Przekaźniki, styczniki, elementy sterownicze

Martyna Żukowska 252877

Maciej Barna 252931

Szymon Kordek 252935

grupa E13-01a, Pn 8:00 TN

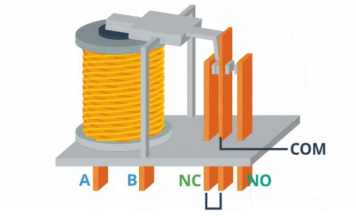
1 Cel pracy

Celem ćwiczenia jest zapoznanie się z metodami zabezpieczenia instalacji elektrycznych, zapoznanie się ze sterownikiem LOGO oraz poznanie podstaw języka FBD.

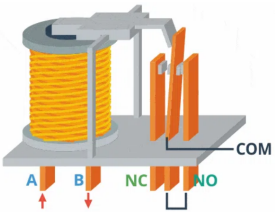
2 Wstęp teoretyczny

2.1 Przekaźnik

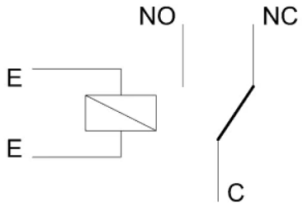
Przekaźnik to urządzenie służące do załączania sygnałów. Składa się z cewki przełączającej oraz trzech styków: COMMON (wspólny), NC (normaly closed) oraz NO (normaly open). Podając prąd na cewkę przyciąga ona ruchowy element, który zaczyna zwierać wyjście COMMON z NO zamiast NC. Poniżej przedstawiono ilustrację przekaźnika oraz jego symbolu:



(a) Cewka wyłączona



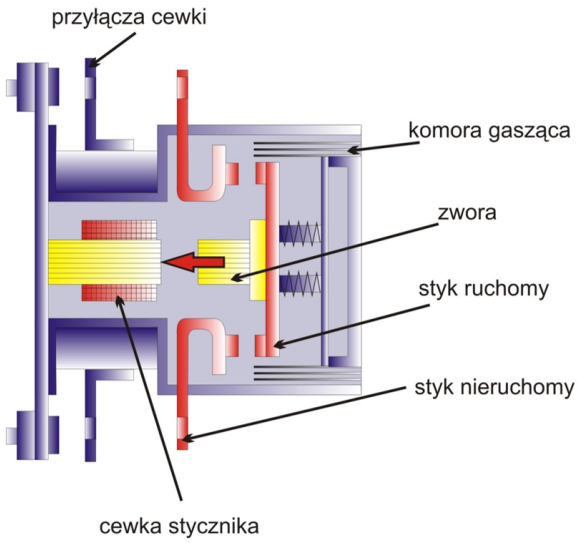
(b) Cewka włączona



(c) Schemat przekaźnika

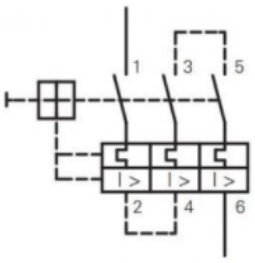
2.2 Stycznik

Stycznik budową i zastosowaniem jest bardzo zbliżony do przełącznika. Przełącznika używa się do przełączania sygnałów sterujących i prądów o małym natężeniu, natomiast styczniki wykorzystuje się tam, gdzie potrzeba przełączyć większy prąd np do zasilenia silnika. Z różnicy w zastosowaniu wyinkają także różnice w budowie. Stycznik ma grubsze łączenia metalowe oraz dwa punkty rozłączenia zamiast jednego, co wpływa także na jego gabaryty. Poniżej przedstawiono schemat ideowy stycznika:

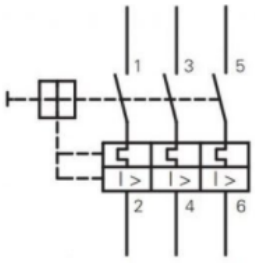


2.3 Wyłącznik silnikowy

Wyłącznik silnikowy jest zabezpieczeniem używanym do ochrony silnika przed przeciążeniem, zwarciem oraz zanikiem fazy w trójfazowym zasilaniu. Posiada wewnątrz wyzwalacz termiczny (termik), który reaguje w przypadku przeciążenia lub zaniku fazy oraz wyzwalacz elektromagnetyczny chroniący przed zwarciem. Wyłącznik silnikowy montuje się przed stykami aby chronić także je w przypadku przeciążenia lub zwarcia. Warto dodać, że wyłącznik ustawia się na prąd znamionowy silnika. Poniżej przedstawiono symbole wyłącznika dla jedno- i trójfazowego silnika:



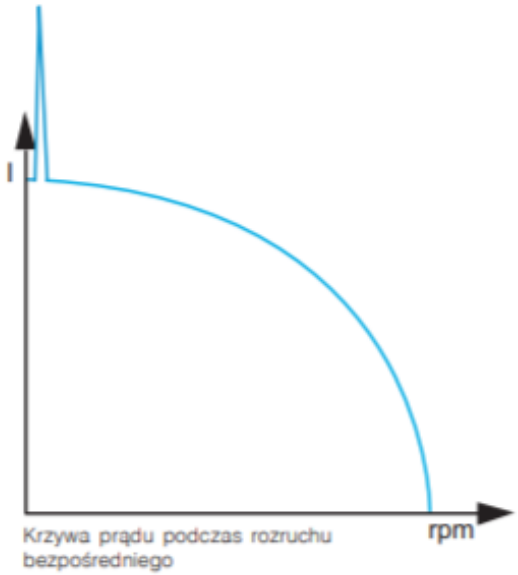
1-fazowy silnik



3-fazowy silnik

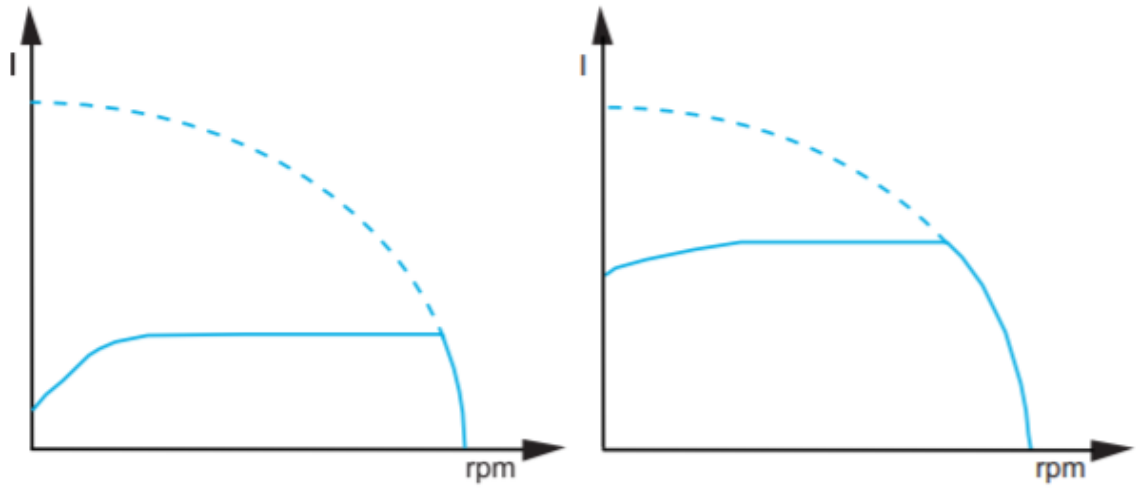
2.4 Układ softstart

Podczas rozruchu silnik pobiera znacznie większy prąd niż w trakcie normalnej pracy. Momentami jest on w stanie osiągnąć wartości od 6 do 20 razy większe niż prąd znamionowy (źródło, str 13). Poniżej przedstawiono wykres przykładowego poboru prądu w trkcie rozruchu silnika (źródło, str 13):



Krzywa prądu podczas rozruchu bezpośredniego

Przy takich wartościach prądu silnik potrzebuje dodatkowych zabezpieczeń. Jednym ze stosowanych rozwiązań jest softstart. Jego zadaniem jest ograniczenie stopniowe zwiększanie napięcia przykładanego do silnika, co przekłada się na stopniowy wzrost prądu. Silnik powoli zaczyna się rozpędzać do momentu, w którym prąd osiągnie wartość znamionową. Poniżej wykres przedsawiający pobór prądu w czasie rozruchu z zastosowaniem układu softstart (źródło, str 19):



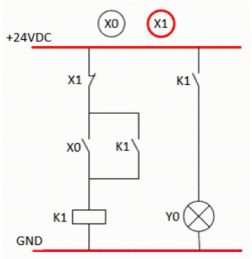
Krzywe momentu obrotowego i prądu podczas rozruchu softstartem silnika pod niskim obciążeniem i silnika pod pełnym obciążeniem.

