienie dotyczy wyjścia Y0. Okres sygnału PWM wynosi 100us, czyli odpowiada to częstotliwości 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. Jeżeli reakcja silnika będzie odwrotna należy zmienić opcję „Output Pulse Logic” na przeciwną.

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.

FBD:



ST:

PWM (M1, D10, K100, Y0) ;

5.9.4 Socket Communication – wysyłanie danych do MATLAB

Konfiguracja parametrów komunikacji została przedstawiona w rozdziale 5.2.9. Należy pamiętać o dodaniu opcji Socket Communication i ustawieniu odpowiedniego portu komunikacyjnego.

Kolejnym etapem jest dodanie programu w grupie SCAN o nazwie np. „Socket”.

Następnie należy dodać w zmiennych lokalnych tego programu następujące zmienne.

Lokalne

	Label Name	Data Type
1	sent	Bit
2	not_sent	Bit
3	auto_send	Bit
4		

Rys. 5.20: Zmienne lokalne

W zmiennych globalnych należy dodać poniższe zmienne do przechowywania łańcucha znaków.

Globalne

	Label Name	Data Type	Class	Assign (Device/Label)
1	string_len	Word [Signed]	VAR_GLOBAL	D3000
2	send_string	String(32)	VAR_GLOBAL	D3001
3	temp_string	String(32)	VAR_GLOBAL	
4				

Rys. 5.21: Zmienne globalne

W programie wykonywanym cyklicznie (program regulatora) należy wpisać kod, który przygotowuje ramkę łańcucha znaków do wysłania przez Ethernet do MATLAB.

//Generacja tesktu do wysłania przez socket communication

temp_string := 'U=';

temp_string := CONCAT(temp_string, REAL_TO_STRING(PID3.MV));

```

temp_string := CONCAT(temp_string, 'Y=');
temp_string := CONCAT(temp_string, REAL_TO_STRING(PID3.PV));
temp_string := CONCAT(temp_string, '$L');
send_string := temp_string;
//Długość tekstu
string_len := LEN(send_string);
SET(TRUE, UDP_Send_TRIG);

```

Po uzupełnieniu ciągu znaków należy uruchomić wysyłanie przy pomocy flagi UDP_Send_TRIG. Należy zwrócić uwagę na sposób konwersji REAL_TO_STRING. W przypadku regulatora PID3, jego zmienne są typu FLOAT. Jeżeli będziemy chcieli wysłać wartości regulatora wbudowanego PID (które są liczbami typu INT) należy użyć instrukcji konwersji INT_TO_STRING.

Komunikacja odbywa się przy użyciu funkcji SP_SOCSND, której przykładową implementację przedstawiono poniżej. Fragment programu musi być umieszczony w grupie programów typu SCAN. Na początku resetowane są znaczniki sukcesu wysłania lub porażki. Następnie właściwa funkcja, która jest uruchomiona pod wpływem zbocza narastającego sygnału UDP_Send_TRIG i obecności sygnału SD10680.0, który mówi o prawidłowym połączeniu z serwerem. Wynik operacji wysłania jest umieszczony w bitach pamięci M300 i M301. Jeżeli flaga sukcesu wysłania 'sent' zostanie ustawiona nastąpi automatyczne zresetowanie flagi UDP_Send_TRIG, dzięki czemu będzie możliwe powtórne wysłanie wiadomości w kolejnej iteracji algorytmu regulacji.

```

RST(LDP(TRUE, UDP_Send_TRIG), sent);
RST(LDP(TRUE, UDP_Send_TRIG), not_sent);
SP_SOCSND( LDP(TRUE, UDP_Send_TRIG) AND SD10680.0 , 'U0' , K1 , D2990 , D3000,
M300 );
SET( LDP(TRUE, M300) AND NOT M301, sent);
SET( LDP(TRUE, M300) AND M301, not_sent);

IF sent THEN
    RST(TRUE, UDP_Send_TRIG);
END_IF;

```

W programie PLC wysyłanie będzie realizowane cyklicznie zgodnie z wykonaniem programu regulatora – w jego końcowej części. Wysyłanie będzie możliwe dopiero po nawiązaniu komunikacji z serwerem, która musi zostać zainicjowana po stronie środowiska MATLAB. Przykładowa realizacja skryptu MATLAB do nawiązania połączenia i następnie cyklicznego rysowania danych na wykresie została przedstawiona poniżej. Proszę zwrócić uwagę, aby nie nadpisywać w programie rejestrów D2990 do D2999, które służą do konfiguracji wysyłania. Domyślnie te rejestry nie powinny być zmieniane.

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 done. ');
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 done. ');
        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(0.01);
end

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

```

UWAGA: Aby operacje składania tekstu działały prawidłowo należy wyłączyć funkcję optymalizacji kodu i przestrzeni zmiennych oraz etykiet. W tym celu należy przejść do Tool -> Options -> Convert -> Basic Setting. Poniżej znajduje się okno z prawidłową konfiguracją, którą należy ustawić.

Program Check		
Execute Program Check after Build or Online Program Change	No	▼
Target the SET instruction for duplicated coil check	Yes	▼
Operational Setting		
Use the Same Label Name in Global Label and Local Label	Min	▼
Optimize the Number of Steps.	NO	▼
Collectively Allocate Temporary Area to Optimize the Number of Steps	Yes	▼
Reassign Labels in Executing Rebuild All	No	▼
Check the data type of instruction argument	No	▼
Language for Instruction Conversion of Character String Operation	User Locale	▼
Convert, Online Program Change Target Setting	Low-speed	▼
Function Block		
Enable to Use MC/MCR to Control EN	No	▼
Conversion Operation		
Enable Rebuild All (Reassignment)	Yes	▼
Enable Rebuild All (Retain)	Yes	▼
Enable Conversion	Yes	▼

5.10 Przykład 1 – logika sterowania binarna

5.10.1 Etap projektowania PLC

W rozdziale tym przedstawiona zostanie realizacja prostego automatu sekwencyjnego. Jego zastosowanie ma szerokie horyzonty z uwagi na łatwą implementację i naturalne odwzorowanie zadanego cyklu maszyny czy procesu. Każdy stan posiada warunki przejścia do kolejnego stanu a także operacje realizowane w danym stanie. Przechodzenie między kolejnymi stanami może być wykonane automatycznie lub można wybrać tryb krokowy działania automatu i wówczas przejścia do kolejnych stanów są realizowane przez użytkownika. Naturalnie wybór kolejnego stanu jest realizowany w sposób programowy na podstawie warunków przejścia a użytkownik wyznacza tylko moment, w którym to przejście następuje.

Pierwszym krokiem do realizacji przykładu jest stworzenie nowych zmiennych i przydzielenie do nich odpowiednich obszarów pamięci. Zmienna Stan_aktualny będzie pokazywała, w jakim aktualnie stanie się znajdujemy, zmienna Stan_nastepny będzie pokazywała do jakiego stanu nastąpi przejście, bit Automat_krokowy służy do wyboru normalnego trybu pracy lub trybu krokowego i ostatni bit Automat_krok będzie służył do wyzwolenia przejścia automatu do kolejnego stanu.

Label				
Global Label				
1	Stan_aktualny	Word [Signed]	...	D100
2	Stan_nastepny	Word [Signed]	...	D101
3	Automat_krokowy	Bit	...	M50
4	Automat_krok	Bit	...	M51
5			...	

Po deklaracji zmiennych należy przygotować program, który będzie oparty na instrukcji warunkowej CASE. Początkowy fragment pozwala na wybór automatu krokowego i obsługę przejść do kolejnych stanów. Następnie kolejne stany zostały przedstawione w postaci cyfr od 0 do n, wybieranych przy pomocy instrukcji CASE.

Warunki przejść zostały zapisane jako instrukcje IF ... THEN. Każdy stan może zawierać instrukcje przypisania, ustawienia SET, skasowania RESET. Należy tylko pamiętać, że instrukcje te będą realizowane tylko podczas realizacji danego stanu aktualnego.

```
IF Automat_krokowy AND LDP(TRUE, Automat_krok) THEN
    Stan_aktualny := Stan_nastepny;
ELSE
    IF NOT Automat_krokowy THEN
        Stan_aktualny := Stan_nastepny;
    END_IF;
END_IF;
```

```
IF Stan_aktualny = Stan_nastepny THEN
    CASE Stan_aktualny OF
```

```
    0 :
        IF M610 THEN
            Stan_nastepny:= 1;
            RST(TRUE, M610);
        END_IF;
        IF M618 THEN
            Stan_nastepny:= 3;
            RST(TRUE, M618);
        END_IF;
```

```
    1:
        IF M611 THEN
            Stan_nastepny:= 2;
            RST(TRUE, M611);
        END_IF;
        MOV(TRUE,2,D610);
```

```
    2:
        SET(TRUE,M650);
        IF M612 THEN
            Stan_nastepny:= 3;
            RST(TRUE, M612);
            RST(TRUE,M650);
        END_IF;
```

```
    IF M615 THEN
        Stan_nastepny:= 0;
        RST(TRUE, M615);
        RST(TRUE,M650);
    END_IF;
    MOV(TRUE,1,D610);
```

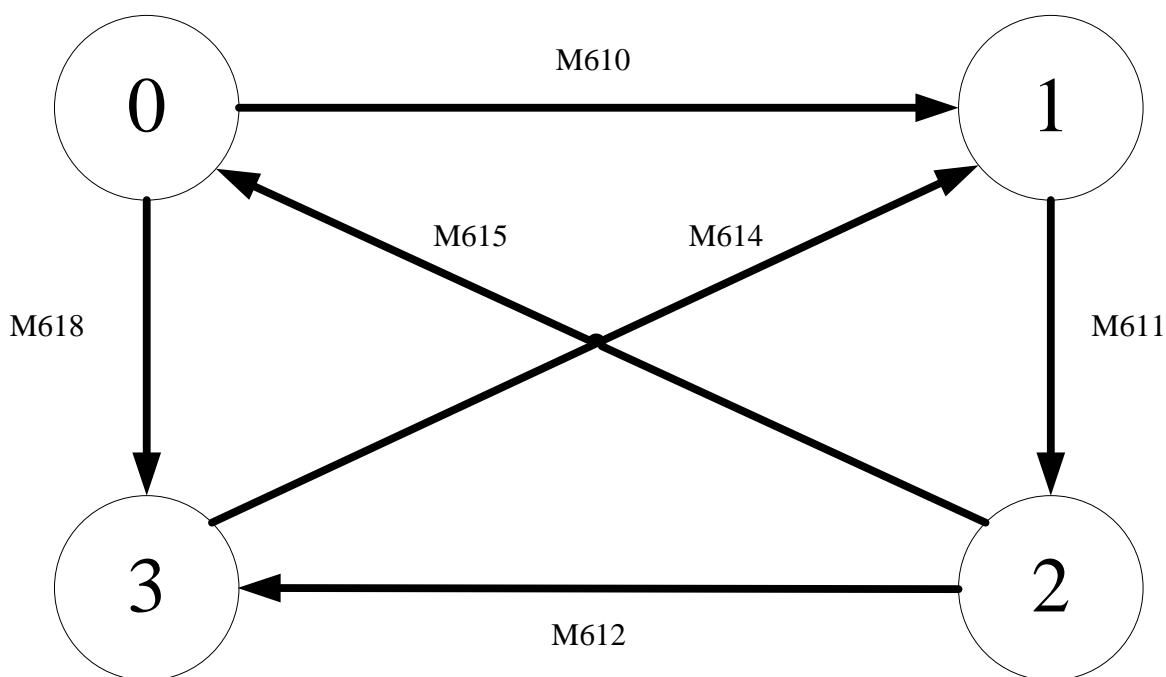
```

