Podstawy programowania PLC w języku drabinkowym - ćwiczenie 5

1. Cel ćwiczenia

Zapoznanie się z podstawowymi elementami języka drabinkowego i zasadami programowania Programowalnych Sterowników Logicznych (Programmable Logic Controller). W ramach ćwiczenia studenci wykonują przykładowe programy na sterownikach.

2. Wykaz aparatury

- Sterownik programowalny typu FX3U firmy Mitsubishi
- Panel operatora firmy Proface
- Oprogramowanie narzędziowe GX Developer

3. Podstawy programowania w języku drabinkowym

Podczas programowania w języku drabinkowym używamy określonych operandów (zmiennych określonego typu). Listę operandów dla sterownika FX3U podano poniżej. Każdy operand ma przypisany numer z dostępnego zakresu np. X0, Y1, M0, M10 itp. (dostępny zakres numeracji podano w tabeli w kolumnie „Liczba adresów")

Nazwa operandu	Symbol operandu	Opis	Liczba adresów
Wejście	X	Wejście dwustanowe sterownika PLC	W sumie 256
Wyjście	Y	Wyjście dwustanowe sterownika PLC	
Znacznik	M	Wewnętrzna 1-bitowa pamięć pomocnicza	7680
Licznik czasu	T	Element odliczający zadany przedział czasu	512
Licznik zdarzeń	C	Element liczający impulsy logiczne	256, 6 HSC
Znacznik stanu	S	Do programowania krokowego (STL)	4096
Stała	K, H	Stała dziesiętna lub szesnastkowa	16-, 32-bitowe
Rejestr danych	D, R	Rejestr danych, rejestr zbioru (16, 32-bitowy)	8000, 32768
Rejestr indeksowy	V, Z	Zawiera adres pośredni; do indeksowej modyfikacji argumentów	16
Wskaźnik	P	Wskaźnik skoku w programie (etykieta)	4095
Przerwanie	I	Przerwanie programu głównego	6 wejść, 3 timery
Zagłębienie	N	Sterowanie sekcjami programu głównego	8

Wejścia „X" i wyjścia „Y" numerowane są w zapisie ósemkowym kolejno niezależnie dla wejść i wyjść np. X0, X1, X2 …X8, X10, X11 …X17 itd. (jak widać nie występują X8 i X9, X18 i X19 itd.) oraz Y0 do Y7, Y10 do Y17 itd. Numerację rozpoczyna się od wejść i wyjść znajdujących się na sterowniku – numeracje wejść i wyjść znajdujących się na modułach rozszerzających rozpoczynamy od nowej oktawy (np. jeżeli ostatnie wejście na sterowniku miało numer X13 to kolejne wejście już na module ma numer X20 a nie X14).

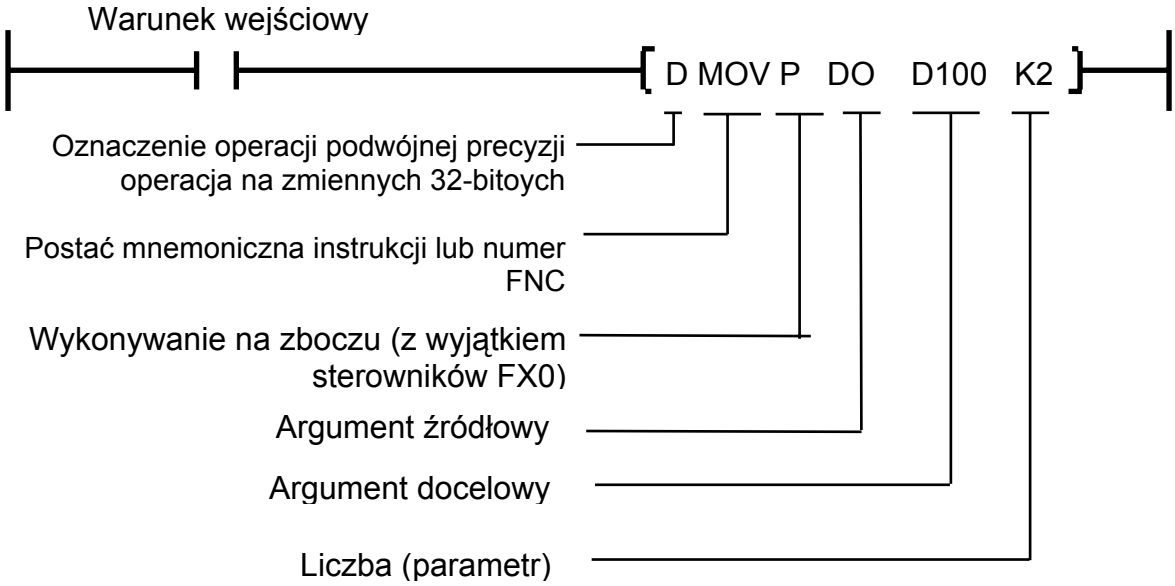
Rolę przekaźników pomocniczych pełnią 1-bitowe zmienne M. Bity M0 do M7679 mogą być używane przez programistę dowolnie (zwykle do zapisania pośrednich wyników –zmiennych 1-bitowych). Bity od M8000 do M8511 zwane są znacznikami (flagami) systemowymi gdyż mają zdefiniowane znaczenie (np. **M8000 przyjmuje stan „1"** zawsze gdy sterownik pracuje