Ponieważ silnik nie wyrywa się na samym początku tylko stopniowo zwiększa prędkość obrotową, wszystkie elementy podłączone do niego są powoli naprężane co powoduje, że zużywają się wolniej i rzadziej wymagają serwisu, co przekłada się na zmniejszenie kosztów eksploatacji.

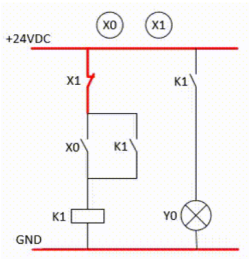
Układ softstartu można także zastosować do realizacji softstopu, dzięki czemu unikniemy gwałtownych zmian w układzie napędowym po odcięciu zasilania. Układ jest stopniowo wyciszany aż zatrzyma się całkowicie.

2.5 Układ załączania z podtrzymaniem

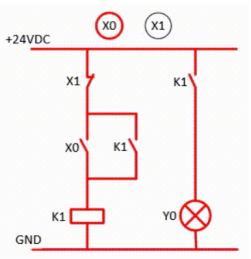
Jest to układ zabezpieczający czynne elementy instalacji elektrycznej (np silniki) przed uszkodzeniem w wyniku tymczasowej utraty zasilania. W przypadku gdy na hali produkcyjnej zostanie nagle odcięte zasilanie, a następnie zostanie przywrócone, układ ten sprawia, że chwilowa utrata napięcia rozłącza styki i trzeba ręcznie uruchamiać każdą maszynę. Dzięki temu unika się włączania wszystkich urządzeń na raz co mogłoby przeciążyć sieć oraz daje gwarancję, że maszyna została wyłączona na dobre i nie uruchomi się spontanicznie po powrocie zasilania w najmniej oczekiwanym momencie, co podwyższa bezpieczeństwo pracowników. Układ składa się z dwóch włączników X_0 i X_1 oraz cewki K_1 załączającej dwa styki K_1. Do układu jest również podłączone obciążenie Y_0. Poniżej prezdstawiono schemat na rysunku (a), a na kolejnych widać działanie układu:



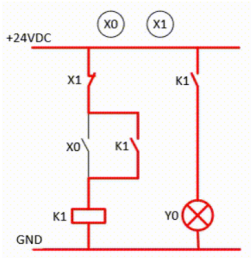
(a) Cewka wyłączona



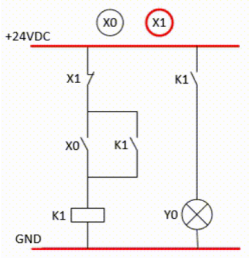
(b) Cewka włączona



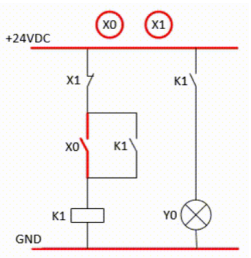
(c) Schemat przekaźnika



(d) Cewka wyłączona



(e) Cewka włączona



(f) Schemat przekaźnika

Opis działania układu:

(a) Przedstawia schemat układu.

(b) Po włączeniu X_1 układ pozostaje wyłączony.

(c) Po włączeniu X_0 układ się załącza. Cewka K_1 zwiera dwa styki K1.

(d) Po rozłączeniu X_0 układ pozostaje włączony dzięki cewce, która zwiera styk równoległy do X_0.

(e) Po rozłączeniu X_1 lub utracie zasilania styki K_1 się rozwierają i układ przestaje przewodzić.

(f) Samo załączenie X_0 przy wyłączonym X_1 nie uruchamia układu.

2.6 LOGO

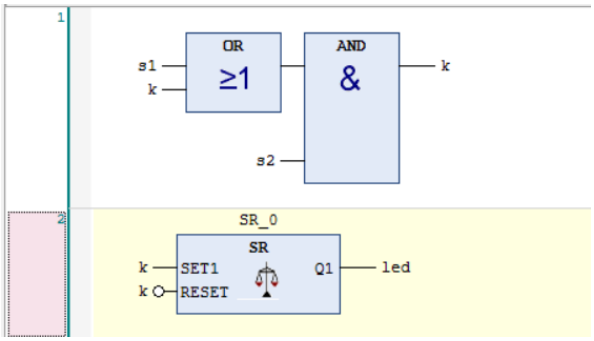
Jest to uniwersalny sterownik logiczny opracowany przez firmę Siemens, służący do przełączania i sterowania w zastosowaniach domowych i przemysłowych.Jest on zastąpieniem tradycyjnego sterowania wykonanego w oparciu o przekaźniki i styczniki. Działa on przy pomocy bloczków funkcyjnych, które w prosty sposób łączy się ze sobą tworząc w ten sposób program. Ten sterownik można sterować kożystając z przycisku i monitora znajdujących się na sterowniku, jak i przy pomocy programu Logo!. Sterownik Logo posiada port internetowy ethernet, dodatkowo nowe Logo 8.3 domyślnie z poziomu środowiska Logo Soft Confort można połączyć z chmurą i przesyłać do niej dane, bez żadnych dodatkowych serwerów/ ruterów.

Rys. 2.3. Siemens Logo 12/24RC

3 Ćwiczenia

3.1 Układ załączania z podtrzymaniem

Program powinien polegać na tym, że jeżeli w stanie spoczynku wciśniemy przycisk S2 to led się nie zapali, dopiero po wciśnieciu przycisku S1 led się zapali i nie zgaśnie do momentu wyłączenia przycisku S2. Przy wcześniejszym wyłączeniu przycisku S1, bez wyłączenia S2 led powinnien dalej się palić. Działa to na zasadzie opisanej w puncie 2.5

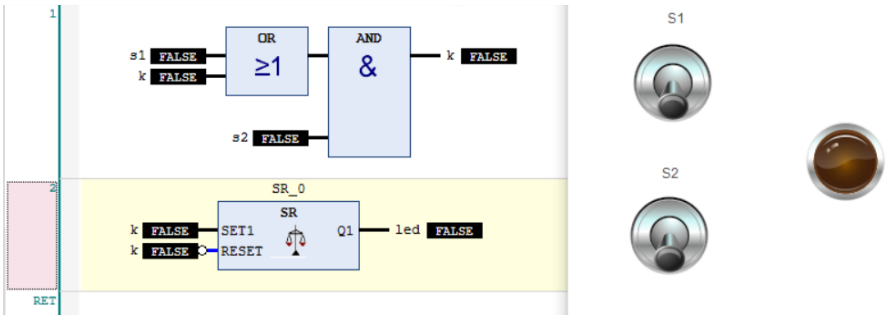


Rys. 3.1. Schemat programu

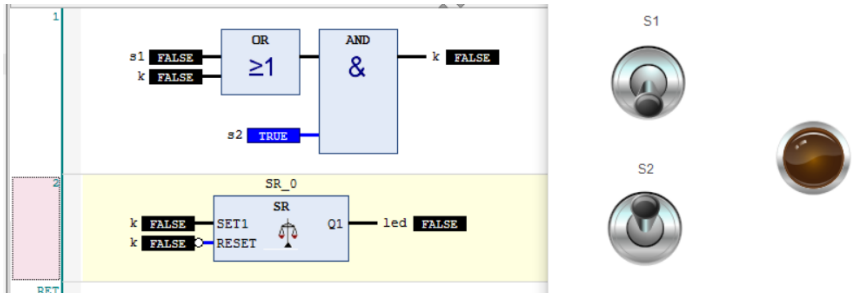
Program działa w prosty sposób. Używamy bloczka OR (lub): gdy przycisk S1 jest ustawiony na ”1” lub gdy zmienna k jest ustawiona na ”1” (ten element umożliwia działanie leda w późniejszym czasie mimo wyłączenia przycisku S1, ale i uzależnia też jego początkowe włączenie od wciśniecia S1) przejdź dalej i (bloczek AND) sprawdź czy przycisk S2 jest ustawiony na ”1”. Dopiero w takim momencie zmienna k jest ustawiana na ”1”.