3:
IF M614 THEN
    Stan_nastepny:= 1;
    RST(TRUE, M614);
END_IF;
MOV(SM412,2,D610);
MOV(NOT SM412, 0, D610);
OUT(SM412,M652);

END_CASE;
END_IF;

```

Przedstawiony program będzie realizował automat stanów przedstawiony na rysunku poniżej.



Rys. 5.22: Graf przejść

5.11 Przykład 2 – logika sterowania ciągłego

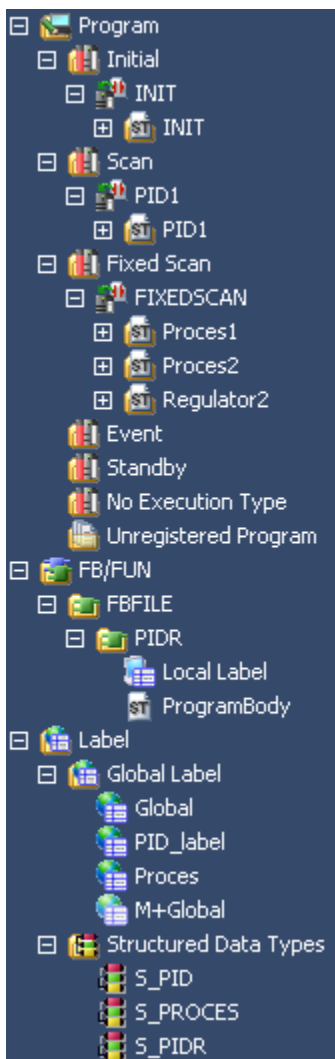
5.11.1 Etap projektowania PLC

W tej części zostanie dokładnie omówiona część programowa sterownika PLC, która pozwoli na symulację działania dwóch procesów inercji pierwszego rzędu.

Następnie zostaną przygotowane regulatory (PID wbudowany w PLC, PID z równania różnicowego).

Pierwsze kroki pracy z programem GXWorks3 zostały wykonane w powyższych rozdziałach. Należy zweryfikować poprawność parametrów z powyższych rozdziałów, dodać diagnostyczne mruganie diodą wyjściową oraz uzupełnić do końca programy symulacji procesów, regulatorów i przestrzeni zmiennych globalnych.

Poniżej przedstawiona została struktura programów, grup zmiennych oraz struktur potrzebnych do realizacji przykładu. W sekcji programów INIT znajduje się jeden program, który ustawia parametry procesów i regulatorów. W sekcji programów SCAN znajduje się program realizujący zadanie wbudowanego w PLC regulatora PID. Sekcja programów FIXED SCAN zawiera programy odpowiedzialne za cykliczne wykonywanie symulacji procesu 1 i 2 oraz regulację PID przy użyciu równania różnicowego. W sekcji FB/FUN widnieje biblioteka PIDR, która wykonuje obliczenia w zaimplementowanym regulatorze PID. W sekcji zmiennych znajdziemy trzy grupy potrzebne do realizacji przykładu: Global, PID_label, Proces. Dla wygody pisania programu zastosowano również struktury danych S_PID, S_PROCES, S_PIDR.



Poniższy rysunek przedstawia elementy i ich typy danych dla struktury S_PID.

	Label Name	Data Type
1	SV	Word [Signed]
2	PV	Word [Signed]
3	MV	Word [Signed]
4	params	Word [Unsigned]/Bit String [16-bit][0..29]
5	Control_ON	Bit

Poniższy rysunek przedstawia elementy i ich typy danych dla struktury S_PROCES.

	Label Name	Data Type
1	sampling_time	FLOAT [Single Precision]
2	Time_const	FLOAT [Single Precision]
3	Alfa	FLOAT [Single Precision]
4	Gain	FLOAT [Single Precision]
5	u_k	FLOAT [Single Precision]
6	u_k_1	FLOAT [Single Precision]
7	y_k	FLOAT [Single Precision]
8	y_k_1	FLOAT [Single Precision]
9	A_p	FLOAT [Single Precision]

Poniższy rysunek przedstawia elementy i ich typy danych dla struktury S_PIDR.

	Label Name	Data Type
1	SV	FLOAT [Single Precision]
2	PV	FLOAT [Single Precision]
3	MV	FLOAT [Single Precision]
4	K_gain	FLOAT [Single Precision]
5	Tl	FLOAT [Single Precision]
6	TD	FLOAT [Single Precision]
7	Rp0	FLOAT [Single Precision]
8	Rp1	FLOAT [Single Precision]
9	Rp2	FLOAT [Single Precision]
10	Ep0	FLOAT [Single Precision]
11	Ep1	FLOAT [Single Precision]
12	Ep2	FLOAT [Single Precision]
13	Control_ON	Bit
14	sampling_time	FLOAT [Single Precision]

Poniższy rysunek przedstawia deklaracje zmiennej Proces1 w grupie Proces.

	Label Name	Data Type		Class	Assign (Device/Label)
1	Proces1	S_PROCES	...	VAR_GLOBAL	Detailed Setting
2	Proces2	S_PROCES	...	VAR_GLOBAL	Detailed Setting
3			...		

Structure Device Setting Window

Proces1 (S_PROCES)

	Label Name	Data Type	Device
1	sampling_time	FLOAT [Single Precision]	D2000
2	Time_const	FLOAT [Single Precision]	D2002
3	Alfa	FLOAT [Single Precision]	D2004
4	Gain	FLOAT [Single Precision]	D2006
5	u_k	FLOAT [Single Precision]	D2008
6	u_k_1	FLOAT [Single Precision]	D2010
7	y_k	FLOAT [Single Precision]	D2012
8	y_k_1	FLOAT [Single Precision]	D2014
9	A_p	FLOAT [Single Precision]	D2016

Poniższy rysunek przedstawia deklaracje zmiennej Proces2 w grupie Proces.

	Label Name	Data Type		Class	Assign (Device/Label)
1	Proces1	S_PROCES	...	VAR_GLOBAL	Detailed Setting
2	Proces2	S_PROCES	...	VAR_GLOBAL	Detailed Setting
3			...		

Structure Device Setting Window

Proces2 (S_PROCES)

	Label Name	Data Type	Device
1	sampling_time	FLOAT [Single Precision]	D2020
2	Time_const	FLOAT [Single Precision]	D2022
3	Alfa	FLOAT [Single Precision]	D2024
4	Gain	FLOAT [Single Precision]	D2026
5	u_k	FLOAT [Single Precision]	D2028
6	u_k_1	FLOAT [Single Precision]	D2030
7	y_k	FLOAT [Single Precision]	D2032
8	y_k_1	FLOAT [Single Precision]	D2034
9	A_p	FLOAT [Single Precision]	D2036

Poniższy rysunek przedstawia deklaracje zmiennej PID1 w grupie PID_label.

	Label Name	Data Type		Class	Assign (Device/Label)
1	PID1	S_PID	...	VAR_GLOBAL	Detailed Setting
2	PID2	S_PIDR	...	VAR_GLOBAL	Detailed Setting
3			...		

Structure Device Setting Window

PID1 (S_PID)

	Label Name	Data Type	Device
1	SV	Word [Signed]	D1000
2	PV	Word [Signed]	D1001
3	MV	Word [Signed]	D1002
4	params	Word [Unsigned]/Bit String [16-bit][0..29]	D1003
5	Control_ON	Bit	D1033.0

Poniższy rysunek przedstawia deklaracje zmiennej PID2 w grupie PID_label.

	Label Name	Data Type	Class	Assign (Device/Label)
1	PID1	S_PID	VAR_GLOBAL	Detailed Setting
2	PID2	S_PIDR	VAR_GLOBAL	Detailed Setting
3				

Structure Device Setting Window

PID2 (S_PIDR)

	Label Name	Data Type	Device
1	SV	FLOAT [Single Precision]	D1100
2	PV	FLOAT [Single Precision]	D1102
3	MV	FLOAT [Single Precision]	D1104
4	K_gain	FLOAT [Single Precision]	D1106
5	TI	FLOAT [Single Precision]	D1108
6	TD	FLOAT [Single Precision]	D1110
7	Rp0	FLOAT [Single Precision]	D1112
8	Rp1	FLOAT [Single Precision]	D1114
9	Rp2	FLOAT [Single Precision]	D1116
10	Ep0	FLOAT [Single Precision]	D1118
11	Ep1	FLOAT [Single Precision]	D1120
12	Fn2	FLOAT [Single Precision]	D1122

Deklaracje zmiennych w grupie Global pozostają do implementacji w ramach dalszych prac nad przykładowym projektem. W tej chwili grupa może pozostać pusta.

Poniżej przedstawiono konfigurację uruchamiania programów (CPU Parameter → Program Setting)

Execute Order	Program Name	Type	
		Type	
1	PID1	Scan	
2	FIXEDSCAN	Fixed Scan	Interrupt:128:1000 ms
3	INIT	Initial	

Przedstawienie przykładowych rozwiązań programów zaczniemy od bloku funkcyjnego PIDR zdefiniowanego w grupie FB/FUN. Poniżej znajduje się deklaracja zmiennej lokalnej wymaganej do pracy ze strukturą danych regulatora.

	Label Name	Data Type	Class
1	PIDR_s	S_PIDR	VAR_IN_OUT

Kod źródłowy bloku funkcyjnego znajduje się poniżej.

//Regulator PID na podstawie rownania roznicowego

IF PIDR_s.Control_ON THEN

 //Wylczenie parametrow

 PIDR_s.Rp0 :=

 PIDR_s.K_gain*(1.0+(PIDR_s.sampling_time/(2.0*PIDR_s.TI))+PIDR_s.TD/PIDR_s.sampling_time); //r0 = K*(1+(Tp/(2*Ti))+Td/Tp);

