*Piotr Wilk*
ORCID https://orcid.org/0000-0003-2794-8339
Akademia Nauk Stosowanych w Raciborzu

# NAUKA JĘZYKÓW PROGRAMOWANIA STEROWNIKÓW PLC W DOBIE OBECNEGO ROZWOJU

**Streszczenie (abstrakt):** Artykuł zawiera opis języków programowania programowalnych sterowników logicznych (PLC), które szczegółowo omówiono w normie IEC61131-3[1]. W treści artykułu można odnaleźć również wskazówki dla początkujących automatyków, dotyczące wyboru języka bądź języków programowania sterowników logicznych. W związku z tym autor wskazuje te języki programowana, które są obecnie najczęściej stosowane przez doświadczonych automatyków. Jednocześnie stara się odpowiedzieć na pytanie, czy warto jeszcze uczyć się języków programowania w czasach dynamicznego rozwoju sztucznej inteligencji.

**Słowa kluczowe:** sterowanie, sterownik PLC, programowanie PLC, języki programowania, sztuczna inteligencja

## LEARNING PLC PROGRAMMING LANGUAGES IN THE ERA OF CURRENT DEVELOPMENT

**Abstract:** The article describes the programming languages for programmable logic controllers (PLCs), which are discussed in detail in the IEC61131-3 standard[2]. The article also contains tips for beginner automation engineers regarding the selection of the programming language or languages for logical controllers. Therefore, the author indicates the programming languages that are currently most often used by experienced automation engineers. At the same time, it tries to answer the question whether it is still worth learning programming languages in times of dynamic development of artificial intelligence.

**Keywords:** control, PLC controller, PLC programming, programming languages, artificial intelligence

## 1. Wstęp

Programowalne sterowniki logiczne PLC (ang. *Programmable Logic Controller*) to komputery przemysłowe czasu rzeczywistego, które zbierają dane wejściowe (ustalają stan wejść sterownika), przetwarzają je zgodnie z programem użytkownika, generując dane

[1] A European Standard EN 61131-3, *Programmable controllers – Part 3: Programming languages (IEC 61131-3:2013)*, Brussels 2013, s. 123-143.

[2] Ibidem, s. 123-143.

wyjściowe (ustalając stan wyjść sterownika) sterujące urządzeniami wykonawczymi (rys. 1)[3].


**Rys. 1.** Zasada działania sterownika PLC

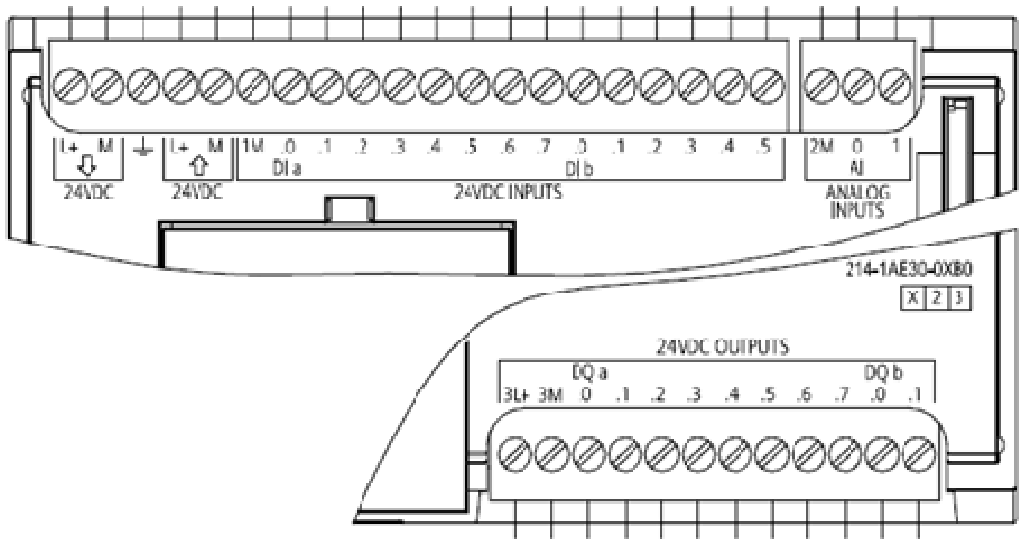
Z definicji wynika, iż sterownik logiczny musi pracować w czasie rzeczywistym. W związku z tym realizacja programu użytkownika, który przetwarza dane wejściowe w dane wyjściowe musi być zrealizowane w wymaganym dla zadania czasie[4].

## 2. Budowa programowalnych sterowników logicznych

Sterowniki PLC stanowią element składowy większości układów sterowania maszyn technologicznych oraz linii i systemów produkcyjnych. Dlatego można stwierdzić, że znajomość budowy i zasady działania programowalnych sterowników logicznych jest niezbędna w pracy każdego automatyka. Znajomość budowy powinna obejmować nie tylko cechy zewnętrzne sterownika, w tym sposób zasilania i dostępne wejścia oraz wyjścia (rys. 2), ale również jego budowę wewnętrzną (rys. 3).

[3] J. Kwaśniewski, *Sterowniki: SIMATIC S7-1200 w praktyce inżynierskiej*, Wydawnictwo BTC, Legionowo 2013, s. 10.

