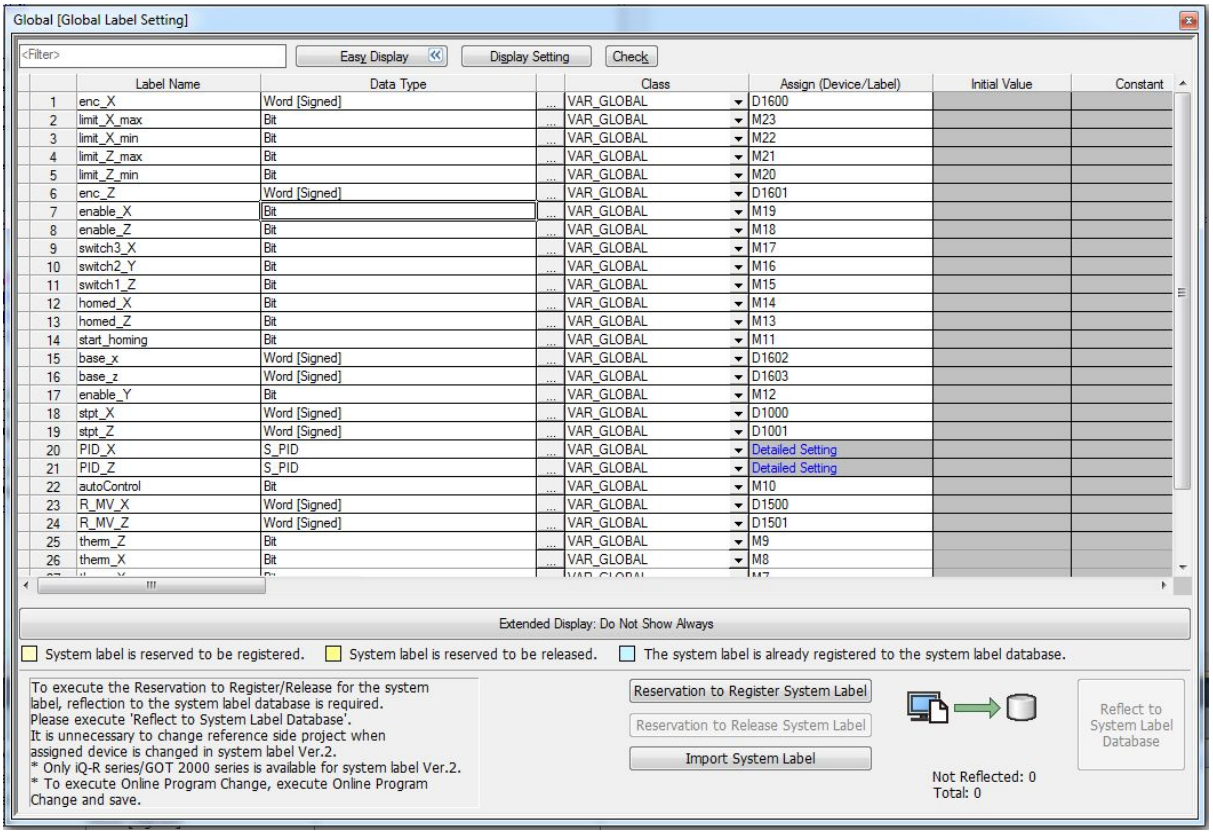
2.1.4 Wyjścia PWM

dodaj linijkę kodu do opisu:
HIOEN(TRUE, K0, K17, K0);
"aktywacja wyjść PWM na kanałach CH1 i CH5"
ustawione bity pierwszy i piąty dają dziesiętnie 17 obecne jako argument HIOEN

2.2. Mechanizm labeli

Zarejestrowaliśmy szereg labeli globalnych zastępujących m.in. wejścia X, ponieważ własne nazwy zwiększają uniwersalność i znacznie zwiększają czytelność kodu.



global_label.png
rys. Ustawienia labeli globalnych

2.3. Skalowanie i bazowanie

2.3.1 Skalowanie osi X

Odczytując wartości enkodera osi X w dwóch skrajnych pozycjach i dzieląc ich różnicę(9421) przez maksymalną założoną wartość pozycji zadanej(100) otrzymaliśmy zaokrąglony współczynnik skalowania(94)

Efektem jest operacja:
MOV(TRUE, stpt_X*K94, PID_X.SV);

2.3.2 Skalowanie osi Z

Odczytując wartości enkodera osi Z w dwóch skrajnych pozycjach i dzieląc ich różnicę(2300) przez maksymalną założoną wartość pozycji zadanej(100) otrzymaliśmy zaokrąglony współczynnik skalowania(23)

Efektem jest operacja:

MOV(TRUE, stpt_Z*K23, PID_Z.SV);

2.3.3 Bazowanie osi

Bazowanie wszystkich osi inicjowane jest stanem wysokim zmiennej *start_homing*. Po spełnieniu tego warunku rozpoczynany jest ruch osi w kierunku krańcówek. Osie poruszają się do chwili gdy osiągnął pozycje swoich krańcówek bazujących. W następnym kroku osie są zatrzymywane, ustawiane są bity informujące o zakończeniu bazowania konkretnej osi i wykonywane jest zerowanie liczników enkoderów osi.

