ARCHIVES of FOUNDRY ENGINEERING

Published quarterly as the organ of the Foundry Commission of the Polish Academy of Sciences

ISSN (1897-3310)
Volume 13
Special Issue
1/2013
189–192

37/1

Programowanie Sterowników PLC Metodą Nauczania

R. Więcławek

Politechnika Wrocławska, Instytut Technologii Maszyn i Automatyzacji, ul. Łukasiewicza 5, 50-370 Wrocław, Polska
*Corresponding author. E-mail address: e-mail: rafal.wieclawek@pwr.wroc.pl

Received 17.04.2012; accepted in revised form 08.05.2012

Streszczenie

Do automatyzacji procesów odlewniczych powszechnie stosowane są programowalne sterowniki logiczne PLC. Języki oraz znane metody programowania sterowników PLC sprawiają duże trudności przy implementacji rozwiązań dotyczących sterowania procedurami sekwencyjnymi. Dlatego podjęto prace nad opracowaniem metody, która umożliwiłaby łatwe programowanie procedur sekwencyjnych przez użytkowników nie będących automatykami. Efektem tych prac jest aplikacja opracowana dla sterowników Simatic S7-300, która umożliwia ich programowanie metodą stosowaną w robotyce – programowaniem poprzez nauczanie. Przeprowadzone badania wykazały możliwość stosowania tej metody do programowania dowolnych procedur sekwencyjnych o dowolnej liczbie kroków. Jedynym ograniczeniem jest rozmiar pamięci operacyjnej sterownika. Aplikację można też implementować w dowolnym sterowniku PLC, którego języki programowania są zgodne z normą IEC61131. Dalsze badania będą dotyczyły opracowania aplikacji umożliwiającej programowanie procedur sekwencyjnych, w których przejście do następnego kroku jest uzależnione od upływu zadanego czasu.

Słowa kluczowe: Mechanizacja i automatyzacja procesów odlewniczych, Komputerowe wspomaganie produkcji odlewniczej, Sterowniki PLC, Języki programowania sterowników PLC

1. Wprowadzenie

Automatyzacja procesów odlewniczych przynosi duże korzyści między innymi dzięki wzrostowi wydajności i jakości produkcji [1,2,3,4,5]. Obecnie podstawowym narzędziem automatyzacji procesów odlewniczych są programowalne sterowniki logiczne PLC [6,7,8,9,10]. Są to uniwersalne urządzenia mikroprocesorowe, których głównym zadaniem jest reagowanie na zmiany wejść przez obliczanie wyjść na podstawie zaprogramowanych reguł sterowania.

Do programowania sterowników PLC stosuje się języki tekstowe (IL, ST) i języki graficzne (LD i FBD). Jednak programowanie procedur sekwencyjnych przy użyciu tych języków jest trudnym i czasochłonnym zadaniem. Znane są metody, które ułatwiają syntezę układów sterowania sekwencyjnego, ale wymagają od użytkownika dużej wiedzy i doświadczenia w stosowaniu tych metod [10,11,12]. Pewnym ułatwieniem jest stosowanie języka SFC do programowania procedur sekwencyjnych, ale i tutaj każda zmiana w algorytmie procesu wymaga odpowiedniej zmiany w kodzie programu, którą może wykonać osoba o odpowiednich kwalifikacjach.

W związku z tym podjęto prace nad opracowaniem aplikacji, która umożliwiłaby łatwe programowanie procedur sekwencyjnych użytkownikom, którzy nie mają specjalistycznej wiedzy z zakresu programowania sterowników PLC, natomiast dobrze znają urządzenie i proces technologiczny przez nią realizowany.

Efektem tych prac jest aplikacja sporządzona dla sterowników Simatic S7-300, napisana w języku STL, która umożliwia programowanie procedur sekwencyjnych metodą nauczania. Jest

to jedna z metod programowania stosowanych w robotyce, znana jako nauczanie przez pokazywanie [13]. Metoda polega na przemieszczaniu robota do pożądanego punktu docelowego i zapisywaniu jego pozycji w pamięci, a następnie jego odczytaniu. W fazie nauczania robota użytkownik może sterować nim za pomocą ręcznego programatora. Programator ręczny jest przenośnym pulpitem z klawiszami, które umożliwiają sterowanie napędami robota a także klawiszami służącymi do: zapamiętania pozycji, wyboru trybu pracy itd.

