dr inż. Krzysztof Oprzędkiewicz
Katedra Automatyki,
Wydział EAIiE AGH Kraków

ZASTOSOWANIE STANDARDOWYCH NARZĘDZI PROGRAMOWANIA PLC DO IMPLEMENTACJI SPECJALNYCH ALGORYTMÓW STEROWANIA

W pracy omówiono zagadnienia praktycznej implementacji pewnej klasy specjalnych algorytmów sterowania procesami przemysłowymi na sterownikach PLC. Rozważono algorytmy sterowania zawierające część realizującą kompensację dynamiczną, zaprojektowane do sterowania procesami z opóźnieniem (skupionym lub rozłożonym). Realizacja kompensacji dynamicznej bazuje na modelu obiektu regulacji. W związku z tym podczas realizacji praktycznej tych algorytmów sterowania niezbędne jest zamodelowanie na sterowniku PLC pewnych podstawowych typów elementów dynamicznych, opisanych równaniami stanu lub transmitancjami. Przedyskutowano możliwości realizacji specjalnych algorytmów sterowania z wykorzystaniem najbardziej rozpowszechnionych narzędzi programowych, do jakich można zaliczyć język drabinkowy LD oraz język listy instrukcji IL.

THE IMPLEMENTATION OF SPECIAL CONTROL ALGORITHMS IN PLCWITH THE STANDRARD TOOLS USE

In paper a practical implementation of a class of special control algorithms is considered. We deal with the algorithms of the dynamic compensation. Those algorithms can be used for control of systems with delay or distributed-parameter systems. As a physical environment for this implementation the PLCs with standard programming tools were used. In paper are discussed the possibilities of use of the standard programming tools: IL language and LD language for realisation of the dynamic compensation algorithms with the serial or the feedback structure.

UWAGI WSTĘPNE

W wielu sytuacjach praktycznych zastosowanie specjalnych algorytmów sterowania zapewnia znacznie lepszą (w sensie wybranego wskaźnika) jakość regulacji, niż algorytmy konwencjonalne (np. algorytm PID). Podczas konstruowania algorytmu specjalnego w dużym stopniu wykorzystuje się informacje dot. modelu matematycznego obiektu (dokładnego lub zastępczego), często też stosuje się model lub jego fragment do realizacji algorytmu sterowania. Realizacja algorytmów specjalnych z użyciem komputera klasy PC lub komputera przemysłowego i zaawansowanych środowisk programowych nie nastręcza praktycznie żadnych trudności, jednak takie rozwiązanie często nie jest uzasadnione w ze względów niezawodnościowych lub (przede wszystkim) ekonomicznych. Z tego względu wydaje się, że z praktycznego punktu widzenia najbardziej uzasadniona jest realizacja układów korekcji dynamicznej na typowych sterownikach programowalnych: regulatorach

wielofunkcyjnych lub sterownikach PLC. Zasadniczym atutem PLC jest w rozważanym przypadku większa „otwartość" oprogramowania oraz jego normalizacja (norma IEC-1131). Ponadto jest to sprzęt powszechnie spotykany w przemyśle i w związku z tym proponowane rozwiązania mogą znaleźć szerokie zastosowanie.
W pracy zostaną omówione następujące zagadnienia:

- Standardowe narzędzia programowe sterowników PLC,
- Struktury rozważanych algorytmów specjalnych i ich realizacja w języku drabinkowym,
- Podstawowe elementy algorytmów specjalnych i ich realizacja,
- Wnioski i uwagi końcowe.

1. ŚRODOWISKO PLC DO REALIZACJI ALGORYTMÓW SPECJALNYCH

Norma IEC 1131-3, której przedmiotem jest oprogramowanie sterowników, definiuje 5 podstawowych języków programowania sterowników PLC, podzielonych na 3 grupy. Do języków tekstowych zaliczamy języki typu lista instrukcji (IL) , będące odpowiednikiem asemblerów oraz języki wysokiego poziomu. Z kolei do języków graficznych zaliczamy najczęściej stosowany w praktyce język drabinkowy (LD) oraz język schematów blokowych (FBD). Ponadto do osobnej klasy zalicza się jeszcze język SFC, stosowany do opisu procesów sekwencyjnych modelowanych z użyciem sieci Petriego. Niektórzy producenci bardziej zaawansowanych systemów sterownikowych oferują opcjonalne (nieznormalizowane) pakiety do realizacji złożonych struktur sterowania, które mogą być przydatne do realizacji algorytmów specjalnych. Jako przykład można tu podać SIMATIC MODULAR PID CONTROL, który jest rozszerzeniem pakietu standardowego STEP-7. Jednak tak zaawansowane narzędzia programowe są jeszcze obecnie spotykane stosunkowo rzadko, a ponadto – dość kosztowne.

