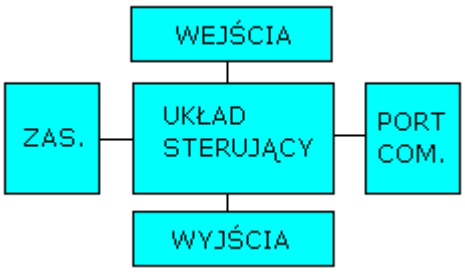
PROSTO o STROWNIKACH PROGRAMOWALNYCH

Sterowniki programowalne PLC mają już ugruntowaną pozycję wśród urządzeń technicznych, co wynika z wszechstronnych możliwości i łatwości ich wykorzystania. Warto więc ogólnie je poznać, aby móc ocenić przydatność do ewentualnych zastosowań.

Prosty sterownik programowalny zbudowany jest z :
- wejść wykrywających zmiany dwustanowe (binarne) lub analogowe (jaką wartość ma sygnał dołączony do wejścia)
- wewnętrznego mikroprocesorowego układu sterującego z programowalną pamięcią
- wyjść w formie styków przekaźnika lub tranzystorów sterowanych układem mikroprocesorowym

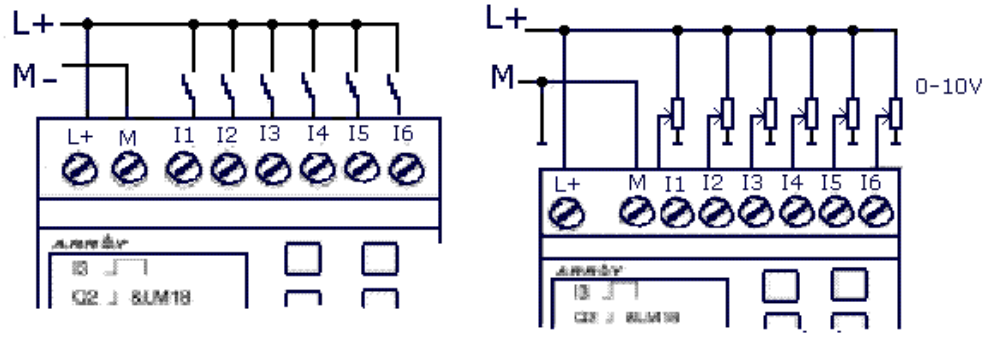
Dodatkowe układy to: wewnętrzny zasilacz dostosowujący sterownik do typowego zasilania 12-24V albo 230V, port do komunikacji z komputerem , niekiedy zegar czasu rzeczywistego RTC , wyświetlacz i klawiatura, różne przetworniki sygnałów.



Rys. Podstawowe bloki prostego PLC

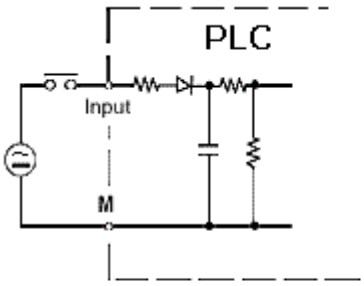
Co to są wejścia, wyjścia sterownika ?
/ nazywane punktami /

Wejścia to zaciski do których przewodami doprowadza się sygnał (np. napięcie, prąd).

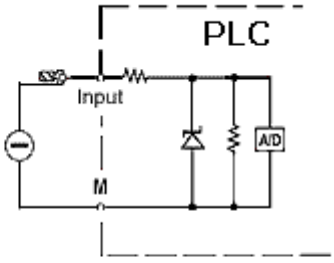


rys. Doprowadzenie do PLC sygnałów wejściowych dwustanowych i analogowych

Poniżej przedstawiono schematy prostych, niesymetrycznych **wejść** napięciowych.



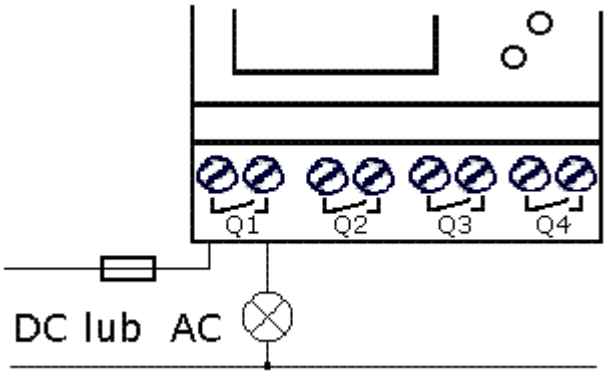
reprezentacja wejścia
dwustanowwgo AC lub DC



reprezentacja wejścia analogowego
typowo 0-10V

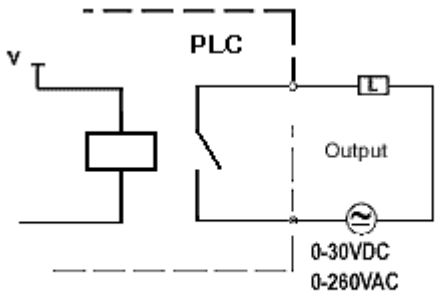
Obwody wejściowe sterownika różnią się, zależnie od tego , czy są dwustanowe (jest sygnał / nie ma), czy analogowe (rozpoznanie wartości sygnału) W pierwszym wypadku w PLC wystarczy zastosować dzielnik rezystancyjny, diodę prostowniczą lub chroniącą przed odwrotną polaryzacją, kondensator filtrujący. Następnie, sygnał bezpośrednio, albo przez układ optoizolacji, doprowadzany jest do wejścia procesora.
W drugim przypadku (we. analogowych), po dzielniku rezystancyjnym, sygnał doprowadzany jest do przetwornika analogowo cyfrowego A/D, który zamienia informację ciągłej wartości analogowej np. 0-10V, na postać cyfrową (odczytywaną przez procesor).
Dioda Zenera jest zabezpieczeniem przepięciowym i przed odwrotną polaryzacją
Pokazane wejścia są typu niesymetrycznego (sygnał względem masy), napięciowe (AC albo DC) i PNP (styki, czujniki załączają dodatnie napięcie względem masy).
W przypadku wejść prądowych, sygnałem jest wartość prądu. Wejście 0-10V można łatwo dostosować do pomiaru prądu 0-20mA, włączając równolegle do wejścia rezystor 500 omów.