```

    PIDR_s.Rp1 := PIDR_s.K_gain*((PIDR_s.sampling_time/(2.0*PIDR_s.TI))-
(2.0*PIDR_s.TD/PIDR_s.sampling_time)-1);//r1 = K*( (Tp/(2*Ti))-(2*Td/Tp)-1 );
    PIDR_s.Rp2 := PIDR_s.K_gain*PIDR_s.TD/PIDR_s.sampling_time;//K*Td/Tp;

    //Wyliczenie uchybu regulacji i przesunięcie historii
    PIDR_s.Ep2 := PIDR_s.Ep1;
    PIDR_s.Ep1 := PIDR_s.Ep0;
    PIDR_s.Ep0 := PIDR_s.SV - PIDR_s.PV;

    //Obliczenie sterowania
    PIDR_s.MV := PIDR_s.Rp2*PIDR_s.Ep2 + PIDR_s.Rp1*PIDR_s.Ep1 +
PIDR_s.Rp0*PIDR_s.Ep0 + PIDR_s.MV;//u = R2*E2 + R1*E1 + R0*E0 + u;

    //ANTI WIND UP
    IF (PIDR_s.MV > 100.0) THEN
        PIDR_s.MV := 100.0;
    END_IF;

    IF (PIDR_s.MV < 0.0) THEN
        PIDR_s.MV := 0.0;
    END_IF;

END_IF;

```

Blok funkcyjny regulatora został użyty w programie Regulator2 w grupie FIXED SCAN. Poniżej przedstawiono deklarację zmiennych lokalnych programu i jego kod źródłowy.

	Label Name	Data Type	
1	PIDR2	PIDR	...

```
EMOV(PID2.Control_ON, Proces2.y_k ,PID2.PV);
```

```
PIDR2(PIDR_s:= PID2);
```

```
EMOV(PID2.Control_ON, PID2.MV, Proces2.u_k);
```

Regulator wbudowany w sterowniku PLC został wykorzystany w programie PID1 z grupy SCAN. Poniżej przedstawiono kod źródłowy.

```
MOV(PID1.Control_ON, REAL_TO_INT( Proces1.y_k ),PID1.PV);
```

```
PID( PID1.Control_ON, PID1.SV , PID1.PV , PID1.params[0] , PID1.MV);
```

```
EMOV(PID1.Control_ON, INT_TO_REAL(PID1.MV), Proces1.u_k);
```

Symulacja dwóch procesów została umieszczona w programach Proces1 i Proces2 umieszczonych w grupie FIXED SCAN. Odpowiednie kody źródłowe przedstawiono poniżej.

Proces 1

```
Proces1.Alfa := EXP(-Proces1.sampling_time/Proces1.Time_const);
    //Współczynnik Alfa
Proces1.A_p := Proces1.Gain * (1 - Proces1.Alfa);
    //Współczynnik Ap
Proces1.y_k := (Proces1.A_p * Proces1.u_k_1) + (Proces1.Alfa * Proces1.y_k_1);
    //Nowa wartosc wyjscia na podstawie u(k-1), y(k-1)
Proces1.y_k_1:= Proces1.y_k;
    //Zapamiętanie wyliczonej wartosci wyjscia

Proces1.u_k_1:= Proces1.u_k;
    //Zapamiętanie ostatniego sterowania
```

Proces 2

```
Proces2.Alfa := EXP(-Proces2.sampling_time/Proces2.Time_const);
    //Współczynnik Alfa
Proces2.A_p := Proces2.Gain * (1 - Proces2.Alfa);
    //Współczynnik Ap
Proces2.y_k := (Proces2.A_p * Proces2.u_k_1) + (Proces2.Alfa * Proces2.y_k_1);
    //Nowa wartosc wyjscia na podstawie u(k-1), y(k-1)
Proces2.y_k_1:= Proces2.y_k;
    //Zapamiętanie wyliczonej wartosci wyjscia

Proces2.u_k_1:= Proces2.u_k;
    //Zapamiętanie ostatniego sterowania
```

Ostatnim z opisywanych programów będzie program inicjalizujący wszystkie potrzebne zmienne. Program znajduje się w grupie Initial a jego kod źródłowy przedstawiono poniżej.

Warto zwrócić uwagę na ostatnią instrukcję EI(TRUE), która uruchamia przerwanie w sterowniku PLC. Dzięki niej uruchamiają się programy z grupy FIXED SCAN z określonym odstępem czasowym.

```
// Ustawienie procesu symulowanego
//Okres probkowania 1s=1000ms - parametr w programie FIXED SCAN
Proces1.sampling_time := 1.0;
// Ustawienie wartosci początkowych procesu 1
Proces1.Gain := 10.0;
Proces1.Time_const := 5.0;
//Okres probkowania 1s=1000ms - parametr w programie FIXED SCAN
Proces2.sampling_time := 1.0;
// Ustawienie wartosci początkowych procesu 2
```

```

Proces2.Gain := 5.0;
Proces2.Time_const := 10.0;

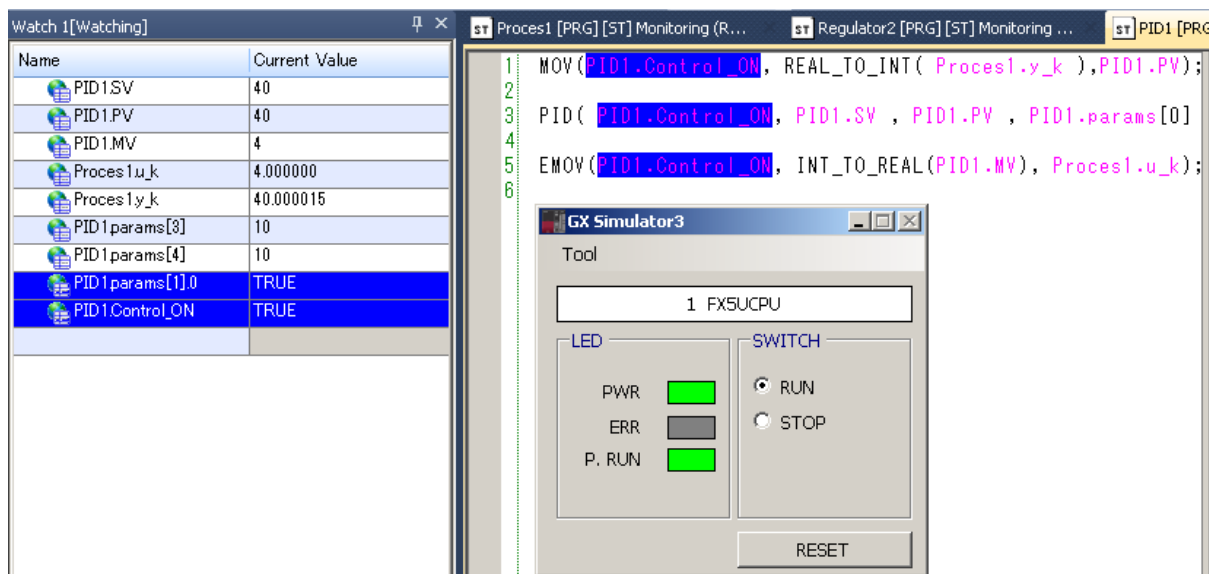
//Parametry regulatora wbudowanego PID1
PID1.params[0] := K1000; //okres regulacji w milisekundach
PID1.params[3] := K1; //wzmocnienie regulatora P
PID1.params[4] := K0; //TI = 0 oznacza nieskonczony czas calkowania - inaczej mowiac
calkowanie wylaczone
PID1.params[5] := K0; //KD = 0 oznacza zerowe wzmocnienie rozniczki
PID1.params[6] := K0; //TD = 0 oznacza wylaczone rozniczki
PID1.params[22] := 100; //gorny limit wartosci wyjsciowej z regulatora - zapobiega
rowniez efektowi wind-up
PID1.params[23] := 0; //dolny limit wartosci wyjsciowej z regulatora - -||-
SET(TRUE, PID1.params[1].5); //aktywacja limitow na wyjsciu regulatora
SET(TRUE, PID1.params[1].0); //trzeba odwrocic kierunek dzialania PID

//Parametry regulatora z dyskretnego rownania roznicowego PID2
PID2.K_gain := 1.0;
PID2.TI := 99999.0;
PID2.TD := 0.00001;
PID2.Ep0 := 0.0;
PID2.Ep1 := 0.0;
PID2.Ep2 := 0.0;
PID2.Rp0 := 0.0;
PID2.Rp1 := 0.0;
PID2.Rp2 := 0.0;
PID2.sampling_time := 1.0;

