

POLITECHNIKA POZNAŃSKA

WYDZIAŁ AUTOMATYKI, ROBOTYKI I ELEKTROTECHNIKI



RAPORT LABORATORYJNY

ZASTOSOWANIE STEROWNIKÓW
PRZEMYSŁOWYCH

RAPORT LABORATORYJNY

JULIA SKOTNICKA, 147542

JULIA.SKOTNICKA@STUDENT.PUT.POZNAN.PL

KRZYSZTOF DOLNY, 147528

KRZYSZTOF.DOLNY@STUDENT.PUT.POZNAN.PL

PROWADZĄCY:

MGR INŻ. KRZYSZTOF

KOLANOWSKI.

KRZYSZTOF.KOLANOWSKI@PUT.POZNAN.PL

07-05-2023

1. Wstęp	3
2. Podział pracy	3
3. Opis istniejących rozwiązań	3
4. Algorytm działania programu	4
5. Program opatrzone komentarzami	4
6. Spis i wykaz Tag, DB, wykorzystanej przestrzeni adresowej	7
7. Wykazanie zajętości programu w sterowniku	10
8. Wykazany czas trwania programu	10
9. Wnioski	10

1. Wstęp

Celem projektu było napisanie programu w Tia Portal wraz z intuicyjną wizualizacją na HMI, który sterowałby ruchem pojazdów i pieszych na skrzyżowaniu ulic Baraniaka i Jana Pawła II w Poznaniu. Program obsługuje przyciski dla pieszych, pętle indukcyjne i pozwala na sterowanie ręczne uniemożliwiające spowodowanie sytuacji zagrożenia w ruchu. Na panelu HMI możemy zobaczyć które światła na skrzyżowaniu aktualnie świecą dla danego pasa ruchu i poszczególnych przejść dla pieszych.

2. Podział pracy

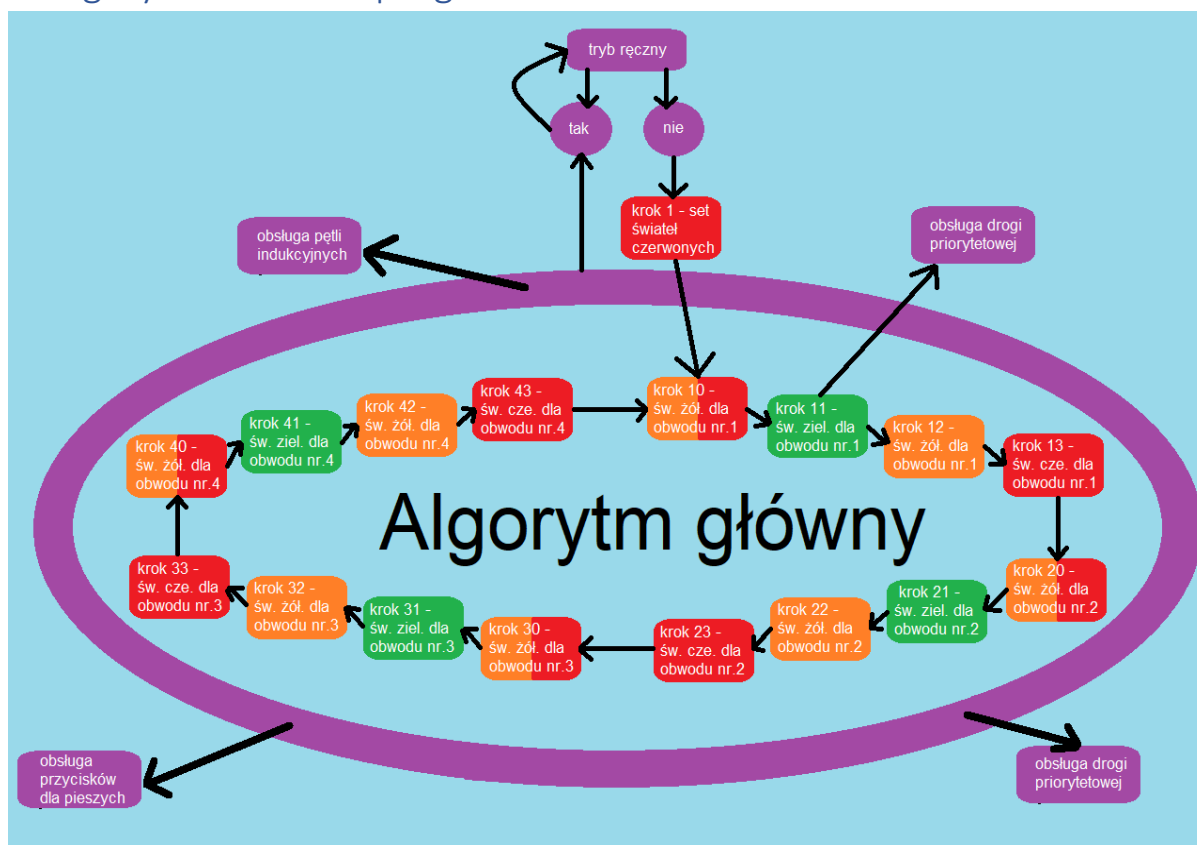
Algorytm główny działania programu został wymyślony i zaimplementowany przez obu członków grupy. Krzysztof Dolny był odpowiedzialny za napisanie obsługi przycisków dla pieszych, pętli indukcyjnych, nadanie trybu priorytetu poszczególnych ulic i ich implementację w LAD. Julia Skotnicka wykonała wizualizację działania skrzyżowania na panelu HMI i testowała działanie programu.