Wyjścia sterownika, w prostym przypadku, to wyprowadzenia styków wewnętrznego przekaźnika PLC.

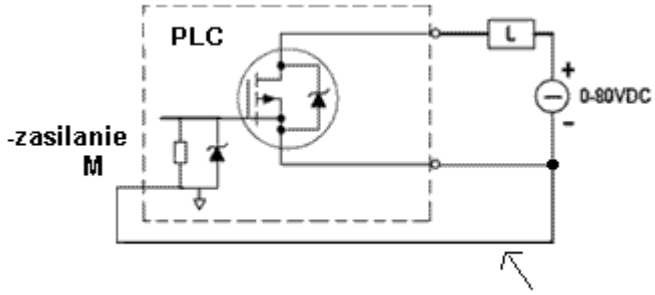


Rys. podłączenie wyjścia sterownika w postaci styku przekaźnika

Przykłady **wyjść** dwustanowych.



wyjscie przekaźnikowe



wyjście tranzystorowe

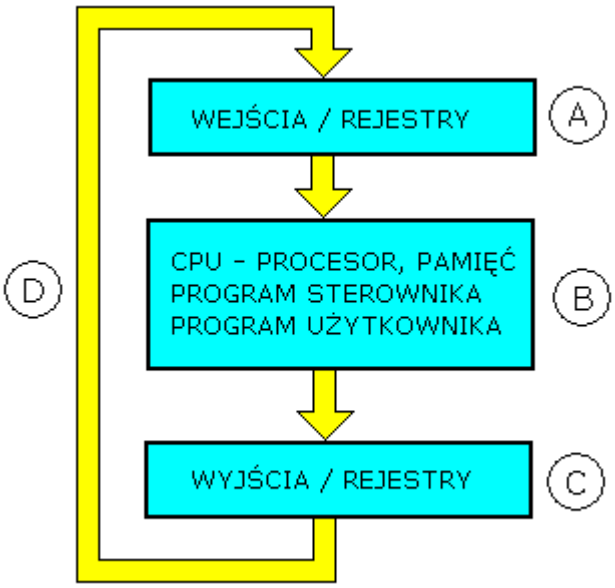
Najprostsze wyjście to styk przekaźnika, uruchamiany sygnałem z procesora (nie bezpośrednio). Zaleta takich wyjść, to możliwość łączenia różnych, niezależnych obwodów (AC i DC), bez konieczności utrzymywania wspólnych potencjałów.
Wadą jest ograniczona mechaniczna i elektryczna trwałość przekaźnika oraz stosunkowo mała szybkość rzędu 10mS.
Drugi, popularny typ wyjść, to tranzystorowe dwustanowe. Tranzystor pracuje tu jak klucz – włączony / wyłączony. Spotykane wyjścia są typu NPN (symbol pochodzi od tranzystorów bipolarnych, choć w sterownikach mogą być zastosowane C-MOS jak na rysunku) oraz typu PNP. Można przyjąć, że wyjścia NPN łączą obciążenie z masą, PNP łączą obciążenie z plusem zasilana.
Zaletą wyjść tranzystorowych jest ich trwałość i szybkość. Możliwe jest generowanie ciągłych przebiegów prostokątnych. Wady, to łączenie tylko obwodów prądu stałego, bez izolacji optycznej, konieczność utrzymywania wspólnego potencjału odniesienia np. masy.
W przypadku wyjść tranzystorowych o większych prądach (np. 2A w APB, AF, SR) stosowane są połączenia zewnętrzne (na rysunku wskazany strzałką), aby prądy obciążeń nie płynęły przez sterownik.

Porty komunikacyjne (interfejsy) sterownika

Podstawowe porty komunikacyjne sterownika (przynajmniej jeden taki musi być o ile programowany jest z komputera), to szeregowe : RS-232, RS-485 albo USB.
Interfejs **RS-232** wykorzystywany do komunikacji między dwoma urządzeniami. Podstawowe sygnały napięciowe to Rx (odbiór), Tx (nadawanie) jako napięcie +/- względem masy GND. Dodatkowe (mogą nie występować) to np. DTR, RTS, CTS.
Porty oznaczane są jako com 1 , com 2 itd.
Do RS-232 mogą być stosowane pętle prądowe, znaczeni wydłużające odległość
USB – nowszy standard interfejsu szeregowego, obecnie powszechny w komputerach. Sygnały transmisji danych to D+ , D- . Dodatkowe pin to GND i V_{BUS} 5V pozwalający na zasilenie niewielkich odbiorników. Cechą charakterystyczną tego interfejsu jest rozpoznawanie urządzeń podłączonych do USB i podstawianie systemowego sterownika programowego (driver). Współczesne laptopy, mające wyłącznie USB, mogą być wyposażane w zewnętrzne konwertery np. USB / RS-232 symulujące porty com.
RS-485 – standard komunikacji szeregowej pozwalający na sieciowe łączenie wielu urządzeń. Jedno urządzenie tzw Master (klient) decyduje o zajętości i wymianie informacji z pozostałymi urządzeniami typu Slave (serwer). Przykład to jeden komputer albo panel operatorski, obsługujący wiele sterowników (o różnych adresach) .
Informacja przekazywana jest jako różnica potencjałów między linią A i B . Łączone są też potencjały GND urządzeń.

