Michał DOLIGALSKI, Marian ADAMSKI
UNIWERSYTET ZIELONOGÓRSKI, ul. Licealna 9, 65-417 Zielona Góra

Specyfikacja sterowników cyfrowych zorientowana na niezawodność

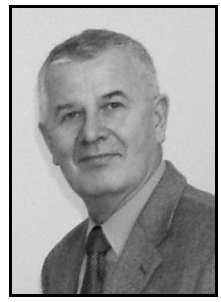
Mgr inż. Michał DOLIGALSKI

Absolwent Uniwersytetu Zielonogórskiego (2006). Ukończył studia o specjalności Inżynieria Komputerowa. Od roku 2006 pracuje jako asystent na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego. Zainteresowania badawcze obejmują metody projektowania i implementacji systemów cyfrowych. Członek Polskiego Towarzystwa Informatycznego.

e-mail: M.Doligalski@iie.uz.zgora.pl

Prof. dr hab. inż. Marian ADAMSKI

Dyrektor Instytutu Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Zainteresowania badawcze obejmują projektowanie mikrosterowników logicznych oraz formalne metodysyntezy, analizy i weryfikacji rekonfigurowalnych układów cyfrowych. Członek IEEE, IET, ACM, Polskiego Towarzystwa Elektrotechniki Teoretycznej i Stosowanej, Polskiego Towarzystwa Informatycznego oraz Brazylijskiego Towarzystwa Mikroelektronicznego.



e-mail: M.Adamski@iie.uz.zgora.pl

Streszczenie

W artykule przedstawiono wdrażaną metodę projektowania rekonfigurowalnych sterowników logicznych, ukierunkowaną na jakość behawioralnej specyfikacji, a tym samym niezawodność pracy. Zamierzone funkcjonowanie sterownika jest opisane z wykorzystaniem równocześnie dwóch dualnych języków graficznych: diagramu maszyny stanów UML oraz komplementarnej sieci Petriego. Synergia spowodowana dwoma wzajemnie się uzupełniającymi podejściami do behawioralnego opisu tego samego sterownika daje szansę na otrzymanie uwiarygodnionej specyfikacji już we wstępnej fazie projektowania.

Słowa kluczowe: Systemy wbudowane, Rekonfigurowalne Sterowniki Logiczne, Diagramy Maszyny Stanów UML, Sieci Petriego, diagramy SFC.

Quality oriented specification of logic controllers

Abstract

In the paper quality oriented approach to the design of digital embedded reconfigurable controllers is presented. The behaviour of a logic controller is described by means of dual related graphical languages: UML State machine diagram and Petri Net graph. The first one is well accepted among designers from the electronic industry, the second one among control engineers taking the advantage from similarities between Petri nets and Sequential Function Charts (SFC). The synergy of the view from two sides into the same project gives a chance to obtain validated specification at the design process beginning. It is shown in the second paragraph. Comparison of elementary models (Tab. 1) and design process with use of the dual specification (Fig. 1) are also presented. The third paragraph deals with mutual conversion of the elementary elements (Tab. 2) and shows their subsets in the form of class diagrams (Figs. 2 and 3). An example of the logic controller dual specification (Figs. 4 and 5) is given in the fourth paragraph. The practical use of dual specification is contingent upon implementation of tools for performing the conversion process in an automatic way.

Keywords: embedded systems, reconfigurable logic controller, UML state machine diagram, Petri net, sequential function chart, quality-driven design, formal verification.

1. Wstęp

W artykule przedstawiono dualne i komplementarne podejście do behawioralnej specyfikacji rekonfigurowalnych sterowników logicznych RLC (ang. Reconfigurable Logic Controller) [2], realizowanych układowo w systemach SoPC (ang. System on Programmable Chip) [1]. Pełne wykorzystanie bogatej funkcjonalności scalonych mikrosystemów cyfrowych wymaga wprowadzenia nowych, sprawniejszych, formalnych metod i technik projektowych, zwiększających niezawodność ich funkcjonowania [9]. Z drugiej strony znacznie wzrasta rola poglądowej oraz akceptowanej przez przemysł standardowej specyfikacji, która umożliwia dokładne i jednoznaczne odwzorowanie zarówno zamiarów projektanta, jak i wymagań użytkownika.