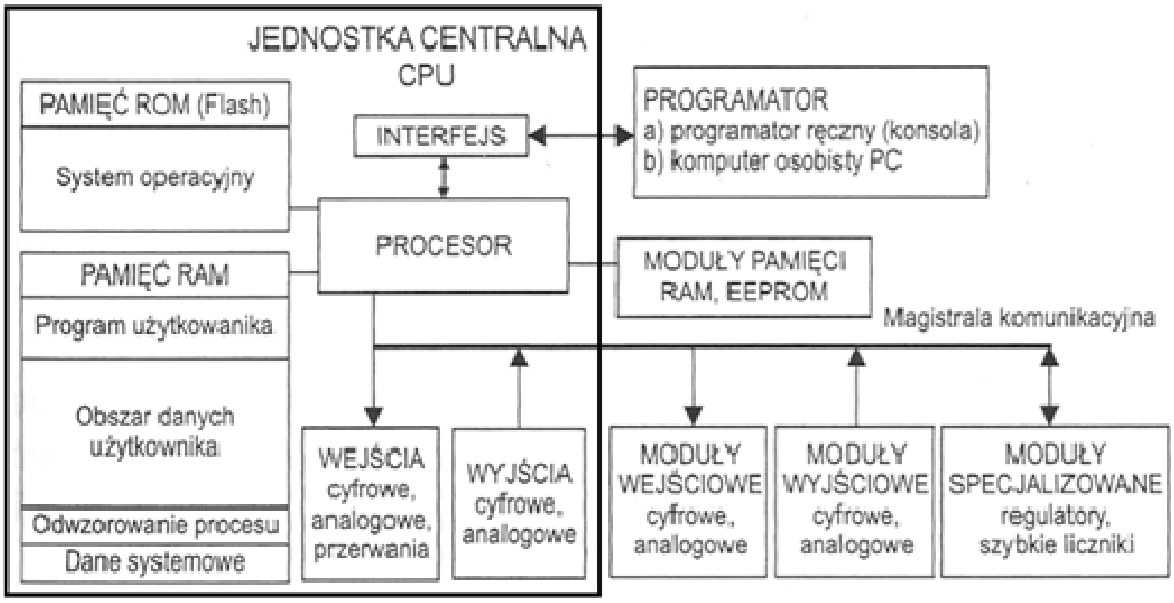
[4] Ibidem, s. 11.




**Rys. 2.** Schemat złącz sterownika PLC Siemens SIMATIC S7-1200 CPU 1214C

**Źródło**: Dokumentacja sterownika Siemens SIMATIC S7-1200 - Easy Book - wydanie 4/2012, https://publikacje.siemens-info.com/pdf/687/S7-1200%20Easy%20Book.pdf [dostęp: 20. 09.2024].

Sterowniki PLC składają się z jednostki centralnej CPU wyposażonej w pamięć oraz modułów wejść i wyjść. System operacyjny sterownika zapisany jest w pamięci trwałej ROM, obecnie najczęściej typu Flash[5].




**Rys. 3.** Schemat blokowy budowy sterownika PLC

**Źródło**: Kacprzak S., *Programowanie sterowników PLC zgodnie z normą IEC61131-3 w praktyce*, Wydawnictwo BTC, Legionowo 2011, s. 21.

[5] S. Kacprzak, *Programowanie sterowników PLC zgodnie z normą IEC61131-3 w praktyce*, Wydawnictwo BTC, Legionowo 2011, s. 21.

Pamięć RAM stanowi zapisywalną pamięć, która jest podtrzymywana bateryjnie lub za pomocą kondensatora. Zawiera ona program użytkownika, dane systemowe i użytkownika oraz obraz wejść i wyjść sterownika[6].

Program użytkownika może być również zapisany w pamięci trwałej. W takim przypadku program jest przepisywany do pamięci RAM w momencie włączenia zasilania[7].

Program zawarty w pamięci sterownika przetwarza dane użytkownika, które stanowią zdefiniowane przez programistę zmienne. Część z nich wymaga podtrzymania zawartości po zaniku zasilania, co ułatwia ponowne uruchomienie sterowanego układu. W takim przypadku programista określa, które zmienne będą podtrzymywane w sytuacji braku zasilania.

Moduły wejściowe i wyjściowe, w zależności od architektury sterownika, znajdują się razem z jednostką centralną lub osobno. Pod tym względem wyróżnia się odpowiednio sterowniki kompaktowe oraz modułowe. Sterowniki kompaktowe umożliwiają obsługę do 128 punktów (wejść i wyjść) i kilkunastu punktów analogowych oraz mogą pracować w sieci komunikacyjnej. Przy czym nie można w nich zmienić typu jednostki centralnej bez wymiany całego sterownika[8].

## 3. Zasada działania programowalnych sterowników logicznych

Jednostka centralna sterownika pracuje w tzw. cyklu programowym (*ang. scan cycle*), który stanowi pętlę bezwarunkową. Taki sposób pracy sterownika PLC umożliwia przetwarzanie danych w trybie rzeczywistym. Dane są aktualizowane w każdym cyklu pracy sterownika, a opóźnienie sterowanego sygnału jest bardzo małe i wynika tylko z czasu skanowania. Długość pętli programowej zależna jest od wielkości aplikacji użytkownika, szybkości przetwarzania CPU, liczby wejść-wyjść i może wynosić od kilkudziesięciu do kilkuset mikrosekund. W przypadku większości sterowników PLC jednostkę centralną można ustawić w cykl o stałej wartości przetwarzania, który nie może być krótszy niż czas niezbędny do realizacji programu. Po wykonaniu wszystkich faz programu sterownik odlicza pozostały czas do pełnego cyklu i wówczas dopiero rozpoczyna kolejny cykl pracy sterownika[9].