Do standardowych elementów organizacyjnych oprogramowania PLC zdefiniowanych przez normę IEC 1131-3 zaliczamy programy, funkcje i bloki funkcyjne. W programach realizujących specjalne algorytmy sterowania konieczne jest użycie wszystkich tych elementów organizacyjnych, ze szczególnym uwzględnieniem bloków funkcyjnych.

Norma IEC-1131 definiuje również typy danych, dostępnych w systemach sterownikowych, przy czym różne typy danych są dostępne na różnym sprzęcie. Możliwość wykonywania operacji zmiennoprzecinkowych może, ale nie musi być zaletą konkretnego sterownika. Należy bowiem pamiętać, że operacje zmiennoprzecinkowe choć z jednej strony znacznie zwiększają dokładność obliczeń i zmniejszają nakład pracy przy pisaniu programu, to z drugiej strony – w większości przypadków znacznie wydłużają czas obliczeń. We wszystkich algorytmach specjalnych realizowanych przez autora stosowano arytmetykę zmiennoprzecinkową, gdyż w rozważanych przypadkach czas obliczeń nie był parametrem krytycznym.

Spośród tych standardowych narzędzi programistycznych, zdefiniowanych w normie, najbardziej predysponowany do realizacji rozważanych metod korekcji dynamicznej jest język schematów blokowych (FBD), ponieważ zgodnie z IEC-1131 jest on przeznaczony do realizacji złożonych struktur sterowania. Niestety, nie wszędzie jest on dostępny w wersji określonej przez normę (np. STEP-7 – budowa pełnego schematu blokowego możliwa jest tylko dla sygnałów binarnych). W praktyce najczęściej stosowane są języki programowania: język drabinkowy LD, który należy uznać za podstawowe narzędzie programowania urządzeń tej klasy oraz język typu lista instrukcji (IL). Ponadto w bardziej zawansowanych

środowiskach można spotkać języki strukturalne wysokiego poziomu. Przykładem języka wysokiego poziomu jest język SCL, dostępny jako rozszerzenie pakietu STEP-7.

W tym miejscu od razu należy dodać, że w/w najczęściej stosowane w praktyce języki nie są optymalnym narzędziem do realizacji algorytmów specjalnych. Wynika to z ich innego przeznaczenia. W związku z tym zaproponowane rozwiązania muszą być uznane za pewien kompromis pomiędzy wymaganiami stawianymi przed algorytmem specjalnym a realnymi możliwościami sprzętu.

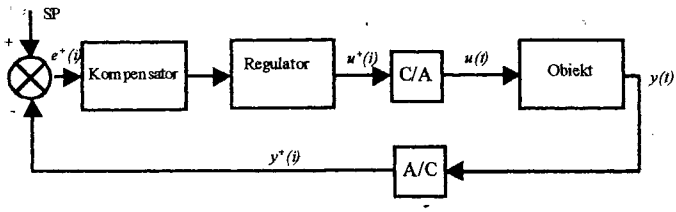
Podczas realizacji algorytmu specjalnego należy w środowisku sterownika PLC zamodelować jedną ze struktur omawianych w następnym rozdziale. Jest oczywiste, że podczas budowy tej struktury należy dokładnie znać sposób interpretacji schematu drabinkowego przez sterownik podczas wykonywania programu. Przypomnijmy więc, że podczas jednego cyklu programowego wykonanie programu napisanego w języku drabinkowym odbywa się zgodnie z kolejnością szczebli z góry na dół, a w ramach szczebla – od strony lewej do prawej, chyba, że schemat zawiera instrukcje skoku, które zmieniają tę kolejność.