Jak pracuje sterownik programowalny ?

Odpowiadając krótko – podobnie jak każde urządzenia z mikroprocesorem.
Podstawowy sposób organizacji pracy sterownika przedstawia rysunek poniżej.



Rys. Schemat blokowy podstaw działania sterownika

A - odczyt stanów wejść binarnych (dwustanowych) lub analogowych - pomiar wartości
B - przetworzenie informacji odczytanych w punkcie A z uwzględnieniem danych wynikających z dotychczasowej pracy, i zapisanych stałych w pamięci programu.
Sposób działania określony jest między innymi przez program użytkownika, przetłumaczony na kod maszynowy procesora - instrukcje. Pierwotną postać programu, użytkownik (instalator), przygotowuje wykorzystując udostępniane metody (języki) programowania.
Poza programem użytkownika, na działanie sterownika wpływ mają właściwości elektryczne CPU, parametry elektryczne wejść i wyjść, jak też program stały (wgrany przez producenta), tzw. „firmware”.
C - wyjścia sterownika , rejestry, aktualizowane po wykonaniu programu w pkt. B
D- powrót do zadań wykonywanych w pkt. A itd.

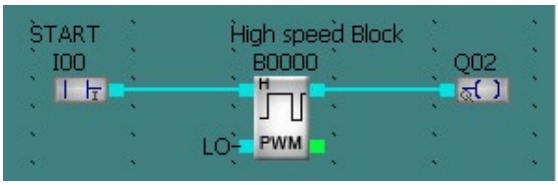
Jak widać, sterownik pracuje cyklicznie, z tak zwanym cyklem programowym (Program Sweep) , niekiedy nazywanym „scanem”. Oznacza to, że badanie stanów wejść, realizacja wewnętrznych funkcji, aktualizacja wyjść, wykonywane są cyklicznie, co jakiś czas.
Z tym wiąże się okresowość odczytów stanów wejść, aktualizacji wyjść i znaczący czas reakcji wyjścia na zmiany stwierdzone na wejściu czyli czas odpowiedzi układu.

Szybkość działania sterownika

W podstawowym działaniu, szybkość uzyskania odpowiedzi na wyjściu, ewentualnie szybkości wykrycia zmiany stanu na wejściach, decydujący wpływ będzie miał czas trwania jednego cyklu (scanu). Skrócenie czasu cyklu osiąga się szybszym procesorem, ale też przez optymalizację programu. Niestety, szybkość programu użytkownika, pozostaje w sprzeczności z łatwością metody . Ale to łatwość programowania jest konieczną cechą łatwego sterownika programowalnego, pozwalającą instalatorowi bezbłędnie i możliwe

szybko osiągać cel. Pewnym kompromisem w opisanej sprzeczności, jest wykorzystanie tzw. bloków szybkich, czyli sprzętowych możliwości samego procesora (jego liczników) albo obsługiwanie fragmentu programu poza cyklem w tzw. przerwaniach. Szybkość takiego wydzielonego bloku już nie zależy od czasu cyklu i jest precyzyjnie podawana przez producenta np. jako maksymalna częstotliwość generatora czy licznika. Należy jednak pamiętać, że wymiana nastaw w sprzętowym bloku szybkim, odbywa się z czasem cyklu czyli wolniej. Naturalnie sterownik może obsługiwać jedynie procesy nieco wolniejsze, niż maksymalny czas jego reakcji (czasu trwania pętli). Dla teoretycznego określenia czasu reakcji sterownika, czyli czasu wykonywania pętli , konieczne jest zsumowanie czasów użytych wszystkich instrukcji. Ale to jest trudne, a w łatwych metodach programowania praktycznie niemożliwe. Niekiedy, producent podaje bardzo orientacyjnie czas, np. 1mS na blok . Można założyć, że szybkość prostych sterowników pozwala zliczyć 2 do 4 Hz . I nie należy sugerować się nieco mylącym parametrom typu czas zmiany stanu z niskiego na wysoki , bo to może być jedynie czas przyciągania kotwicy zastosowanego przekaźnika. Wpisywane wartości w funkcyjnych blokach czasowych dotyczą zależności między ich wejściem a wyjściem i nie określają możliwości całego sterownika.

Przykłady szybkich bloków (sprzętowych do 10kHz) można zobaczyć w programie APB Soft, dostępnym na stronie www.telmatik.pl … (**na końcu więcej o APB)**.



rys. Przykład szybkiego bloku, wytwarzającego przebieg PWM (prostokąt do 10kHz o zmiennym współczynniku wypełnienia) na wybranym wyjściu sterownika APB.

UWAGA Niekiedy można wyeliminować zwłokę czasową sterownika odpowiednio łącząc czujniki.