Poniżej został dodany funkyjny bloczek SetReset, który większą wagę przywiązuje do elementu podawanego w wejściu SET1, niż RESET, co oznacza ze resetuje tylko w momencie kiedy SET1=”0” i RESET=”1”, a setuje zawsze gdy SET1=”1”. W tym przypadku na wejsciu SET1 dajemy zmienna k, a na RESET jej negacje, po to, żeby led który jest na wyjściu tego bloczka ustawiał się na ”1” do momentu ustawienia k na ”0”.

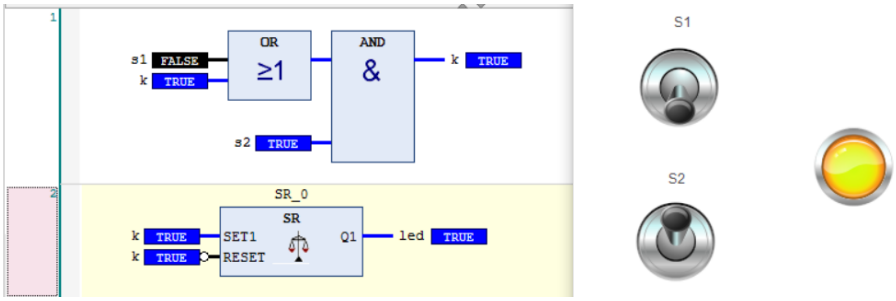
Przeprowadzony został test działania programu



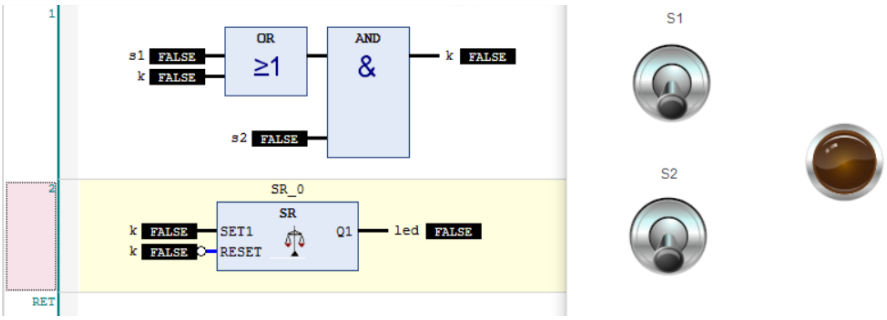
Rys. 3.2. Działanie programu w stanie spoczynku



Rys. 3.3. Działanie programu po wciśnięciu S2



Rys. 3.4. Działanie programu po wciśnieciu i odciśnięciu S1

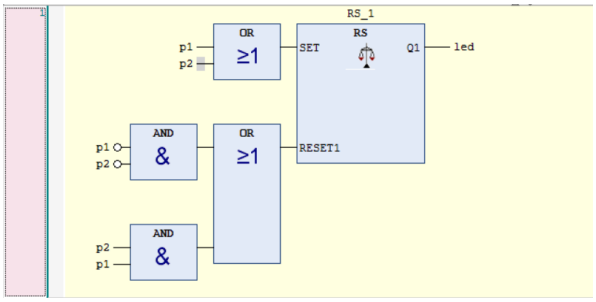


Rys. 3.5. Działanie programu odciśnięciu przycisku S2

Warto zauważyć, że led nie zapalił się do momentu wciśnięcia S1 po wciśnięciu S2, jak i że nie zgasł wcześniej niż po odciśnięciu S2, co świadczy o dobrze przeprowadzonym zadaniu.

3.2 Wyłącznik schodowy

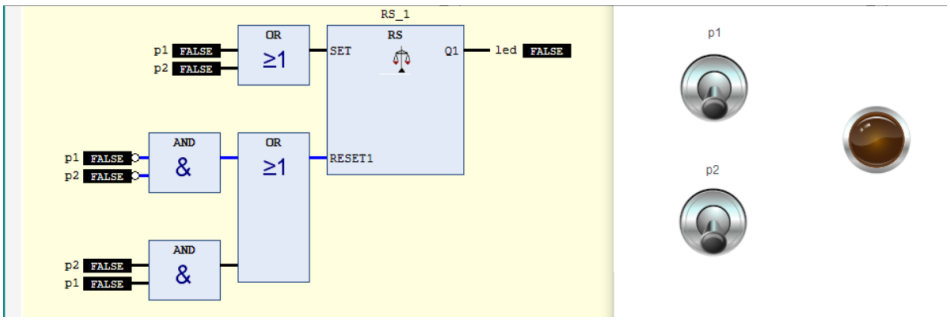
Zadanie polegało na stworzeniu układu, który miał działać jak instalacja światlna na klatce schodowej. To znaczy po zapaleniu światła na którymkolwiek piętrze zapala się ono na całej klatce i można je zgasić na każdym poziomie. Utoworzony do tego został poniższy schemat.



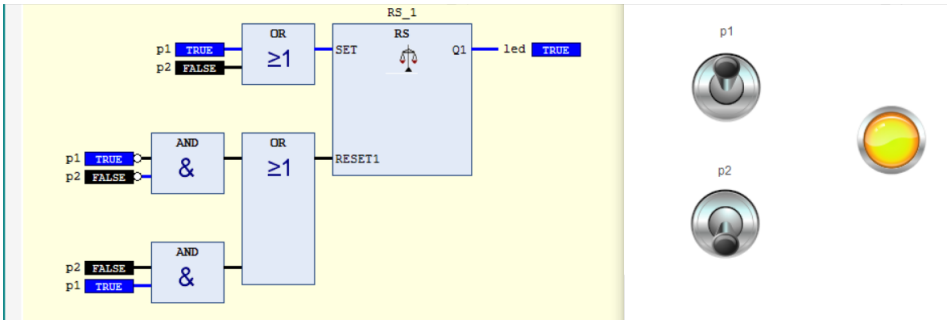
Rys. 3.6. Schemat układu

Polega on na tym ze zakladamy, że są 2 piętra na jednym jak przycisk p1 na drugim p2, światlo na całej klatce schodowej symbolizuje jedno wyjscie ”led”. W tym porogramie wykorzystany został bloczek funkyjny ResetSet, kóry działa podobnie jak SetReset opisany w zadaniu 1, z tą różnicą, że tym razem Reset ma większą wagę niż Set. Jednak tak samo jak wcześniej własnie ten bloczek ostatecznie decyduje o wartości zmiennej led.

