Szkolenie PLC cz. 9
Programowanie sterownika z wykorzystaniem diagramu bloków funkcyjnych (z ang. FBD – Function Block Diagram)

W aktualnym wydaniu kwartalnika „Pod Kontrolą" zajmiemy się nieporuszanym do tej pory zagadnieniem projektowania prostych i średnio skomplikowanych aplikacji logicznych w środowisku SG2 Client z wykorzystaniem diagramu bloków funkcyjnych.

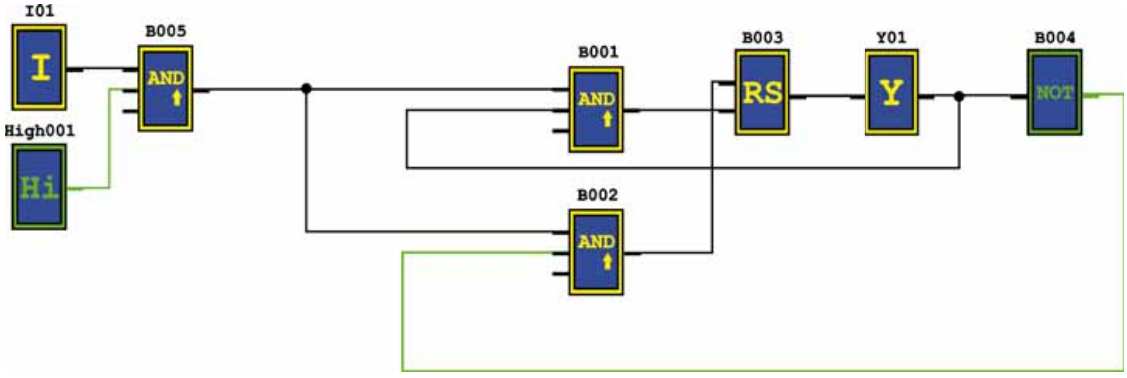
Przez ostatnie kilka szkoleń zajmowaliśmy się opisem realizacji mniej lub bardziej zaawansowanych problemów programistycznych z wykorzystaniem sterownika programowalnego SG2. Charakterystyka dotyczyła zarówno zagadnień bezpośrednio związanych z logiką tworzonych algorytmów jak i z komunikacją pomiędzy urządzeniami wchodzącymi w skład prezentowanych aplikacji. Do programowania sterowników arbitralnie wykorzystany został język drabinkowy LAD, który jest rozwiązaniem domyślnym w przypadku środowiska SG2 Client. Sytuacja ta nie była przypadkowa, gdyż bazując na dotychczasowych doświadczeniach aplikacyjnych, wykorzystanie opisanego języka, a co z tym związane, sposobu tworzenia oprogramowania okazało się najsensowniejszym wyborem zarówno dla prostych jak i bardziej zaawansowanych problemów programistycznych. Nie oznacza to jednak, że producent urządzenia oraz środowiska programistycznego SG2 Client nie udostępnił użytkownikowi innego lub innych języków/ sposobów programowania aplikacji logicznych.

DIAGRAM BLOKÓW FUNKCYJNYCH

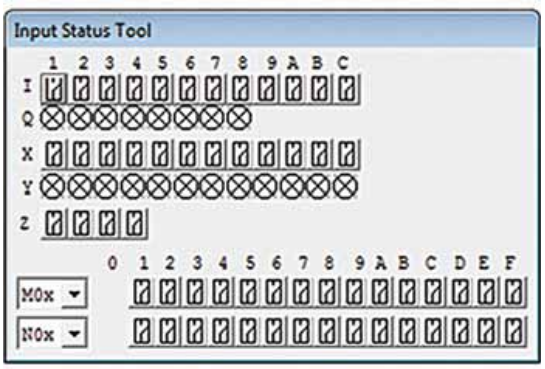
Swoistą alternatywą dla języka drabinkowego LAD jest diagram bloków funkcyjnych (z ang. FBD – Function Block Diagram). Programowanie w FBD w dużym uproszczeniu polega na umieszczaniu odpowiednich bloczków realizujących konkretne funkcje matematyczne oraz logiczne na schemacie i łączenie ich za pomocą linii sygnałowych. Parafrazując słowa producenta można powiedzieć, iż ta koncepcja projektowania oprogramowania jest dużo bardziej wydajna oraz „humanistyczna", gdyż prezentuje kod programu w formie bardziej zrozumiałej od języka drabinkowego. Ma to szczególne zastosowanie w przypadku osób uczących się programowania tego typu aplikacji, dla których język drabinkowy nie jest rozumiany i przyswajany w sposób automatyczny (co ma miejsce w przypadku bardziej zaawansowanych użytkowników oraz osób posiadających doświadczenie programistyczne).