3. Opis istniejących rozwiązań

Obecnie programowanie sygnalizacji świetlnej realizowane na wiele różnych sposobów w zależności od wielu czynników takich jak analiza wymagań, dostępne zasoby oraz natężenia ruchu w danym miejscu. Oto kilka rozwiązań:

- Technologia Vehicle-to-Infrastructure (V2I), czyli komunikacji między pojazdami a infrastrukturą drogową. Pojazdy przesyłają dane dotyczące swojego położenia, prędkości, kierunku jazdy. W ten sposób, sygnalizacja świetlna może dostosowywać swoje działanie do bieżącej sytuacji na drodze i wprowadzać zmiany w czasie rzeczywistym.
- Programowalne sterowniki Ruchu Miejskiego - to systemy sterowania sygnalizacją świetlną opracowane przez firmę ITS Polska programowane zazwyczaj w języku drabinkowym. PRM składa się z dwóch części: sprzętowej i oprogramowania.
- Systemy wykorzystujące algorytmy adaptacyjne np. SCOOT-umożliwia dynamiczne dostosowanie cyklu świateł do aktualnego natężenia ruchu, co pozwala na optymalne wykorzystanie drogi i minimalizację korków, na ogół w takich systemach stosuje się programowanie w C++, Java lub Python
- Sterowniki PLC połączone z technologią IoT- pozwala to na komunikację z innymi urządzeniami takimi jak czujniki ruchu, aplikacje czy kamery co pozwala na bardziej efektywne zarządzanie ruchem drogowym
- Programowalne sterowniki sygnalizacji świetlnej - działają w trybie stałego czasu i wykorzystywane są na skrzyżowaniach o małym natężeniu ruchu