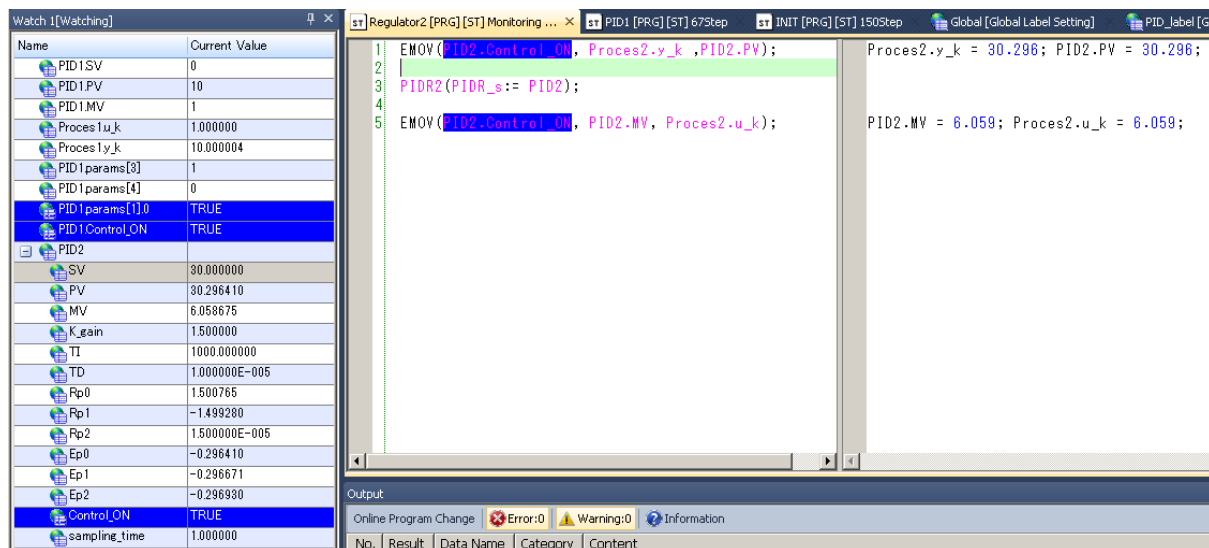
EI(TRUE);

```

Poniżej przedstawiono możliwości symulacji i monitorowania napisanych programów. Do okna Watch można dodawać pojedyncze zmienne jak również całe struktury.



Rys. 5.23: Monitorowanie zmiennych systemu w oknie Watch



Rys. 5.24: Monitorowanie zmiennych programu ST

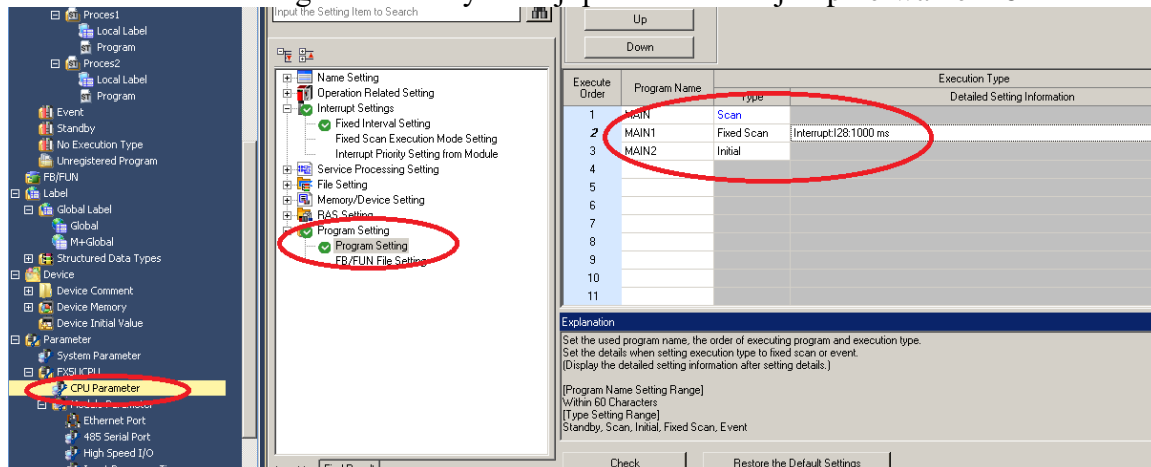
//////* OLD VERSION

5.12 Pierwszy program PLC – regulacja ciągła

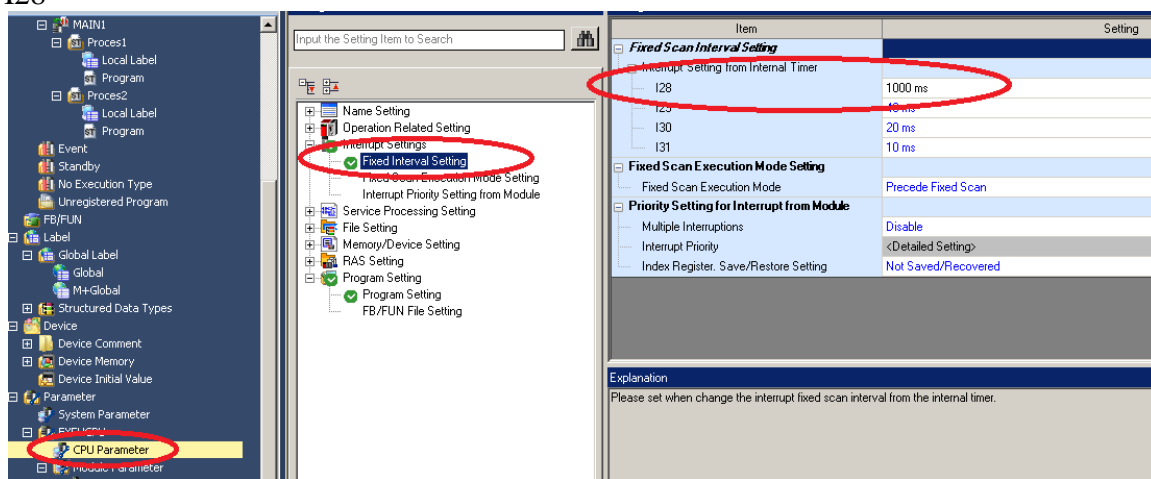
W tej części zostanie dokładnie omówiona część programowa sterownika PLC, która pozwoli na realizację regulatora (PID wbudowany, PID z równania różnicowego). Na laboratorium należy uzupełnić podany przykładowy program.

Poniżej przedstawiono najważniejsze punkty programów.

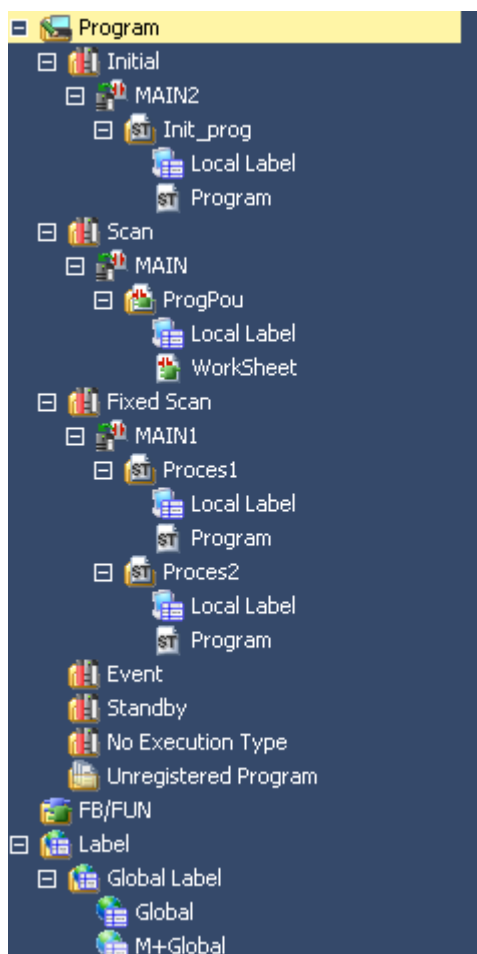
Okres uruchamiania regulatorów i symulacji procesów inercji – przerwanie I28



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 poczatkowych procesu 1
EMOV(TRUE, 5.0, stala_czasowa1);
EMOV(TRUE, 10.0, K_p_proces1);
```

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

```
//Parametry regulatora wbudowanego PID
parametry1[0] := K1000; //okres regulacji w milisekundach
parametry1[3] := K1; //wzmocnienie regulatora P
parametry1[4] := K0; //TI = 0 oznacza nieskonczony czas calkowania - inaczej mowiac
calkowanie wylaczone
parametry1[5] := K0; //KD = 0 oznacza zerowe wzmocnienie rozniczkowania
parametry1[6] := K0; //TD = 0 oznacza wylaczone rozniczkowanie
```



```
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;
```

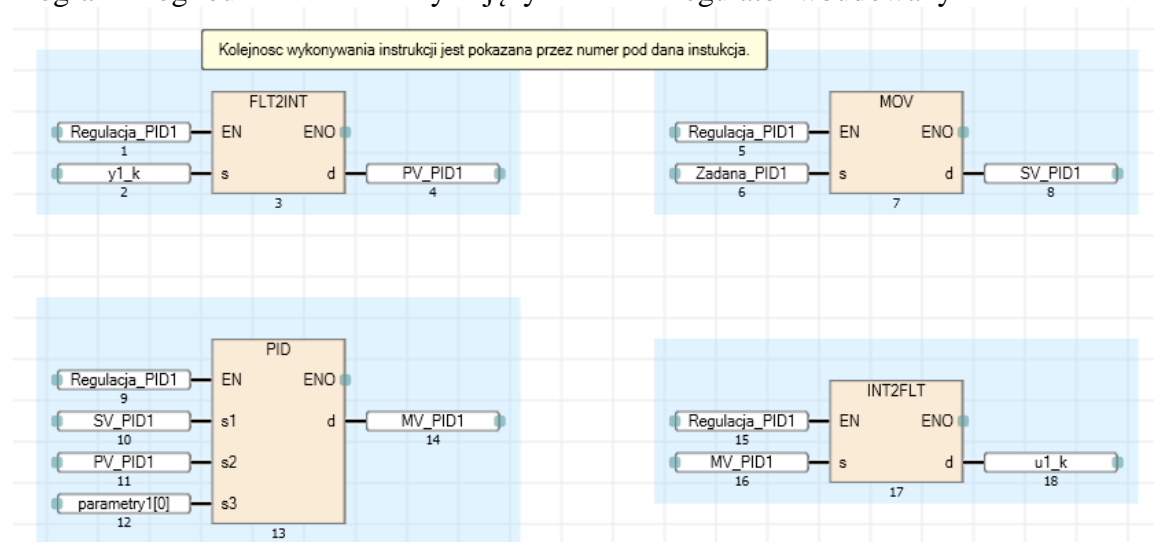
```
R1_PID3 := 0.0;
```

```
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/Le
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

```

```

*//////// OLD VERSION END – chyba do wyrzucenia – nie wiem tylko czy nie dać im
do samodzielnego opisanie równań PID o ile w ogóle teraz będziemy robić tu
PID??

