Porównanie języków programowania sterowników PLC pod kątem spełnienia wymagań czasu rzeczywistego

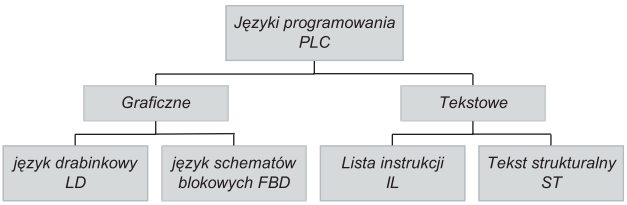
Krzysztof Oprzędkiewicz

Celem artykułu jest udzielenie odpowiedzi na pytanie: czy wybór języka programowania podczas realizacji zadania hard real time ma jakiekolwiek znaczenie, a jeśli tak, to który z tych języków należy wybrać. Jako miarę spełnienia wymagań czasu rzeczywistego przyjęto najdłuższy czas oraz rozrzut czasów wykonania określonej funkcji testowej.

Każdy cyfrowy system sterowania, a w szczególności PLC, jest systemem czasu rzeczywistego. Często stawia się mu wymagania *hard real time*, tzn. że podczas pracy systemu muszą być spełnione określone uwarunkowania czasowe dotyczące przede wszystkim determinizmu i powtarzalności czasów wykonania określonych części programu. Często wymagania *hard real time* muszą być spełnione podczas realizacji funkcji o znacznym stopniu złożoności obliczeniowej, których czas wykonania w środowisku PLC jest długi, a jednocześnie jest on parametrem krytycznym podczas pracy systemu sterowania. Z kolei sprzęt i oprogramowanie zapewniające spełnienie wymagań czasu rzeczywistego są znacznie droższe niż wersje podstawowe sprzętu i oprogramowania (np. cena oprogramowania soft PLC SIEMENS Win AC RTX zapewniającego spełnienie wymagań czasu rzeczywistego jest o ok. 50 % wyższa niż wersji podstawowej BASIS).

Języki programowania PLC

PN-EN 61131-3:2004 (U) *Sterowniki programowalne – Część 3. Języki programowania,* identyczna z normą europejską EN 61131-3:2003, definiuje 4 podstawowe języki programowania sterowników PLC, podzielone na 2 grupy [5, 6] (rys. 1).



Rys. 1. Klasyfikacja języków programowania sterowników PLC wg PN-EN 61131-3:2004 (U)

dr inż. Krzysztof Oprzędkiewicz – Katedra Automatyki, Akademia Górniczo-Hutnicza, Kraków

Każdy z języków programowania pokazanych na rys. 1 jest ukierunkowany na pewną, konkretną grupę zastosowań:

- schemat drabinkowy (LD) jest przeznaczony do realizacji systemów sterowania logicznego
- schemat bloków funkcji (FBD) jest przeznaczony do realizacji systemów regulacji ciągłej
- lista instrukcji (IL) – język typu asembler – jest przeznaczona do realizacji niewielkich, krytycznych pod względem jakości fragmentów programu; umożliwia pełny dostęp do wszystkich zasobów systemu operacyjnego sterownika
- tekst strukturalny (ST) – język wysokiego poziomu, bardzo podobny do pascala – jest przeznaczony do realizacji złożonych procedur obliczeniowych, jego stosowanie jest uzasadnione wszędzie tam, gdzie zachodzi konieczność stosowania pętli i rozbudowanych instrukcji wyboru.

Ponadto norma definiuje schemat *funkcji sekwencyjnej* (SFC) przeznaczony do programowania procesów sekwencyjnych, którego ogólna struktura bazuje na sieciach Petriego typu P/T. Żaden spośród omówionych języków programowania nie jest zorientowany na spełnienie wymagań czasu rzeczywistego.