2. STRUKTURY ALGORYTMÓW SPECJALNYCH I ICH REALIZACJA W JĘZYKU DRABINKOWYM

Struktury algorytmów specjalnych rozważymy na przykładzie algorytmów kompensacji własności dynamicznych obiektu. Ta klasa algorytmów specjalnych jest znana w automatyce od dość dawna. Przykładem mogą tu być: predyktor Smitha i regulator Reswicka dla układów z opóźnieniem (zob. [2]), układy kompensacji dynamicznej dla systemów nieskończenie wymiarowych (zob.[5]) W przypadku zastosowania algorytmów specjalnych tej klasy poprawa jakości regulacji uzyskiwana jest na drodze kompensacji dynamiki obiektu regulacji, opisywanej w zależności od przyjętego modelu obiektu poprzez: widmo układu (w przypadku opisu równaniem stanu) lub też zastępcze stałe czasowe i czasy opóźnień (przy opisie transmitancją). Układ kompensatora dynamicznego jest najczęściej uzupełnieniem regulatora konwencjonalnego (P, PI, PID). Ponadto należy tu dodać, że rozważane w pracy układy kompensacji dla potrzeb ich realizacji praktycznej należy opisać modelami dyskretnymi. Struktury algorytmów specjalnych mogą być bardzo różne, w zależności od konkretnej sytuacji. W dalszej części pracy rozważymy dwa podstawowe typy tych struktur: szeregową oraz ze sprzężeniem zwrotnym.

2.1. Struktura szeregowa

Schemat układu kompensacji dynamicznej o strukturze szeregowej pokazany jest poniżej:

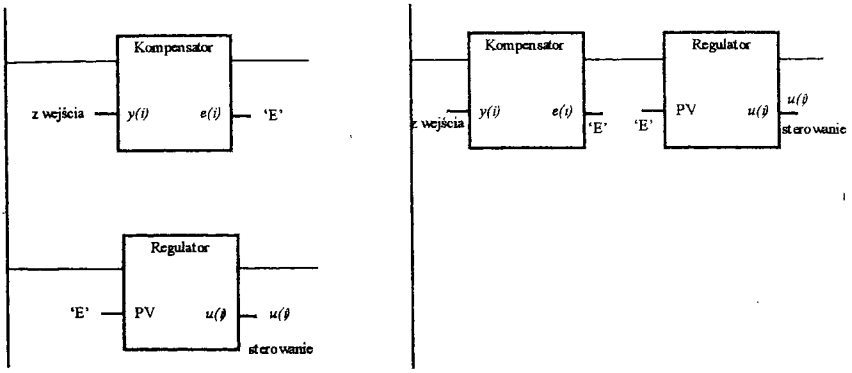


Rys. 1. Układ z szeregowym kompensatorem dynamicznym.

W układzie o strukturze szeregowej kompensator jest połączony szeregowo z regulatorem i sygnał sterujący jest wyznaczany przez regulator na podstawie sygnału wyjściowego z

kompensatora. Jest to sytuacja, z którą spotykamy się np. w układzie regulacji z obserwatorem. Szczególnym przypadkiem takiego układu jest układ kompensatora dynamicznego dla systemu parabolicznego (zob. [5]). Innym przypadkiem układu rozważanej klasy jest układ zmodyfikowanego regulatora redukcyjnego, opisany w pracy [7].

Dwa warianty realizacji drabinkowej struktury szeregowej pokazanej na rys. 1 są pokazane na schemacie 2. Jak widzimy, w przypadku takiej struktury nie ma praktycznie żadnych trudności z jej realizacją, gdyż elementy połączone szeregowo można umieścić w kolejnych szczeblach schematu drabinkowego lub w obrębie tego samego szczebla jeden za drugim. Ten drugi wariant ma tę cechę, że jeśli np. podczas wykonywania kompensatora wystąpi błąd, to wówczas algorytm regulatora również nie zostanie wykonany. W obu przypadkach wymiana danych pomiędzy blokami realizującymi obie części odbywa się z wykorzystaniem nazw symbolicznych lub zmiennych bezpośrednio reprezentowanych.

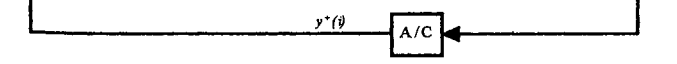


a b

Rys. 2 . Warianty realizacji regulatora specjalnego w wersji szeregowej.