natomiast **M8002 przyjmuje stan „1" tylko w pierwszym cyklu** po uruchomieniu sterownika.

W języku drabinkowym używamy następujących elementów

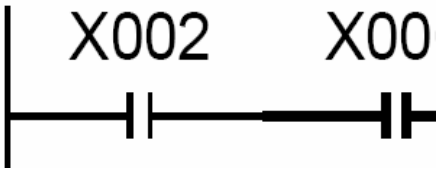
Instrukcja	Symbol drabinkowy	Opis instrukcji
LD		**LoaD** **Rozpoczyna połączenie - załączana sygnałem logicznym ' 1 '**
LDI		**LoaD Inverse – negacja** **Rozpoczyna połączenie - załączana sygnałem logicznym ' 0 '**
LDP		**LoaD Pulse** **Rozpoczyna połączenie - załączana zboczem narastającym**
LDF		**LoaD Falling pulse** **Rozpoczyna połączenie - załączana zboczem opadającym**
OUT		**Output** **Wysterowanie wyjścia (cewki),**

Możliwości sterownika wzbogaca szereg (dla FX3U 209) instrukcji aplikacyjnych. Ogólną postać instrukcji dla języka drabinkowego w programie GX developer przedstawiono poniżej

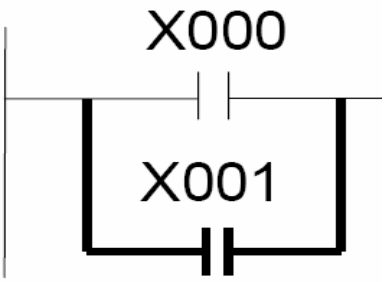


Podstawowe funktory logiczne realizujemy poprzez wzajemny układ styków

AND -iloczyn logiczny

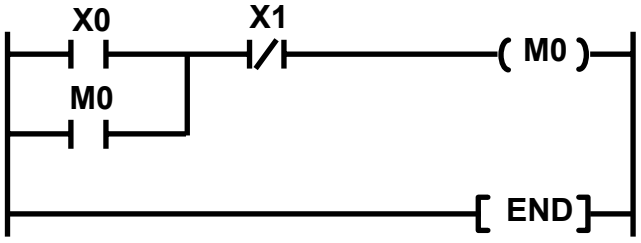


OR – suma logiczna



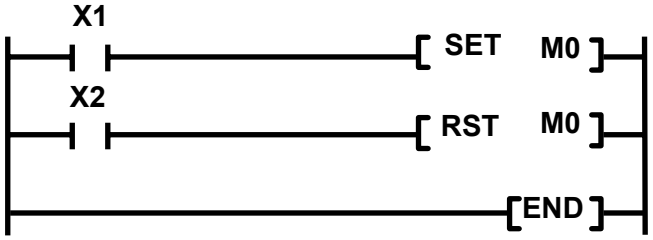
Uwaga: Zastanów się jak zrealizować funkcję NOR ,NAND i XOR – na zaliczenie przygotuj się z zapisu funkcji logicznych (przełączających) w języku drabinkowym

Przykłady programów:



Przycisk X0 służy do załączenia urządzenia a X1 do jego wyłączenia obydwa są niestabilne w związku z tym wykorzystamy nowa zmienna M0, która przyjmuje stan „1” po chwilowym załączeniu styku X0 i stan „0” po chwilowym załączeniu styku X1. Jest to odpowiednik przerzutnika RS. Jeżeli jest to potrzebne to zamiast przekaźnika pomocniczego M0 możemy bezpośrednio załączyć wyjście np. Y0.
Uwaga: Zastanów się jak wykorzystując metodę drabinkową zrealizować przerzutnik SR