Oszacowanie czasu wykonania procedur obliczeniowych

Zagadnienia oszacowania czasu cyklu, czasu odpowiedzi i czasu wykonania programu użytkownika były przedmiotem rozważań wcześniejszych prac autora, m.in: [7, 8]. W tym miejscu należy przypomnieć oszacowanie czasu wykonania procedury obliczeniowej w środowisku PLC:

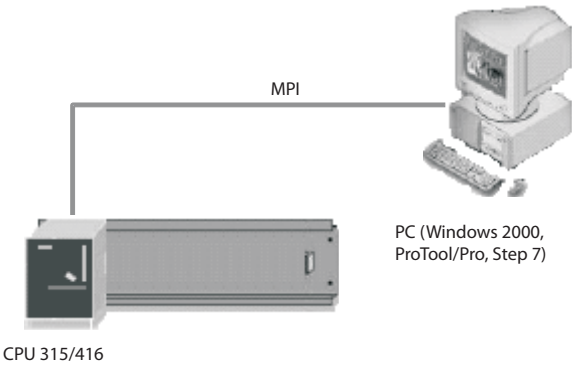
$$T_e \le F \sum T_{instr} \qquad (1)$$

gdzie: T_e oznacza szacowany czas wykonania procedury, T_{instr} – maksymalne czasy wykonania poszczególnych instrukcji, zależne od typu CPU i typu danych, na jakich są wykonywane działania, F >1 jest stałą zależną od typu użytej CPU.

Należy zaznaczyć, że relacja (1) jest tylko górnym oszacowaniem czasu wykonania procedury. W praktyce te czasy są krótsze, a ich wartość zmienia się w dość znacznym zakresie.

Badania eksperymentalne

Do badań doświadczalnych wykorzystano sprzęt i oprogramowanie firmy Siemens. Schemat układu doświadczalnego jest przedstawiony na rys. 2.



Rys. 2. Układ doświadczalny

Stanowisko składa się z dwu zasadniczych części:

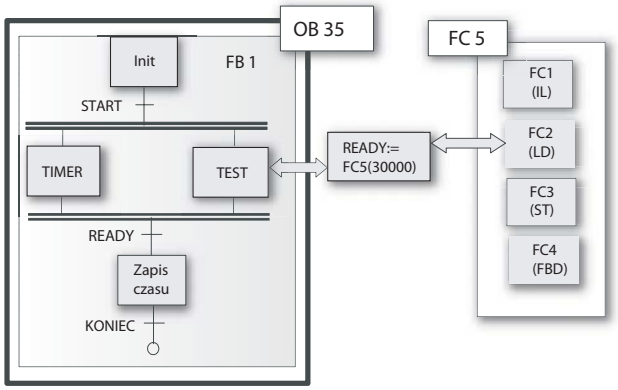
1. jednostka centralna Siemens: S-7 300 typ 315 lub S-7 400 typ 416
2. komputer PC z kartą komunikacyjną SIEMENS CP-5611 oraz oprogramowaniem konfiguracyjnym STEP-7 i systemem SCADA ProTool/Pro do zbierania wyników doświadczalnych.

Obie części są połączone z wykorzystaniem interfejsu MPI. Należy podkreślić, że do eksperymentu wykorzystano wyłącznie jednostkę centralną bez interfejsu procesowego, który był zbędny w rozważanej sytuacji.

Do badań czasu wykonania działania wykorzystano metodę omówioną w [8]. Metoda ta wymaga zastosowania narzędzia programowego schemat funkcji sekwencyjnej (SFC) i do pomiaru czasu wykonania procedury – timera uruchamianego współbieżnie z testowaną procedurą. Mierzony jest czas wykonania pętli zawierającej dużą liczbę wykonań następującego działania testowego:

$$Y:=ASIN\ (SIN\ (0{,}25*\pi) + 0{,}01) \qquad (2)$$

Działanie (2) jest zapisane w funkcji zbudowanej z wykorzystaniem jednego z języków programowania. Wzajemne powiązanie poszczególnych elementów programu testującego jest pokazane na rys. 3.