Metody programowania

Programowalny charakter sterowników polega na możliwości zapisania do układu sterującego B informacji, jak ma działać, czyli programu. Oczywiście w ramach udostępnionych możliwości. W niniejszej informacji nie podejmujemy się porządkowania nazewnictwa metod, a tym bardziej oceny zgodności z istniejąca normą. Rozwój wymusza na producentach odstępstwa, łączenie metod, wprowadzanie własnych, korzystnych rozwiązań i takie postępowanie wydaje się właściwe. Celem jest przyjazne dla człowieka, graficzne lub opisowe projektowanie działania sterownika z ukrytym tłumaczeniem na „język procesora”. Tylko orientacyjnie, znane metody programowania to:

- pisanie listy instrukcji, skrót STL (Statement List)
Zbliżona do asemblera - metody programowania procesorów. A więc, choć optymalna, to trudna , bez możliwości łatwego wykrycia miejsc popełnionych błędów logicznych. Metoda dobra dla osób często programujących, gromadzących własne gotowe procedury .

- rysowanie drabinki o szczeblach składających się z odpowiednio łączonych styków NO , NC, sterowanych wskazanymi sygnałami i włączających wybrane wyjście. Skrót nazwy to LAD (Ladder Diagram). Metoda prosta co do zasady, jednak trudna przy budowie dłuższych

algorytmów . Nie wykorzystuje funktorów, wprowadzonych dla układów cyfrowych jako łatwiejszych przy analizie logicznej, ale i tak siłą rzeczy musi korzystać z bloków funkcyjnych np. liczników. Metoda dla osób dość często programujących, wymaga doświadczenia, trudna w przypadku szukania błędów logicznych.

- wykorzystanie graficznych bloków funkcyjnych odpowiednio łączonych między sobą. Skrót nazwy to FBD (Function Block Diagram). Niekiedy rysowanie pełnego schematu zbliżonego do schematu ideowego układów cyfrowych z logicznymi funktorami typu AND , OR . Skrót CSF (Control System Flowchart) albo FBD
Metody wysokiego poziomu tj. łatwe dla człowieka, ale dalekie od kodu maszynowego procesora. Rozbudowane możliwości graficzne pozwalają na zwięzłe umieszczanie diagramu (schematu) na ekranie, jego łatwe testowanie a i często późniejszy podgląd rzeczywistego działania w sterowniku. Metoda łatwa do budowy układów logicznych . Trudne, albo niemożliwe wyliczanie czasów reakcji i odpowiedzi sterownika, utrudnione wykonywanie złożonych operacji arytmetycznych. W tej metodzie najłatwiej szukać błędów logicznych.
Wydaje się, że ktoś kto widział skomplikowane schematy układów przekaźnikowych (np. fragmentów dawnych central telefonicznych) nie ma wątpliwości, dlaczego wraz z techniką cyfrową wprowadzono symbole logiczne (funktory), zamiast stosować pozornie prostszą symbolikę styków.

Sterownik , czy przekaźnik programowalny ?

Niektórzy stosują określenie „przekaźnik programowalny” w odniesieniu do małych sterowników, szczególnie programowanych drabinkowo. Ale podział jest nieprecyzyjny. Trudno powiedzieć, czy sterownik z procesorem 32bitowym i 320 zaawansowanymi blokami funkcyjnymi ale o 8 we/4wy jest przekaźnikiem, czy sterownikiem, szczególnie, gdy można dołączyć do niego rozszerzenia.

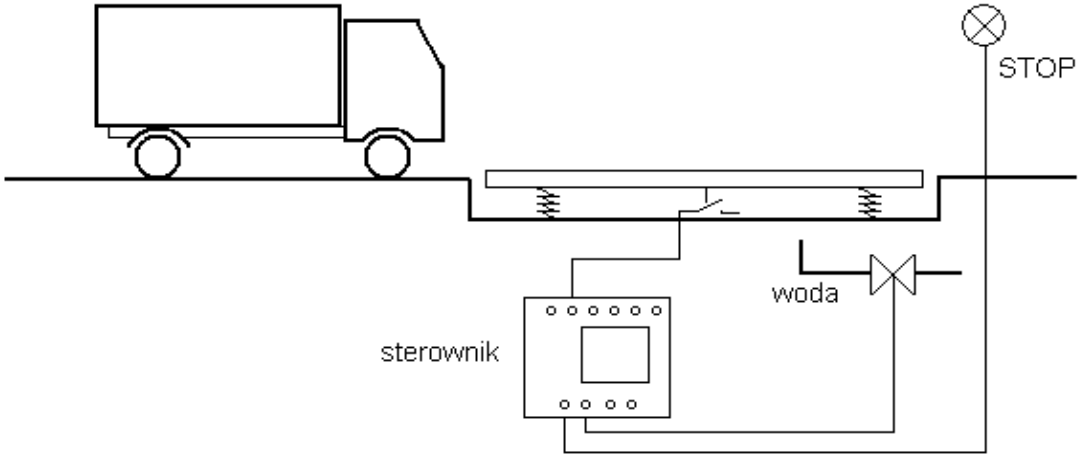
Jeśli „PLC prosto” - to programowanie metodą bloków funkcyjnych

Według nas, do zastosowań logicznych, rysowanie na ekranie komputera schematu (diagramu) z użyciem dostępnych bloków funkcyjnych jest najłatwiejsze. Szczegóły metody FBD, CSF opisują instrukcje programów, dlatego w dalszej części proponujemy tylko ogólne, szybkie, ale praktyczne poznanie jednej metody programowania sterowników. Tworzone diagramy podobne będą do schematów ideowych układów cyfrowych. Sygnały wejściowe odpowiadają wejściom sterownika albo wewnętrznym zmiennym, a wyjściowe są wyjściami sterownika (np. przekaźniki) albo zmiennymi wewnętrznymi. Dostępne w metodzie FBD bloki działają podobnie do elementów techniki cyfrowej, choć realizowane są na drodze programowej. Programy komputerowe przeznaczone do programowania sterowników wyposażane są w symulacje działania bloków i całego diagramu a więc umożliwiają naukę metodą próby i błędów. Największą zaletą FBD w stosunku do metody drabinkowej LAD , czy pisanych komend, jest możliwość łatwego śledzenia przebiegu sygnałów od wejścia do wyjścia sterownika, w formie ciągłej, „podświetlanej” linii. Nie musimy więc pamiętać i uwzględniać niewidocznych skoków programu, wykonywanych operacji w odległych szczeblach drabinki. Nawiasem , nie ma już „czystej drabinki” , bo współczesne zawierają bloki funkcyjne.
Nie można jednak zapominać, że zawsze działanie sterownika polega na wykonywaniu programu w powtarzającej się pętli, który analizuje stany wejść, analizuje i aktualizuje stany wszystkich zastosowanych bloków (programowych modeli funkcyjnych) w kolejności ich numerów, a na koniec ustawia wyjścia sterownika. Oznacza to, że np. aktualizacja wyjść