2.2. Struktura ze sprzężeniem zwrotnym

W układzie ze sprzężeniem zwrotnym regulator konwencjonalny jest objęty dodatkową wewnętrzną pętlą sprzężenia zwrotnego. Klasycznym przykładem takiej struktury jest układ predyktora Smitha (zob. np. [2]).Schemat tej struktury jest pokazany poniżej:

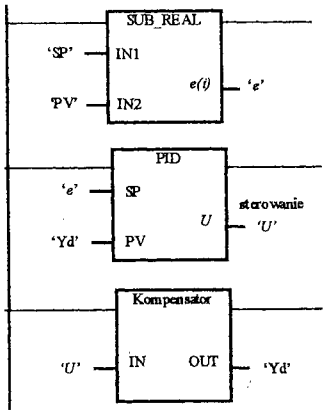


Rys. 3. Układ kompensacji dynamicznej ze sprzężeniem zwrotnym.

Poprawna realizacja algorytmu specjalnego ze sprzężeniem zwrotnym przy użyciu języka drabinkowego jest znacznie bardziej kłopotliwa, niż struktury szeregowej, omawianej powyżej. Aby pokazać problem, rozważmy najprostszą możliwą realizację struktury z rys. 3 w języku drabinkowym. Jest ona pokazana na rys. 4 . Interpretacja schematu z rys. 4. jest

następująca: Sygnał sterujący wyznaczony w bloku PID jest podawany na obiekt oraz jednocześnie (w tym samym cyklu programowym) na wejście bloku kompensatora. Jednak sygnał wyjściowy (będący sygnałem sprzężenia zwrotnego) z bloku kompensatora jest podawany na wejście regulatora dopiero w **następnym** cyklu. Ten efekt może być interpretowany jako obecność w torze sprzężenia zwrotnego dodatkowego elementu opóźniającego. Okazuje się, że opisany powyżej efekt jest nie do uniknięcia w języku drabinkowym (zob. [4], s. 160) i należy w związku tym zaproponować metody pozwalające na poprawną realizację zadanego algorytmu. W takiej sytuacji można zaproponować dwie metody postępowania:

1) Przekształcenie schematu blokowego algorytmu specjalnego do postaci bez sprzężenia zwrotnego jeszcze przed etapem realizacji na PLC. W wyniku takiego przekształcenia otrzymujemy algorytm równoważny zadanemu, lecz możliwy do poprawnej realizacji na PLC. Ten sposób postępowania został zastosowany podczas implementacji praktycznej regulatora redukcyjnego (zob.[7]).
2) Wykorzystanie dodatkowego opóźnienia wnoszonego przez sprzężenie zwrotne podczas realizacji algorytmu. Ten sposób został zastosowany podczas realizacji na PLC predyktora Smitha (zob.[6]). W tym przypadku opóźnienie modelowane przez blok opóźniający zostało celowo zmniejszone o 1 krok, a krok brakujący został „dodany" przez tor sprzężenia zwrotnego.



Rys. 4. Realizacja struktury ze sprzężeniem zwrotnym w języku drabinkowym.

Dyskretny algorytm specjalny musi być wykonywany z zadanym (określonym na etapie projektu) okresem próbkowania. W praktyce osiągnięcie tego założenia jest proste, ale uzależnione od sprzętu. Dla przykładu – w sterownikach SIEMENS najprościej jest umieścić cały algorytm specjalny w bloku organizacyjnym uruchamianym przerwaniem cyklicznym (np. w bloku OB 35), a w sterownikach GE FANUC – można wykorzystać bity systemowe T_SEC, T_1MS, T_100MS w połączeniu z prostym układem licznikowym.

4. PODSTAWOWE ELEMENTY REGULATORÓW SPECJALNYCH I ICH REALIZACJA

Układ realizujący algorytm specjalny rozważanej klasy zawiera dwie zasadnicze części:

- Regulator konwencjonalny (P, PI, PID)
- Kompensator dynamiczny

Obie w/w części regulatora specjalnego muszą być zrealizowane na sterowniku PLC. W przypadku regulatora konwencjonalnego, najwygodniej jest zastosować gotowy blok funkcyjny PID dostarczany przez producenta, natomiast kompensator dynamiczny, ze względu na swą nietypowość musi być w każdym przypadku zbudowany indywidualnie. Kompensator dynamiczny bazuje na modelu matematycznym obiektu i opisany jest równaniem stanu lub transmitancją. W związku z tym, podczas realizacji kompensatora niezbędne jest zamodelowanie w środowisku sterownika PLC pewnych podstawowych typów elementów dynamicznych. Elementy te podczas swego działania w środowisku PLC muszą mieć cechy bloków funkcyjnych (pewnie zbiór zmiennych musi być zapamiętany pomiędzy wywołaniami bloku w programie) Realizacja cyfrowa tych elementów bazuje na dyskretnym modelu kompensatora, który może mieć dwie następujące postaci: równanie stanu lub transmitancję.