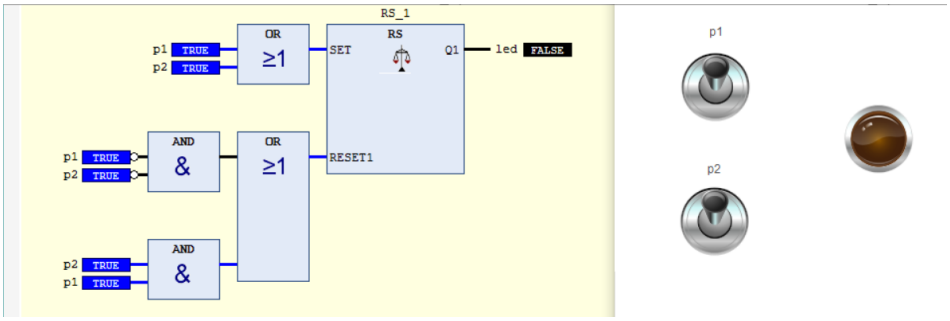
Na wejściu SET1 podłączony został bloczek OR (lub), który przepuszcza sygnał do SET1 w momencie, gdy przycisk p1 lub p2 jest ustawiony na stan ”1”.
Na wejściu RESET1 również został podłączony bloczek OR (lub), jednak z bardziej rozbudowanymi do niego wejściami o bloczek AND (i). Bloczek funkcyjny ma się resetować w momencie gdy przycisk p1 i p2 są ustawione na stan ”0” lub gdy są one ustawione na stan ”1”
Przeprowadzony został test działania programu



Rys. 3.7. Działanie w stanie spoczynku



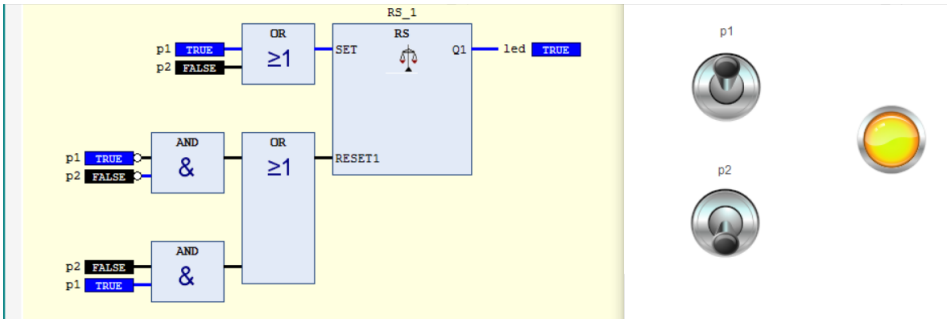
Rys. 3.8. Działanie programu po zapaleniu światła na dole (włączenie przycisku p1)



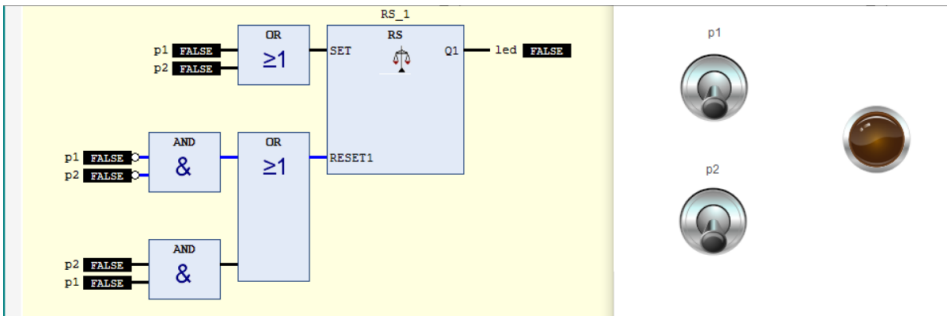
Rys. 3.9. Działanie programu po zgaszeniu światła na górze (włączenie przycisku p2)

I w tym momencie są możliwe 2 przypadki

1. Zapalanie światła na pietrze (przycisk p2)

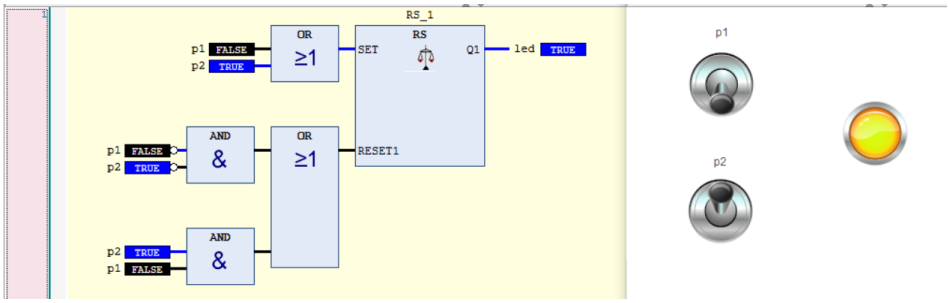


Rys. 3.10. Działanie po zapaleniu światła (w tym przypadku wyłaczenie przycisku p2)

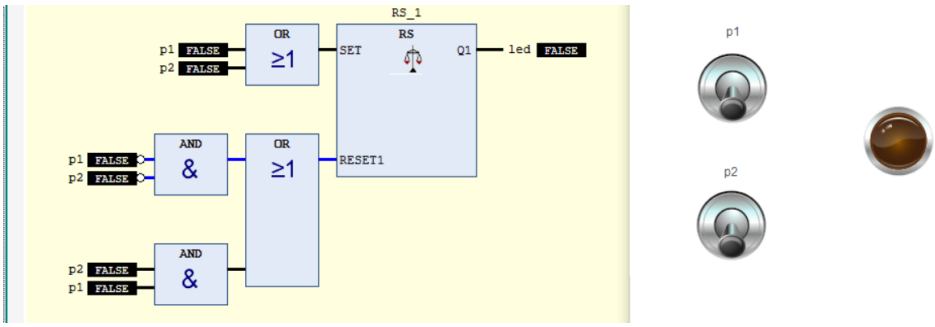


Rys. 3.11. Działanie po zgaszeniu światła na parterze (wyłaczenie przycisku p1)

2. Zapalenie światła na parterze



Rys. 3.12. Działanie po zapaleniu światła na parterze (wyłaczenie przycisku p1)

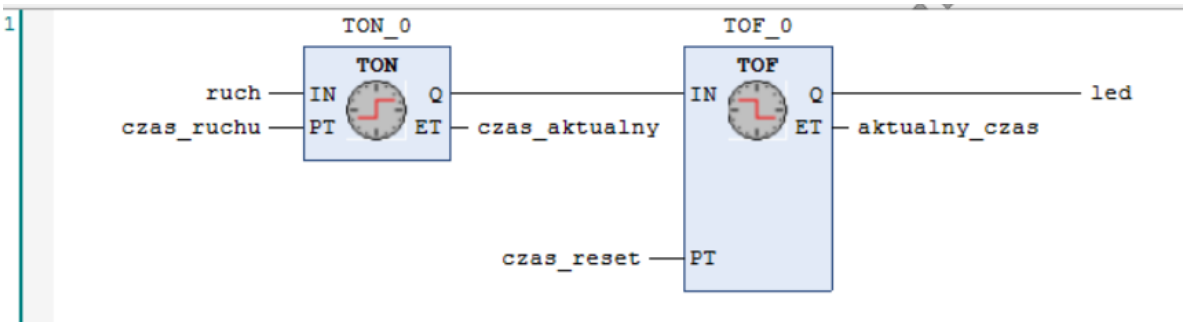


Rys. 3.13. Działanie po zgaszeniu światła na piętrze (wyłaczenie przycisku p2)

Program działa poprawnie.

3.3 Detekcja ruchu

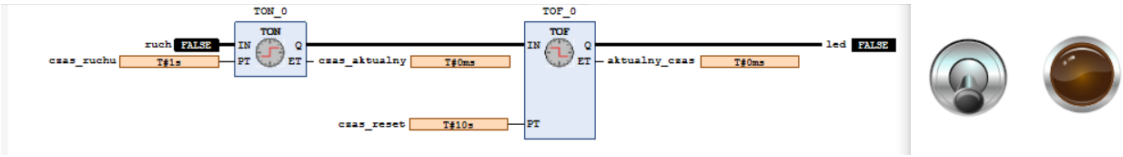
Zadanie polega na tym, aby utworzyć układ który zapala led po 1 sekundzie ciągłego ruchu i świeci sie przez cały czas jego trwania, a nawet po zakończeniu ruchu led powinnien się jeszcze świecić przez 10 sekund. Niestety na symulacji nie możemy skorzystac z czujnika ruchu, więc używać będziemy przycisku "ruch", którego stan ustawiony na "1" oznacza, że występuje jakiś ruch, a stan ustawiony na "0" oznacza jego brak.