Taki sposób programowania został zrealizowany w aplikacji opisanej poniżej.

2. Stanowisko badawcze

Stanowisko, na którym testowano opracowaną aplikację, jest zbudowane z następujących elementów:

- jednostki centralnej CPU 313C-2DP z wbudowanym modułem 16 kanałów wejściowych i 16 kanałów wyjściowych,
- 16-kanałowego modułu wyjść cyfrowych,
- dwóch 32-kanałowych modułów wejść cyfrowych,
- panelu operatorskiego.

Do sterowania automatyzowanym urządzeniem zarezerwowano 32 wejścia i 32 wyjścia zwane dalej odpowiednio wyjściami i wejściami procesu. Zostały one skonfigurowane w programie HW Config w taki sposób, aby zajmowały w przestrzeni adresowej ciągły obszar 32 bitów. W związku z tym wyjścia procesu obejmują adresy począwszy od I0.0 do I3.7, czyli jedno podwójne słowo ID0. Podobnie zostały skonfigurowane wejścia procesu i obejmują adresy począwszy od Q0.0 do Q3.7, czyli jedno podwójne słowo QD0.

Panel operatorski służy do programowania układu sterowania. Jest wyposażony w stabilny przycisk *Auto* (I8.0) do przełączania między pracą w trybie automatycznym a trybem programowania oraz przycisk niestabilny *Zapis* (I8.1), który w trybie programowania służy do zapisu aktualnego stanu wejść i wyjść procesu w pamięci sterownika.

Ponadto panel operatorski jest wyposażony w 32 stabilne przyciski, które są podłączone do wejść sterownika zwanych dalej wyjściami panelu, obejmujących adresy począwszy od I4.0 do I7.7 (ID4). Wyjścia panelu mają zastosowanie do sterowania sygnałami wejściowymi procesu w trybie programowania. Odbywa się to poprzez kopiowanie zawartości pamięci wskazywanej przez ID4 do obszaru QD0. Aby np. ustawić wejście procesu o adresie Q0.0 w stan 1, należy wcisnąć przycisk panelu operatorskiego podłączonego do wejścia o adresie I4.0, aby ustawić Q0.1 w stan 1, należy wcisnąć przycisk panelu operatorskiego podłączonego do wejścia o adresie I4.1 itd.

3. Aplikacja do programowania PLC metodą nauczania

Opracowana aplikacja jest złożona z następujących elementów: bloku danych DB1, bloku programowego OB1, oraz funkcji FC1 i FC2 (tabela 1).

W bloku DB1 zadeklarowano jednowymiarową tablicę o 128 wierszach do przechowywania elementów typu DWORD. Tablica służy do przechowywania informacji o stanie kolejnych tranzycji i kroków. Krok określa zestaw działań sterownika PLC skojarzonych z etapem procesu, natomiast tranzycja reprezentuje warunki logiczne realizacji poszczególnych kroków programu. A zatem w wierszu o numerze $2n$ jest przechowywana tranzycja T_n, czyli stan wyjść procesu w momencie, w którym rozpoczęła się realizacja kroku S_n, natomiast w wierszu o numerze $2n+1$ jest zapisany stan wejść procesu w momencie realizacji kroku S_n. W wierszu $n=0$ jest przechowywana informacja o liczbie zapisanych wierszy w tejże tablicy a tym samym o liczbie kroków występujących w algorytmie sterowania.

Zawartość bloku DB1 jest modyfikowana w trybie programowania przez funkcję FC2, po naciśnięciu przycisku *Zapisz*. W pierwszym szczeblu funkcji FC2 (Network 1) następuje wyzerowanie informacji o liczbie zapisanych wierszy w DB1. Ta operacja jest wykonywana tylko jeden raz w ciągu całego cyklu uczenia. W trzecim szczeblu następuje skopiowanie stanu wyjść panelu operatorskiego, ustawionych przez użytkownika, do wejść procesu. Użytkownik musi zadbać o odpowiednie ustawienie przycisków panelu operatorskiego, a tym samym o odpowiedni stan wejść procesu, aby umożliwić realizację danego kroku i przejście do kroku następnego. Szczebel czwarty odpowiada za zapisanie w bloku danych DB1 aktualnego stanu wyjść procesu (tranzycji) i aktualnego stanu wejść procesu (kroku). W wyniku tych operacji rozmiar tablicy danych w bloku DB1 powiększa się o dwa wiersze. W związku z tym wskaźnik wolnego wiersza tablicy w bloku DB1 jest zwiększany o 8 bajtów (gdyż jeden wiersz tablicy zajmuje 4 bajty) i ta informacja jest zapisywana w pierwszym wierszu tejże tablicy. Użytkownik może przystąpić do zapisu następnej pary tranzycja-krok po ustabilizowaniu procesu, czyli w momencie, w którym nie zachodzą już żadne zmiany wywołane zapisem poprzedniego stanu.