4.1. Dyskretne równanie stanu

Dyskretne równanie stanu ma następującą postać ogólną:

$$\begin{cases} x^+(i+1) = A^+x^+(i) + B^+u^+(i) \\ y^+(i) = C^+x(i) + D^+u^+(i) \end{cases} \qquad (1)$$

W równaniu (1) $x^+(i)$, $u^+(i)$, $y^+(i)$ – oznacza dyskretne: stan, sterowanie i wyjście, A^+, B^+, C^+, D^+ - oznaczają macierze: stanu, sterowań, wyjść i bezpośrednich sterowań układu dyskretnego. Warto tu zwrócić uwagę, że równanie w postaci (1) jest zapisane w postaci, która może być bezpośrednio zrealizowana cyfrowo, należy też tu dodać, że realizacja tego typu jest w większości przypadków optymalna ze względu na poprawność numeryczną (zob. [1]). Realizacja cyfrowa (1) wymaga zapamiętania tylko aktualnej i poprzedniej wartości wektor stanu. Wektor stanu wyznaczony w danym kroku po wyznaczeniu wyjścia bloku jest zapamiętywany jako poprzedni.

4.2. Transmitancja dyskretna

Transmitancja dyskretna opisuje bezpośrednio związek pomiędzy wejściem i wyjściem bloku realizującego dany element dynamiczny. Może ona być zapisana następująco:

$$G(z^{-1}) = \frac{Y(z^{-1})}{U(z^{-1})} = \frac{a_n z^{-n} + \ldots + a_1 z^{-1} + a_0}{b_m z^{-m} + \ldots + b_1 z^{-1} + b_0} \qquad (2)$$

Przejście od zapisu (2) do realizacji cyfrowej będzie mieć postać następującą:

$$y(i) = -\frac{1}{b_0}\left(b_1 y(i-1) + \ldots + b_{m-1} y(i-m+1) + b_m y(i-m)\right) + \\ + \frac{1}{b_0}\left(a_0 u(i) + a_1 u(i-1) + \ldots + a_m u(i-n)\right) \qquad (3)$$

Realizacja cyfrowa opisana przez (3) jest prosta do implementacji praktycznej, gdyż opisuje bezpośredni związek między wejściem i wyjściem i nie musimy przechowywać w pamięci żadnych zmiennych wewnętrznych, jak w przypadku równania stanu. Do wyznaczania wyjścia elementu niezbędne jest zapamiętanie m poprzednich wartości wyjścia oraz n poprzednich wartości sterowania. Wartości te powinny być zapamiętane w strukturze danych działającej tak, jak stos. Wadą realizacji transmitancji opisanej przez (3) jest to, że przy

rzędzie transmitancji wyższym od 3 (m>3) nie jest ona optymalna z punktu widzenia poprawności numerycznej (zob. [1], s. s. 351 i dalej) Element opisany transmitancją wyższego rzędu lepiej jest w takiej sytuacji opisać w jakiejś innej równoważnej postaci. Przykładowa realizacja elementu inercyjnego I rzędu z wykorzystaniem języka IL dla sterowników GE FANUC jest pokazana na wydruku z rys.5 .

Szczególnym przypadkiem równania (3) jest równanie opisujące element opóźniający, mające postać (4):

$$y(i) = u(i-n) \qquad (4)$$

Przy jego realizacji praktycznej należy przyjąć założenie, że modelowany czas opóźnienia oraz okres próbkowania układu są wzajemnie współmierne, tj. czas opóźnienia może być wyrażony jako wielokrotność okresu próbkowania. Można to zapisać, że $\tau = nT_p$. Przy tym założeniu element opóźniający może być zrealizowany podobnie, jak inne elementy transmitancyjne, przy czym wielkość stosu do przechowywania poprzednich wartości musi być równa n.