PROGRAMOWANIE W NOWYM JĘZYKU

Na ilustracji nr 1, przedstawiony został przykładowy program stworzony w oparciu o środowisko FBD. Program ten realizuje bardzo dobrze znany czytelnikom problem dydaktyczny – zapalanie oraz gaszenie żarówki jednym przyciskiem monostabilnym. Jak łatwo zauważyć, elementy sterujące takie jak wejścia cyfrowe, markery oraz cewki a także funkcje realizowane na podstawie wymienionych sygnałów zaznaczone są jako niebieskie bloczki. Wykonanie (skan) programu odbywa się poprzez przepływ sygnału z jednego elementu do kolejnego, z którym jest połączony. Elementy mogą blokować sygnał lub go przepuszczać i w ten sposób sterują aktualnym stanem programu oraz stanem każdego z bloczków wyjściowych. W momencie gdy sygnał „biegnący" przez wszystkie elementy osiągnie element ostatni, skan rozpoczyna się od nowa.



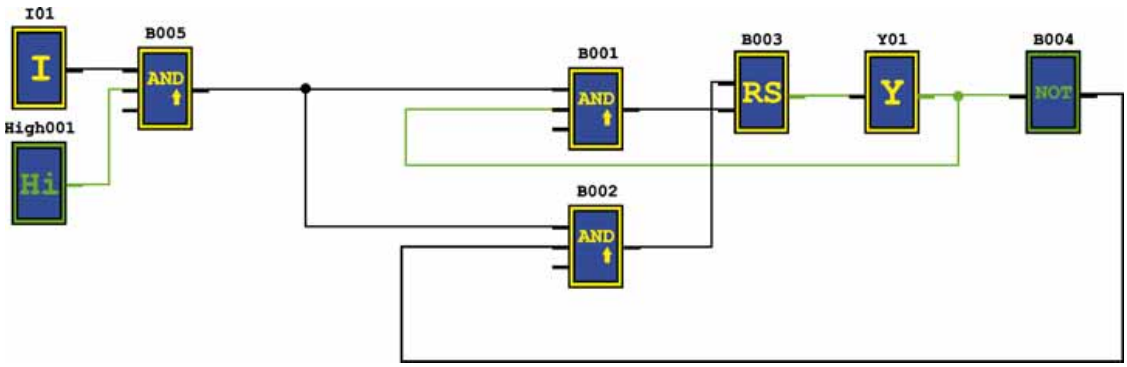
Rysunek 1
Stan początkowy programu (lampka zgaszona).



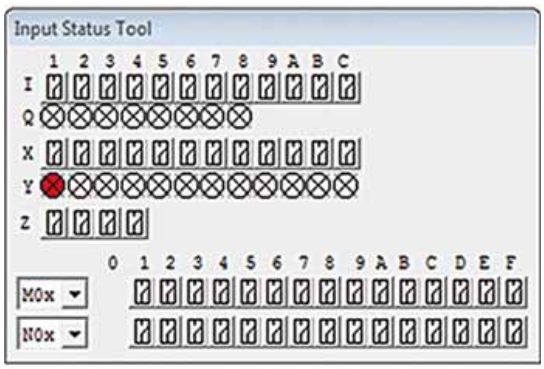
Analizę rozpoczynamy z lewej strony schematu. Pierwszym elementem jest wejście binarne **I01**, do którego podłączony jest monostabilny przycisk inicjujący zapalenie i zgaszenie żarówki. Poniżej znajduje się enigmatycznie nazwany i wyglądający ele-