Funkcja FC2 jest wywoływana cyklicznie z poziomu bloku OB1 pod warunkiem, że przycisk *Auto*=0 (tabela 1). Aby wyjść z trybu programowania do trybu pracy automatycznej, należy wcisnąć przycisk *Auto* i wówczas z poziomu bloku OB1 będzie cyklicznie wywoływana funkcja FC1.

Funkcja FC1 (tabela 1) odpowiada za realizację sterowania zgodnie z algorytmem zapisanym trybie programowania. Pierwszy szczebel jest wykonywany tylko raz, po przejściu z trybu programowania do trybu automatycznego i służy do ustawienia wskaźnika aktualnie realizowanego kroku ($n=1$). W szczeblu drugim następuje sprawdzenie, czy jest spełniona tranzycja T_n. A zatem następuje porównanie aktualnego stanu wyjść procesu ze stanem wyjść zapisanym w bloku DB1 dla danej tranzycji T_n. Jeśli wynik porównania jest równy 1 (tranzycja T_n spełniona), to realizowane są instrukcje ze szczebla trzeciego, które ustawiają wejścia procesu zgodnie ze stanem zapisanym w DB1 dla danego kroku S_n. Jeśli wynik porównania jest negatywny, wówczas realizowany jest szczebel piąty, w którym następuje sprawdzenie, czy jest spełniona tranzycja T_{n+1}. Niespełnienie tranzycji T_{n+1} powoduje zakończenie realizacji FC1 i powrót do OB1. Spełnienie tranzycji T_{n+1} powoduje inkrementację wskaźnika aktualnie realizowanego kroku ($n=n+1$), co odpowiada uaktywnieniu kroku S_{n+1}. Jeśli wskaźnik jest równy liczbie kroków zapisanych w bloku DB1, wówczas przypisuje mu się wartość 1 i rozpoczyna się nowy cykl programowy, w przeciwnym razie wskaźnik aktualnie realizowanego kroku jest zwiększany o 1 (network 8) .

Tabela 1.
Bloki programowe i funkcje aplikacji do programowania sterowników Simatic S7-300 metodą nauczania

FC2

```
Network 1:
      A     "Zapis"
      AN    M     0.0
      JNB   _001
      L     0
      T     DB1.DBD     0
      SET
      SAVE
      CLR
_001: A     BR
      S     M     0.0
Network 2:
      A     "Zapis"
      FP    M     0.1
      =     M     0.2
Network 3:
      A     M     0.2
      JNB   _003
      L     ID4
      T     QD0
_003: NOP   0
Network 4:
      A     M     0.2
      JNB   _002
      OPN   DB    1
      L     DBD   0
      SLD   3
      LAR1
      L     ID0
      T     DBD [AR1,P#4.0]
      L     QD0
      T     DBD [AR1,P#8.0]
      L     L#8
      L     DBD   0
      +D
      T     DBD   0
_002: NOP   0
```

FC1

```
Network 1:
      A     M     0.0
      JNB   _001
      L     L#0
      T     MD    1
      SET
      SAVE
      CLR
-001: A     BR
      R     M     0.0
Network 2:
      OPN   DB    1
      L     MD    1
      SLD   3
      LAR1
      L     DBD[AR1,P#4.0]
      L     ID0
      ==D
      JCN   et1
Network 3:
      L     MD    1
      SLD   3
      LAR1
      L     DBD[AR1,P#8.0]
      T     QD0
      NOP   0
Network 4:
et1:  NOP   0
Network 5:
      L     MD    1
      SLD   3
      LAR1
      L     DBD[AR1,P#12.0]
      L     ID0
      ==D
      JCN   et2
Network 6:
      L     MD    1
      L     L#8
      +D
      T     MD    1
      NOP   0
Network 7:
et2:  NOP   0
Network 8
      A(
      L     MD    1
      L     L#8
      +D
      T     #wskaźnik
      AN    OV
      SAVE
      CLR
      A     BR
      )
      A(
      L     #wskaźnik
      L     DB1.DBD
            0
      ==D
      )
      JNB   _002
      L     0
      T     MD    1
_002: NOP   0
```