Powyższy program możemy również zrealizować używając instrukcji



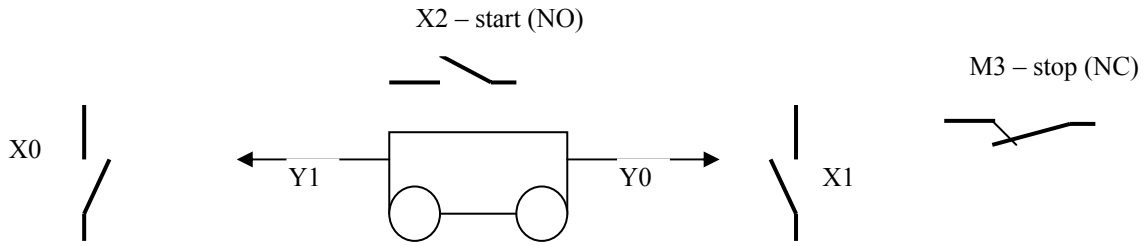
Żeby zabezpieczyć się przed niezamierzonym załączeniem spowodowanym np. zacięciem przycisku X0 korzystniej używać go, jako styk impulsowy 

Uwaga: Cewka o tym samym numerze w jednym przebiegu programu nie powinna występować więcej niż jeden raz. Załączenie cewki lub instrukcji bezpośrednio do linii zasilania jest traktowane jako błąd. Jeżeli dana linia ma być zawsze w stanie „1” to zastosuj styk specjalny M8000.

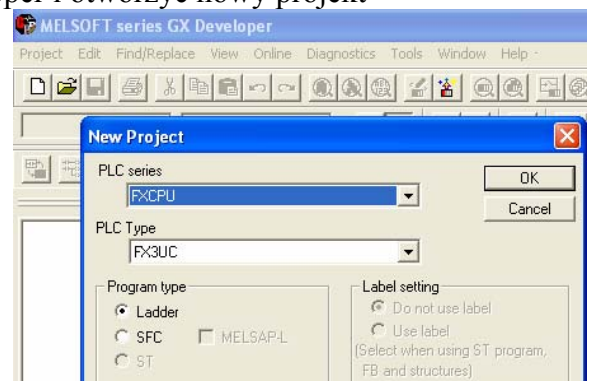
4. Cwiczenia

Zadanie 1. Używając tylko podstawowych elementów języka drabinkowego napisać program realizujący następujące zadanie:

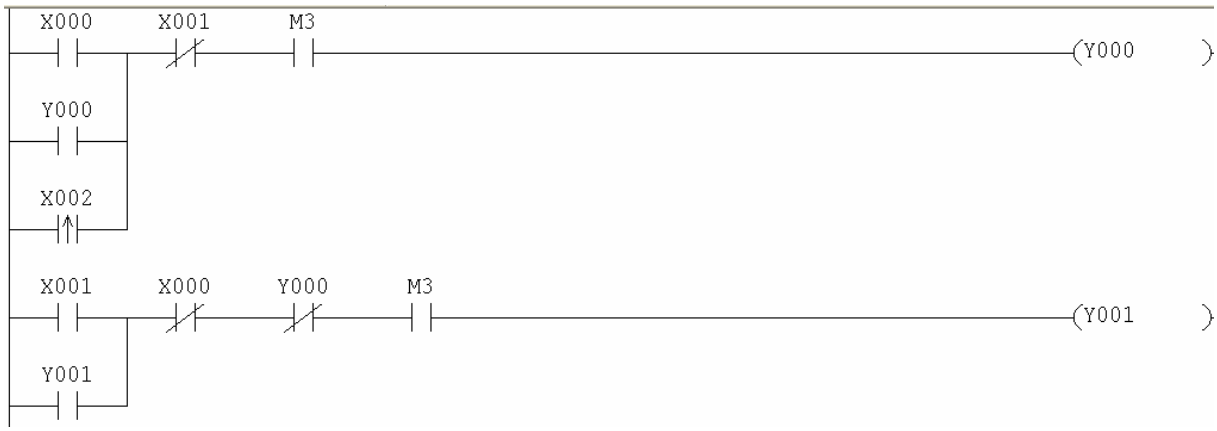
Wózek napędzany silnikami załączanymi z wyjść Y0 i Y1 porusza się pomiędzy krańcówkami X0 i X1. Start ruchu w kierunku Y0 przyciskiem X2 a zatrzymanie przyciskiem M0 (normalnie zwarty). Wszystkie przyciski niestabilne.