Rys. 3. Struktura programu testowego

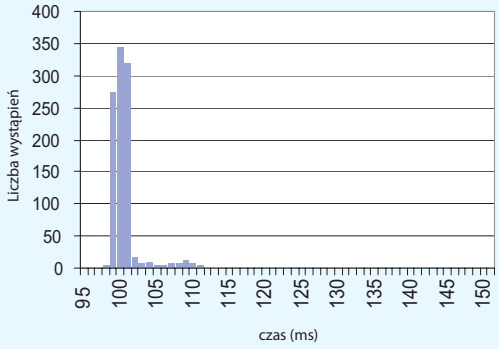
Działanie testowe (2) zostało wykonane na danych typu REAL. Zostało ono zapisane w funkcjach FC1 – FC4 typu VOID (niezwracających żadnego wyniku), przy czym każda z tych funkcji jest zbudowana z wykorzystaniem jednego z testowanych języków programowania PLC:

- FC1 – z użyciem listy instrukcji (IL)

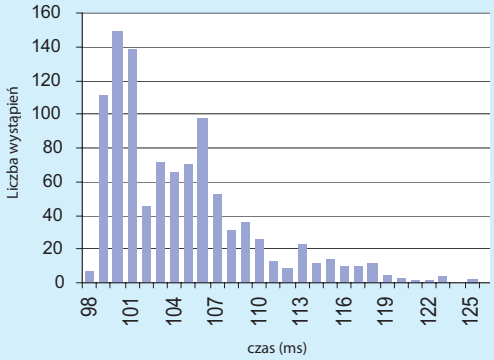
Tabela 1. Wartości czasów wykonania funkcji testowych

CPU 315					
Język	Czas minimalny (ms)	Czas maksymalny (ms)	Czas najbardziej prawdopodobny (ms)	Wartość średnia czasu (ms)	Odchylenie standardowe (ms)
Lista instrukcji (IL)	95	151	100	100,523	2,8428
Schemat drabinkowy (LD)	181	259	183	188,411	6,0402
Tekst strukturalny (ST)	86	166	100	100,094	2,3712
Schemat bloków funkcji (FBD)	189	428	191	198,426	17,9936
CPU 416					
Język	Czas minimalny (ms)	Czas maksymalny (ms)	Czas najbardziej prawdopodobny (ms)	Wartość średnia czasu (ms)	Odchylenie standardowe (ms)
Lista instrukcji IL	98	125	100	104,291	4,9031
Schemat drabinkowy (LD)	200	235	213	209,99	6,6207
Tekst strukturalny (ST)	95	129	100	102,888	50403
Schemat bloków funkcji (FBD)	206	248	211	216,576	7,2518