```

6 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;
```

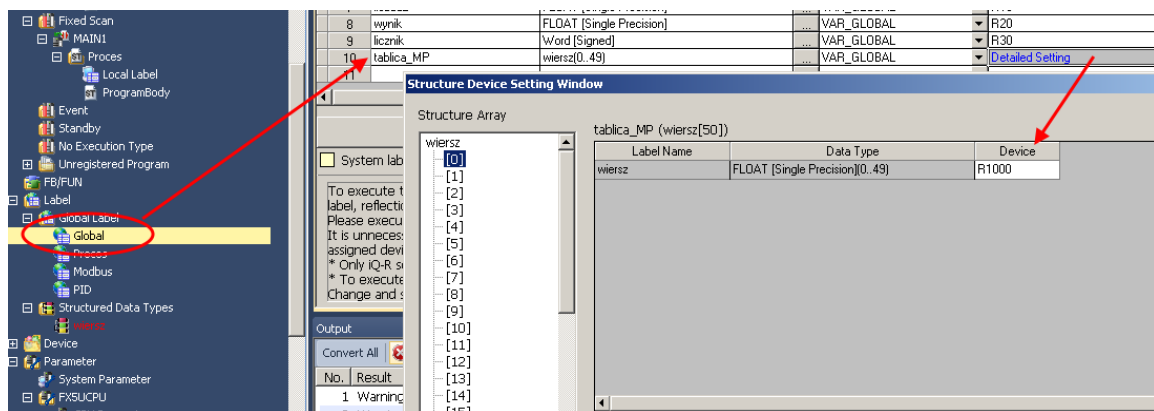
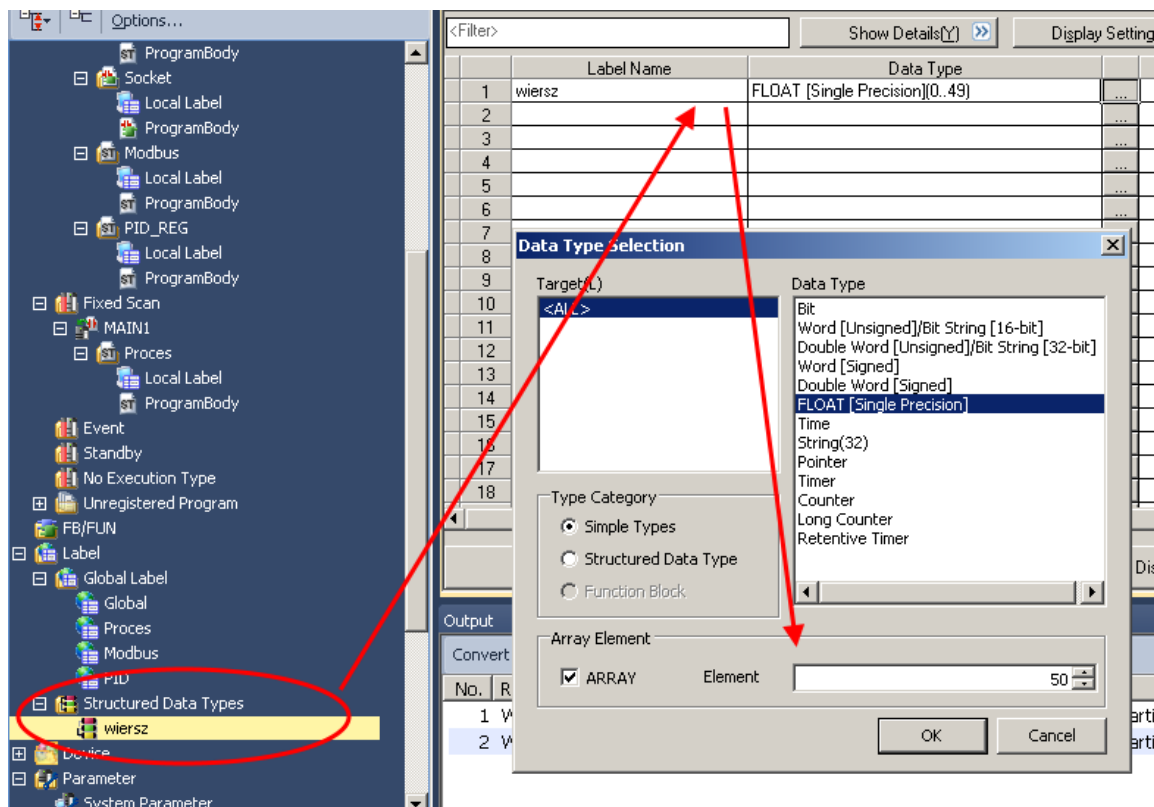
Użycie tablicy w obliczeniach:

```
FOR licznik := 0 TO 49 BY 1 DO
```

```
    wynik := liczba1 * liczba2;
```

```
    wynik := tablica_MP[licznik].wiersz[licznik] * INT_TO_REAL(licznik);
```

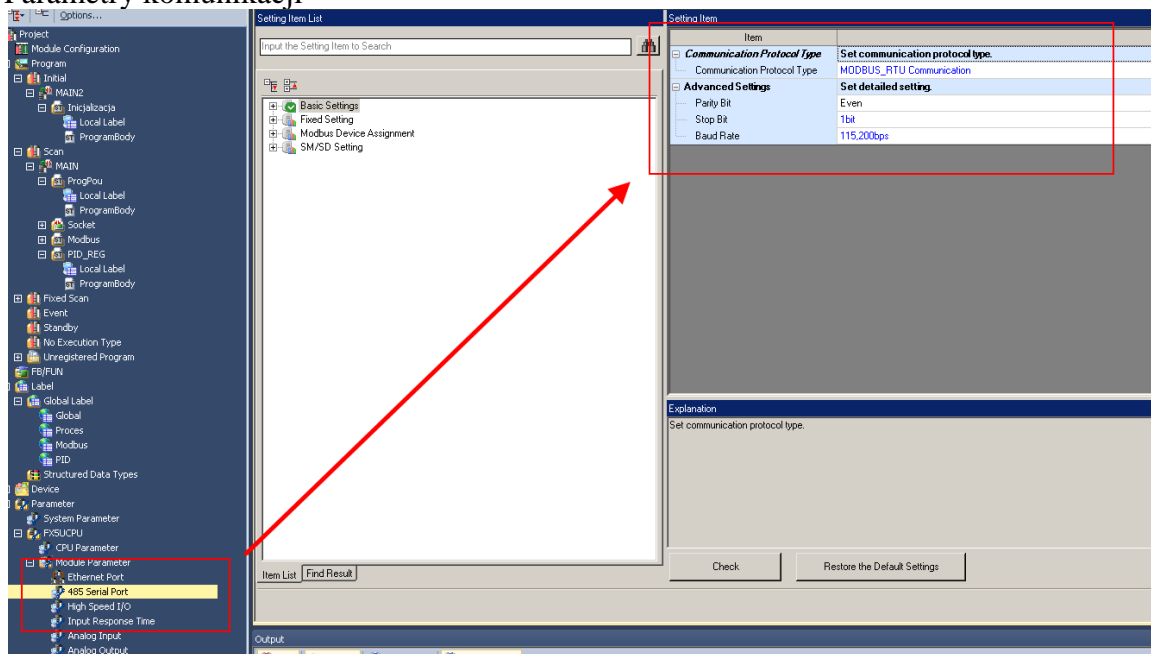
```
END_FOR;
```



7 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

		Show Details	Display Setting
	Label Name	Data Type	
1	wyslałem	Bit	...
2	nie_wyslałem	Bit	...
3	zmienna_int	Word (Signed)	...
4	auto_send	Bit	...
5			...

Globalne

		Easy Display	Display Setting	Check
	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 wysłania 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)));
```

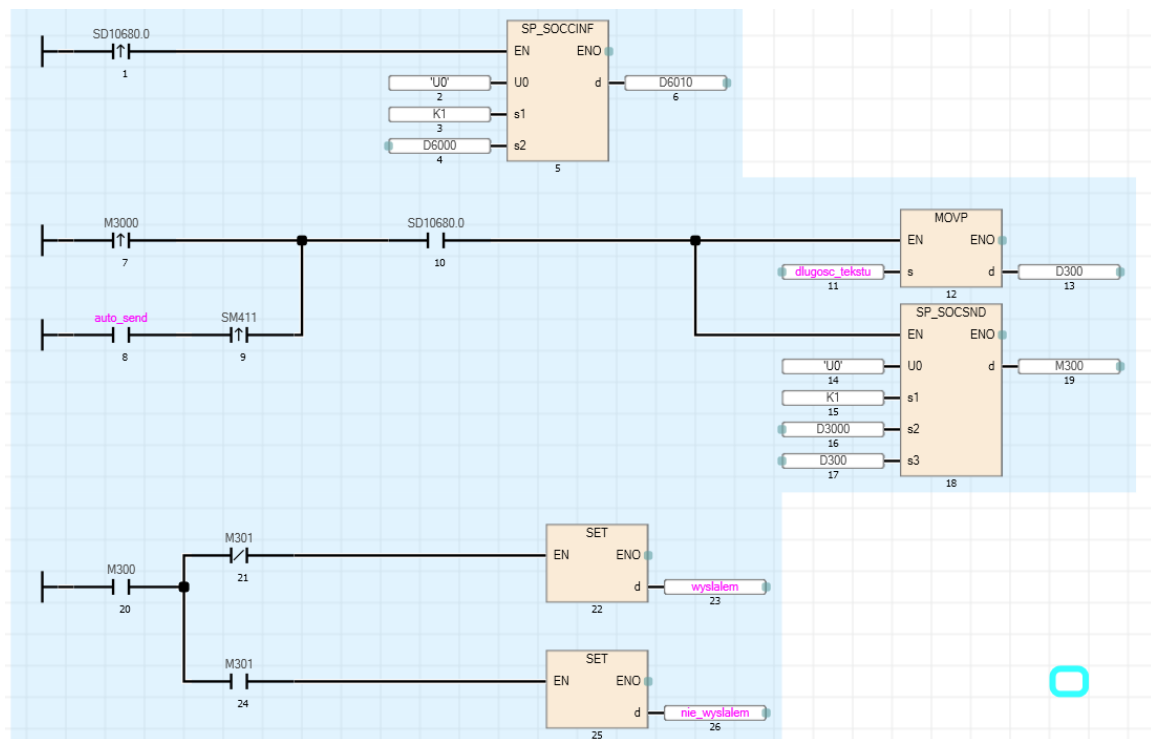
```
tekst_temp := CONCAT(tekst_temp, '$L');
```

```
send_string := tekst_temp;
```

```
//Długość 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;