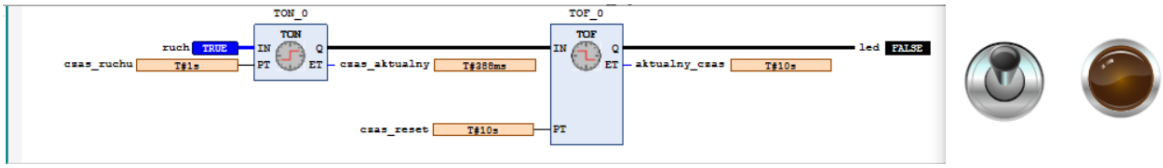
Rys. 3.14. Schemat układu

Do wykonania tego zadania wykorzystaliśmy dwa timery jeden TON i drugi TOF. Ten pierwszy ma zliczać czas od momentu ustawienia przycisku ruch na stan "1", do momentu odliczenia 1 sekundy zadeklarowanej w zmiennej czas_ruchu (T#1s), po tym czasie sygnał pójdzie dalej i uruchomi timer TOF_0, który przepuszcza sygnał dalej i ustawia stan led na "1".Timer TOF_0 uruchamia się dopiero po utracie sygnału wejściowego (czyli w naszym układzie, gdy ruch zostanie ustawiony na stan "0" - stop ruchu), wtedy przepuszcza on sygnał dalej jeszcze przez 10 s zadeklarowanych w zmiennej czas_reset

(T#10s), co powoduje ze led jest dalej ustawiony w tym czasie na stan ”1”, mimo że przycisk ruch jest ustawiony na stan ”0”.
Przeprowadzony został test działania programu

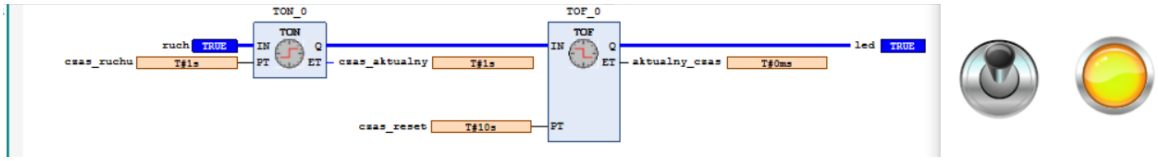


Rys. 3.15. Działanie układu w stanie spoczynku

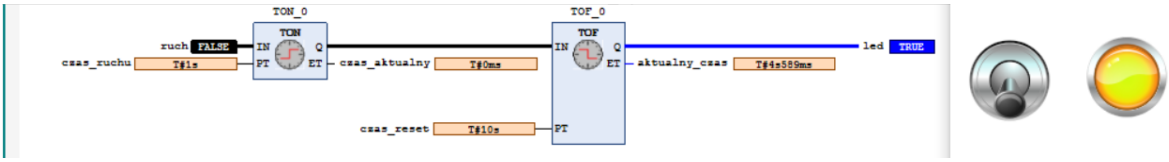


Rys. 3.16. Działanie układu po ruchu (mniejszym niż 1 sekunda)

Jak widać led się jeszcze nie pali bo ruch jest jeszcze za krótki.

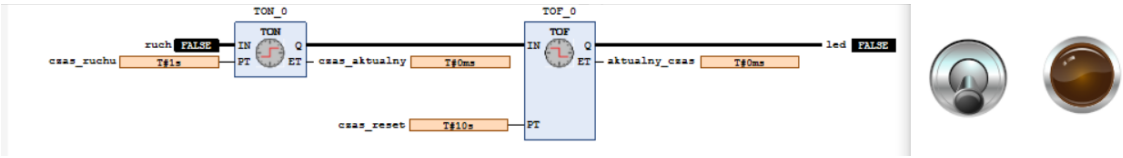


Rys. 3.17. Działanie układu po 1 sekundzie ruchu (led się zapala)



Rys. 3.18. Działanie układu po zakończeniu ruchu

Led dalej się pali mimo, że przycisk ruch jest ustawiony na stan ”0”. Timer zaczyna odliczanie 10 sekund.

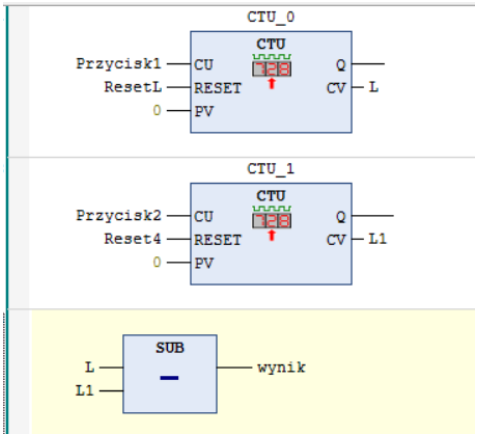


Rys. 3.19. Działanie układu po 10 sekundach braku ruchu (led gaśnie)

Program działa poprawnie.

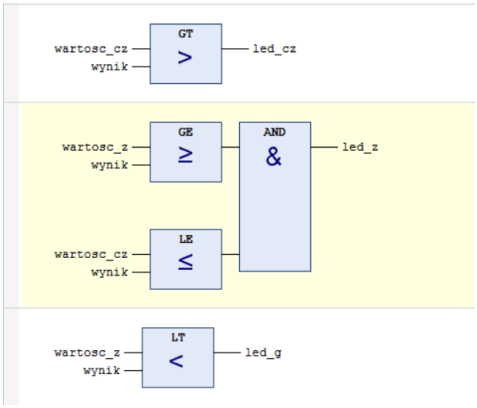
3.4 Licznik

Zadanie polegało na stworzeniu układu w którym można zwiększyć i zmiejszyć zmienną przy pomocy counterów. W momencie znalezienia się zmiennej w odpowienim zakresie powinnien zapalić się przypisany led.



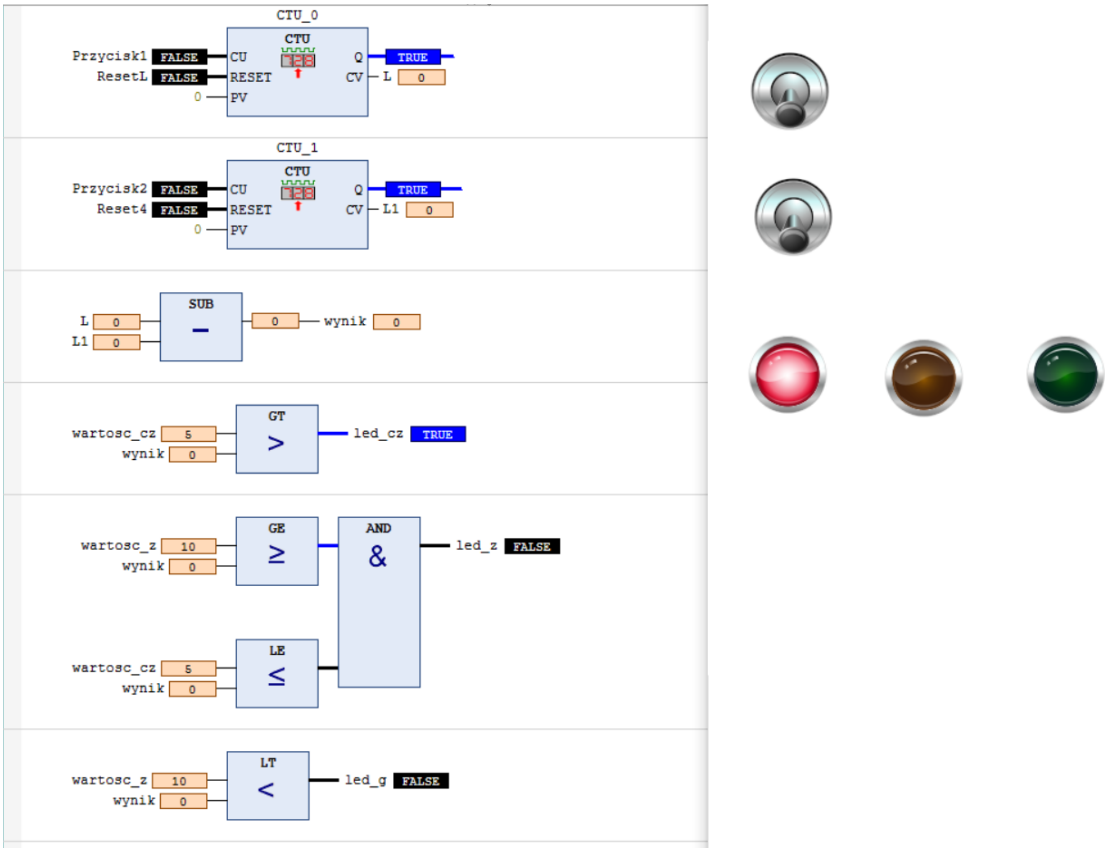
Rys. 3.20. Pierwsza część schematu układu