a) **CPU 315**

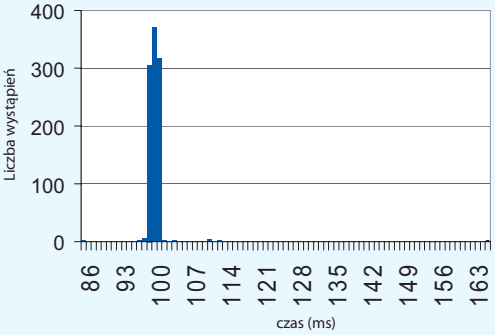


b) **CPU 416**

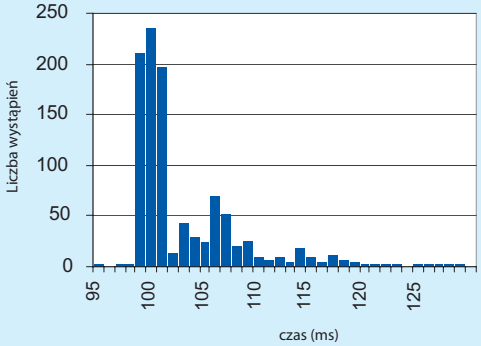


Rys. 3. Rozkład czasów wykonania testu dla listy instrukcji

a) **CPU 315**

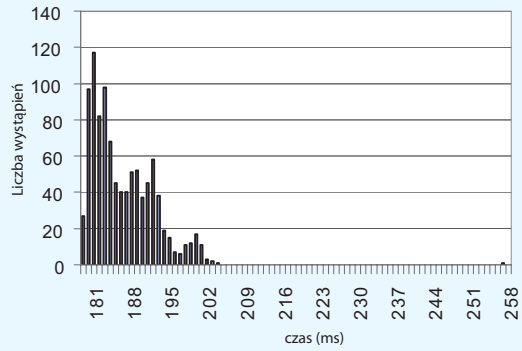


b) **CPU 416**

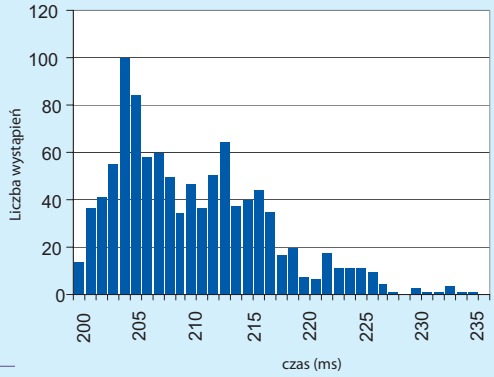


Rys. 4. Rozkłady czasów wykonania testu dla tekstu strukturalnego

a) **CPU 315**

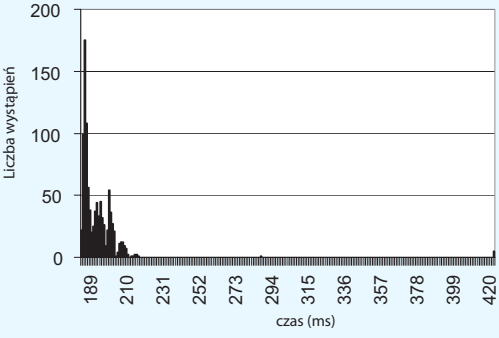


b) **CPU 416**

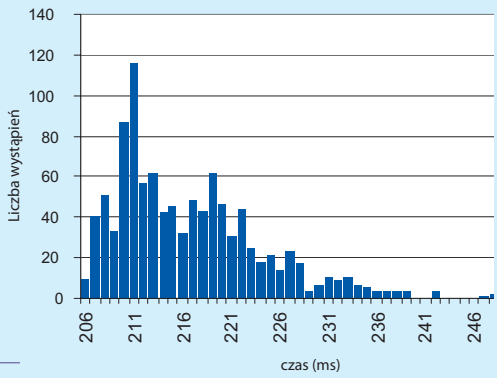


Rys. 5. Rozkłady czasów wykonania testu dla schematu drabinkowego

a) **CPU 315**



b) **CPU 416**



Rys. 6. Rozkłady czasów wykonania testu dla języka schematów blokowych

- FC2 – z użyciem schematu drabinkowego (LD)
- FC3 – z użyciem tekstu strukturalnego (ST) – języka wysokiego poziomu
- FC4 – z użyciem schematów bloków funkcji (FBD).

Każda z funkcji FC1 – FC4 została wykonana w pętli FOR DO, w funkcji FC5 typu BOOL, zwracającej wartość TRUE po zakończeniu wykonywania pętli. Ustawienie wyniku działania FC5 na TRUE sygnalizuje zakończenie wykonywania pętli z testowanym działaniem. Argumentem wejściowym funkcji FC5 jest liczba wykonań pętli z testowaną funkcją – w rozważanym przypadku przyjęto 30.000.

Funkcja FC5 jest wywoływana w akcji związanej z etapem TEST schematu funkcji sekwencyjnej, który jest uaktywniany równolegle z etapem TIMER uruchamiającym timer do pomiaru czasu wykonania. Ustawienie zmiennej wyjściowej READY na TRUE powoduje zakończenie współbieżnych etapów TIMER i TEST, przejście do następnego etapu i zapis bieżącej wartości czasu timera do komórki pamięci wewnętrznej, z której jest on następnie odczytywany przez aplikację SCADA i zapisywany w pliku MS EXCEL.

Wykonanie instalacji bloku funkcyjnego FB1 zawierającej schemat jest uaktywniane z poziomu bloku organizacyjnego OB35, uaktywnianego przerwaniem zegarowym z okresem 2 s.

W ramach badań doświadczalnych dla każdej badanej CPU i dla każdego z języków wykonano 1000 pomiarów czasu wykonania funkcji FC5. Wyniki eksperymentów są podane w tabeli 1 i na histogramach (rys. 3 – 6).