możliwa jest dopiero po jakimś czasie od momentu zmiany sygnału na wejściu czyli po czasie jednej pętli .

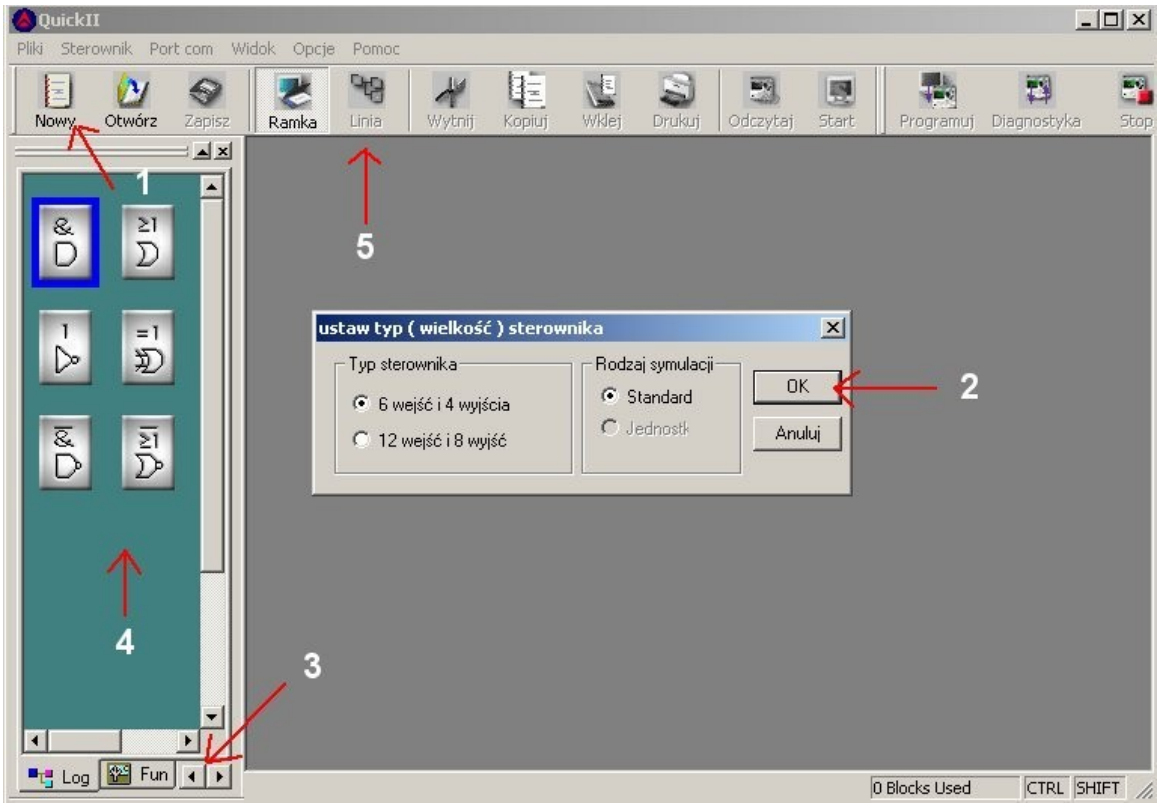
Ćwiczenie praktyczne dla osób początkujących :

Zadaniem jest wykonanie prostego projektu metodą FBD i sprawdzenie działania poprzez symulację programową , a realizującego następujące funkcje:
Pojazd (np. śmieciarka opuszczająca wysypisko) wjeżdżający na platformę z czujnikiem obciążenia ma być spryskany wodą przez czas 20 sek. Włączenie natrysku należy opóźnić o 5 sek. od chwili wykrycia obciążenia pojazdem. W czasie oczyszczania pojazdu powinno zapalić się czerwone światło sygnalizacyjne. Poniżej przedstawiono rysunek obiektu do opisanego zadania .