Cykl pracy sterownika PLC składa się z następujących faz (rys. 4):

- odczyt stanów wejść,
- wykonanie programu użytkownika,
- aktualizacja stanów wyjść,

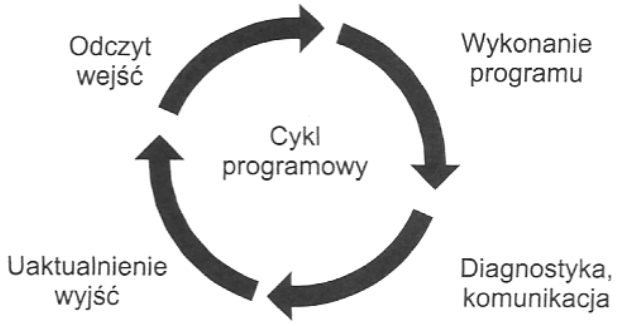
[6] J. Kwaśniewski, *Programowalny sterownik SIMATIC S7-300 w praktyce inżynierskiej*, Wydawnictwo BTC, Legionowo 2009, s. 25.

[7] S. Kacprzak, *Programowanie sterowników…*, *op.cit.*, s. 21.

[8] Ibidem, s. 21.

[9] Ibidem.

- wykonanie diagnostyki i obsługa komunikacji[10].




**Rys. 4.** Cykl programowy w sterowniku PLC

**Źródło**: Gilewski T., *Szkoła programisty PLC: sterowniki przemysłowe*, Wydawnictwo Helion, Gliwice 2017, s. 22.

W momencie uruchomienia sterownika następuje odczyt stanów wejść, czyli przepisanie z modułów wejściowych stanu bufora do obrazu pamięci CPU. Dopiero wówczas rozpoczyna się realizacja kolejnych instrukcji zawartych w programie użytkownika pomiędzy pierwszą i ostatnią linią. Po zakończeniu tego etapu następuje aktualizacja stanów wyjść poprzez przepisanie z pamięci obrazu jednostki centralnej do buforów modułów wyjściowych aktualnych wartości. Działania związane z diagnostyką sterownika obejmują sprawdzenie konfiguracji sprzętowej oraz długości realizowanego programu. W fazie tej jest wykonywana również obsługa urządzeń zewnętrznych, czyli komunikacja z urządzeniem programującym i obsługa modułów komunikacyjnych[11].

## 4. Języki programowania programowalnych sterowników logicznych

Norma IEC61131-3 zawiera wymagania dotyczące języków programowania sterowników PLC. Dzięki wprowadzonym zasadom użytkownik może programować dowolny system PLC. W normie IEC61131-3 przedstawiono pięć języków programowania, w tym języki tekstowe oraz graficzne. Do języków tekstowych zalicza się:

- język listy rozkazów IL (*ang. Instruction List*) – jest podobny do języka typu assembler, w którym są dostępne rozkazy: logiczne, arytmetyczne, obsługi czasomierzy oraz liczników itp.;
- język tekstu strukturalnego ST (*ang. Structured Text*) – w języku ST jest stosowana struktura programu podobna do tych wykorzystywanych w językach wysokiego poziomu, takich jak: Pascal i C[12].

[10] Ibidem, s. 23.

[11] Ibidem, s. 23-24.

[12] A European Standard…, *op, cit.*, s. 123-134.

Pozostałe trzy to języki graficzne:

- język schematów drabinkowych LD (*ang. Ladder Diagram*) – program opracowany w tym języku jest zbliżony do schematu elektrycznego, Dopuszcza się w nim używanie oprócz symboli styków i cewek również funkcji oraz bloków funkcjonalnych;
- język funkcjonalnych schematów blokowych FBD (*ang. Function Block Diagram*) – w języku tym schemat przedstawiony jest w postaci połączonych symboli bramek, funkcji oraz bloków funkcjonalnych;
- język sekwencyjnego schematu funkcjonalnego SFC (*ang. Sequential Function Chart*) – w tym przypadku program przedstawiony jest w postaci grafów zawierających kroki i warunki przejścia pomiędzy tymi krokami. Definicja kroków i warunków przejścia są programowane w jednym z czterech wymienionych wcześniej języków[13].

**Tabela 1.** Języki programowania sterowników Siemens SIMATIC S7-1200 i S7-1500

| Język programowania | Simatic S7-1200 | Simatic S7-1500 |
|---|---|---|
| **Ladder diagram (LAD)**<br>**Język schematów drabinkowych (LD)** | + | + |
| **Function block diagram (FBD)**<br>**Język funkcjonalnych schematów blokowych (FBD)** | + | + |
| **Structured Control Language (SCL)**<br>**Język tekstu strukturalnego (ST)** | + | + |
| **Graph Language**<br>**Język sekwencyjnego schematu funkcjonalnego (SFC)** | - | + |
| **Statement list (STL)**<br>**Język listy rozkazów (IL)** | - | + |

**Źródło**: Opracowanie własne, na podstawie: *Przewodnik programowania dla S7-1200/S7-1500. Opis systemu – wydanie 3/2019r.*, https://publikacje.siemens-info.com/ebook/50/przewodnik-programowania-dla-s7-1200-s7-1500, s. 13 [dostęp: 21.09.2024].

W tabeli 1. zawarto wykaz znormalizowanych języków programowania sterowników PLC, wraz z ich odpowiednikami w środowisku TIA Portal, które jest przeznaczone do programowania sterowników firmy Siemens między innymi SIMATIC S7-1200 i SIMATIC S7-1500. W przypadku sterownika SIMATIC S7-1500 programista ma do wyboru aż pięć języków programowania, natomiast dla sterownika SIMATIC S7-1200 liczba dostępnych języków ogranicza się do trzech: LAD, SCL oraz FBD. Oznacza to, że możliwość wyboru języka programowania może być zależna od modelu sterownika PLC. Z taką sytuacją mamy do czynienia chociażby w przypadku sterowników firmy Siemens.