Po analizie tabeli 1 i histogramów 3 – 6 można sformułować następujące uwagi:

- Wartości minimalnych, maksymalnych, najbardziej prawdopodobnych oraz średnich czasów wykonania testowej funkcji są bardzo do siebie zbliżone dla obu języków testowych i obu testowanych CPU. Fakt ten można wytłumaczyć tym, że każdy element oprogramowania napisany w języku IL jest zawsze kompilowany do postaci ST i w tej postaci jest wykonywany przez CPU.
- Maksymalny, najbardziej prawdopodobny oraz średni czas wykonania funkcji testowej przy użyciu obu języków tekstowych jest znacznie krótszy niż przy użyciu języków graficznych. Również odchylenie standardowe (*jitter*) wartości czasu wykonania są mniejsze przy użyciu języka tekstowego, niż graficznych. Wydaje się, że zjawisko to można wytłumaczyć różnymi mechanizmami wykonywania instrukcji w obu grupach języków:
  - przy użyciu języków tekstowych operacje są wykonywane wyłącznie na danych zapisanych w akumulatorach jednostki centralnej, do których czas dostępu jest krótki
  - przy użyciu języków graficznych wykonanie złożonych operacji wymaga stosowania dodatkowych zmiennych do zapisu pośrednich wyników, ponieważ zmienne te są zapisywane w pamięci o dłuższym czasie dostępu. Jest to źródło dodatkowych losowych opóźnień w wykonywaniu programu
  - przy użyciu języków graficznych dodatkowym czynnikiem pogarszającym szybkość i determinizm czasowy działania jest sposób realizacji każdej operacji innej, niż logiczna: jedyną formą ich wykonania w językach graficznych jest wywołanie ich jako funkcji.
- Przy użyciu języków tekstowych oraz języka drabinkowego mniejszą wartość odchylenia standardowego czasu wykonania testu zapewnia zastosowanie jednostki centralnej serii 300, a przy użyciu języka FBD mniejszą wartość odchylenia standardowego zapewnia zastosowanie CPU serii 400.
- Mniejszy rozrzut minimalnej i maksymalnej wartości czasów wykonania testu jest zapewniony przy zastosowaniu CPU serii 400, natomiast należy zaznaczyć, że w dużym obszarze zastosowań praktycznych (systemy *firm real time* – np. systemy regulacji ciągłej procesów wolnozmiennych) pojedyncze długie czasy wykonania algorytmu nie wpływają na poprawność działania systemu sterowania, gdyż w takim wypadku znaczenie ma wartość średnia tego czasu, która powinna być krótsza od zadanego okresu próbkowania, który jest wartością stałą podczas realizacji algorytmu.

Uwagi końcowe

- Na podstawie analizy wyników badań doświadczalnych należy jednoznacznie stwierdzić, że dla rozważanego systemu PLC (SIEMENS) do realizacji zadań czasu rzeczywistego jest zalecane stosowanie języków tekstowych, przy czym nie ma znaczenia, czy jest to lista instrukcji (IL), czy język wysokiego poziomu – tekst strukturalny (ST), gdyż dla spełnienia wymagań czasu rzeczywistego oba te języki można uznać za równoważne. Z kolei z punktu widzenia użytkownika, lista instrukcji (IL) jest znacznie bardziej przyjazna, ale jeśli chodzi o możliwości funkcjonalne, to tylko tekst strukturalny (ST) zapewnia dostęp do wszystkich zasobów CPU.
- Gdy niezbędne jest użycie języka graficznego, wtedy zaleca się schemat drabinkowy (LD), gdyż charakteryzuje się on nieco lepszym determinizmem czasowym i krótszymi czasami wykonania funkcji niż schemat bloków funkcji (FBD).
- Gdy spełnienie wymagań czasu rzeczywistego jest kluczowym parametrem podczas projektowania układu sterowania i mamy do wyboru sterowniki serii S7 300 i S7 400, wtedy można zalecić wybór sprzętu serii S7 300, gdyż zapewnia on lepsze parametry w tej dziedzinie, przy znacznie niższej cenie.
- Zaproponowana w artykule metodyka postępowania może być przydatna podczas projektowania układu sterowania ze sterownikiem PLC przy doborze CPU sterownika pod kątem spełnienia wymagań czasu rzeczywistego.