```

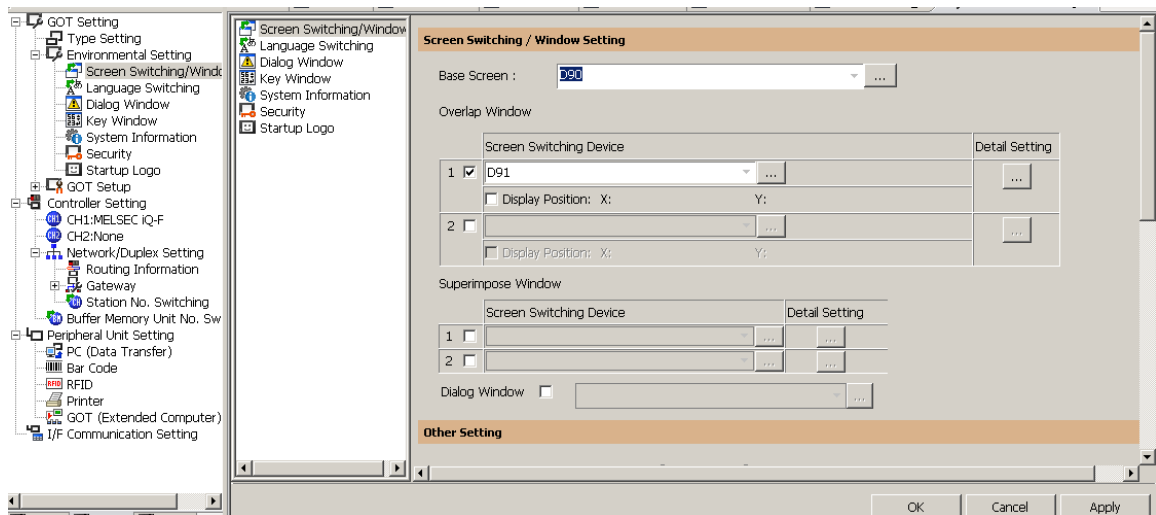
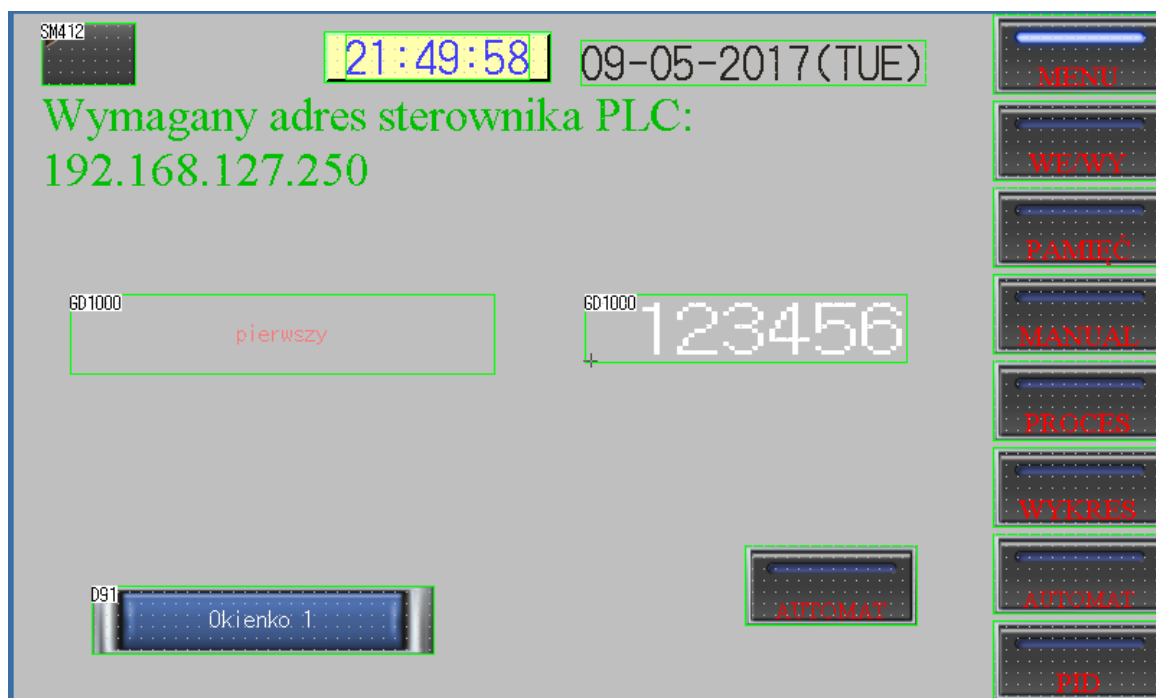
8 Tworzenie grafik operatorskich w środowisku GT Designer 3

8.1 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.

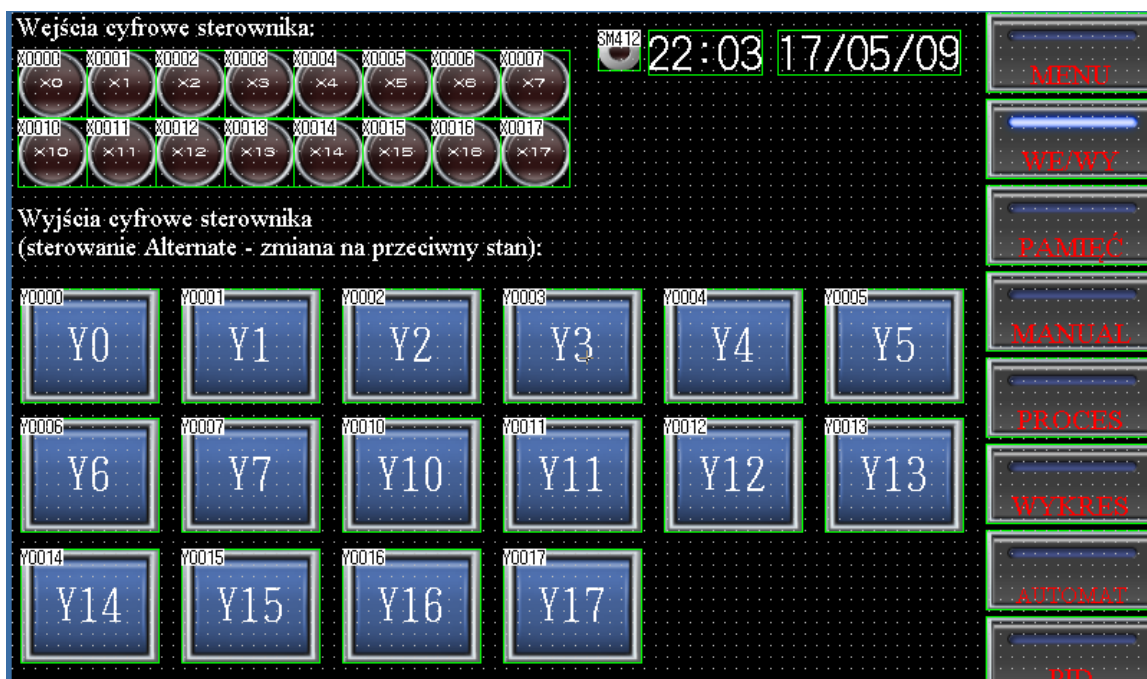
8.2 Panel MENU – 1

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.



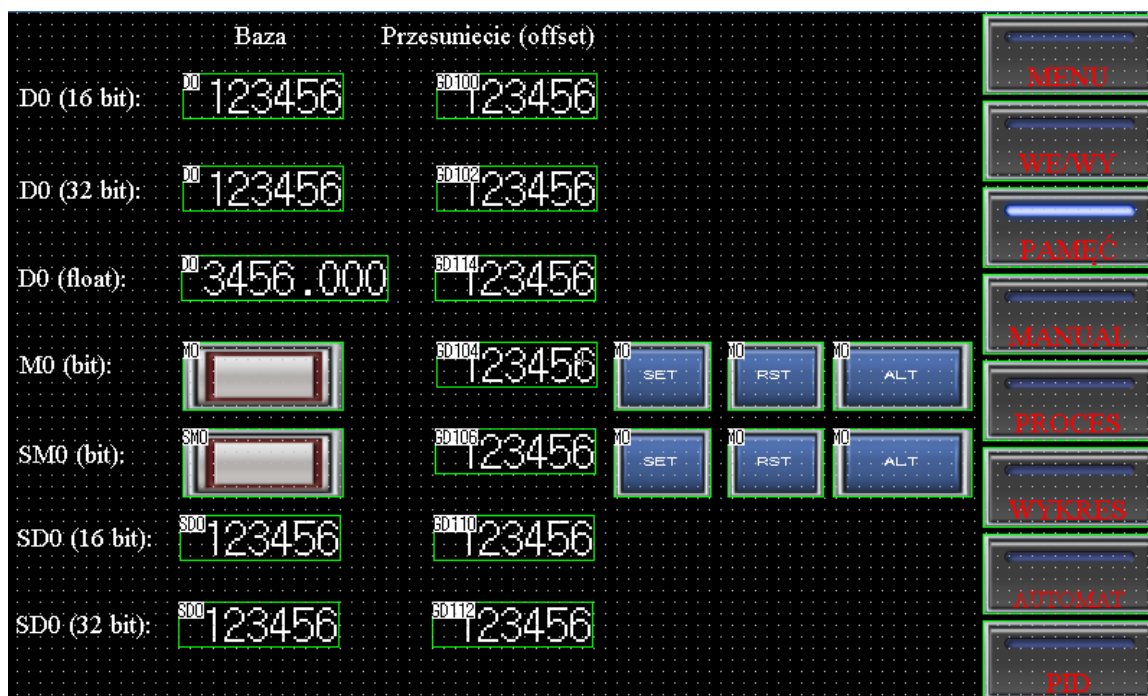
8.3 Panel WEWY – 2

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 wciskać i zmieniać tym samym stan wyjścia w trybie „alternate” – zawsze zmiana na stan przeciwny.