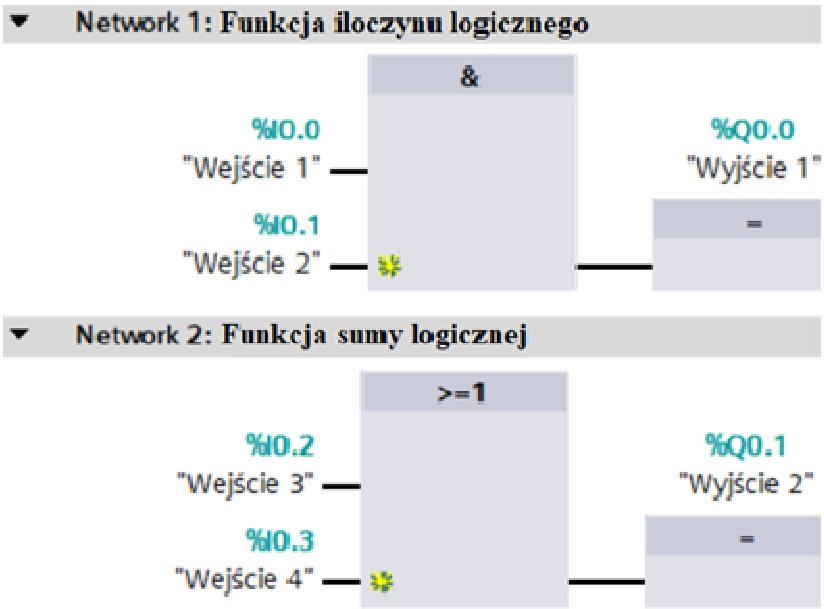
Wybór dostępnego języka programowania zależy przede wszystkim od preferencji programisty, ale w wielu przypadkach to zleceniodawca wskazuje, jaki język programowania należy zastosować w danym projekcie. Najczęściej tym językiem programowania

[13] Ibidem, s. 135-143.

jest język schematów drabinkowych LD/LAD. W takim przypadku uzupełnieniem możliwości języka graficznego może być język tekstowy, którym najczęściej jest obecnie język ST/SCL. Język programowania, w którym powstaje dana aplikacja może być również uzależniony od możliwości środowiska programistycznego. Przykładowo sterowniki PLC firmy EL-Piast programuje się w środowisku Macrocontrol jedynie w języku FBD, co całkowicie wyklucza możliwość wyboru języka programowania. Z kolei programowanie sterowników logicznych firmy Fatek Polska odbywa się w środowisku WinProLadder umożliwiającym programowanie w języku LAD oraz za pomocą odpowiednika języka SFC. Szczegółowy opis środowisk Macrocontrol i WinProLadder umieszczono na stronach internetowych producentów: EL-Piast[14] oraz Fatek Polska[15].

Każdy język programowania posiada swoje indywidualne cechy, które w mniejszym lub większym stopniu odpowiadają preferencjom konkretnego automatyka programisty.

**Język funkcjonalnych schematów blokowych (FBD)** – w języku tym realizacja programu opiera się na przepływie sygnału przez elementy przetwarzania sygnałów. Przepływ sygnału następuje z wyjścia jednej funkcji lub bloku funkcjonalnego do przyłączonego wejścia następnego elementu. Elementy składowe programu reprezentowane są przez połączone liniami prostokąty tworzące tzw. obwód (rys. 5). Oznacza to, że obwód (network) stanowi zespół połączonych ze sobą elementów, któremu można przypisać etykietę (nazwę).