Uwiarygodnienie specyfikacji względem oczekiwań użytkownika wpływa zdecydowanie na zmniejszenie liczby błędów projektowych już na początkowym etapie specyfikacji funkcjonowania mikrosterownika. Unika się w ten sposób wielokrotnego, wtórnego poprawiania implementacji, przez co wyraźnemu skróceniu ulega czas potrzebny na wytworzenie sterownika i wprowadzenie nowego produktu na rynek. Nie bez znaczenia jest mniejsza pracochłonność progresywnie przeprowadzanego procesu syntezy, w naturalny sposób połączonego z weryfikacją formalną projektu. Mając do dyspozycji dwa równoważne behawioralnie modele, kontrolę poprawności projektu można przeprowadzać w sposób ciągły z użyciem narzędzi logicznych typu „model checker"[8].

Powszechnie stosowane w praktyce metody behawioralnej specyfikacji sterowników cyfrowych nie spełniają jednocześnie kryterium transparentności i ścisłości, stąd nie sprzyjają syntezie i weryfikacji formalnej, metodami rozwijanymi w instytucjach naukowych i przemysłowych.

Sposoby graficznej specyfikacji behawioralnej z wykorzystaniem zunifikowanego języka modelowania UML (Unified Modelling Language) takie jak: diagramy aktywności lub diagramy maszyny stanów pozwalają na przejrzyste modelowanie złożonych systemów sterowania. Umożliwiają one modelowanie procesów hierarchicznych i współbieżnych. Pozorną zaletą jest nadmiarowość elementów języka graficznego, która w połączeniu z intuicyjnym, ale nieumiejętnym i zbyt spontanicznym ich wykorzystaniem przez projektanta może doprowadzić do niejednoznaczności specyfikacji. Metody formalnej weryfikacji modeli nie są wystarczająco rozwinięte, zaakceptowane i rozpowszechnione w sposób umożliwiający ich wykorzystanie w realizacji rzeczywistych zadań projektowych [2].

Popularne w środowisku akademickim sieci Petriego są wsparte przez wypróbowane metody, algorytmy i narzędzia komputerowe do ich weryfikacji. Wymagają one od projektantów nie tylko intuicji ale również przygotowania matematycznego. Zawiłość i liczne warianty interpretowania sieci Petriego i ich pozorne ulepszenia, ukierunkowywane na rozwiązywanie konkretnych klas problemów, sprawiają że cieszą się one stosunkowo małą popularnością w środowisku inżynierskim. Dodatkowym powodem braku akceptacji jest językowa „wieża Babel" spowodowana nadmierną inwencja niektórych grup naukowców. Dla projektantów z przemysłu niezwykle istotna jest jednoznaczność, przejrzystość oraz spójność – rozumiana jako zgodność specyfikacji poszczególnych bloków systemu, przygotowywanych przez różne współpracujące ze sobą grupy projektowe. Próba standaryzacji sieci Petriego w formie normy światowej jeszcze się nie powiodła.

2. Synergia dualnej specyfikacji

Koncepcja diagramu maszyny stanów UML ma swoje oparcie w diagramach hierarchicznej maszyny stanów HCFSM (ang. Hierarchical Concurent Finite State Machine), będących rozwinięciem mapy stanów Harela (statecharts) [2, 5, 6, 7, 11]. Hierar-

chiczne sieci SFC (ang. Sequential Function Chart) wykorzystywane do programowania dla sterowników PLC mogą być uważane za dedykowaną formę interpretowanej sieci Petriego sterowania [1, 2]. Wartą rozpatrzenia propozycją w obszarze metod specyfikacji behawioralnej mikrosterowników logicznych jest ich dualna specyfikacja, oparta na połączeniu dwóch zaaprobowanych już modeli: diagramu maszyny stanów UML (SM) oraz hierarchicznej sieci Petriego (HPN). Model dualny SM-HPN powstaje przez zintegrowanie dwóch modeli elementarnych, które powinny być względem siebie jednoznacznie i łatwo przekształcalne. Modele cząstkowe są wzajemnie komplementarne, przez co niwelują one swoje wady i jednocześnie uwypuklają najbardziej istotne zalety. Każdy z modeli cząstkowych spełnia określoną rolę w procesie projektowania: diagram maszyny stanów UML pełni funkcję podstawowego, standardowego narzędzia specyfikacji dla współpracującej Se sobą grupy projektantów, zadaniem hierarchicznej sieci Petriego jest umożliwienie łatwiejszej weryfikacji z wykorzystaniem istniejących już metod i narzędzi. Nie bez znaczenia jest dostępność narzędzi komputerowych do syntezy logicznej układów cyfrowych opisanych sieciami Petriego [1, 3, 10, 11, 12].