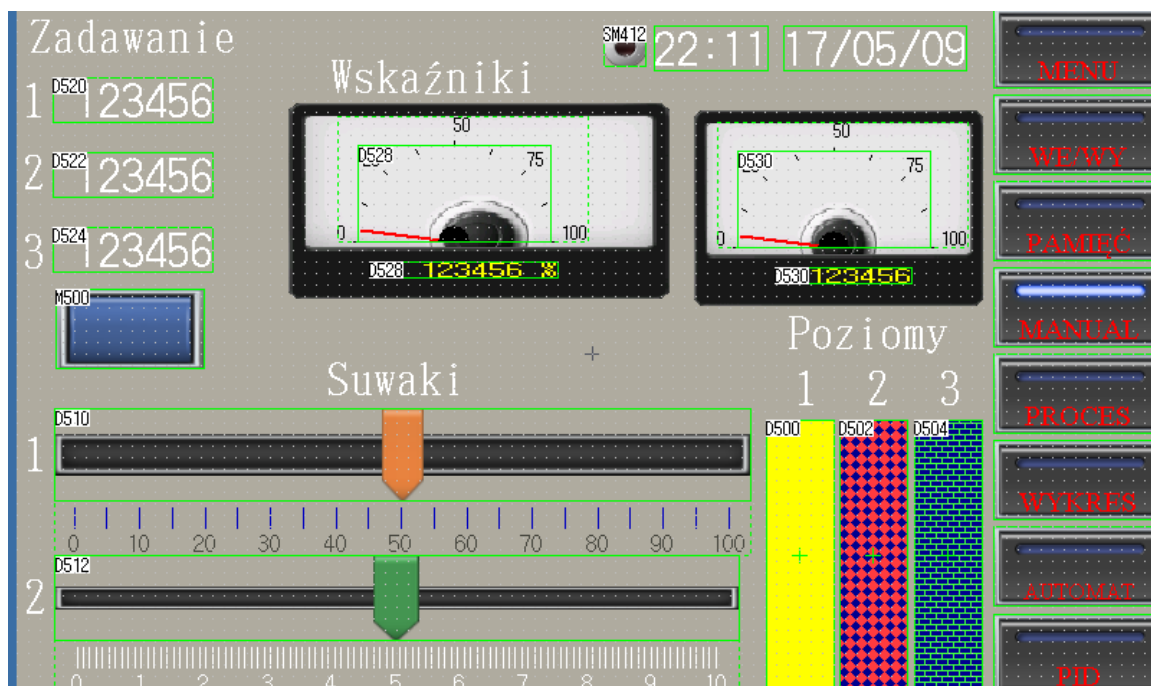
8.4 Panel PAMIEC – 3

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.



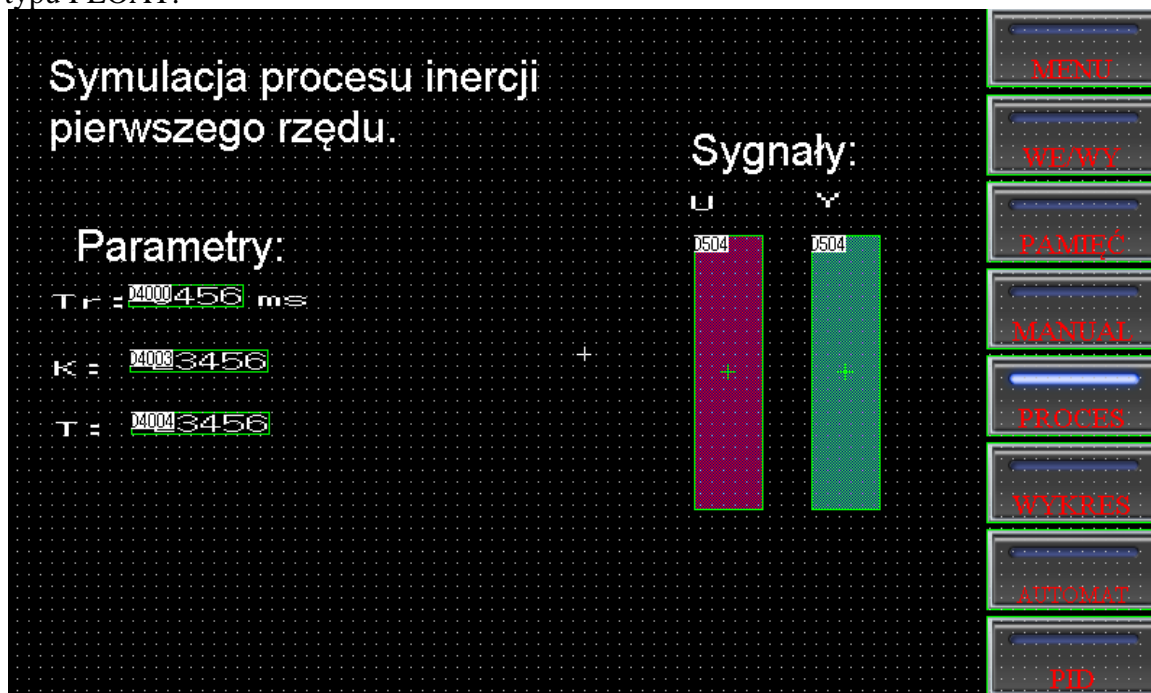
8.5 Panel MANUAL – 4

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.



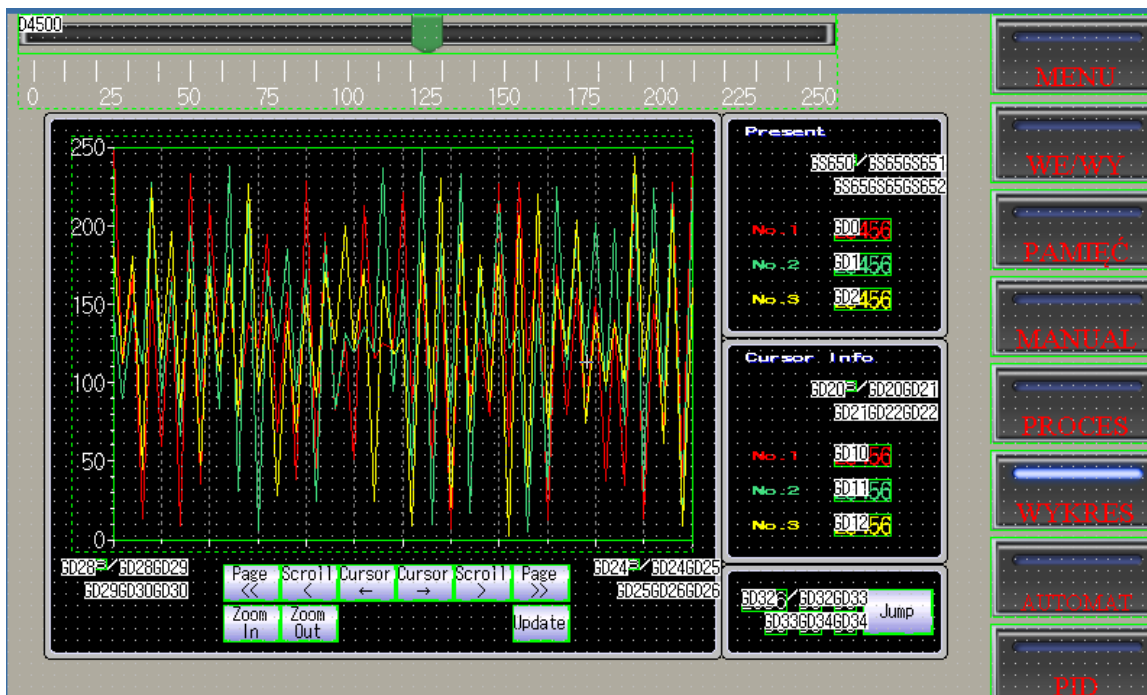
8.6 Panel PROCES – 5

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.



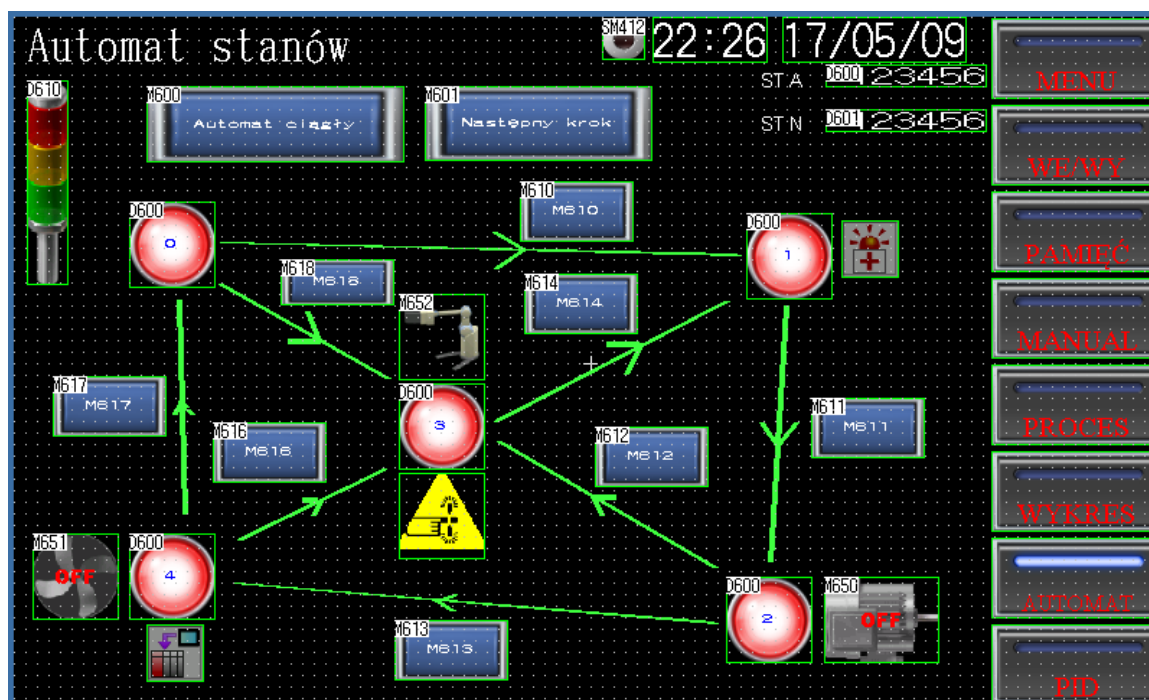
8.7 Panel WYKRES – 6

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.