Praca naukowa finansowana ze środków na naukę w latach 2006 – 2007 w ramach projektu badawczego nr 3 T11A 007 30.

Bibliografia

1. S. Bennet, *Real – Time Computer Control*, An Introduction Prentice Hall 1994.
2. H. Berger, *Automating with STEP7 in STL and SCL MCD*, Corporate Publishing 2001.
3. W. Grega, *Sterowanie cyfrowe w czasie rzeczywistym*, Wyd. Wydz. EAIiE AGH, 1999.
4. W. Grega, *Metody i algorytmy sterowania cyfrowego w układach scentralizowanych i rozproszonych* Wyd. AGH 2004.
5. T. Legierski, J. Wyrwał, J. Kasprzyk, J. Hajda, *Programowanie sterowników PLC*, Wyd. Prac. Komp. J. Skalmierskiego, Gliwice 1998.
6. R.W. Lewis, *Programming industrial control systems using IEC1131-3*, Revised Edition, IEE 1998.
7. K. Oprzędkiewicz, *Uwarunkowania czasowe realizacji specjalnych algorytmów sterowania w systemach PLC*, PAR 4/2003 s. 48 – 52.
8. K. Oprzędkiewicz, *Programowy pomiar czasu realizacji złożonych procedur obliczeniowych w środowisku PLC*, PAR 2/2006, s. 22 – 25.
9. SIEMENS SIMATIC S7-300, *Programmable Controller. Manual. Installation and Hardware*, SIEMENS AG 1998.
10. SIEMENS SIMATIC S7-300, *Programmable Controller. CPU-s Instruction List*, Siemens AG 1998.
11. SIEMENS SIMATIC S7-300, *Programmable Controller. Reference Manual. Module Specifications*, Siemens AG 1998. ■

FORESIGHT MAZOVIA opi

Szanowni Państwo,
przesyłamy drugi newsletter projektu Foresight Mazovia, adresowany do osób zainteresowanych zrównoważonym rozwojem województwa mazowieckiego i projektami typu foresight. Nowości Foresight Mazovia. Nabór ekspertów w projekcie Foresight Mazovia

Informujemy, że rozpoczął się nabór Ekspertów biorących udział w realizacji projektu Foresight Mazovia (Monitorowanie i prognozowanie (foresight) priorytetowych, innowacyjnych technologii dla zrównoważonego rozwoju województwa mazowieckiego).

Nabór ekspertów prowadzony będzie w 7 obszarach tematycznych: ■ poziom życia społeczeństwa ■ energia ■ ekologia ■ ochrona środowiska ■ zasoby naturalne i nowe materiały ■ infrastruktura ■ wzrost gospodarczy.

Do udziału w projekcie w charakterze Eksperta zapraszamy osoby związane z sektorem przedsiębiorstw, naukowców oraz osoby związane z działalnością administracyjną i społecznie użyteczną. Informacje odnośnie dokumentów, które powinni złożyć kandydaci na Ekspertów, można znaleźć na stronach: **http://www.piap.pl/aktualnosci/zamowienia_publiczne.php, http://www.formazovia.pl**

O projekcie: Projekt "Monitorowanie i prognozowanie (Foresight) priorytetowych, innowacyjnych technologii dla zrównoważonego rozwoju województwa mazowieckiego" jest realizowany przez Przemysłowy Instytut Automatyki i Pomiarów oraz Ośrodek Przetwarzania Informacji. Celem projektu jest identyfikacja wiodących technologii o znaczeniu strategicznym, których rozwijanie w następnych 20 latach będzie priorytetowe dla regionu województwa mazowieckiego.

Więcej informacji znajdą Państwo na stronie internetowej www.formazovia.pl.

Jeżeli nie chcą Państwo otrzymywać newslettera Foresight Mazovia, prosimy o poinformowanie nas o tym na adres: **biuro@formazovia.pl.**

Z poważaniem,
Zespół projektu Foresight Mazovia

Projekt współfinansowany przez UNIĘ EUROPEJSKĄ ze środków Europejskiego Funduszu Rozwoju Regionalnego

UNIA DLA PRZEDSIĘBIORCZYCH
PROGRAM KONKURENCYJNOŚĆ