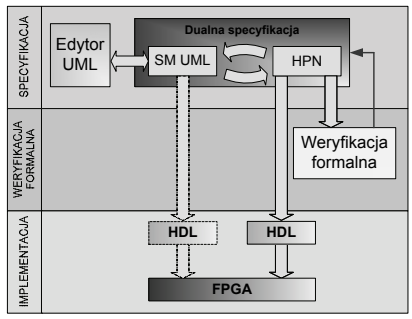
Podział ról między komplementarnymi modelami elementarnymi w ramach modelu dualnego wynika z ich predyspozycji przedstawionych w tabeli 1.

Tab. 1. Porównanie modeli elementarnych
Tab. 1. Comparison of elementary models

	Diagram maszyny stanów UML	Hierarchiczna sieć Petriego
Zalety	• Przejrzystość. • Akceptacja w środowisku inżynierskim. • Powszechność i akceptowalność standardu. • Uniwersalność i bogactwo języka. • Bezpośrednia obsługa wyjątków. • Wsparcie hierarchii i współbieżności.	• Oparcie semantyki na ugruntowanej teorii matematycznej. • Bogate wsparcie w postaci narzędzi do analizy formalnej. • Jednoznaczność specyfikacji. • Wsparcie hierarchii i współbieżności.
Wady	• Brak akceptowanych narzędzi do weryfikacji formalnej. • Duża dowolność w interpretacji języka utrudniająca implementację sprzętową.	• Mało popularna w środowisku inżynierskim. • Utrudniona obsługa wyjątków. • Różnorodne dialekty języka graficznego. • Mniej przydatna w projektach grupowych.

Diagram maszyny stanów ze względu na przejrzystą formę modularnej specyfikacji, powszechność standardu UML oraz jego uniwersalność jest doskonałym narzędziem do graficznego modelowania behawioralnego. Ma on bezpośrednio wbudowany mechanizm wywłaszczeń, który ułatwia obsługę sytuacji wyjątkowych. Hierarchiczne sieci Petriego nie mają podobnego mechanizmu, a realizacja wywłaszczeń w sytuacjach wyjątkowych na ogół polega na wprowadzeniu miejsc spoczynkowych oraz wielu dodatkowych tranzycji. Powoduje to znaczne zmniejszenie czytelności modelu. Niewątpliwą zaletą hierarchicznych sieci Petriego jest bogactwo różnorodnych metod formalnej weryfikacji, co jest niezwykle istotne w przypadku projektowania systemów o podwyższonej niezawodności (ang. dependable systems).

W literaturze przedmiotu najczęściej sugeruje się że wystarczy zapewnić możliwość przekształcenia diagramu maszyny stanów w hierarchiczną sieć Petriego [1, 4]. Pozwala to na przeprowadzanie jednostronnego procesu weryfikacji, modyfikacji i implementacji systemu, zrywając przejrzystą więź między specyfikacją i realizacją układu. Brak możliwości przekształcenia modelu opartego na hierarchicznej sieci Petriego w kierunku odwrotnym, utrudnia projektantom wykorzystywanie metod inżynierii wstecznej. Największą niedogodnością jest brak możliwości otrzymania informacji zwrotnej z procesu weryfikacji formalnej w postaci diagramu maszyny stanów UML. Dla szerokiego grona projektantów z przemysłu istotne jest wskazanie, bezpośrednio na diagramie maszyny stanów, elementów wymagających korekty lub modyfikacji. Dodatkowo możliwość przekształcania wzajemnego ułatwia import wcześniej wdrożonych projektów i ponowne ich wykorzystanie w nowych zadaniach.