Zbazowanie każdej z osi kończy proces bazowania dzwigu. Możliwe staje się sterowanie za pomocą pozycji zadanych poprzez regulatory PID.

```
IF start_homing = TRUE THEN
        IF homed_X = FALSE THEN
                IF switch3_X = FALSE THEN
                        Y7:=TRUE; // DEAKTYWACJA HAMULCA
                        PWM(TRUE, K30, K100, Y0); // PWM OSI X - KARETKA
                        ELSE
                        PWM(FALSE, K1, K100, Y0);
                        Y7:=FALSE; // AKTYWACJA HAMULCA
                        DHCMOVP(TRUE, 0, 0, SD4500); // WYZEROWANIE ENKODERA OSI X
                        homed_X:=TRUE;
                END_IF;
        END_IF;

        IF homed_Z = FALSE THEN
                IF switch1_Z = FALSE THEN
                        Y5:=TRUE; // DEAKTYWACJA HAMULCA
                        PWM(TRUE, K30, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU
                        ELSE
                        PWM(FALSE, K1, K100, Y2);
                        DHCMOVP(TRUE, 0, 0, SD4620); // WYZEROWANIE ENKODERA OSI Z
                        Y5:=FALSE; // AKTYWACJA HAMULCA
                        homed_Z:=TRUE;
                END_IF;
        END_IF;

        IF homed_X AND homed_Z THEN
                start_homing := FALSE;
        END_IF;
END_IF;
```

2.4. Obsługa I/O cyfrowych

2.4.1 Odczyt wejść cyfrowych

Stany wejść cyfrowych w celu zwiększenia czytelności kodu są przepisywane do uprzednio zdefiniowanych labeli, które następnie są wykorzystywane w programie.

Przykładowy odczyt stanu krańcówek osi:

```
MOVB(TRUE, X13, switch3_X);
```

```
MOVB(TRUE, X14, switch2_Y);
MOVB(TRUE, X12, switch1_Z);
```

Dalsze przykładowe wykorzystanie labeli do sterowania:

```
IF switch3_X = FALSE THEN
        Y7:=TRUE; // DEAKTYWACJA HAMULCA
        PWM(TRUE, K30, K100, Y0); // PWM OSI X - KARETKA
        ELSE
        PWM(FALSE, K1, K100, Y0);
        Y7:=FALSE; // AKTYWACJA HAMULCA
        DHCMOVP(TRUE, 0, 0, SD4500); // WYZEROWANIE ENKODERA OSI X
        homed_X:=TRUE;
END_IF;
```

2.4.2 Zapis wyjść cyfrowych

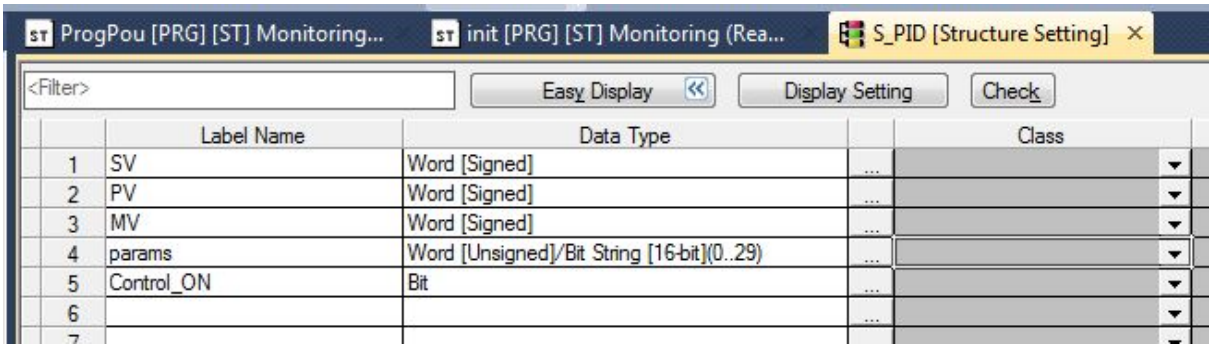
Zapis wyjść cyfrowych odbywa się poprzez bezpośrednie przypisanie stanu wyjścia. Nie zdecydowaliśmy się na zastosowanie dedykowanych labeli, ponieważ wyjść było stosunkowo niewiele.