Uruchomić Gx developer i otworzyć nowy projekt



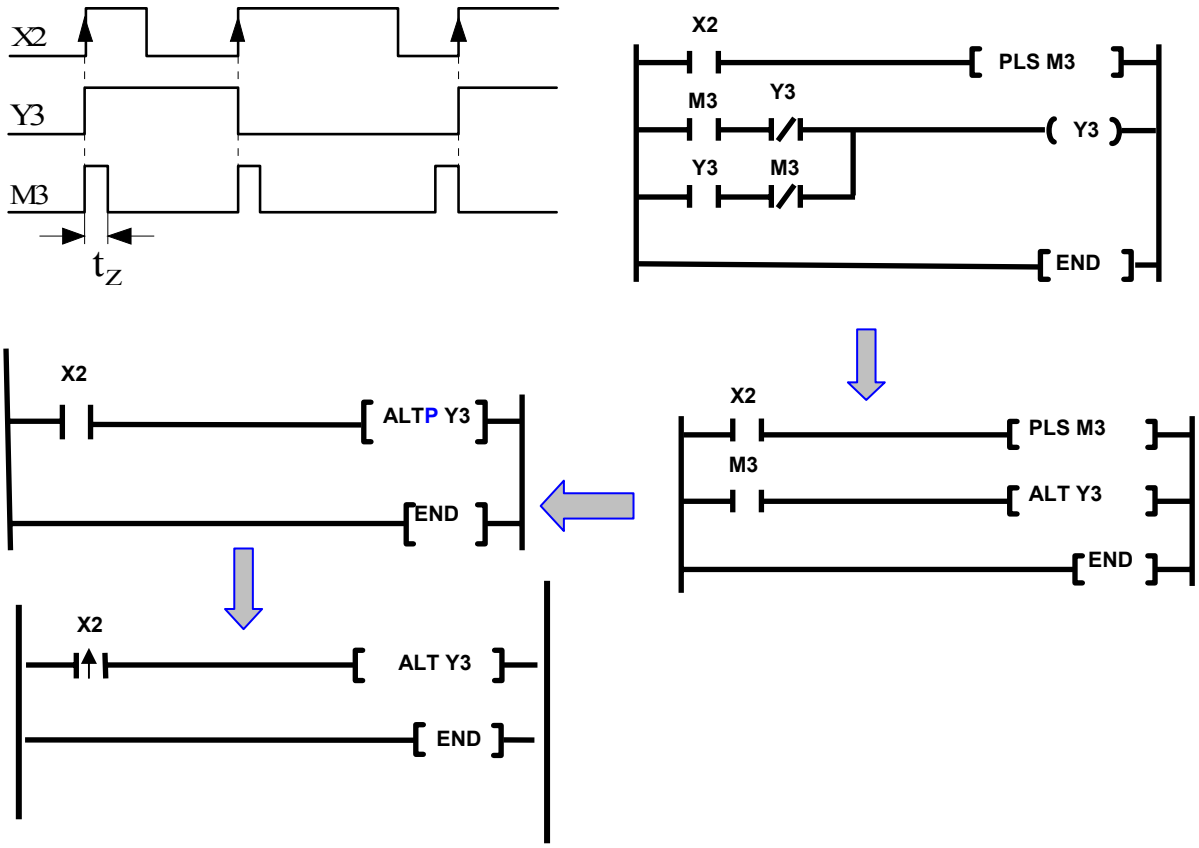
Rozwiązanie:
Korzystając z symboli drabinki napisać następujący program. Program piszemy od lewej do prawej.



Po zakończeniu edycji skonwertować program „F4" lub opcja „Convert – convert"
Aby sprawdzić działanie programu wgrywamy go do sterownika:
Opcje **Online / Write to PLC -** zaznaczamy MAIN i wciskamy przycisk Execute

Uwaga:
Krańcówki zgodnie z zasadami powinny występować, jako normalnie zamknięte do zatrzymania ruchu i normalnie otwarte to uruchomienia ruchu w przeciwną stronę. Zastanów się jak zmienić powyższy program zakładając, że krańcówki normalnie zamknięte nazywają się M0 i M1.