Zostały dodane dwa countery. Obydwa countery zostały ustawione tak aby przepuszczać sygnał w momencie przekroczenia 0. W ten sposób nalicza nam się wartość L i L1 za każdym razem gdy wciśniemy przycisk odpowiednio Przycisk1 i Przycisk2. Dodatkowo został dodany bloczek SUBTRACTION, który pozwoli nam na to, aby nasz wynik końcowy (czyli zmienna wynik) był równy różnicy L i L1. Zostało to wykonane po to aby w tym momencie Przycisk 1 zwiekszał wartość zmiennej wynik, a Przycisk2 analogicznie ją zmiejszał.



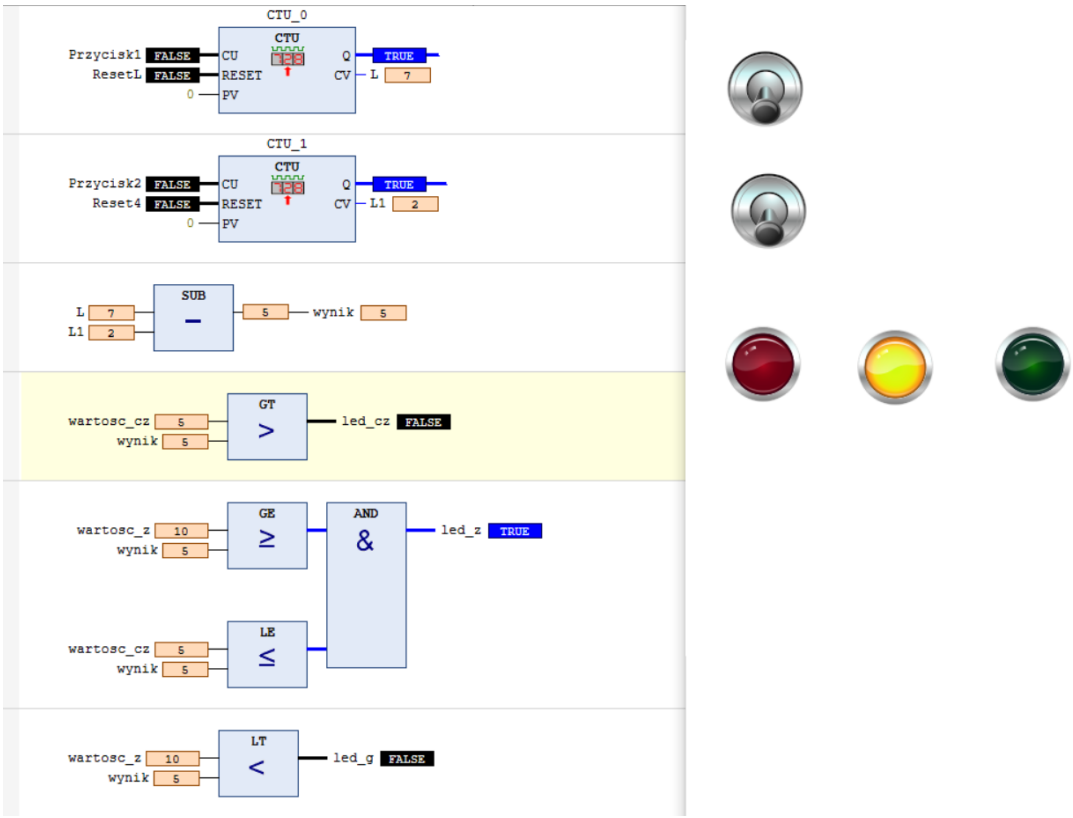
Rys. 3.21. Druga część schematu układu

W drugiej części schematu działamy już na zakresach w jakich ma się znajdować zmienna wynik, zapalając przy tym odpowiednie ledy. Zmienna wartosc_cz odpowiada za górną granicę zakresu odpowiadającego za świecenie leda czerwonego, jest on bowiem równy od minus nieskończoności do 5, dlatego został użyty do schematu bloczek GT który sprawdza czy zmienna wynik jest mniejsza niż zmienna wartosc_cz



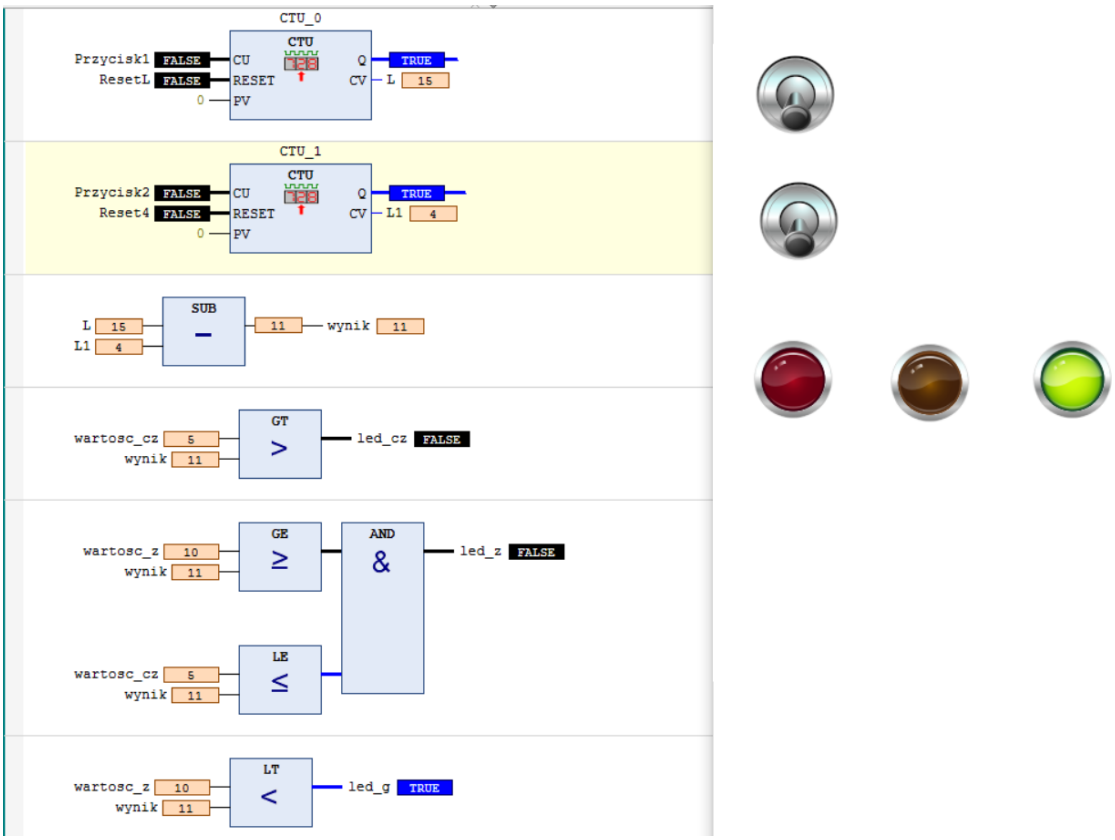
Rys. 3.22. Działanie programu na początku wartość wyniku=0 - zapalony czerwony led