OB1

```
Network 1:
      A     "Auto"
      CC    FC    1
Network 2:
      AN    "Auto"
      CC    FC    2
```

4. Przykład

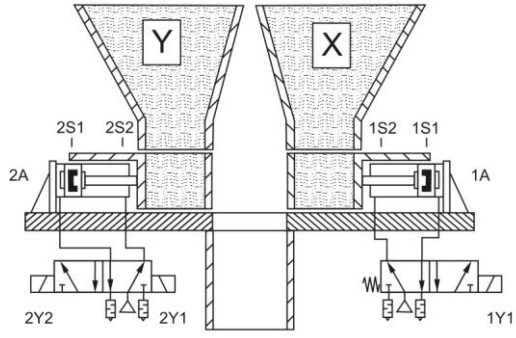
Programowanie sterownika Simatic S7-300 metodą nauczania zostanie pokazane na przykładzie dozownika materiałów sypkich (rys.1).

Dozownik realizuje następujący algorytm [14]:
krok 1: dozowanie materiału X,
krok 2 napełnienie dozownika materiału X,
krok 3: dozowanie materiału Y,
krok 4: napełnienie dozownika materiału Y.

W tabeli 2 pokazano przyporządkowanie adresów wejścia i wyjścia sterownika PLC poszczególnym sygnałom wejściowym i wyjściowym procesu i panelu operatorskiego.

W celu zaprogramowania sterownika PLC do realizacji powyższego algorytmu należy zwolnić przycisk *Auto*. Jeśli urządzenie nie jest w pozycji wyjściowej, to należy je do niej sprowadzić przy pomocy panelu operatorskiego. W tym celu należy ustawić: *2A*+=0, *2A*-=1, *1A*+=0 i nacisnąć *Zapis*. Po sprowadzeniu do pozycji wyjściowej należy ponownie nacisnąć i zwolnić przycisk *Auto*. Jeśli urządzenie znajduje się już w pozycji wyjściowej *(2S1*=1 i *1S1*=1*)*, należy wcisnąć przycisk *Start* i pozostawić w tym stanie do końca programowania. Dzięki temu przyciskiem *Start* będzie można zatrzymać i uruchomić realizację procesu w dowolnym momencie. Następnie należy wykonać poniższe czynności w podanej kolejności:

1. Ustawić: $1A^+$=1, $2A^-$=0, $2A^+$=0 i nacisnąć *Zapis.* Ta czynność spowoduje wysuw siłownika 1A.
2. Jeśli siłownik 1*A* jest wysunięty (1*S2*=1), to ustawić: $1A^+$=0, $2A^-$=0, $2A^+$=0 i nacisnąć *Zapis* – nastąpi wsuw siłownika *1A.*



Rys.1. Schemat dozownika materiałów sypkich

3. Jeśli siłownik *1A* jest wsunięty (1*S1*=1), to ustawić: $1A^+$=0, $2A^-$=0, $2A^+$=1 i nacisnąć *Zapis*, co spowoduje wysuw siłownika *2A*.
4. Jeśli siłownik *2A* jest wysunięty (*2S2*=1), to ustawić: $1A^+$=0, $2A^-$=1, $2A^+$=1 i nacisnąć *Zapis*. Dzięki temu urządzenie znajdzie się w pozycji wyjściowej.
5. Jeśli siłownik *2A* jest wsunięty (*2S1*=1), to ustawić: $1A^+$=0, $2A^-$=0, $2A^+$=0 i nacisnąć *Zapis*. Ta czynność odpowiada za ustawienie takiego stanu sygnałów wejściowych procesu, jaki jest wymagany w pozycji wyjściowej.

Po wykonaniu czynności 1-5 układ sterowania jest już zaprogramowany i gotowy do działania w trybie automatycznym.