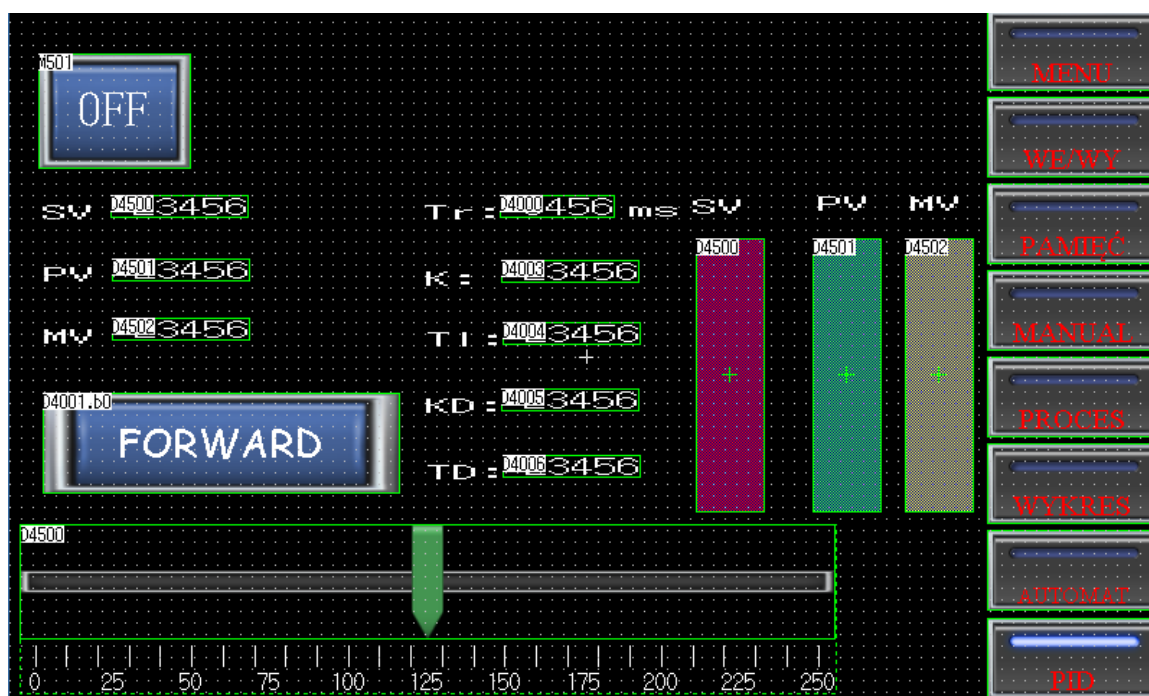
8.8 Panel AUTOMAT – 7

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.

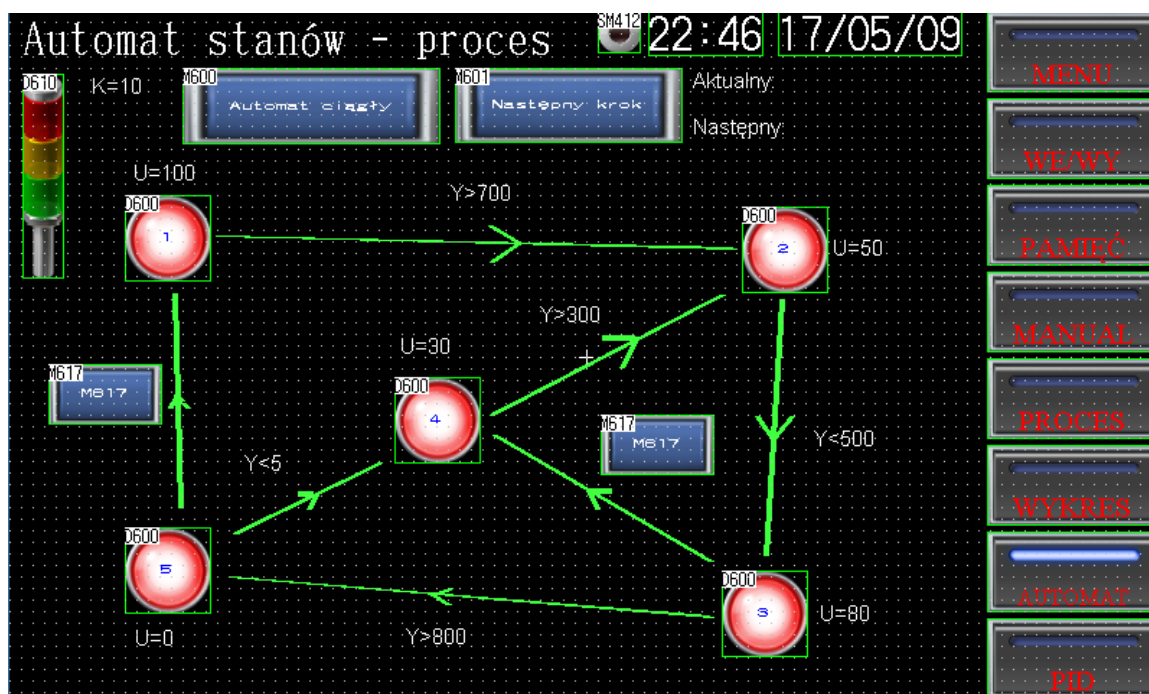


8.9 Panel AUTOMAT P – 8

Na panelu ósmym przedstawiono parametry oraz odpowiednie wartości wejściowe/wyjściowe dla regulatora PID. Znajdują się tutaj wartości zadana SV, mierzona procesa 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.



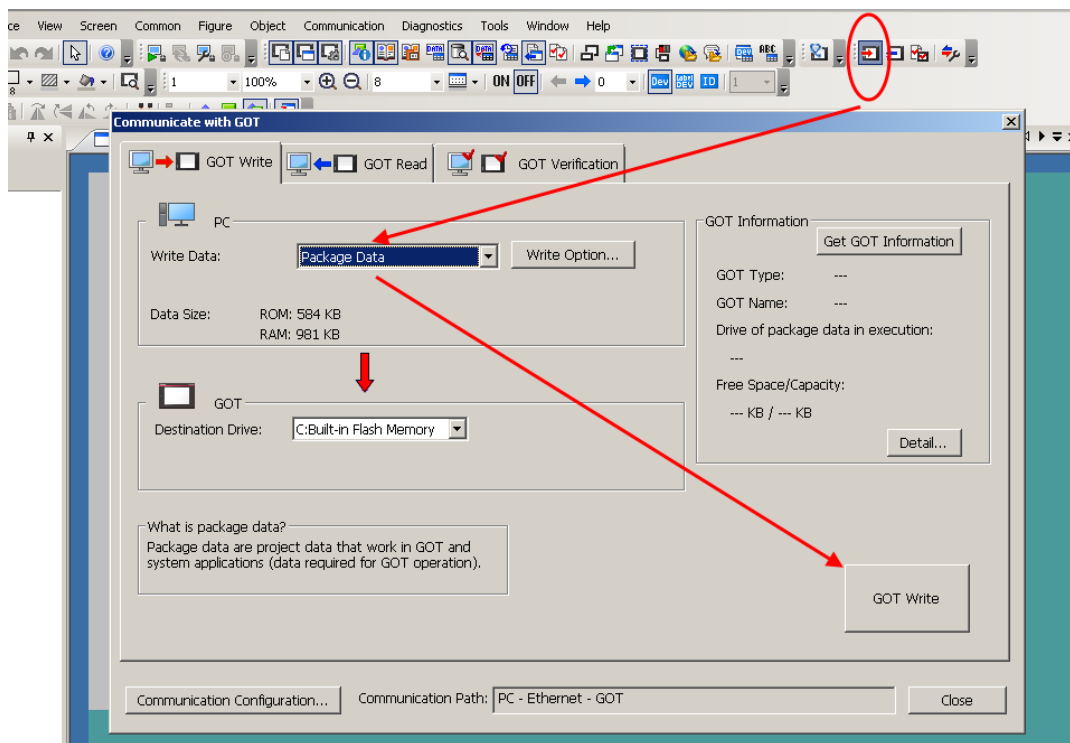
8.10 Panel AUTOMAT P – 10



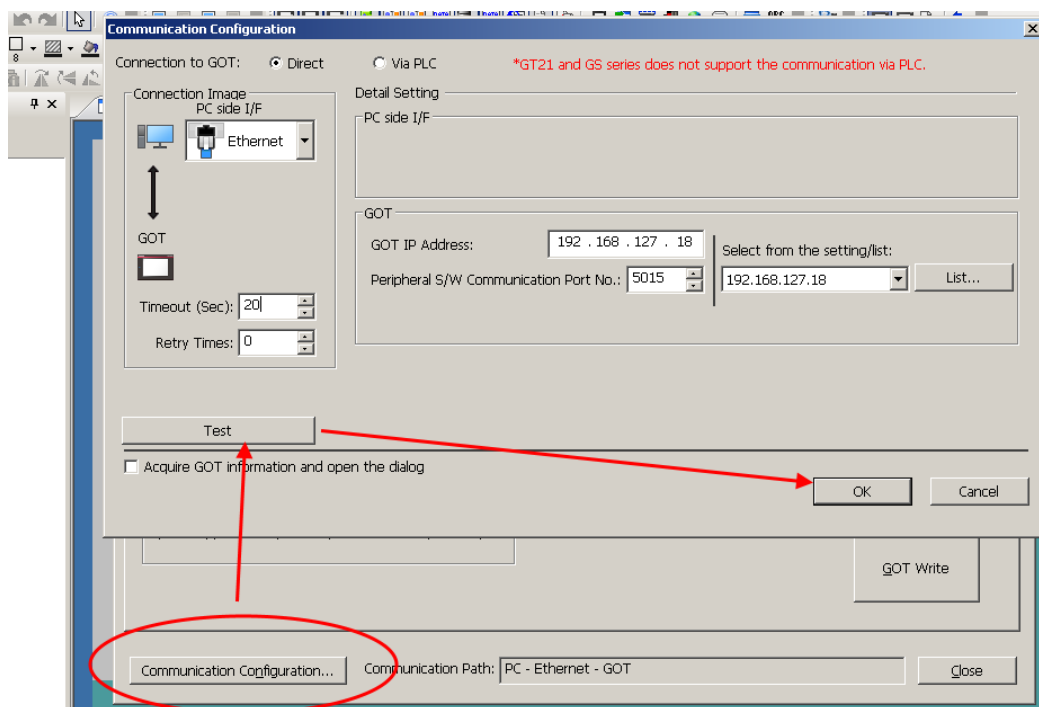
Na panelu dziesiątym przedstawiono graficzną reprezentację automatu stanów zrealizowane w języku ST w sterowniku PLC. Automat ten powinien zostać użyty do zmiany wartości zadanych w algorytmach regulacji w celu całkowitej automatyzacji pomiarów. Automat ten powinien rozpoczynać działanie tylko wtedy,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

8.11 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.

9 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

10 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 PRZYCISK „GRZYP” AWARYJNY. WŁĄCZAMY ZASILANIE ZESTAWU PLC, NASTĘPNIE WŁĄCZAMY ZASILANIE STANOWISKA (PRZEŁĄCZNIK Z TYŁU OBUDOWY PUDEŁKA Z ELEKTRONIKĄ). JEŻELI PLC JEST WŁĄCZONE MOŻNA WYCIĄGNĄĆ PRZYCISK „GRZYP” AWARYJNY I WCISNĄĆ NA STANOWISKU CZERWONY PRZYCISK 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 krańcowek (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 – pokrętko ręczne);
tryb pozycjonowania; tryb prędkościowy ->
użycie pomiaru prędkości z tachoprądnicy,
użycie pomiaru pozycji z enkodera

3. MULTI TANK



- Sterowanie poziomu 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.