Przykłady programowania – flip-flop



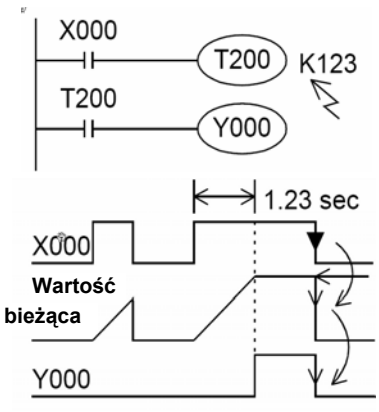
Uwaga:
Zastanów się jak zmienić program z zadania 1, gdy do załączenia i wyłączenia wózka używamy jednego przycisku (pierwsze naciśnięcie załącz a kolejne wyłącz).

Przekaźniki czasowe

Liczniki czasu (TIMER) odpowiadają przekaźnikom czasowym w układach przekaźnikowych ich działanie polega na zliczaniu impulsów o określonym okresie. Okres impulsów przypisany jest do Timer'ów w nastepujący sposób:
T0 do T199 – 100ms, T200 do T245 – 10ms, T246 do T249 – 1ms, T250-T255 – 100ms (z pamięcią), T256-T511- 1ms.

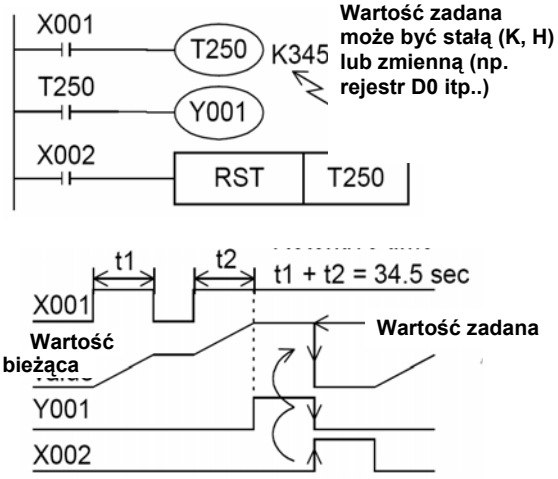
Zasada działania przekaźników czasowych:
Przekaźniki czasowe dzielimy na zwykłe, które kasują się po rozłączeniu styku załączającego i z pamięcią, które zatrzymują liczenie i kontynuują je po ponownym załączeniu

Timer'y zwykłe



Kasowanie timera poprzez rozłączenie gałęzi załączającej lub instrukcję [RST T200]

Timer'y z pamięcią



Wartość zadana może być stałą (K, H) lub zmienną (np. rejestr D0 itp..)

Kasowanie timera z pamięcią tylko poprzez instrukcję [RST T250]. Stan timera pamiętany również po wyłączeniu napięcia zasilania

Zadanie 2
Napisać program realizujący po załączeniu przycisku X0 załączenie Y0 na 3 sekundy

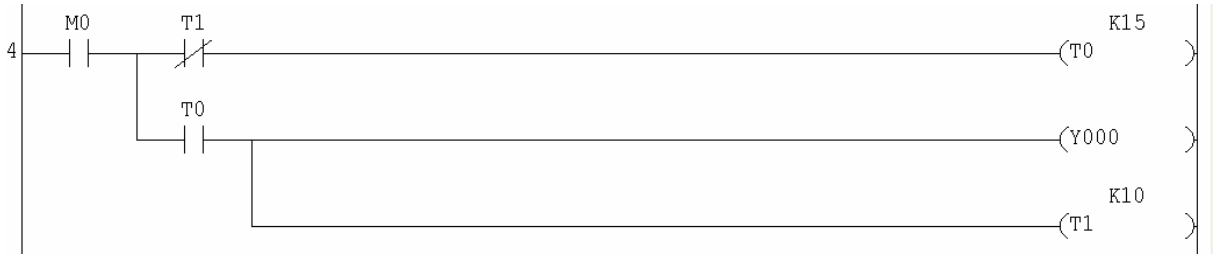
Lub

Zadanie 3

Napisać program realizujący po załączeniu przycisku X0 załączenie Y0 na 2 sekundy z opóźnieniem 3 sekundy

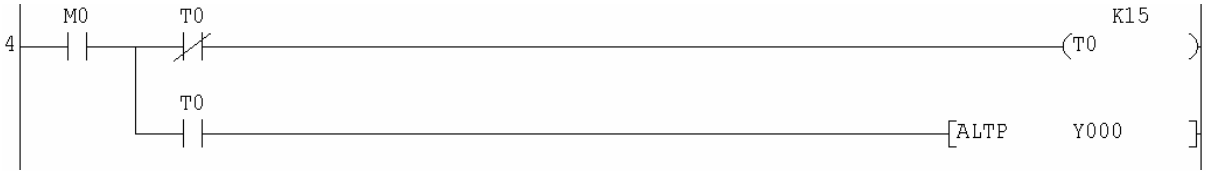
Sprawdzić, co zmieni się w działaniu programu, jeżeli **zamiast RST M0 wpiszemy RST T0**
Taki program będzie realizował generator sygnału prostokątnego o czasach „0”=T0 i „1”=T1