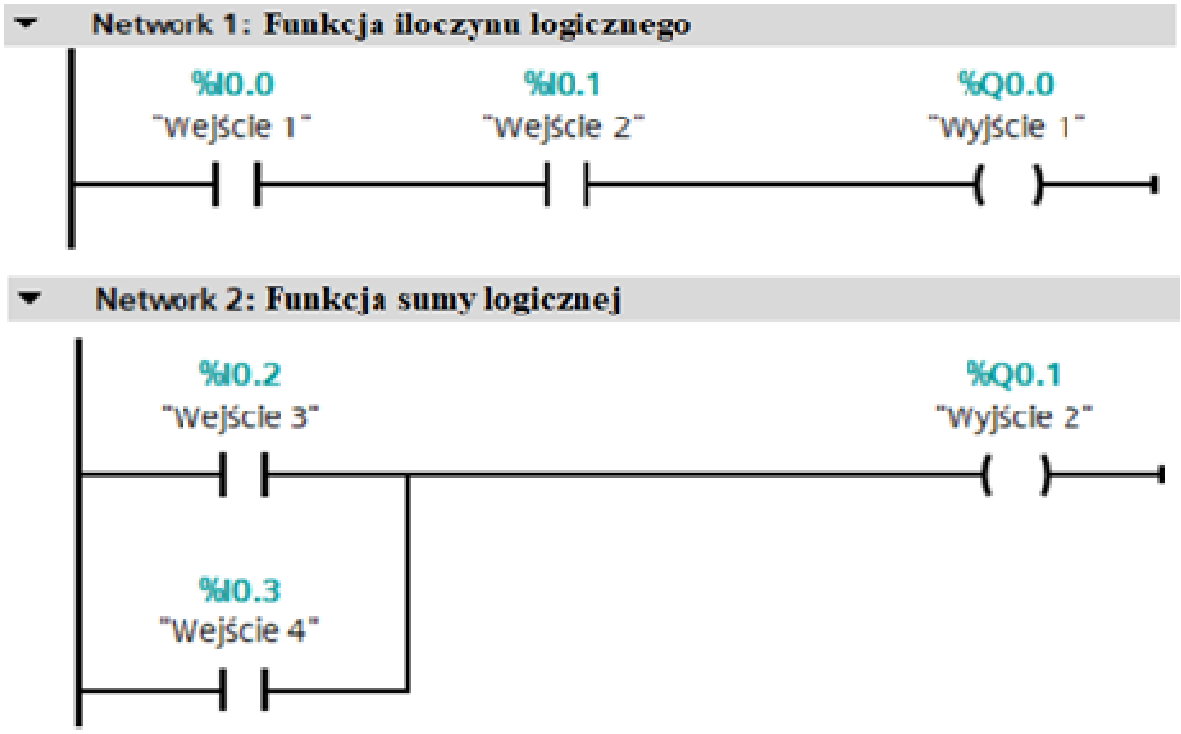

**Rys. 5.** Funkcja sumy logicznej i iloczynu logicznego w języku FBD

[14] Strona internetowa firmy EL-Piast Sp. z o. o. https://el-piast.com/macrocontrol [dostęp: 21.09.2024].

[15] Strona internetowa firmy FATEK Polska Sp. z o. o. https://www.fatek.pl/oprogramowanie-winproladder-po-polsku [dostęp: 21.09.2024].

Program jest wykonywany w kolejności wprowadzania obwodów, przy czym kolejność wykonywania obwodów może być zmieniona przez zastosowanie skoków warunkowych lub bezwarunkowych. W języku FBD nie może być realizowana operacja sumy logicznej OR poprzez równoległe łączenie wyjść elementów. Zamiast tego stosuje się jawną postać funkcji OR. W programie opracowanym w języku FBD nie występują symbole cewek i styków, tak jak w języku LD[16].

**Język schematów drabinkowych LD** – jest graficznym językiem programowania, gdzie program jest przedstawiony w postaci symboli, podobnie jak w schemacie drabinkowym układu przekaźnikowego. Elementy składowe programu są wzajemnie połączone (pionowo lub poziomo), tworząc odpowiedni algorytm sterowania. Podobnie jak w przypadku języka FBD elementy są umieszczone w obwodach, ale ograniczone z lewej i prawej strony przez szyny prądowe. Zgodnie z normą IEC61131-3 prawa szyna nie musi być rysowana i może pozostać w domyśle, a stan lewej szyny prądowej jest zawsze ustawiony na wartość ON. W programie napisanym w języku schematów drabinkowych instrukcje są wykonywane od lewej strony do prawej, przy czym wyjścia z obwodu mogą być wyznaczone tylko wtedy, gdy wszystkie wejścia zostaną odczytane oraz zakończy się przetwarzanie całego obwodu. Kolejne obwody są przetwarzane po kolei zgodnie ze schematem drabinkowym. Wyjątek stanowi sytuacja, w której do programu wprowadzono elementy zmieniające kolejność wykonywania obwodów (skoki bezwarunkowe i/lub warunkowe)[17].




**Rys. 6.** Funkcja sumy logicznej i iloczynu logicznego w języku LD

[16] S. Kacprzak , *Programowanie sterowników…*, *op. cit.*, s. 204.

[17] Ibidem, s. 186-187.

Przetwarzanie obwodu rozpoczyna się od przepływu sygnału przez styki, które oprócz cewek stanowią podstawowe elementy składowe programu (rys. 6). Styk jest elementem, który przekazuje stan do połączenia w schemacie, lecz nie modyfikuje wartości przypisanej do niego zmiennej. Każde połączenie występujące w schemacie elektrycznym służy do zasilenia (włączenia lub wyłączenia) określonego elementu wykonawczego. W języku LD takim elementem jest cewka, która stanowi zakończenie obwodu. Cewka przekazuje stan połącznia ze schematu do zmiennej, która jest do niej przypisana[18].

W języku LD są dostępne następujące rodzaje standardowych styków i cewek:

- styk normalnie otwarty lub normalnie zamknięty,
- styk reagujący na zbocze narastające lub opadające,
- cewka normalna oraz negująca,
- cewka ustawiająca oraz kasująca,
- cewka reagująca na zbocze narastające oraz opadające[19].

**Język listy rozkazów IL** – jest tekstowym językiem programowania o składni zbliżonej do języka niskiego poziomu typu assembler. Program napisany w języku IL składa się z ciągu rozkazów zawierających: operator, modyfikator oraz operand. Operator określa jakie działanie ma być wykonane, natomiast operand reprezentuje stałe lub zmienne. W języku IL dostępnych jest szereg operatorów wykonujących określone działanie: przesłanie, mnożenie, dzielenie, dodawanie, odejmowanie, porównywanie, skoki, operacje boolowskie (AND, OR, XOR), ustawienie i zerwanie wyjścia. Operandy działają w ten sposób, że bieżąca wartość jest wyznaczona jako wynik działania operatora z wartością aktualną oraz wartością operandu. Operator uwzględnia poprzednią wartość wyrażenia i przyjmuje ją jako wartość aktualną. W przypadku operatora **LD** aktualna wartość przyjmie wartość operandu. Przykładowo rozkaz **OR %IX0.0** odpowiada zapisowi **Wynik:=Wynik OR %IX0.0** w języku ST. W danym rozkazie można również zastosować modyfikator **N**, który oznacza negację operandu np. **ANDN %IX0.1** jest równoważne z zapisem **Wynik:=Wynik AND NOT %IX0.1** w języku ST[20].