Rys. 1. Projektowanie z użyciem dualnej specyfikacji
Fig. 1. Design with use of the dual specification

Zastosowanie dualnej specyfikacji pociąga za sobą konieczność stworzenia nowej metodyki projektowania sterowników logicznych oraz wspomagającego ją oprogramowania. Najbardziej oczywistym podejściem jest wykorzystanie modelu elementarnego sieci Petriego przede wszystkim na etapie weryfikacji formalnej a następnie implementacji w rekonfigurowanym układzie cyfrowym z wykorzystaniem języków opisu sprzętu. Model maszyny stanów może również zostać wykorzystany na etapie implementacji jako alternatywna forma odwzorowywania w strukturze mikrosterownika z wykorzystaniem nowopowstającego autorskiego systemu CAD (na rys. 1). Redundantna, niezależna implementacja sterownika na podstawie dwóch odrębnych specyfikacjikorzystna jest w systemach podwyższonego bezpieczeństwa [12].

3. Wzajemna konwersja diagramów maszyny stanów UML i sieci Petriego

Wzajemna konwersja diagramów sieci Petriego i maszyny stanów UML będzie znacznie łatwiejsza, jeśli ze zbiorów elementów graficznych obydwu modeli zostaną wybrane tylko elementy podstawowe, wystarczające do modelowania wymaganej klasy systemów sterowania dyskretnego. Konsekwencją takiego posunięcia jest konieczność dokładniejszego sprecyzowania zasad specyfikacji behawioralnej sterownika przy użyciu celowo ograniczonego podzbioru diagramów maszyny stanów UML. Pozwoli to w następnym kroku na opracowanie formalnych reguł konwersji dwukierunkowej. W literaturze przedmiotu przedstawiono już wiele propozycji odwzorowywania diagramów maszyny stanów na diagramy sieci Petriego. Proponowane elementarne sposoby transformacji zamieszczono w tab. 2.

Tab. 2. Wzajemna konwersja elementów podstawowych
Tab. 2. Mutual conversion of the elemetary elements

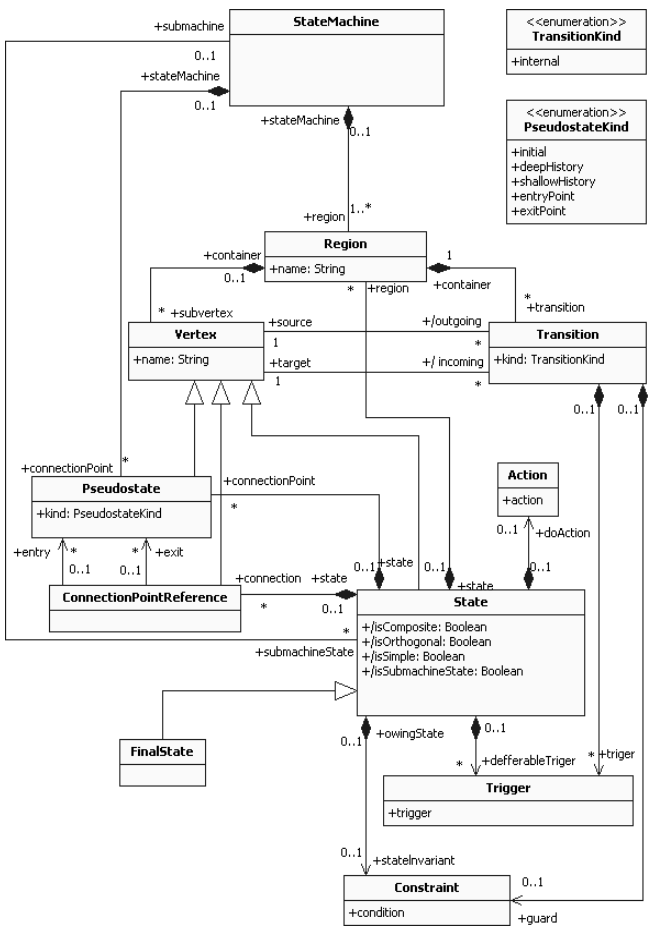
Maszyna stanów UML		Sieć Petriego	
Stan prosty		Miejsce	
Przejście z określonym wyzwalaczem		Tranzycja z określonym warunkiem realizacji	
Stan oznaczony stanem początkowym		Miejsce oznakowane znacznikiem	
Stan końcowy		Miejsce bez określonego przejścia wyjściowego	