rys. automatyczna myjnia podwozia

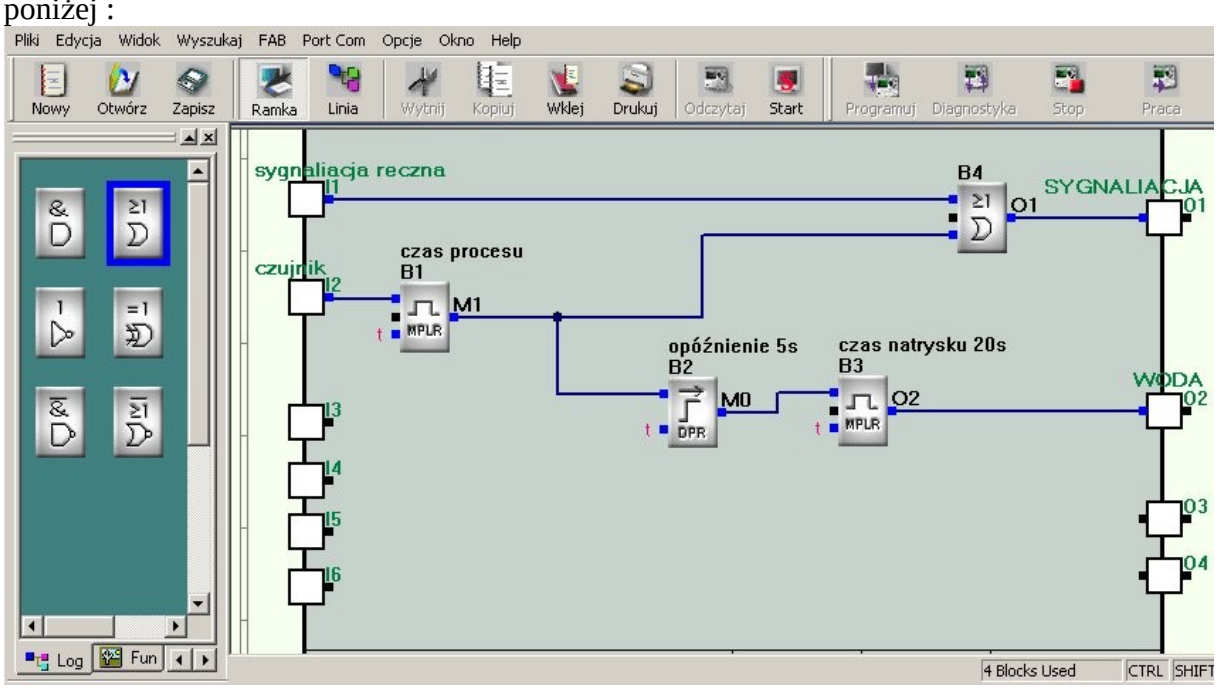
Aby wykonać i sprawdzić projekt, należy ze strony www.telmatik.pl pobrać demonstracyjny lub pełen program Quick II dla sterowników AF (Array-FAB). Dokładniej, to pliki umieszczone są na stronie pod przyciskiem … „więcej o AF”.
Wersja demonstracyjna nie wymaga instalacji, ale nie można nią programować sterownika. Niezależnie od wersji, wykonany, przetestowany i zapisany projekt jest pełnowartościowy. Uruchomienie programu QuickII.exe spowoduje pojawienie się pola z odpowiednimi narzędziami do rysowania diagramu, lub otwierania gotowych już plików.
Należy więc, na początku, zdecydować i wybrać polecenie „Otwórz” lub „Nowy” a następnie wykonywać czynności zgodnie z naniesioną na rysunku numeracją.



rys. początkowy wygląd uruchomionego programu QuickII

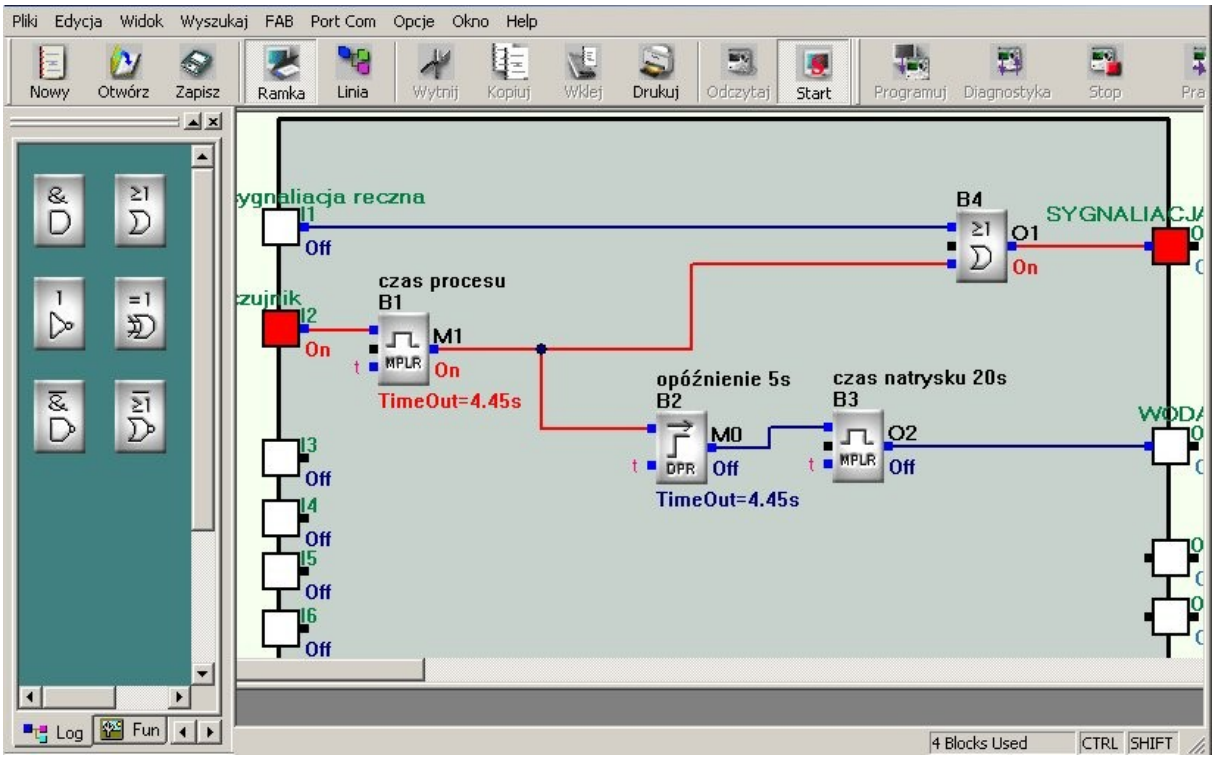
1- decyzja o wykonaniu nowego projektu
2- wybór wielkości sterownika (np. 6wejśc , 4 wyjścia) i akceptacja przyciskiem OK.
3- wybór grupy dostępnych bloków funkcyjnych (pole wyżej)
4- aktualnie dostępne bloki funkcyjne , przenoszone na pole rysowania diagramu
5- przycisk uruchamiający funkcję rysowania połączeń między blokami