	Label	Instruction	Operand	Ref. Address	Value	Comment
1		MUL_REAL(				
2		IN1 :=	u	%R00009		"wejście bloku
3		IN2 :=	0.02			"stała B
4		)				
5		ST_REAL	uw	%R00011		"zapamiętanie po wymnożeniu
6		MUL_REAL(				
7		IN1 :=	yp	%R00007		"poprzednia wartość wyjścia
8		IN2 :=	0.09			"stała A
9		)				
10		ST_REAL	ypw	%R00003		"zapamiętanie po wymnożeniu
11		ADD_REAL(				"sumowanie
12		IN1 :=	uw	%R00011		
13		IN2 :=	ypw	%R00003		
14		)				
15		ST_REAL	wyjscie	%R00005		"zapis wyniku na wyjście
16		MOVE_REAL(				"zapis yp do następnego kroku.
17		IN :=	wyjscie	%R00005		
18		Length :=	1			
19		Q =>	yp	%R00007		
20		)				
21						
22	<End Of					

Rys. 5. Realizacja elementu inercyjnego I rzędu w języku IL.

5. UWAGI KOŃCOWE

Jako podsumowanie niniejszej pracy oraz wcześniejszych prac autora z rozważanego zakresu [5], [6], [7], można sformułować następujące uwagi końcowe:

- Środowisko programowe typowego sterownika PLC pozwala na realizację algorytmów specjalnych rozważanej klasy, przy czym zdecydowanie prostszy do realizacji jest algorytm o strukturze szeregowej.
- Do realizacji całej struktury algorytmu najbardziej uzasadnione wydaje się zastosowanie jednego z języków graficznych, głównie ze względu na czytelność programu i prostotę jego interpretacji przez sterownik. W praktyce najczęściej jest to język LD, natomiast zalecane jest korzystanie z języka FBD, jeśli tylko pozwala na to konkretny sprzęt.

- Realizacja mniejszych elementów algorytmu specjalnego (tj. pojedynczych bloków) jest możliwa z wykorzystaniem każdego ze standardowych języków programowania PLC, natomiast w odczuciu autora najbardziej celowe jest wykorzystanie w tym wypadku języków tekstowych ze względu na optymalność oprogramowania (język IL) lub dużą wygodę (język strukturalny wysokiego poziomu).
- W przypadku, gdy proces podlegający sterowaniu z wykorzystaniem algorytmu specjalnego ma charakter wolnozmienny (tak jest w wypadku większości procesów przemysłowych), to zalecane jest stosowanie arytmetyki zmiennoprzecinkowej ze względu na większą dokładność obliczeń oraz mniejszy nakład pracy podczas programowania i większą czytelność programu.

Praca została zrealizowana w ramach grantu KBN nr 7 T11A 022 21.

LITERATURA

[1] Astrom K. J., Wittenmark B. „*Computer-Controlled Systems. Theory and Design*", Prentice Hall 1997.

[2] Górecki H. „*Analiza i synteza układów regulacji z opóźnieniem*", PWN W-a 1971.

[3] Legierski T. i inni „*Programowanie sterowników PLC*" Wyd. Prac. Komp. J. Skalmierski, Gliwice 1998.

[4] Lewis R. W. „*Programming industrial control systems using IEC 1131 – 3*" , IEE 1998.

[5] Mitkowski W. „*Stabilizacja liniowego systemu parabolicznego za pomocą dyskretnego kompensatora dynamicznego*" Kwart. AGH „Elektrotechnika" T 4, (1985) Z. 2 s 189-197

[5] Oprzędkiewicz K. „*Praktyczna realizacja kompensatora dynamicznego na sterowniku PLC*" Pomiary, Automatyka, Robotyka (PAR) , nr 12/2000, s. 5 – 10.

[6] Oprzędkiewicz K. „*Praktyczna realizacja predyktora Smitha na sterowniku PLC*" Automatyka, Tom 5, Zeszyt ½, 2001, s. 457 – 464.

[7] Oprzędkiewicz K. „*Dyskretny regulator redukcyjny i jego praktyczna implementacja*". Pomiary, Automatyka, Robotyka (PAR) , nr 11/2001, s. 8-15.