ment, a mianowicie **Hi**. Jest to bloczek, który przez cały czas zwraca sygnał wysoki – inaczej mówiąc, jest cały czas aktywny. Jego zastosowanie wymuszone zostało poprzez konieczność wykrycia zbocza narastającego na wejściu **I01**. Funkcja ta realizowana jest poprzez zastosowanie bloczka **AND** z wbudowaną analizą tego zbocza, jednakże aby opisywany bloczek działał poprawnie konieczne jest podłączenie do niego minimum dwóch sygnałów binarnych, z których tworzony jest iloczyn logiczny. Jednym z nich jest sygnał wejściowy z przycisku, drugim jest cały czas aktywny sygnał **Hi**. W przypadku standardowej funkcji iloczynu logicznego, sygnał za bloczkiem byłby dokładnie taki sam jak sygnał wejściowy, jednakże zastosowanie funkcji **AND** z dodatkowym wykrywaniem zbocza powoduje, iż sygnał wyjściowy jest aktywny tylko w przypadku wykrycia zbocza narastającego na wejściu **I01**.

Kolejnymi dwoma elementami, są równolegle połączone bloczki realizujące iloczyn logiczny z wykryciem zbocza. W każdym z nich pierwszym sygnałem sterującym jest opisany wcześnie sygnał zbocza narastającego na wejściu cyfrowym, natomiast drugi pochodzi ze sprzężenia zwrotnego. Aby opisać zasadę działania tego mechanizmu, musimy wyjść nieco w przód i określić skąd pochodzą owe sygnały.



Rysunek 2 Drugi z dwóch stanów układu – lampka zapalona.



Pierwszy z nich jest naszym sygnałem wyjściowym z aplikacji, czyli sygnałem zapalającym lub gaszącym żarówkę, natomiast drugi jest chwilowym zanegowanie sygnału pierwszego. Jeśli lampka w danej chwili jest zapalona, to aktywny jest pierwszy z nich, w przeciwnym przypadku – drugi. Wracając do pozostawionych dwóch bloczków **AND**, możemy stwierdzić iż w zależności od aktualnego stanu programu (lampka zapalona lub zgaszona), sygnał w danym cyklu przejdzie tylko przez jeden z nich.

Kolejnym elementem jest funkcja realizująca w dużym uproszczeniu działanie przerzutnika SR. Posiada ona dwa wejścia: jedno sterujące (aktywujące) oraz drugie kasujące, które to jest wejściem dominującym. W momencie pojawienia się wysokiego sygnału na wejściu sterującym, przerzutnik ustawi swoje wyjście w stan wysoki, natomiast podanie wysokiego sygnału na drugie z wyjść(kasujące), zresetuje bloczek i skasuje stan wysoki na wyjściu, niezależnie od aktualnego stanu wejścia sterującego. Doszukując się analogii do języka drabinkowego, można powiedzieć iż funkcja te realizuje operacje SET oraz RESET dla konkretnego sygnału wyjściowego.

Ostatecznie wyjście elementu **RS** jest podawane na fizyczne wyjście binarne sterownika **Y01**. Zasada działania układu jest więc następująca: w momencie wykrycia zbocza narastającego na elemencie **I01**, analizowany jest aktualny stan układu, pochodzący ze sprzężenia zwrotnego. W zależności od niego, aktywowany jest jeden z dwóch równoległych elementów **AND**, podanych odpowiednio na wejście sterujące oraz kasujące bloczka RS. Jeśli żarówka była zgaszona,a co za tym idzie wyjście **Y01** nieaktywne to inicjowany jest sygnał sterujący przerzutnika i lampka się zapala. W przeciwnym przypadku, lampka która była zapalona – gaśnie, gdyż aktywowane jest wejście kasujące elementu **RS**.