Przykładowy zapis wyjścia cyfrowego:

```
Y5:=TRUE; // DEAKTYWACJA HAMULCA OSI Z
```

2.5. PID

Zastosowaliśmy wbudowaną strukturę regulatora PID. Do jej wykorzystania potrzebne było stworzenie struktury *S_PID* zawierającej SV, PV, MV, parametry regulatora oraz bit aktywujący regulator.



s_pid_struct.png
struktura S_PID

W programie inicjującym znajduje się inicjalizacja parametrów regulatorów PID

```
//Parametry regulatora wbudowanego PID osi X
PID_X.params[0] := K100; //okres regulacji w milisekundach
PID_X.params[3] := K3; //wzmocnienie regulatora P
PID_X.params[4] := K5; //TI = 0 oznacza nieskonczony czas calkowania - inaczej
//mowiac calkowanie wylaczone
PID_X.params[5] := K0; //KD = 0 oznacza zerowe wzmocnienie rozniczkowania
PID_X.params[6] := K0; //TD = 0 oznacza wylaczone rozniczkowanie
PID_X.params[22] := K100; //gorny limit wartosci wyjsciowej z regulatora - zapobiega
//rowniez efektowi wind-up
PID_X.params[23] := K1; //dolny limit wartosci wyjsciowej z regulatora - -||-
SET(TRUE, PID_X.params[1].5); //aktywacja limitow na wyjsciu regulatora
SET(TRUE, PID_X.params[1].0); //trzeba odwrocic kierunek dzialania PID
```

```
//Parametry regulatora wbudowanego PID osi Z
PID_Z.params[0] := K100; //okres regulacji w milisekundach
PID_Z.params[3] := K1; //wzmocnienie regulatora P
PID_Z.params[4] := K2; //TI = 0 oznacza nieskonczony czas calkowania - inaczej
//mowiac calkowanie wylaczone
PID_Z.params[5] := K0; //KD = 0 oznacza zerowe wzmocnienie rozniczkowania
PID_Z.params[6] := K0; //TD = 0 oznacza wylaczone rozniczkowanie
PID_Z.params[22] := K100; //gorny limit wartosci wyjsciowej z regulatora - zapobiega
//rowniez efektowi wind-up
PID_Z.params[23] := K0; //dolny limit wartosci wyjsciowej z regulatora - -||-
SET(TRUE, PID_Z.params[1].5); //aktywacja limitow na wyjsciu regulatora
SET(TRUE, PID_Z.params[1].0); //trzeba odwrocic kierunek dzialania PID
```

regulatory PID pracują w głównym programie. Kolejne wartości sterowania MV wyznaczane są poprzez wywołania funkcji PID(). Jeśli wyznaczone MV jest większe od 50 wykonywany jest ruch w przód z zadanym wypełnieniem PWM, a w przeciwnym wypadku wykonywany jest ruch w tył z zadanym wypełnieniem PWM.

Sekcja regulatora PID osi X:

```
MOVB(TRUE, TRUE, PID_X.Control_ON);
MOV(TRUE, enc_X, PID_X.PV);
MOV(TRUE, stpt_X*K94, PID_X.SV);
PID(PID_X.Control_ON, PID_X.SV , PID_X.PV , PID_X.params[0] , PID_X.MV);
PWM(TRUE, PID_X.MV, K100, Y0); // PWM OSI X - KARETKA
IF PID_X.MV < K50 THEN
        MOVB(TRUE, FALSE, Y10);
        MOVB(TRUE, TRUE, Y7);
        PWM(TRUE, K51-PID_X.MV, K100, Y0); // PWM OSI X - KARETKA
        ELSE
        IF PID_X.MV > K50 THEN
                MOVB(TRUE, TRUE, Y10);
                MOVB(TRUE, TRUE, Y7);
                PWM(TRUE, PID_X.MV-K49, K100, Y0); // PWM OSI X - KARETKA
                ELSE
                MOVB(TRUE, FALSE, Y7);
                PWM(FALSE, K1, K100, Y0); // PWM OSI X - KARETKA
        END_IF;
END_IF;
```

Sekcja regulatora PID osi Z:

```
MOVB(TRUE, TRUE, PID_Z.Control_ON);
MOV(TRUE, enc_Z, PID_Z.PV);
MOV(TRUE, stpt_Z*K23, PID_Z.SV);
PID(PID_Z.Control_ON, PID_Z.SV , PID_Z.PV , PID_Z.params[0] , PID_Z.MV);
PWM(TRUE, PID_Z.MV, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU

IF PID_Z.MV < K50 THEN
        MOVB(TRUE, FALSE, Y6);
        MOVB(TRUE, TRUE, Y5);
        PWM(TRUE, K51-PID_Z.MV, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU
        ELSE
        IF PID_Z.MV > K50 THEN
                MOVB(TRUE, TRUE, Y6);
                MOVB(TRUE, TRUE, Y5);
                PWM(TRUE, PID_Z.MV-K49, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU
                ELSE
                MOVB(TRUE, FALSE, Y5);
                PWM(FALSE, K1, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU
```

```
        END_IF;
END_IF;
```