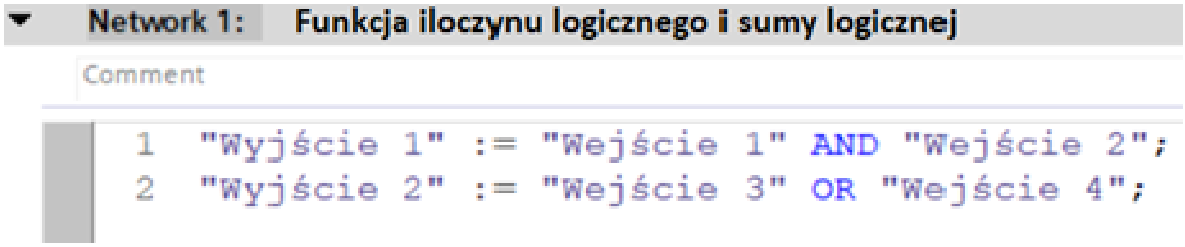
**Język tekstów strukturalnych ST** – jest językiem wysokiego poziomu, który operuje na sygnałach wejściowych i wyjściowych, bitach pamięci, zegarach, licznikach, jak również na wyrażeniach połączonych operatorami. Program składa się z podstawowych elementów, takich jak: wyrażenia i instrukcje, co sprawia że struktura programu jest bardzo przejrzysta. Poszczególne wyrażenia składają się z tzw. operandów i operatorów. Operandem może być zmienna, stała, wywoływana funkcja lub inne wyrażenie. Operatorami są symbole, które decydują, w jaki sposób ma być obliczany wynik np.: potęgowanie, negacja, mnożenie, dzielenie, dodawanie, odejmowanie, porównywanie, iloczyn i suma boo-

[18] T. Gilewski, *Podstawy programowania sterowników SIMATIC S7-1200 w języku LAD*, Wydawnictwo BTC, Legionowo 2017, s. 82.

[19] S. Kacprzak, *Programowanie sterowników…*, *op. cit.*, s. 188-189.

[20] Ibidem, s. 179-180.

lowska (rys. 7). Wyrażenia mogą być połączone ze sobą lub zagnieżdżone jedne w drugim przy użyciu operatorów[21].




**Rys. 7.** Funkcja sumy logicznej i iloczynu logicznego w języku ST

W języku ST programista ma do dyspozycji proste instrukcje: przypisania oraz wywołania bloku funkcyjnego, jak również instrukcje: wyboru oraz powtarzania instrukcji (pętle). Do zapisu nowej wartości w zmiennej lub przypisania wartości wynikającej z działania funkcji służy instrukcja przypisania oznaczona symbolem „ := ”. Dostępnych jest również kilka instrukcji warunkowych (IF oraz CASE) oraz pętli, które umożliwiają wielokrotne wykonanie określonego ciągu instrukcji[22].

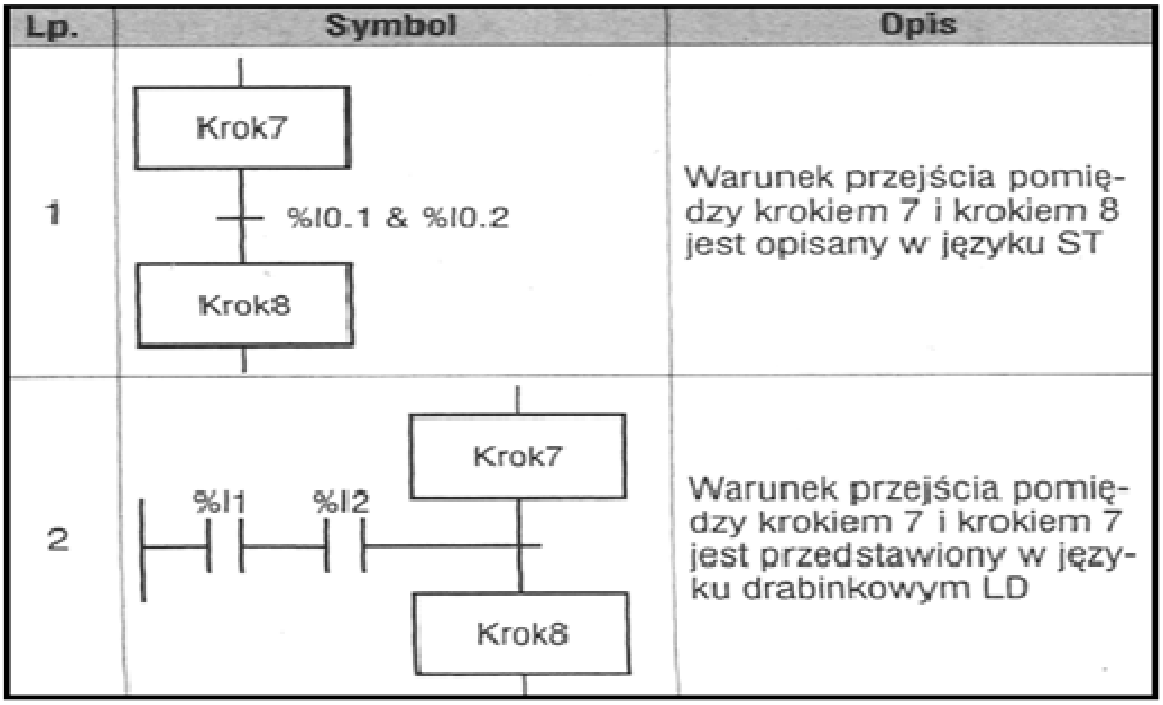
Składnia języka ST pozwala na szybkie opracowanie kodu źródłowego programu oraz realizację skomplikowanych obliczeń matematycznych i statystycznych. Wyrażenia i instrukcje zawarte w programie są zbliżone do powszechnie znanych języków programowania i posiadają cechy wspólne z językiem C i Pascal. Dlatego też język ST jest o wiele prostszy do nauki i zrozumienia dla początkujących i średniozaawansowanych programistów niż język IL[23].