OCENA MOŻLIWOŚCI

Jak zostało pokazane, wykorzystanie schematu bloków funkcyjnych nie generuje żadnych istotnych ograniczeń w sposobie programowania aplikacji w porównaniu do języka LAD. Warto zauważyć, że język FBD umożliwia dużo lepszy nadzór nad aktualnie wykonywana aplikacją niż ma to miejsce w przypadku języka drabinkowego. Jak już wcześniej zaznaczono, jest to szczególnie istotne i pomocne w przypadku osób zaczynających dopiero swoją przygodę z programowaniem sterowników. W mojej, osobistej ocenie wykorzystanie języka LAD jest rozwiązaniem szybszym, gdyż wiele prostych aplikacji można zrealizować poprzez kilka kliknięć myszy, podczas gdy w przypadku FBD wymaga to więcej czasu i energii.

CO DALEJ?

Aktualne wydanie kwartalnika „Pod kontrolą” zawiera ostatnią już odsłonę szkolenia dotyczącego programowania inteligentnych przekaźników sterowalnych SG2. Nie oznacza to jednak zakończenia i zamknięcia tej rubryki. W kolejnych wydaniach zajmiemy się charakterystyką, opisem funkcji i sposobem programowania bardziej zaawansowanego sterownika PLC z rodziny TECO, a mianowicie kontrolera TP03.

Szkolenie prowadzi:
Dominik Szewczyk

tel.: 32 789 00 13

Jednoczesny pomiar przepływu różnych mediów

W jednym ze znanych instytutów badawczych pojawił się problem opomiarowania kilku mediów na jednej aplikacji (powietrze, gaz ziemny, gaz kopalniany, gaz zaazotowany oraz gaz z dużą zawartością wodoru). Procesy te odbywały się okresowo, a medium oraz inne parametry procesowe były skrajnie zróżnicowane. Naturalnym rozwiązaniem był zakup pięciu przepływomierzy, co wiązało się bezpośrednio z dużym nakładem finansowym. Rozwiązaniem niestandardowym a jednocześnie prostym do realizacji było zakupienie jednego przepływomierza termicznego model **ST100** (firmy FCI). Przepływomierz serii ST100 ma możliwość aż 5 różnych kalibracji. Przełączanie może dokonywać się manualnie lub automatycznie wykorzystując dodatkowe wejście w przepływomierzu. Ekonomiczny montaż urządzenia (jeden króciec), regulowane przyłącze pozwalające na zastosowanie na rurociągach o różnych średnicach, certyfikat ATEX oraz zgodność z deklaracją SIL 1 rozwiała ewentualne wątpliwości o słuszności tego rozwiązania.

Detekcja tlenku węgla w ratownictwie medycznym

W ostatnim czasie w mediach pojawiła się informacja o zatruciu czadem ratowników medycznych wezwanych do akcji ratunkowej w mieszkaniu, w którym się ulatniał. Ratownicy nie byli wyposażeni w detektory, a zatem nie mieli pojęcia, że ratując człowieka zatrutego tlenkiem węgla sami zostali poddani działaniu tego niebezpiecznego gazu, który już w niskich stężeniach jest dla człowieka silnie toksyczny. Wszyscy trafili do szpitala. W Europie już od dawna detektory tlenku wegla stosowane są również w takich służbach jak ratownictwo medyczne czy straż pożarna.

Detektory ochrony osobistej GasAlert produkcji BW Technologies by Honeywell od lat sprawdzają się w takich branżach jak paliwowa, wodno-kanalizacyjna, ciepłownicza, spożywcza, a także w oczyszczalniach ścieków, biogazowniach itp. Są także świetnym rozwiązaniem dla karetek pogotowia. Dokładność i stabilność wskazań, prosta obsługa oraz wysokiej jakości obudowa, a zarazem najmniejsze wymiary na rynku powodują, że detektory te zapewniają bezpieczną pracę i nie przeszkadzają w wykonywaniu czynności ratunkowych. Jako jedyne są też odporne na działanie oparów alkoholu, na bazie którego środkami czyszczone są karetki pogotowia.

Zapraszamy do odwiedzenia strony;
www.detektorydlapogotowia.pl