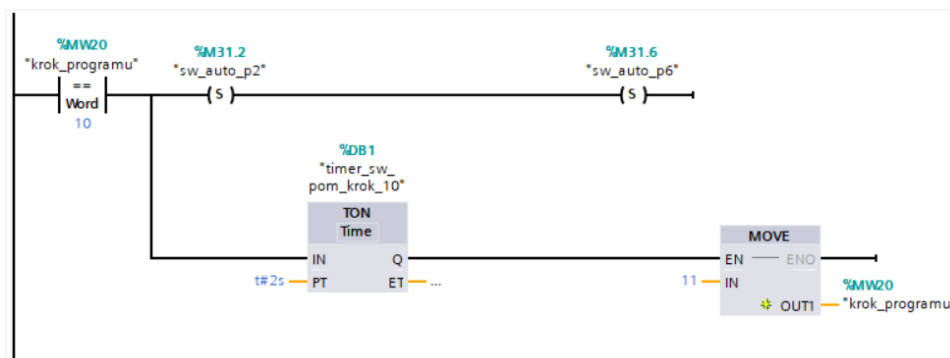
4. Algorytm działania programu



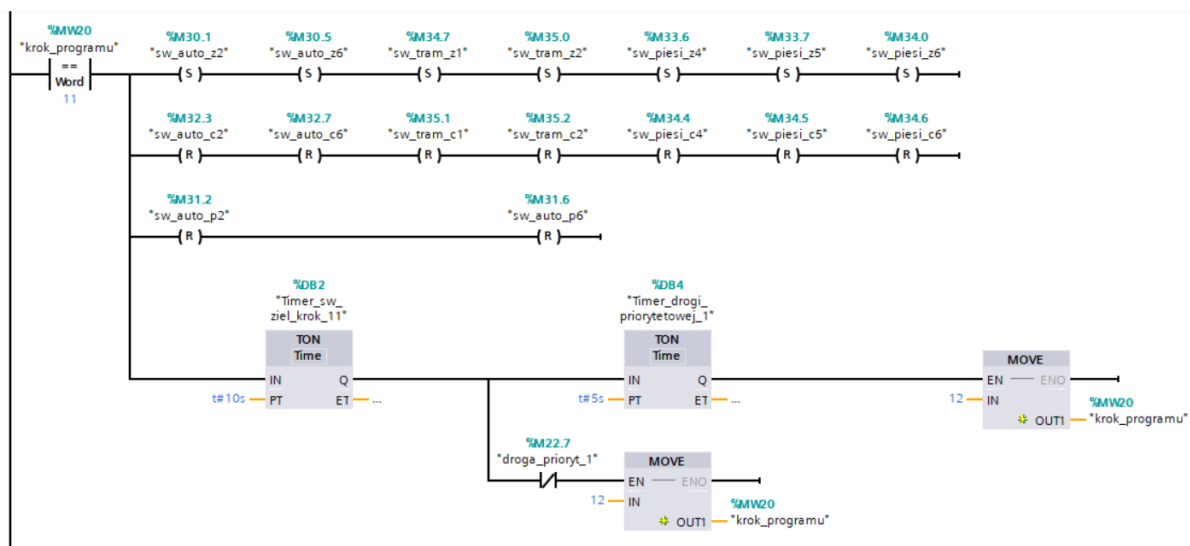
5. Program opatrzony komentarzami

Algorytm główny jest wywoływany cyklicznie z bloku Main [OB1] i został on podzielony na pięć głównych części, czyli start algorytmu głównego i cztery obwody, które załączają w odpowiedniej kolejności (czerwone-żółte, zielone, żółte, czerwone) światła sygnalizacji świetlnej dla danych pasów ruchu i pieszych. Poniżej znajduje się przykład działania pierwszego obwodu:

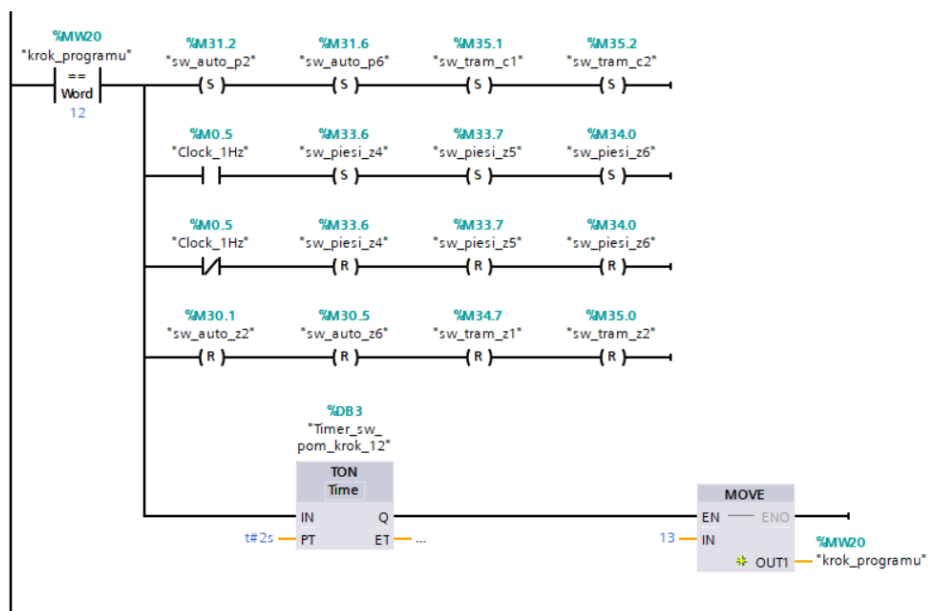
Krok 10 – zapalenie się świateł żółtych dla samochodów z obwodu nr. 1:



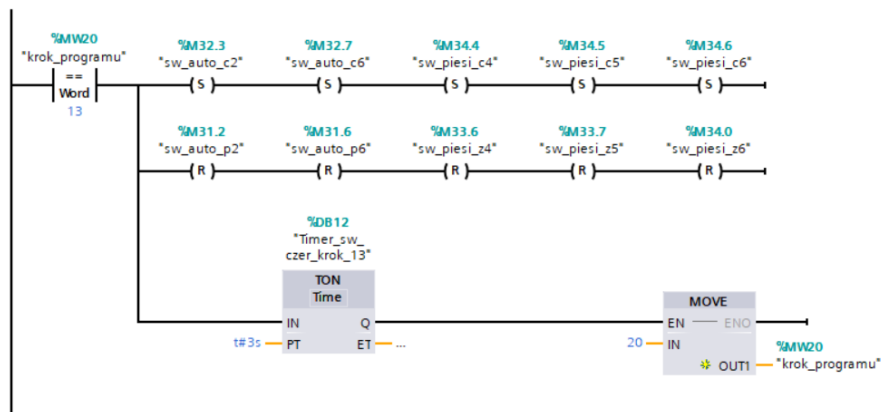
Krok 11 – reset świateł żółtych i czerwonych, set świateł zielonych:



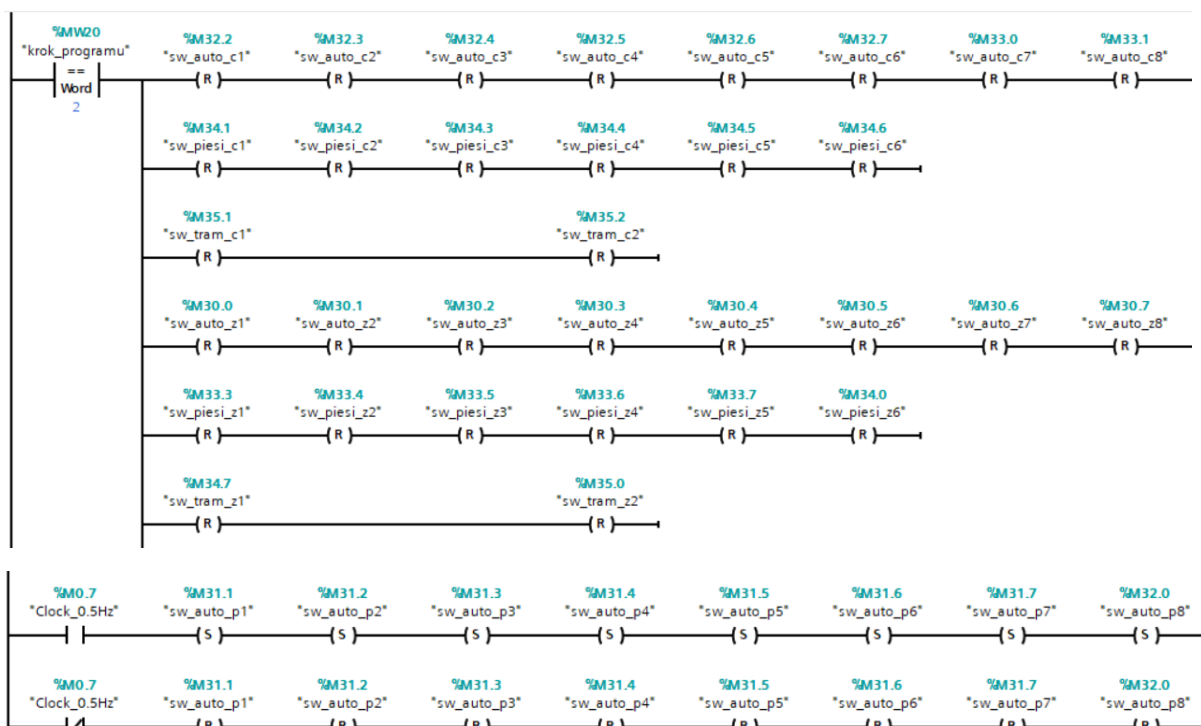
Krok 12 – reset świateł zielonych, set świateł żółtych dla aut i czerwonych dla tramwajów, migające światło zielone dla pieszych:



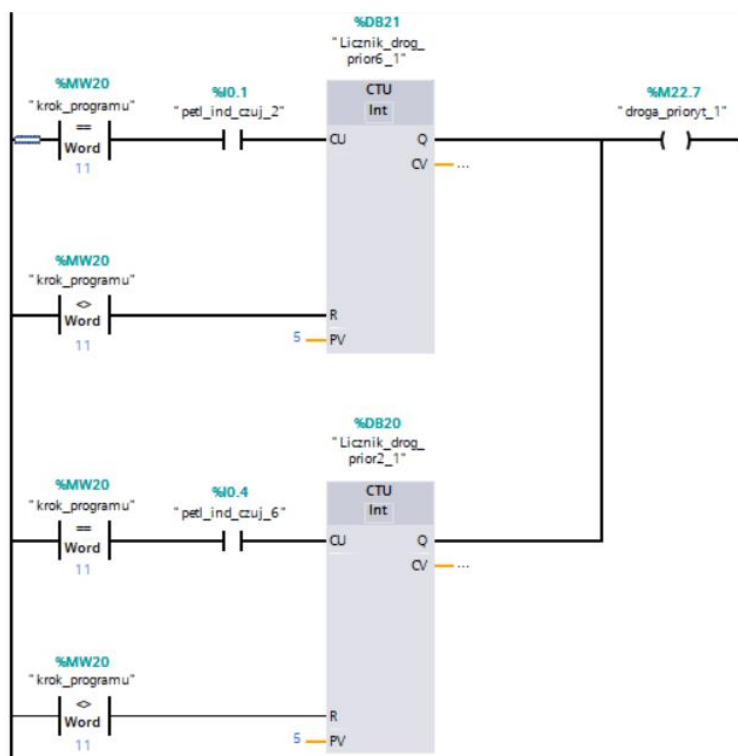
Krok 14 – set świateł czerwonych:



Tryb ręczny - włączenie trybu ręcznego powoduje zresetowanie się wszystkich sygnalizatorów i następnie okresowo załączane są światła żółte. W trakcie działania trybu ręcznego nie są wykonywane żadne kroki z algorytmu głównego.



Obsługa drogi priorytetowej – polega na tym, że jeśli podczas światła zielonego na drodze priorytetowej przejdzie co najmniej 5 pojazdów (wykrywanych przez pętle indukcyjne) to czas działania tego światła zostanie wydłużony:



[illegible]

```

graph LR
    Start(( )) --> NO1["*krok_programu*  
21"]
    NO1 --> NC1["*petl_ind_czuj_1*"]
    NC1 --> NC2["*petl_ind_czuj_5*"]
    NC2 --> NC3["*petl_ind_czuj_9*"]
    NC3 --> TON["TON  
Time  
IN PT 4s Q ET ..."]
    TON --> MOVE["MOVE  
EN ENO  
IN 22 OUT1 *krok_programu*"]
    MOVE --> End(( ))

```

Zmienne PLC 1:

Totally Integrated Automation Portal		
Table of contents		
PLC tags		
Default tag table [23]		
PLC tags		2 - 1
User constants		3 - 1
Tablica_wejsc [14]		
PLC tags		4 - 1
User constants		5 - 1
Tablica_wyjsc [43]		
PLC tags		6 - 1
User constants		7 - 1
Tablica_zmiennych_programowych [10]		
PLC tags		8 - 1
User constants		9 - 1

Zmienne domyślne:

PLC tags / Default tag table [23]							
PLC tags							
PLC tags							
	Name	Data type	Address	Retain	Visible in HMI	Accessible from HMI	Comment
⌵	Clock_Byte	Byte	%MB0	False	True	True	
⌵	Clock_10Hz	Bool	%M0.0	False	True	True	
⌵	Clock_5Hz	Bool	%M0.1	False	True	True	
⌵	Clock_2.5Hz	Bool	%M0.2	False	True	True	
⌵	Clock_2Hz	Bool	%M0.3	False	True	True	
⌵	Clock_1.25Hz	Bool	%M0.4	False	True	True	
⌵	Clock_1Hz	Bool	%M0.5	False	True	True	
⌵	Clock_0.625Hz	Bool	%M0.6	False	True	True	
⌵	Clock_0.5Hz	Bool	%M0.7	False	True	True	












































Tablica wejść:

PLC tags / Tablica_wejsc [14]							
PLC tags							
PLC tags							
	Name	Data type	Address	Retain	Visible in HMI	Accessible from HMI	Comment
⌵	przycisk_piesi_1	Bool	%M10.0	False	True	True	
⌵	przycisk_piesi_2	Bool	%M10.1	False	True	True	
⌵	przycisk_piesi_3	Bool	%M10.2	False	True	True	
⌵	przycisk_piesi_4	Bool	%M10.3	False	True	True	
⌵	przycisk_piesi_5	Bool	%M10.4	False	True	True	
⌵	przycisk_piesi_6	Bool	%M10.5	False	True	True	
⌵	petl_ind_czuj_1	Bool	%I0.0	False	True	True	
⌵	petl_ind_czuj_2	Bool	%I0.1	False	True	True	
⌵	petl_ind_czuj_3	Bool	%I0.2	False	True	True	
⌵	petl_ind_czuj_5	Bool	%I0.3	False	True	True	
⌵	petl_ind_czuj_6	Bool	%I0.4	False	True	True	
⌵	petl_ind_czuj_8	Bool	%I0.5	False	True	True	
⌵	petl_ind_czuj_9	Bool	%I0.6	False	True	True	
⌵	przycisk_trybu_recznego	Bool	%I0.7	False	True	True	

Tablica zmiennych programowych:

PLC tags / Tablica_zmiennych_programowych [10]							
PLC tags							
PLC tags							
	Name	Data type	Address	Retain	Visible in HMI	Accessible from HMI	Comment
⌵	krok_programu	Word	%MW20	False	True	True	
⌵	tryb_sterowania_recznego	Bool	%M22.0	False	True	True	
⌵	zmienna_piesi_1	Bool	%M22.1	False	True	True	
⌵	zmienna_piesi_2	Bool	%M22.2	False	True	True	
⌵	zmienna_piesi_3	Bool	%M22.3	False	True	True	
⌵	zmienna_piesi_4	Bool	%M22.4	False	True	True	
⌵	zmienna_piesi_5	Bool	%M22.5	False	True	True	
⌵	zmienna_piesi_6	Bool	%M22.6	False	True	True	
⌵	droga_prioryt_1	Bool	%M22.7	False	True	True	
⌵	droga_prioryt_2	Bool	%M23.0	False	True	True	

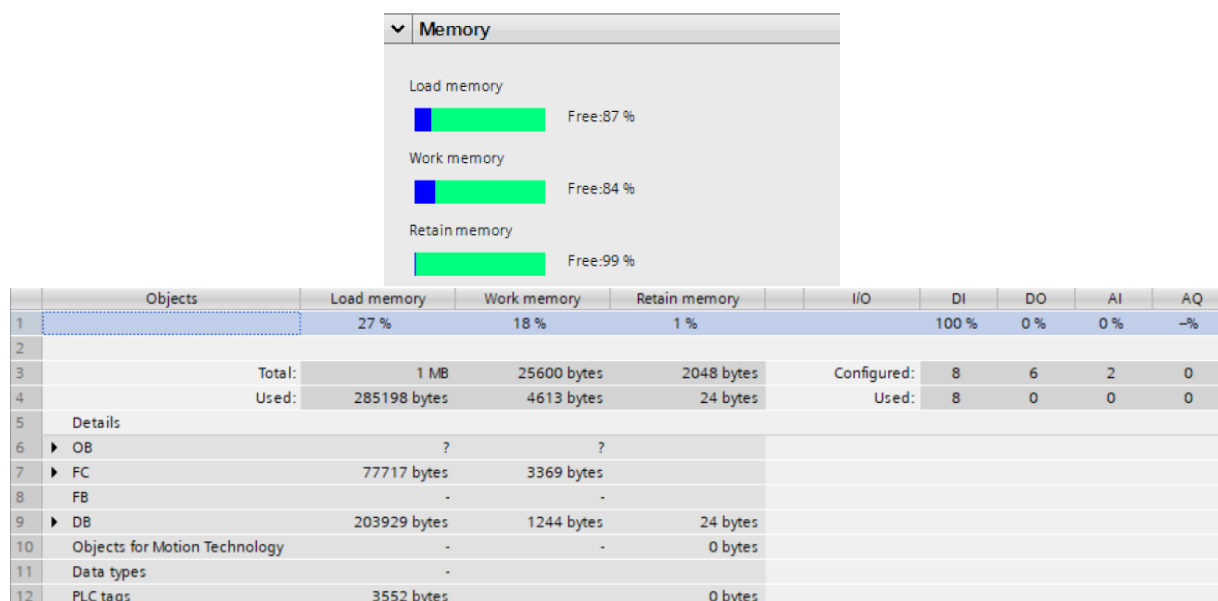
Tablica wyjść:

PLC tags / Tablica_wyjsc [43]							
PLC tags							
PLC tags	Name	Data type	Address	Retain	Visible in HMI	Accessible from HMI	Comment
	sw_auto_z1	Bool	%M30.0	False	True	True	
	sw_auto_z2	Bool	%M30.1	False	True	True	
	sw_auto_z3	Bool	%M30.2	False	True	True	
	sw_auto_z4	Bool	%M30.3	False	True	True	
	sw_auto_z5	Bool	%M30.4	False	True	True	
	sw_auto_z6	Bool	%M30.5	False	True	True	
	sw_auto_z7	Bool	%M30.6	False	True	True	
	sw_auto_z8	Bool	%M30.7	False	True	True	
	sw_auto_z9	Bool	%M31.0	False	True	True	
	sw_auto_p1	Bool	%M31.1	False	True	True	
	sw_auto_p2	Bool	%M31.2	False	True	True	
	sw_auto_p3	Bool	%M31.3	False	True	True	
	sw_auto_p4	Bool	%M31.4	False	True	True	
	sw_auto_p5	Bool	%M31.5	False	True	True	
	sw_auto_p6	Bool	%M31.6	False	True	True	
	sw_auto_p7	Bool	%M31.7	False	True	True	
	sw_auto_p8	Bool	%M32.0	False	True	True	
	sw_auto_p9	Bool	%M32.1	False	True	True	
	sw_auto_c1	Bool	%M32.2	False	True	True	
	sw_auto_c2	Bool	%M32.3	False	True	True	
	sw_auto_c3	Bool	%M32.4	False	True	True	
	sw_auto_c4	Bool	%M32.5	False	True	True	
	sw_auto_c5	Bool	%M32.6	False	True	True	
	sw_auto_c6	Bool	%M32.7	False	True	True	
	sw_auto_c7	Bool	%M33.0	False	True	True	
	sw_auto_c8	Bool	%M33.1	False	True	True	
	sw_auto_c9	Bool	%M33.2	False	True	True	
	sw_piesi_z1	Bool	%M33.3	False	True	True	
	sw_piesi_z2	Bool	%M33.4	False	True	True	
	sw_piesi_z3	Bool	%M33.5	False	True	True	
	sw_piesi_z4	Bool	%M33.6	False	True	True	
	sw_piesi_z5	Bool	%M33.7	False	True	True	
	sw_piesi_z6	Bool	%M34.0	False	True	True	
	sw_piesi_c1	Bool	%M34.1	False	True	True	
	sw_piesi_c2	Bool	%M34.2	False	True	True	
	sw_piesi_c3	Bool	%M34.3	False	True	True	
	sw_piesi_c4	Bool	%M34.4	False	True	True	
	sw_piesi_c5	Bool	%M34.5	False	True	True	
	sw_piesi_c6	Bool	%M34.6	False	True	True	
	sw_tram_z1	Bool	%M34.7	False	True	True	
	sw_tram_z2	Bool	%M35.0	False	True	True	
	sw_tram_c1	Bool	%M35.1	False	True	True	
	sw_tram_c2	Bool	%M35.2	False	True	True	