Diagram maszyny stanów UML 2.2 ukierunkowany na modelowanie funkcjonalne w programowaniu obiektowym, zawiera wiele elementów nadmiarowych, bądź zupełnie nieprzydatnych w projektowaniu sterowników logicznych. Ich zdecydowane usunięcie pozwoliło na opracowanie elementarnego, zredukowanego do rozsądnego minimum diagramu maszyny stanów. Usunię-

to, między innymi punkt zniszczenia (ang. terminate node) w kontekście programowania obiektowego oznaczający zniszczenie obiektu opisywanego maszyną stanów. Zrezygnowano z przejść związanych z przekraczaniem granicy stanów (ang. local, external transitions). Celowo nie wykorzystano potencjalnie użytecznych punktów rozwidlenia i zespolenia, mimo że wyraźnie przypominają one tranzycje sieci Petriego. Przyczyną jest to, że wprowadzają one pewien nieład w diagramie maszyny stanów, w którym z założenia ważniejsza jest modułowość i hierarchia, niż formalne i poglądowe pokazanie współbieżności. Współbieżności, można z powodzeniem modelować za pomocą stanów złożonych. W takim przypadku na danym poziomie hierarchii aktywny będzie tylko jeden stan w maszynie lub podmaszynie stanów. Kolejnymi nadmiarowymi elementami są: punkt scalenia (ang. junction), oraz punkt selekcji (ang. choice). Oba pseudostany można wyeliminować, wykorzystując stan prosty i etykietowane przejścia z odpowiednio przypisanymi warunkami.

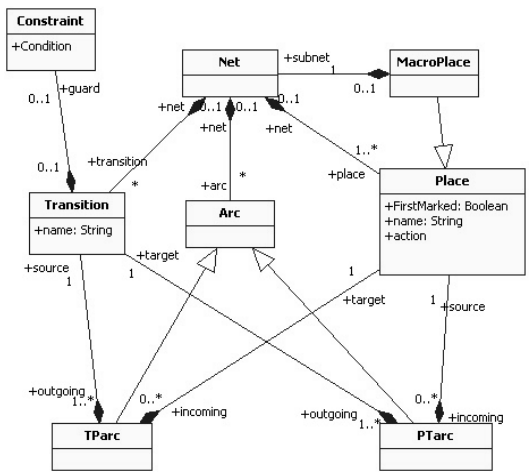
W celu dalszego uproszczenia modelu przyjęto, że do stanu może być przypisana jedynie akcja typu *do*, natomiast przejścia nie mogą mieć przypisanych akcji. Ograniczenie to predestynuje model dualny do realizacji hierarchicznych automatów Moore'a.

Struktura proponowanego modelu elementarnego maszyny stanów została przestawiona schematycznie na rysunku 2. Zaprezentowany diagram klas została oparty na pełnym diagramie zawartym w specyfikacji języka UML w wersji 2.2.



Rys. 2. Diagram klas elementarnego modelu maszyny stanów
Fig. 2. Class diagram of the elementary state machine model

Struktura komplementarnego modelu elementarnego hierarchicznej sieci Petriego została przedstawiona na rysunku 3. Z modelu zostały wyłączone takie elementy jak: łuki zezwalające i zabraniające, które łatwo jest zastąpić odpowiednimi etykietami stanów, umieszczonych przy tranzycjach. Ze względu na założoną implementację układową sterownika opartą na lokalnych współbieżnych stanach wewnętrznych i jednym stanie globalnym, w pełni odzwierciedlającym konfigurację systemu, należało zrezygnować z makrotranzycji, stosując wyłącznie makromiejsca. Aby konfiguracje wyznaczone na podstawie każdego z modeli elementarnych oddzielnie były w pełni izomorficzne, dopuszczono wprowadzanie do sieci Petriego dodatkowych miejsc konfiguracyjnych, precyzyjnie określających aktywność wszystkich zagnieżdżających się stanów złożonych.

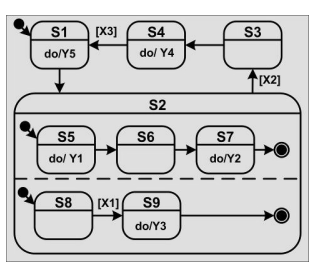


Rys. 3. Diagram klas elementarnego modelu hierarchicznej sieci Petriego
Fig. 3. Class diagram of the elementary Petri net model

Model elementarny maszyny stanów UML powstał poprzez usunięcie zbytecznych elementów ze standardowej wersji języka UML, przez co została zachowana pełna zgodność specyfikacji maszyny stanów z formą akceptowaną przez dostępne, przemysłowe lub uniwersyteckie narzędzia do modelowania lub syntezy układowej.