Taki generator można również zrealizować wykorzystując do kasowania timera T0 styk zamknięty timera T1

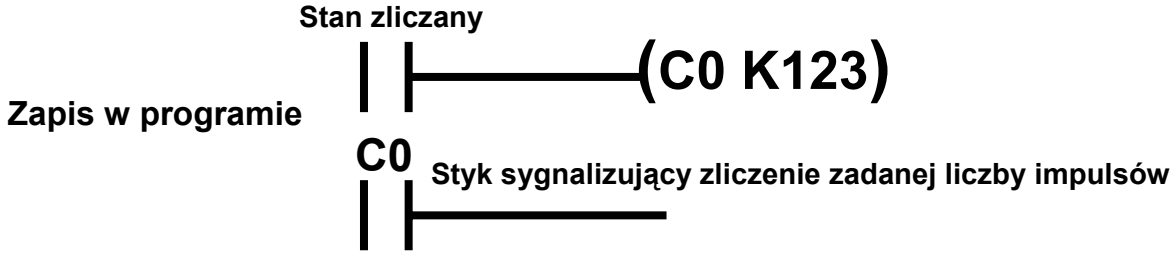


Jeżeli chcemy zrobić generator impulsów o jednakowym czasie jedynki i zera możemy wykorzystać pojedynczy timer

Lub (początek jak wyżej)



Licznik zdarzeń (Counter) – Dzielimy na zwykłe i szybkie. Liczniki zwykłe zliczają impulsy, których czas trwanie jest nie krótszy niż czas cyklu programu. Jako liczniki zwykłe używamy liczników C0 do C199.



Pamietać, że counter należy skasować instrukcją [RST C0]

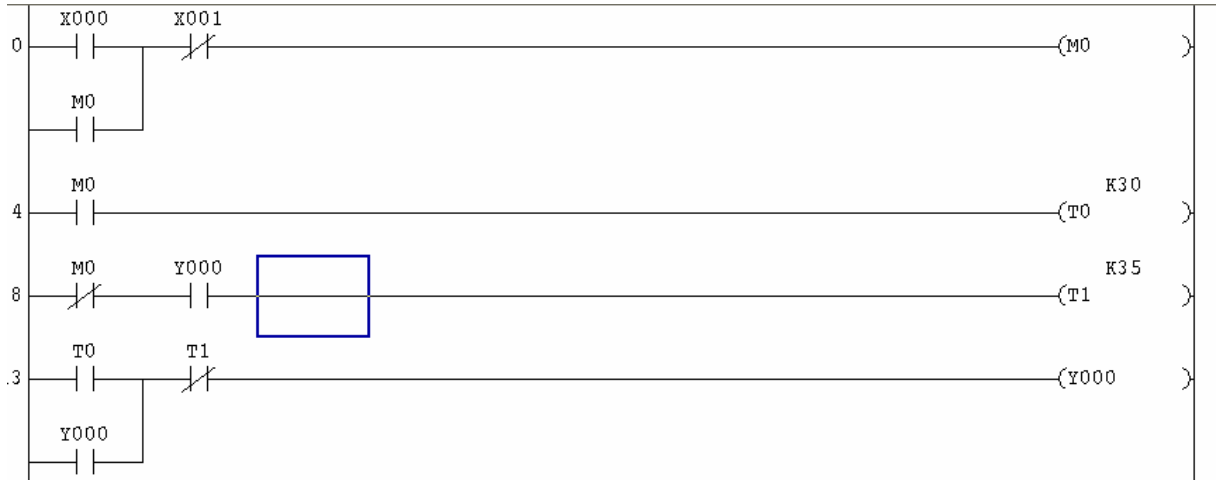
Zadanie 4

Załóżmy, że generator przebiegu prostokątnego z zadania 3 ma wykonać 5 impulsów

Zadanie 5

Zrealizować program realizujący opóźnienie załączenia i opóźnienie wyłączenia, Załączenie niestabilnym przyciskiem X0 a wyłączenie przyciskiem X1

Sposób 1



lub