2.6. Tryb sterowania ręcznego

Sterowanie ręczne możliwe jest po zresetowaniu flagi AUTO_CONTROL(flaga domyślnie ustawiona). Jeśli flaga jest zresetowana program przechodzi do sekcji kodu w której odbywa się bezpośrednie wysterowanie wyjść PWM wartościami zmiennych R_MV_Z dla osi Z i R_MV_X dla osi X

sekcja kodu sterowania ręcznego osią Z:

```
IF enc_Z > K2350 OR enc_Z < -K50 THEN
        // PRZEKROCZENIE ZAKRESÓW OSI Z
        MOVB(TRUE, FALSE, Y5);
        PWM(FALSE, K1, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU
        MOVB(TRUE, FALSE, homed_Z);
        ELSE
        //STEROWANIE RĘCZNE OSI Z
        PWM(TRUE, R_MV_Z, K100, Y2);       // PWM OSI Z - OBRÓT DZWIGU
        IF R_MV_Z < K50 THEN
                MOVB(TRUE, FALSE, Y6);
                MOVB(TRUE, TRUE, Y5);
                PWM(TRUE, K51-R_MV_Z, K100, Y2);  // PWM OSI Z - OBRÓT DZWIGU
                ELSE
                IF R_MV_Z > K50 THEN
                        MOVB(TRUE, TRUE, Y6);
                        MOVB(TRUE, TRUE, Y5);
                        PWM(TRUE, R_MV_Z-K49, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU
                        ELSE
                        MOVB(TRUE, FALSE, Y5);
                        PWM(FALSE, K1, K100, Y2);
                END_IF;
        END_IF;
END_IF;
```

sekcja kodu sterowania ręcznego osią X:

```
IF enc_X > K9450 OR enc_X < -K50 THEN
        // PRZEKROCZENIE ZAKRESÓW OSI X
        MOVB(TRUE, FALSE, Y7);
        PWM(FALSE, K1, K100, Y0); // PWM OSI X - KARETKA
        MOVB(TRUE, FALSE, homed_X);
        ELSE
        // STEROWANIE RĘCZNE OSI X
        PWM(TRUE, R_MV_X, K100, Y0); // PWM OSI X - KARETKA

        IF R_MV_X < K50 THEN
        MOVB(TRUE, FALSE, Y10);
                MOVB(TRUE, TRUE, Y7);
                PWM(TRUE, K51-R_MV_X, K100, Y0); // PWM OSI X - KARETKA
                ELSE
                IF R_MV_X > K50 THEN
                        MOVB(TRUE, TRUE, Y10);
                        MOVB(TRUE, TRUE, Y7);
                        PWM(TRUE, R_MV_X-K49, K100, Y0); // PWM OSI X - KARETKA
                        ELSE
                        MOVB(TRUE, FALSE, Y7);
```

```
                    PWM(FALSE, K1, K100, Y0); // PWM OSI X - KARETKA
                END_IF;
        END_IF;
END_IF;
```

2.7. Zabezpieczenia ruchów krańcowych

W projekcie założyliśmy, że sytuacja w której któraś z osi wyjdzie poza założony dopuszczalny zakres ruchów jest ona natychmiast zatrzymywana i gaszona jest flaga zbazowania danej osi. Działaniem naprawczym w takiej sytuacji jest przejście na tryb sterowania automatycznego i ponowne zainicjowanie bazowania osi ustawiając flagę start_homing
sekcja kodu zabezpieczająca ruchy krańcowe osi X:

```
IF homed_X = TRUE THEN
        IF enc_X > K9450 OR enc_X < -K50 THEN
                MOVB(TRUE, FALSE, Y7);
                PWM(FALSE, K1, K100, Y0); // PWM OSI X - KARETKA
                MOVB(TRUE, FALSE, homed_X);
        ELSE
                //automatyczne lub ręczne sterowanie osi X
                …
        END_IF;
END_IF;
```

sekcja kodu zabezpieczająca ruchy krańcowe osi Z:

```
IF homed_Z = TRUE THEN
        IF enc_Z > K2350 OR enc_Z < -K50 THEN
                MOVB(TRUE, FALSE, Y5);
                PWM(FALSE, K1, K100, Y2); // PWM OSI Z - OBRÓT DZWIGU
                MOVB(TRUE, FALSE, homed_Z);
        ELSE
                //automatyczne lub ręczne sterowanie osi Z
                …
        END_IF;
END_IF;
```

2.8. Język ST

Listingi kodu zawarte powyżej jednoznacznie wskazują na pomyślne wykorzystanie języka ST.