W konkretnym, naszym zadaniu, gotowy diagram może wyglądać jak na rysunku poniżej :



rys. przykład projektu działania sterownika myjni podwozia

Przykład jest bardzo skromny, ale pozwala poznać ważniejsze zasady . Z lewej strony symbolicznie zaznaczone są wejścia sterownika I1-I6, z prawej wyjścia Q1-Q4. Do wejść podawane jest napięcie jako sygnał sterujący. Wyjściami są niezależne styki przekaźników, zamykające obwód sygnalizacji i elektrozaworu wody. Do realizacji użyto zaledwie 4 bloki (licznik w prawym dolnym rogu ekranu), z możliwych do wykorzystania 127 . Pokazuje to potencjalne możliwości nawet bardzo prostego sterownika.
Jak to działa ? Po podaniu napięcia na wejście I2 , czyli pojawieniu się stanu wysokiego, blok B1 na swoim wyjściu wygeneruje impuls o czasie trwania 30sek (ustawiany we właściwościach bloku B1) . Sygnał z wyjścia bloku B1, poprzez element sumujący sygnały (podobnie jak równolegle połączone styki) B4, przepisywany jest na wyjście Q1 sterownika (zapalanie sygnalizacji). Jak widać, możliwe jest również ręczne zapalenie sygnalizacji, przez drugie wejście I1. Sygnał z wyjścia B1 podawany jest również do wejścia bloku B2 opóźniającego pojawienie się jego na wejściu B3 o 5 sekund. Blok B3 wygeneruje impuls o czasie 20sek , czyli wymaganym czasie natrysku wody.
Działanie całego projektu , czy jeszcze wcześniej pojedynczych bloków, najłatwiej sprawdzić uruchamiając w programie Quick symulację działania sterownika (przyciskiem „Start”). Nasz komputer wykona narysowane zadanie, dodatkowo sygnalizując zmianą koloru stany wszystkich linii.



rys. Widok programu Quick II z włączoną symulacją pracy sterownika.

Obserwując zmieniające się stany wejść, wyjść, linii, czy odliczane w blokach czasy , jako reakcje na wymuszenia od strony wejść sterownika (przez kliknięcie myszką) możemy analizować działanie naszego projektu.
Podczas symulacji innych diagramów również będziemy mogli obserwować zmieniające się stany liczników, wyjść bloków zegarowych , kalendarza czy efekty porównań sygnałów analogowych w komparatorach itp.
Po przesłaniu projektu do sterownika (przez port RS232, USB itp.), jego mikroprocesor

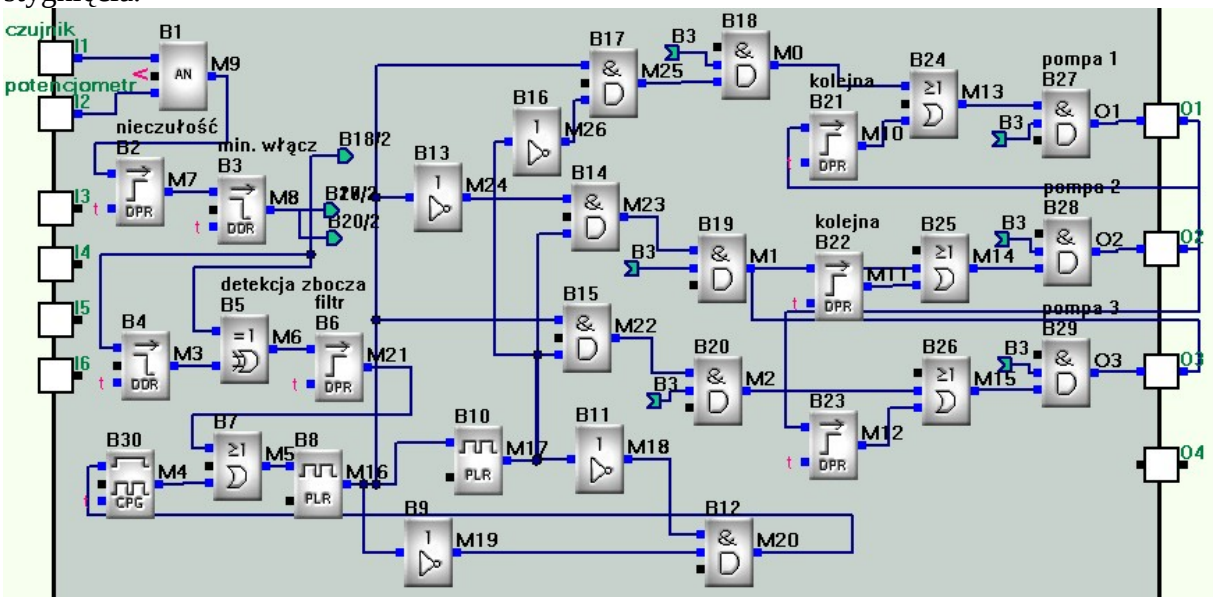
będzie wykonywał ten sam program analizując w niekończącej się pętli blok po bloku. Jednak dla zapewnienia właściwej kolejności analizowania bloków przez sterownik, przez analogię z kierunkiem przepływem prądu od wejścia do wyjścia sterownika, pod koniec prac projektowych należy uporządkować numery bloków. Zasadą powinno być, aby blok bliżej źródła sygnału miał numer niższy (był przez to wcześniej analizowany) , niż blok następujący po nim . Na rysunku bliżej wejścia (źródła sygnału) jest blok B1, po nim następują bloki z numerami B2 i B4. W programie Quick zmianę numeru wykonuje się „ręcznie" ale niekiedy realizowane jest to automatycznie , albo w sposób niejawny. Ręczne przenumerowanie, mimo pewnej niewygody, pozwala lepiej kontrolować działanie programu szczególnie przy sprzężeniach zwrotnych (zawracaniu sygnału).