Drugi zakres to zakres świecenia żółtej diody led jej zakres to 5-10, aby sprawdzić czy zmienna wynik do niego wpada wykorzystujemy bloczek AND (i) na którego wejściach są odpowiednio bloczki GE i LE, które działają w następujacy sposób: Gdy wynik jest mniejszy-równy wartosc_z (zdefiniowana na 10) i gdy wynik jest większy-równy wartość_c (z wcześniejszego zakresu świecenia czerwonego leda) wtedy ustaw stan led_z na "1"



Rys. 3.23. Działanie programu po siedmiokrotnym wcisnięciu Przycisku1 i dwókrotnym wciśnieciu Przycisku 2, powoduje wejście do zakresu 5-10 czyli zapala się zółty led

Ostatni etap programu to trzeci zakres od 10 do nieskończoności aby sprawdzić czy zmienna wynik do niego należy wykorzystujemy bloczek LT, który sprawdza czy wynik jest większy niż wartość_z (z wcześniejszego zakresu świecenia żółtego leda), gdy tak jest stan led_g ustawia się na "1"



Rys. 3.24. Działanie programu po piętnastokrotnym wcisnięciu Przycisku1 i czterokrotnym wciśnięciu Przycisku 2, powoduje wejście do zakresu od 10 czyli zapala się zielony led

Program działa poprawnie.

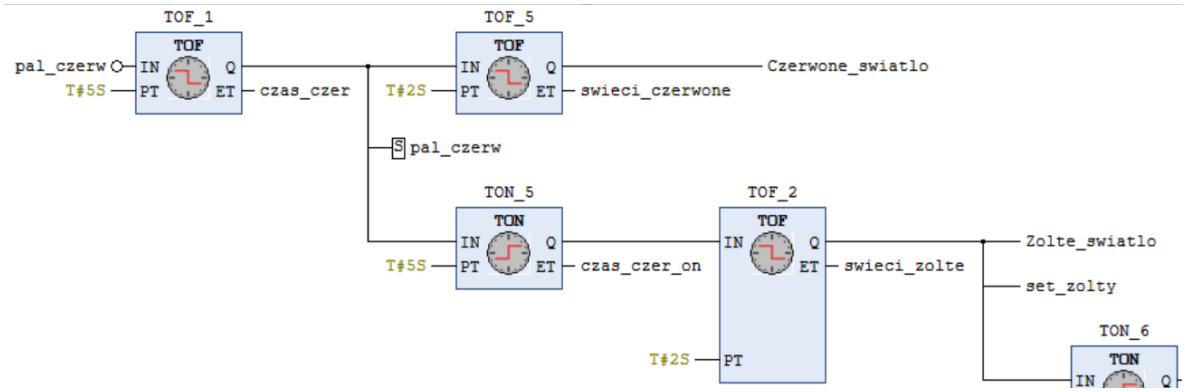
3.5 Sygnalizacja świetlna

3.5.1 Założenia i krótki opis

Zgodnie z założeniami przedstawionymi na zajęciach sygnalizacja świetlna powinna działać w następujący sposób. Na początku przez 5 sekund świeci się samo czerwone światło. Następnie przez 2 sekundy obserwujemy światło czerwone oraz żółte. Przez kolejne 4 sekundy pali się światło zielone. A na koniec zapalamy światło żółte na 2 sekundy. Po ostatniej akcji cały cykl powtarza się w nieskończoność.

3.5.2 Realizacja działania programu

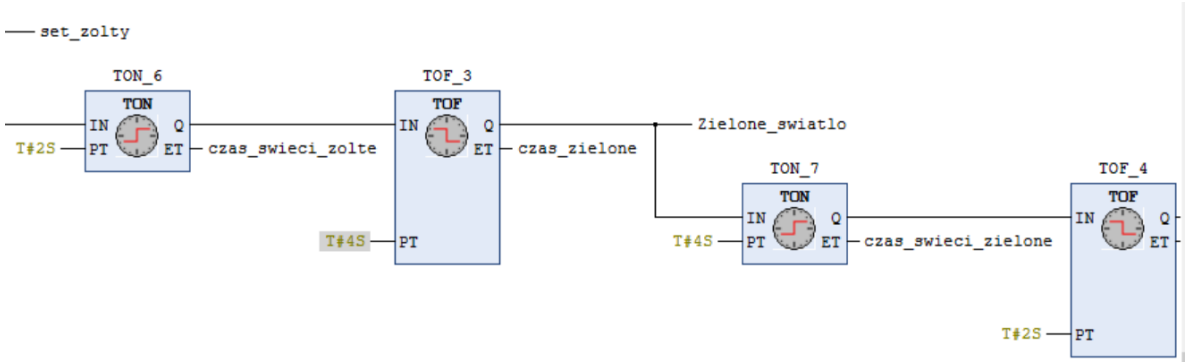
Aby zrealizować zadane zadanie podzieliliśmy je na mniejsze etapy, całość programu natomiast znajduje się w jednej linii.
Zadanie opiera się na odpowiednim wykorzystywaniu timerów typu TOF oraz TON.
Na początek zajmijmy się sposobem uruchamiania programu oraz pierwszymi 2 etapami - zapalaniem się czerwonego światła na 5 sekund i zapalania się po nim czerwonego z żółtym przez 2 sekundy.
Cały cykl zaczynamy od uruchomienia timera TOF utrzymującego sygnał przez 5 sekund. Położy nam do tego zmienna "pal_czerw". Zmienna ta bazowo ma wartość "0", dlatego na wejściu timera podajemy negację. Dzięki temu timer dostaje sygnał i zaczyna działać. Sygnał od razu po przejściu przez timer ustawia zmienną "pal_czerw" na "1" za pomocą funkcji "set" oraz przepuszcza sygnał przez następny timer i zapala "Czerwone_swiatlo". Przez ustawienie zmiennej "pal_czerw" na "1" pierwszy timer traci sygnał więc zaczyna się odliczanie trwające 5 sekund, jednocześnie załącza się timer "TON", mający za zadanie przepuścić sygnał dalej po upływie 5 sekund jego trwania. Po 5 sekundach działania sygnał z pierwszej linii znika. Sygnał przechodzi przez timer "TON" i uruchamia kolejny "TOF" trwający 2 sekundy i uruchamia "Zolte_swiatlo' oraz "set_zolty" (powód użycia tej zmiennej wytłumaczymy później). Dzięki timerowi "TOF_5" po minięciu 5 sekund czerwone światło pali się jeszcze przez 2 sekundy (razem ze światłem żółtym).



Rys. 3.25. Fragment pierwszych 2 etapów działania programu

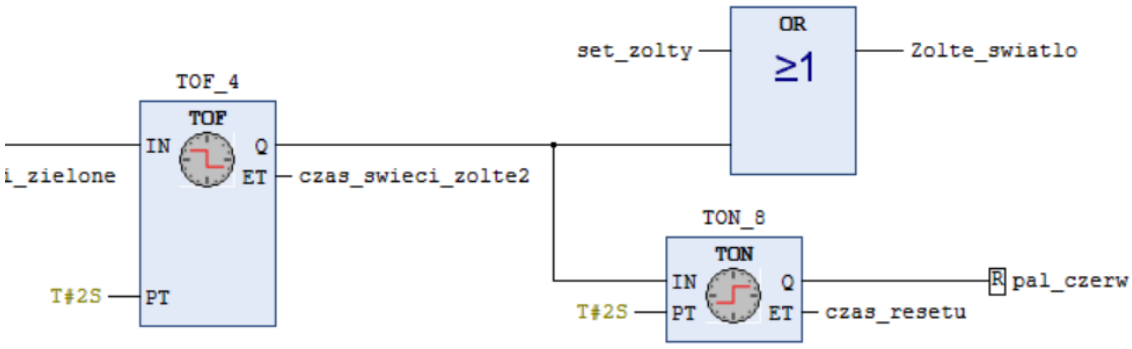
Następnym etapem będzie zapalenie zielonego światła. Po upłynięciu 2 sekund wyłącza się timer "TOF_2" i uruchamia się timer "TON_6". Przepuszcza on sygnał uruchamiając timer "TOF_3" służący uruchomieniu się "Zielonego_swiatla" i utrzymaniu go przez 4

sekundy. Po 4 sekundach sygnał prezchodzi dalej za pomocą timera "TON_7" i zaczyna się ostatni etap.