Tabela 2.
Adresy wejść i wyjść procesu oraz panelu operatorskiego w PLC

Wyjścia panelu operatorskiego		Wejścia i wyjścia procesu	
Adres	Nazwa zmiennej	Adres	Nazwa zmiennej
Q4.0	$1A^+$	Q0.0	*1Y2*
Q4.1	$2A^+$	Q0.1	*2Y2*
Q4.2	$2A^-$	Q0.2	*2Y1*
Q8.0	*Auto*	I0.0	*Start*
Q8.1	*Zapis*	I0.1	*1S1*
		I0.2	*1S2*
		I0.3	*2S1*
		I0.4	*2S2*

5. Podsumowanie

Zaprezentowano aplikację do programowania sterowników Simatic S7-300 metodą nauczania, ale można ją łatwo przystosować do programowania dowolnego sterownika PLC, którego oprogramowanie jest zgodnie z normą IEC 61131. Ograniczeniem tej aplikacji jest liczba 32 wejść i 32 wyjść, natomiast liczba kroków jest ograniczona jedynie rozmiarem pamięci sterownika. Metoda może być przydatna, gdy istnieje potrzeba częstych zmian algorytmu procesu, natomiast obsługa nie posiada kwalifikacji do programowania sterowników PLC przy użyciu standardowych narzędzi. Dalsze badania będą dotyczyły opracowania aplikacji umożliwiającej programowanie procedur sekwencyjnych, w których przejście do następnego kroku jest uzależnione od upływu zadanego czasu.

Literatura

[1] Herman, A. (2012). The Rationalization of Automatic Units for HPDC Technology. *Archives of Foundry Engineering.* 12(2), 29-34.

[2] Ziółkowski, E. & Śmierciak, P. (2012). Comparison of Energy Consumption in the Classical (PID) and Fuzzy Control of Foundry Resistance Furnace. *Archives of Foundry Engineering*. 12(3), 129-132.

[3] Fedoryszyn, A. (2007). Assessment of systems for mechanisation of casting production. *Archives of Foundry Engineering*. 7 (3), 83-86.

[4] Kukla, S. (2007). Evaluation and verification of time and costs of production activities in foundry industry. *Archives of Foundry Engineering.* 7 (3), 79-82.

[5] Fedoryszyn, A. (2007). A characteristic of design solutions for flask moulding lines. *Archives of Foundry Engineering.* 7(1), 5-8.

[6] Fedoryszyn, A. (2010). Control of rebonding sand mixing as a condition for optimisation of the sand feeding system in the casting line. *Archives of Foundry Engineering*. 10(2), 47-50.

[7] Fedoryszyn A. (2007). Flask casting lines with home-produced moulding devices. *Archives of Foundry*. 6(21), 31-36.

[8] Jopkiewicz, A. (2002). Control and monitoring systems of Cupola processes. *Archives of Foundry.* 1(1).

[9] Chmielorz, W. & Dombek, A. (1999). Control of system Polko pneumatic conveying devices. *Solidification of Metals and Alloys*. 1(39), 24-32.

[10] Dworzak, Ł. & Mikulczyński, T. (2009). Synthesis of sequential control algorithms for pneumatic drives controlled by monostable valves. *Archives of Foundry Engineering.* 9(3), 35-40.

[11] Pigiel, M., Więcławek, R. & Wikiera, R. (2007). Application of Graftech method for programming of discrete technological process. *Archives of Foundry Engineering.* 7(1), 9-12.

[12] Nowak, D. & Więcławek, R. (2011). Synthesis of pneumatic controll systems. *Archives of Foundry Engineering*. 11(2), 159-164.

[13] Craig, J. (1989). *Introduction to Robotics: Mechanics and Control*. Reading: Addison-Wesley Publ. Co

[14] Samsonowicz, Z. (1985). *Automation of foundry processes*. Warszawa: WNT.

Teaching-in programmable logic controllers

Abstract

In automation of foundry processes, commonly used are programmable logic controllers (PLC). Languages and known methods of programming the PLC controllers cause big difficulties at implementing solutions concerning control of sequential procedures. This is why undertaken were the works on developing a method that would facilitate programming sequential procedures by the users not being automatic specialists. These works resulted in an application developed for the controllers Simatic S7-300 written in the STL language that makes possible their programming with the method commonly used in robotics, i.e. by teaching-in. The carried-out examinations showed a possibility to use this method for programming any sequential procedures with any number of steps. The only restriction is size of operational memory of the controller. The application can be also implemented in any PLC controller whose programming language is compliant with the standard IEC 61131. Further research will be aimed at developing an application that would permit programming sequential procedures where transition to the following step is dependent on expiry of the preset time.