**Język sekwencyjnego schematu funkcjonalnego SFC** – jest szczególnie przydatny w przypadku realizacji zadań, w których występuje sekwencja działań. W sekwencyjnych układach sterowania aktualny stan wyjść nie zależy tylko od stanu wejść, ale również od historii zdarzeń w poprzednich odcinkach czasu. Dlatego też konieczne jest przechowywanie informacji potrzebnych do realizacji założonego algorytmu sterowania. Program sterujący w języku SFC przedstawia się za pomocą kroków i warunków przejścia między tymi krokami (rys. 8). W ten sposób tworzona jest sieć, która jest podstawową strukturą programu.

[21] J. Kwaśniewski, *Język tekstu strukturalnego w sterownikach SIMATIC S7-1200 i S7-1500*, Wydawnictwo BTC, Legionowo 2014, s. 77.

[22] S. Kacprzak, *Programowanie sterowników…*, *op. cit.*, s. 199.

[23] T. Gilewski, *Podstawy programowania sterowników SIMATIC S7-1200 w języku SCL*, Wydawnictwo BTC, Legionowo 2015, s. 80.




**Rys. 8.** Warunki przejścia w języku SFC

**Źródło**: Kacprzak S., Programowanie sterowników PLC zgodnie z normą IEC61131-3 w praktyce, Wydawnictwo BTC, Legionowo 2011, s. 211.

Kroki i przejścia wymagają przechowywania informacji o ich stanie i mogą być zrealizowane np. za pomocą bloków funkcjonalnych. Każdy krok składa się z zestawu instrukcji, który nazywa się akcjami. Przy przejściach pomiędzy krokami, które mogą być aktywne lub nieaktywne, znajdują się warunki przejścia. Zmiana stanu polega na przechodzeniu pomiędzy krokami aktywnymi a następnymi zależnie od spełnienia warunków przejścia. Przejście to warunek przepływu sygnału pomiędzy kolejnymi elementami i jest on wynikiem rozwiązania wyrażenia boolowskiego. Jeśli przejście jest dozwolone i jednocześnie spełniony jest warunek przejścia, to następuje kasowanie przejścia oraz wyłączenie kroków poprzedzających i aktywacja kroków występujących po symbolu przejścia. Kolejne kroki i przejścia muszą występować naprzemiennie, tzn. dwa kroki muszą być rozdzielone przejściem oraz dwa przejścia muszą być rozdzielone krokiem. Jak już wspomniano definicję kroków i warunków przejścia programuje się w jednym z czterech wymienionych wcześniej języków[24].

## Podsumowanie

Każdy projekt zawierający program pracy sterownika PLC można zrealizować z wykorzystaniem dowolnego języka programowania. Przy czym w praktyce zabierze to programiście mniej lub więcej czasu, w zależności od złożoności i charakteru zadania. Dlatego też przyjęło się, że chcąc realizować projekty w sposób wydajny, automatyk programista powinien znać przynajmniej jeden język graficzny oraz tekstowy.

[24] S. Kacprzak, *Programowanie sterowników…*, *op. cit.*, s. 209-210.

Język LD/LAD jest najczęściej wykorzystywany przez programistów ze względu na swoje podobieństwo do schematów elektrycznych, jak również przejrzystość kodu źródłowego programu. Taka struktura programu pozwala na łatwą analizę programu w przypadku wystąpienia błędów. Za nauką języka drabinkowego przemawia również fakt, iż wielu zleceniodawców wymaga realizacji projektu właśnie w tym języku. Czasem jednak wybór języka programowania jest ograniczony przez środowisko programistyczne. Z taką sytuacją mamy do czynienia w programie Macrocontrol, gdzie programowanie sterowników firmy EL-Piast odbywa się za pomocą języka FBD. Dlatego znajomość również tego języka programowania czyni z automatyka programistę bardziej uniwersalnego.

W przypadku konieczności przeprowadzenia złożonych obliczeń warto również sięgnąć po język tekstowy. Programiści sterowników PLC mają do wyboru język IL/STL lub ST/SCL. Jak już wspomniano język IL jest językiem niskopoziomowym wymagającym od programisty szczegółowej architektury sterownika, wraz z rejestrami i rozkazami procesora. Z jednej strony operowanie rozkazami procesora daje programiście dużą kontrolę nad tym, co znajduje się w programie, ale z drugiej strony staje się to uciążliwe i czasochłonne w przypadku złożonych programów. Język IL uważany jest za trudny w zrozumieniu, więc szczególnie dla początkujących programistów optymalnym zdaje się język ST. Osoby mające styczność z programowaniem za pomocą języków wysokiego poziomu nie będą miały większych problemów z opracowaniem programu w tym języku dla sterownika PLC.