Rys. 3.26. Fragment 3 etapu działania programu

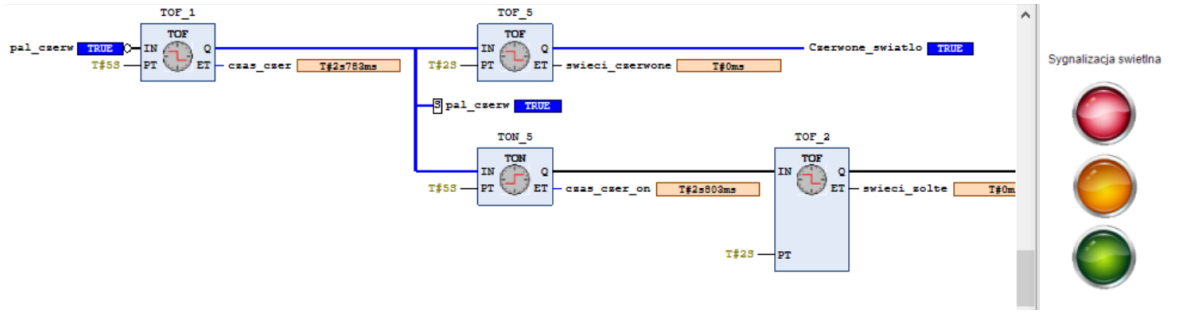
Ostatnim etapem programu jest zapalenie się samego żółtego światła na 2 sekundy i puszczenie cyklu od początku. Na tym etapie wyjaśnia się dlaczego przy zapalaniu żółtego światła razem z czerwonym w drugim etapie programu ustawiamy również zmienną "set_zolty" na "1". Zmienna "Zolte_swiatlo" jest używania w dwóch różnych miejscach programu. Normalnie powodowałoby to, że przy zapalaniu się żółtego światła wystąpiłaby sprzeczność. Zmienna "Zolte_swiatlo" miałaby jednocześnie przypisaną logiczną "1" oraz "0" w dalszej części programu. W naszym wypadku ten problem został rozwiązany za pomocą bloczka "OR" oraz dodatkowej zmiennej "set_zolty". Po upływie 2 sekund świecenia się żółtego światła sygnał zostaje przepuszczony przez timer "TON_8" i wygaszony przez timer "TOF_4". Po przepuszczeniu sygnału ten resetuje stan zmiennej "pal_czerw", tym samym przyjmuje ona logiczne "0" i uruchamia następny cykl programu.



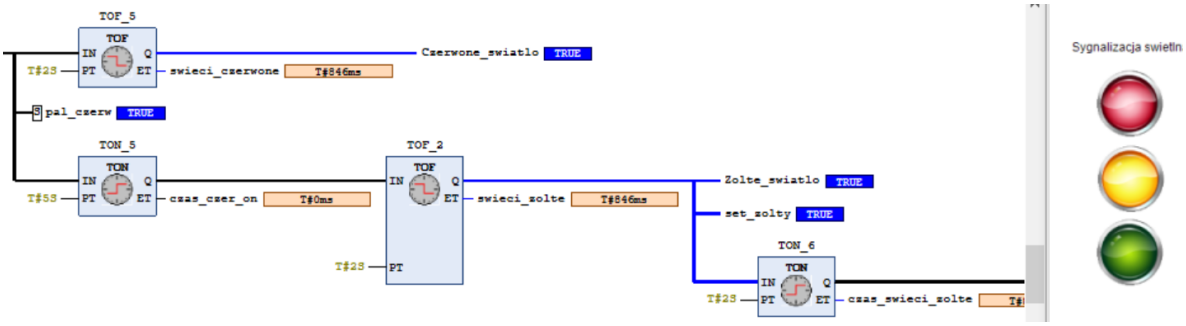
Rys. 3.27. Fragment 4 etapu działania programu

3.5.3 Symulacja

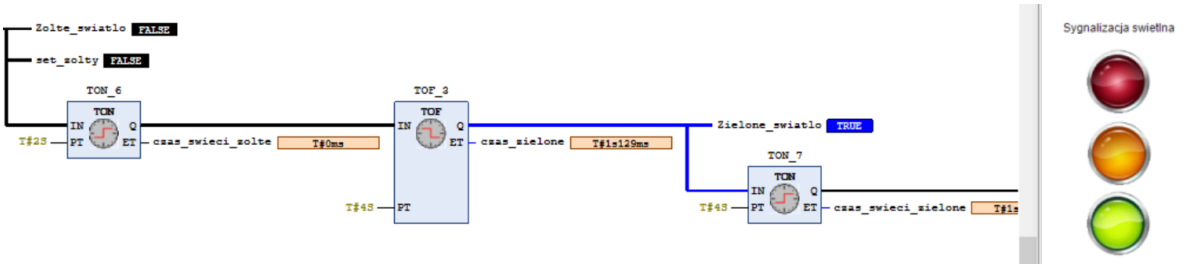
Na poniższych screenach został przedstawiony cały cykl działania programu. Zgodnie z kolejnością opisanych powyżej etapów.



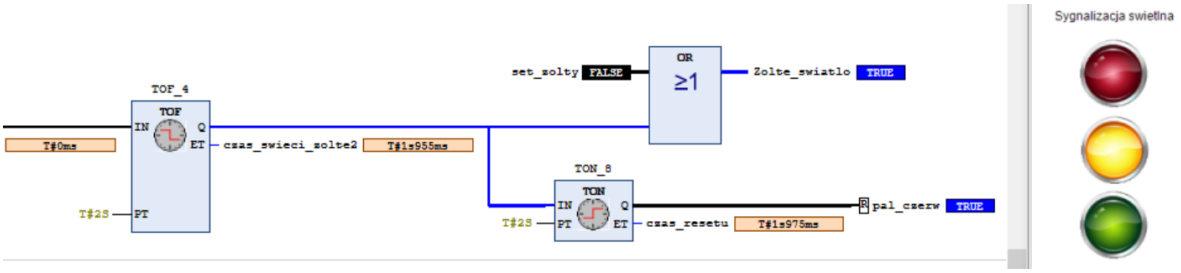
Rys. 3.28. Symulacja 1 etapu działania programu



Rys. 3.29. Symulacja 2 etapu działania programu



Rys. 3.30. Symulacja 3 etapu działania programu



Rys. 3.31. Symulacja 4 etapu działania programu

4 Wnioski

Język użyty w tym ćwiczeniu jest bardzo intuicyjny i dość podobny do laddera. Jest przyjemny w użyciu.
Wszystkie ćwiczenia udało się wykonać prawidłowo. W czasie zajęć nauczyliśmy się podstaw obsługi sterownika LOGO oraz poznaliśmy różne metody zabezpieczania układów i linii przed niepożadanymi zjawiskami takimi jak chwilowa utrata zasilania/fazy lub przeciążenie.

5 Bibliografia

- Styczniki: https://www.tim.pl/strefa-porad/Przekaznik-czy-stycznik—-na-co-zwrocic-uwage-Poruszamy-najwazniejsze-zagadnienia
- Wyłączniki silnikowe: https://automatykablog.pl/2019/10/natomiast-do-czego-sluza-wylaczniki-silnikowe-czesc-ii/
  https://portalprzemyslowy.pl/utrzymanie-ruchu-produkcja/elektryka-elektronika/wylaczniki-silnikowe-forma-skutecznej-ochrony-napedow/
- Softstart: https://library.e.abb.com/public/edb193ba078d901a482579b30042af9c/Softstarty%20kompendium%20wiedzy.pdf
- Układ załączania z podtrzymaniem: https://iautomatyka.pl/podstawy-sterownikow-plc-operacje-programu-drabinkowego/