4. Przykładowa dualna specyfikacja sterownika

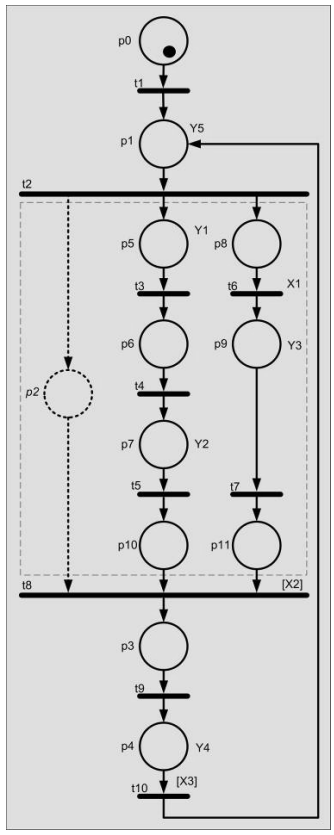
Na rys. 4 przedstawiono przykładowy, bardzo uproszczony diagram maszyny stanów UML. Na najwyższym poziomie hierarchii diagram jest zbudowany z czterech stanów S1,S2,S3,S4, Stan S2 jest stanem złożonym, z zaznaczonymi w nim dwoma obszarami współbieżnymi. Każdy z tych obszarów ma wyróżniony pseudostan początkowy i końcowy. Dwa współbieżne względem siebie podprocesy opisane są sekwencjami stanów prostych S5, S6, S7 oraz S8, S9. Z braku miejsca zrezygnowano z przedstawiania obsługi sytuacji wyjątkowych, w tym wywłaszczania i historii. Przedstawione przejścia pomiędzy stanami są albo bezwarunkowe, albo opatrzone warunkiem dozoru (ang. guard), w przeciwieństwie do wywłaszczeń w sytuacjach wyjątkowych, które winny być przedstawione jako wyzwalacze przejść (ang. trigger).



Rys. 4. Diagram maszyny stanów
Fig. 4. State machine diagram

Na rys. 5 zamieszczono sieć Petriego odpowiadającą wprost rozpatrywanej maszynie stanów UML. Można zauważyć, że liczba stanów prostych w tej sieci odpowiada liczbie wszystkich stanów prostych i złożonych w diagramie maszyny stanów. Do-

datkowe miejsce p2, w podsieci między tranzycjami t2 i t8 pełni rolę znacznika konfiguracji. Pozwala ono w łatwy sposób rozpoznawać i odwzorowywać układowo aktywność stanu złożonego S2. Realizacja przejścia między stanem złożonym S2 a stanem S3 będzie możliwa, gdy w obu podmaszynach stanu S2 osiągnięte zostaną stany końcowe a następnie zostanie spełniony warunek X2. Odpowiada to oznakowaniu miejsc p10 i p11 a następnie odpaleniu tranzycji t8.



Rys. 5. Nadrzędna sieć Petriego
Fig. 5. Top level Petri net

5. Wnioski

Dualna specyfikacja bazująca na dwóch komplementarnych modelach elementarnych: Maszyny stanów UML i hierarchicznej sieci Petriego może stać się akceptowalną i stosowaną praktyce formą specyfikacji sterowników cyfrowych. Wyniki analizy SWOT dualnej specyfikacji przestawiono w tabeli 3. Aby w pełni wykorzystać zalety dualnej specyfikacji konieczna jest implementacja narzędzia realizującego wzajemną konwersję w sposób automatyczny, jest to warunek konieczny szerszej akceptacji metody w środowisku przemysłowym.

W klasycznym podejściu projektowym, błędy powstałe na etapie specyfikacji propagowane były na kolejne etapy i mogły być wykryte i usunięte dopiero na etapie testowania.

Zastosowanie modelu dualnego zapobiega powstawaniu nieścisłości między oczekiwaniami w stosunku do systemu sterowania a specyfikacją. Pozwala również na przeprowadzenie analizę specyfikacji z wykorzystaniem metod formalnych.

Mniejsza liczba błędów specyfikacji przedkłada się bezpośrednio na zwiększenie niezawodności sterownika. Dodatkowym atutem jest zmniejszenie kosztów projektu – zarówno pod względem czasowym jak i finansowym. Oszczędności mogą być częściowo wykorzystane do przeprowadzenia kolejnych iteracji procesu projektowego co również wpływa na zwiększenie jakości.