Wiele osób stawia sobie pytanie, czy warto jeszcze uczyć się języków programowania sterowników PLC w czasach dynamicznego rozwoju dużych modeli językowych LLM (*ang. Large Language Model*), które mogą wykonywać szereg zadań przetwarzania języka naturalnego. Duże modele językowe są tworzone w oparciu o gigantyczne zbiory danych zawierających również wiedzę specjalistyczną. Dzięki temu mogą rozpoznawać, tłumaczyć, przewidywać, generować tekst lub innego rodzaju treści. Wykorzystanie metod sztucznej inteligencji w automatyce nie jest czymś nowym, czego przykładem mogą być regulatory bazujące na logice rozmytej[25], jak również regulatory zawierające w swojej strukturze sieci neuronowe[26]. Nowość stanowią wspomniane duże modele językowe typu ChatGPT, które całkiem dobrze radzą sobie z generowaniem kodu źródłowego w tekstowych językach programowania. Oznacza to, że proces programowania w językach tekstowych, takich jak: ST oraz IL można wspomagać za pomocą sztucznych sieci neuronowych. Przy czym automatyk korzystający z tego narzędzia musi znać dany język programowania celem weryfikacji kodu źródłowego wygenerowanego przez sztuczną inteligencję.

W trakcie pisania niniejszego artykułu trwały testy pilotażowe narzędzia The Engineering Copilot, dedykowanego dla środowiska Tia Portal firmy Siemens. Jest to chatbot stanowiący pomost, który łączy środowisko TIA Portal z usługami Azure OpenAI. Narzędzie to generuje kod źródłowy dla sterowników PLC SIMATIC w języku tekstowym SCL. Odpowiedzi na zadane pytania są udzielane za pośrednictwem usług Azure OpenAI,

[25] J. Kwaśniewski, *Sterowniki SIMATIC S7-1200 i S7-1500 w zaawansowanych systemach sterowania*, Wydawnictwo BTC, Legionowo 2018, s. 193-206.

[26] Ibidem, s. 267-328.

a użytkownicy mają możliwość zintegrowania wyników bezpośrednio z trwającymi projektami w środowisku TIA Portal, co szczegółowo omówiono na stronie internetowej firmy Siemens[27]. Na tym etapie trudno stwierdzić jaka będzie skuteczność Engineering Copilot w przypadku generowania kodu źródłowego dla złożonych zadań automatyzacji, lecz z pewnością będzie to narzędzie skracające czas niezbędny do wdrożenia aplikacji.

Odpowiadając na pytanie, czy warto obecnie uczyć się języków programowania sterowników PLC, można stwierdzić, że obecnie z pewnością tak, gdyż istnieje duże zapotrzebowanie na tego typu specjalistów. Jednakże w dłuższej perspektywie czasowej trudno na to pytanie jednoznacznie odpowiedzieć.

## Bibliografia

1. A European Standard EN 61131-3, *Programmable controllers - Part 3: Programming languages (IEC 61131-3:2013)*, Brussels 2013.
2. Gilewski T., *Podstawy programowania sterowników SIMATIC S7-1200 w języku LAD*, Wydawnictwo BTC, Legionowo 2017.
3. Gilewski T., *Podstawy programowania sterowników SIMATIC S7-1200 w języku SCL*, Wydawnictwo BTC, Legionowo 2015.
4. Gilewski T., *Szkoła programisty PLC: sterowniki przemysłowe*, Wydawnictwo Helion, Gliwice 2017.
5. Kacprzak S., *Programowanie sterowników PLC zgodnie z normą IEC61131-3 w praktyce*, Wydawnictwo BTC, Legionowo 2011.
6. Kwaśniewski J., *Język tekstu strukturalnego w sterownikach SIMATIC S7-1200 i S7-1500* , Wydawnictwo BTC, Legionowo 2014.
7. Kwaśniewski J., *Programowalny sterownik SIMATIC S7-300 w praktyce inżynierskiej*, Wydawnictwo BTC, Legionowo 2009.
8. Kwaśniewski J., *Sterowniki SIMATIC S7-1200 i S7-1500 w zaawansowanych systemach sterowania*, Wydawnictwo BTC, Legionowo 2018.
9. Kwaśniewski J., *Sterowniki SIMATIC S7-1200 w praktyce inżynierskiej*, Wydawnictwo BTC, Legionowo 2013.
10. Publikacje firmy Siemens, Dokumentacja sterownika Siemens SIMATIC S7-1200 - Easy Book - wydanie 4/2012, https://publikacje.siemens-info.com/pdf/687/S7-1200%20Easy%20Book.pdf [dostęp: 20.09.2024].
11. Publikacje firmy Siemens, Przewodnik programowania dla S7-1200/S7-1500. Opis systemu - wydanie 3/2019, https://publikacje.siemens-info.com/ebook/50/przewodnik-programowania-dla-s7-1200-s7-1500 [dostęp: 21.09.2024].
12. Strona internetowa firmy EL-Piast Sp. z o. o., https://el-piast.com/macrocontrol [dostęp: 21.09. 2024].
13. Strona internetowa firmy FATEK Polska Sp. z o. o., https://www.fatek.pl/oprogramowanie-winproladder-po-polsku [dostęp: 21.09.2024].
14. Wsparcie online firmy Siemens, Opis narzędzia The Engineering Copilot w TIA Portal, https://support.industry.siemens.com/cs/document/109955813/siemens-industrial-copilot-for-engineering?dti=0&lc=en-MW [dostęp: 20.09.2024].

[27] Wsparcie online firmy Siemens, Opis narzędzia The Engineering Copilot w TIA Portal, https://support.industry.siemens.com [dostęp: 20.09.2024]

**Dane kontaktowe**

Piotr Wilk, piotr.wilk@akademiarac.edu.pl