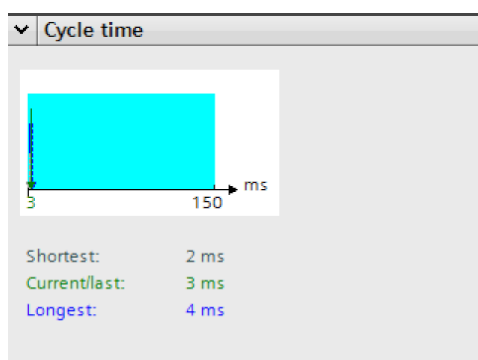
Timery:

Table of contents	
Program resources	
1Timer_petli_ind_9 [DB18]	2 - 1
2Timer_petli_ind_9 [DB19]	3 - 1
1Timer_zmiany_sw_piesi_1 [DB25]	4 - 1
1Timer_petli_ind_1 [DB26]	5 - 1
2Timer_petli_ind_1 [DB27]	6 - 1
1Timer_zmiany_sw_piesi_3 [DB28]	7 - 1
1Timer_petli_ind_3 [DB29]	8 - 1
2Timer_petli_ind_3 [DB30]	9 - 1
1Timer_zmiany_sw_piesi_4 [DB31]	10 - 1
1Timer_petli_ind_8 [DB32]	11 - 1
Timer_obsi_petli_ind_1 [DB33]	12 - 1
Timer_obsi_petli_ind_2 [DB34]	13 - 1
Timer_obsi_petli_ind_4 [DB36]	14 - 1
Timer_obsi_petli_ind_3 [DB35]	15 - 1
Licznik_drog_prior2_1 [DB20]	16 - 1
Licznik_drog_prior6_1 [DB21]	17 - 1
Licznik_drog_prior1_2 [DB22]	18 - 1
Licznik_drog_prior5_2 [DB23]	19 - 1
timer_sw_pom_krok_10 [DB1]	20 - 1
Timer_sw_ziel_krok_11 [DB2]	21 - 1
Timer_sw_pom_krok_12 [DB3]	22 - 1
timer_sw_pom_krok_20 [DB5]	23 - 1
Timer_sw_ziel_krok_21 [DB6]	24 - 1
Timer_sw_pom_krok_22 [DB7]	25 - 1
timer_sw_pom_krok_30 [DB8]	26 - 1
Timer_sw_ziel_krok_31 [DB9]	27 - 1
Timer_sw_pom_krok_32 [DB10]	28 - 1
Timer_sw_czer_krok_33 [DB11]	29 - 1
Timer_sw_czer_krok_13 [DB12]	30 - 1
Timer_sw_czer_krok_23 [DB13]	31 - 1
timer_sw_pom_krok_40 [DB14]	32 - 1
Timer_sw_ziel_krok_41 [DB15]	33 - 1
Timer_sw_pom_krok_42 [DB16]	34 - 1
Timer_sw_czer_krok_43 [DB17]	35 - 1
Timer_drogi_prioritetowej_1 [DB4]	36 - 1
Timer_drogi_prioritetowej_2 [DB24]	37 - 1

7. Wykazanie zajętości programu w sterowniku



8. Wykazany czas trwania programu



9. Wnioski

Dzięki zastosowaniu odpowiedniego podziału programu na kilka bloków funkcyjnych, gdzie naszym trzonem jest blok algorytmu głównego wywoływany z Main [OB1], to po odpowiedniej jego implementacji łatwo można było dodawać kolejne funkcjonalności takie jak obsługa przycisków dla pieszych. W naszym przypadku mogliśmy się przekonać, że panel HMI służy nie tylko do przykładowo obsługi danej maszyny, lecz także był bardzo pomocny przy wizualizacji działania całego programu dzięki czemu dużo łatwiej mogliśmy wyłapać błędy w naszym kodzie. Dodatkowo podjęliśmy próbę implementacji Web servera dla naszego PLC przy użyciu html, css i javascript, lecz niestety nie udało nam się pobierać kilkudziesięciu danych z sterownika PLC jednocześnie przez co niestety nie udało nam się napisać automatycznie aktualizującej się strony, na której byłaby dostępna wizualizacja działania skrzyżowania, lecz mimo wszystko dzięki temu nauczyliśmy się pewnych podstaw pisania stron internetowych.