Określenie stopnia niezawodności tak projektowanych sterowników w stosunku do metod tradycyjnych jest niezwykle trudne, gdyż proces specyfikacji jest w dużej mierze obarczony czynnikiem ludzkim. Nawet najdoskonalsza metoda specyfikacji wykorzystana w sposób nieumiejętny nie przyczyni się do zwiększenia niezawodności.

Tab. 3. Analiza SWOT dualnej specyfikacji
Tab. 3. SWOT analysis of the dual specification

	Pozytywne	Negatywne
Wewnętrzne	*Mocne strony* • Akceptowalność graficznej formy specyfikacji w środowisku inżynierskim. • Transparentność specyfikacji. • Uniwersalność modelu. • Powszechność standardu UML. • Wsparcie hierarchii i współbieżności. Obsługa wyjątków. • Istniejące już narzędzia do weryfikacji formalnej sieci Petriego oraz ich implementacji z wykorzystaniem języków HDL.	*Słabe strony* • Brak narzędzi do wzajemnej konwersji modeli podstawowych. • Subiektywne poczucie ograniczania swobody i inwencji projektanta. • Niechęć projektantów do formalnego dokumentowania swoich pomysłów, zwłaszcza na etapie specyfikacji. • Pozorne marnotrawstwo czasu, wymaganego do sporządzenia dwóch komplementarnych modeli formalnych.
Zewnętrzne	*Szanse* • Aprobowane wykorzystanie dualnej specyfikacji w projektowaniu rzeczywistych, złożonych systemów SoPC. • Rozwój systemów CAD wspierających dualną specyfikację.	*Zagrożenia* • Brak akceptacji w wyniku niezadowalających narzędzi do wzajemnej konwersji. • Problemy z zastosowaniem inżynierii wstecznej dla istniejących już projektów.

6. Literatura

[1] Adamski M., Karatkevich A., Węgrzyn M. (red): Design of embedded control systems, Springer, New York, 2005.
[2] Adamski M.: Logic synthesis of reconfigurable controllers," Industrial Embedded Systems, 2007. IEEE SIES '07, pp.373-376.
[3] Andreu D., Souquet G., Gil T.: Petri Net Based Rapid Prototy-ping of Digital Complex System, IEEE Computer Society Annual Symposium on VLSI, IEEE, 2008.
[4] Basile F., Chiacchio P., Del Grosso D.: Modelling automation systems by UML and Petri Nets, Proceedings of the 9th International Workshop on Discrete Event Systems, Gooteborg, IEEE, 2008.
[5] Bazydło G.: From UML state machine diagrams to FPGA, International Workshop Control and Information Technology - IWCIT 2007. Ostrava, Czechy, 2007 .- Ostrava, VSB - Technical University of Ostrava, 2007, 183-186.
[6] Doligalski M.: Specyfikacja programów sterowania oparta na hierarchicznym modelu maszyny stanów UML, IX International PHD Workshop - OWD 2007. Wisła, Polska, 2007, Warszawa, 2007. Conference Archives PTETiS, nr. 23, 299-304.
[7] Gajski D. D., Vahid F., Narayan S., Gong J.: Specification and Design of Embedded Systems, P T R Prentice Hall, New Jersey 1994.
[8] Grobelna I.: Formal verification of logic controller specification using NuSMV model checker, X International PHD Workshop - OWD 2008. Wisła, Polska, 2008, Conference Archives PTETiS , nr. 25, 459 – 464.
[9] Jóźwiak L.: Quality-driven System on Chip Design, Proceedings of the 1st International Symposium on Quality of Electronic Design, IEEE Computer Society Washington, USA, 2005, 93.
[10] Karatkevich A.: Dynamic Analysis of Petri Net-Based Discrete Systems, Springer, New York, 2007, LNCiS 353.
[11] Łabiak G., Wykorzystanie hierarchicznego modelu współbieżnego automatu w projektowaniu sterowników cyfrowych, Oficyna Wydawnicza Uniwersytetu Zielonogórskiego, Zielona Góra 2005.
[12] Węgrzyn M.: Implementation of Safety Critical Logic Controller by Means of FPGA. Annual Reviews in Control, 2003, Vol.27, pp.55-61.

otrzymano / received: 08.05.2010
przyjęto do druku / accepted: 04.06.2010 *artykuł recenzowany*