Uruchomienie obiektu

Właściwie, po zapisaniu programu do sterownika i wykonaniu połączeń elektrycznych, powinno już wszystko działać. A jeśli nie, to pomocna przy uruchomieniu będzie funkcja programu Quick II, nazwana „monitor". Tym razem, „podświetlane" wejścia i wyjścia czy wewnętrzne linie diagramu odzwierciedlać będą faktyczne stany pracy sterownika (już nie symulacji). Brak oczekiwanej zmiany sygnału z czujnika , czy prawidłowe wysterowanie przekaźnika, możemy zaobserwować również na wyświetlaczu LCD. Ale jest to zdecydowanie mniej wygodne, podobnie, jak wprowadzanie całego diagramu przez LCD .

Kolejne projekty

W instrukcjach zamieszczane są opis i przebiegi graficzne wyjaśniające działanie wszystkich dostępnych bloków. Ale również wykorzystując symulację komputerową, możemy badać działanie pozostałych (tu nieopisanych) bloków i wykonywać bardziej złożone projekty. Efektem może być np. układ sterowania pracą trzech pomp (rys. niżej) zapewniających wymagane podciśnienie w zbiorniku.
Z chwilą , gdy sterownik wykryje zmniejszenie podciśnienia w zbiorniku, (przez wejście analogowe z podłączonym czujnikiem) włączy jedną pompę, ale ostatnio nieużywaną . Jeżeli w wymaganym czasie jedna pompa nie zapewni potrzebnego podciśnienia, uruchamiana jest Druga, ewentualnie trzecia. Do pracy wyznacza jest pompa , która w poprzednim cyklu nie pracowała po to, aby zapewnić równomierne zużycie pomp i możliwie długi czas ich stygnięcia.



Co jeszcze możemy spotkać w prostych PLC ?

- **różne typy wejść** dostępnych w jednostce głównej (podstawowej) albo w formie rozszerzeń czyli oddzielnych modułów. Poza wejściami dwustanowymi (czynnymi albo pasywnymi) , analogowymi typowo 0-10V albo 4-20mA (0-20mA) można spotkać wejścia przystosowane do czujników temperatur rezystancyjnych PT100, PT100 albo termistorów NTC, PTC albo termopar K, J, T, R ewentualnie czujników półprzewodnikowych np. KTY . Jeśli nie ma oryginalnych modułów albo są zbyt drogie, z powodzeniem można stosować powszechnie dostępne przetworniki temperatury z wyjściami 0-10V albo 4-20mA . Często, jedyne dostępne w sterowniku wejście 0-10V, można łatwo zamienić na wejście 0-20mA, łącząc równolegle rezystor 500omów. W sterownikach (szczególnie w szybkich) możemy spotkać wejścia symetryczne tj. takie w których żaden zacisk nie jest potencjałem masy sterownika , wejścia izolowane optycznie (wejścia odseparowane transoptorami od połączeń galwanicznych procesora)
- **różne typy wyjść**. Poza dwustanowymi przekaźnikowymi , tranzystorowymi (typy NPN , PNP) można trafić na tzw. wyjścia szybkie, a więc dostosowane do szybkich przełączeń generowania przebiegów prostokątnych . Spotykane są wyjścia prądowe 4-20mA albo 0-20mA, a więc realizujące funkcję sterowanych źródeł prądowych . Wyjścia mogą być izolowane lub nie
- **wyświetlacze LCD** o mniej lub bardziej rozbudowanych możliwościach
- **wbudowane zegary czasu rzeczywistego RTC** pozwalające na uzależnienie działania sterownika od aktualnej godziny, dnia tygodnia , daty itp.
- **różne funkcje dodatkowe**. Poza funkcjami dostępnymi w formie bloków, mogą pojawiać się możliwości pamiętania stanów po wyłączeniu zasilania (albo pojedynczych bloków albo całego programu) , dostęp do rejestrów czyli udostępnionych komórek pamięci pozwalających na przesuwanie określonych danych, wykonywania operacji arytmetycznych , wymiany informacji z zewnętrznymi obiektami itp. Rejestry mogą dzielić się na tymczasowe (operacyjne) czyli zerowane po zaniku zasilania i trwałe (dane zapisywane do nielotnej pamięci flash). Za dodatkową funkcje można uznać wymianę informacji z zewnętrznymi urządzeniami w określonym standardzie np. Modus RTU. Standardowy protokół komunikacyjny pozwala na łączenie urządzeń różnych producentów, tworzenie sieci RS-485. Więcej informacji o **Modus RTU** znajduje się pod przyciskiem o takiej samej nazwie na :

www.telmatik.pl