Michał DOLIGALSKI [1], **Przemysław MROWIEC** [2]
[1] UNIWERSYTET ZIELONOGÓRSKI,
[2] ITSERWIS SP. Z O.O.

System SMCAD wspomagający projektowanie częściowo rekonfigurowalnych sterowników logicznych

Mgr inż. Michał DOLIGALSKI

Absolwent Uniwersytetu Zielonogórskiego (2006). Ukończył studia o specjalności Inżynieria Komputerowa. Asystent na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego. Prace badawcze prowadzone w Instytucie Informatyki i Elektroniki skupiają się przede wszystkim na nowoczesnych metodach projektowania sterowników logicznych. Członek Polskiego Towarzystwa Informatycznego.

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

Przemysław MROWIEC

Student V roku Informatyki na Uniwersytecie Zielonogórskim, specjalizacja Inżyniera komputerowa. W pracy zawodowej pracy zajmuję się tworzeniem oprogramowania na platformę Lotus Notes oraz wirtualizacją aplikacji przy wykorzystaniu Citrix XenApp.

e-mail: Przemyslaw.Mrowiec@gmail.com

Streszczenie

Rozwój technologii półprzewodnikowych, dynamiczny wzrost pojemności i funkcjonalności układów FPGA stwarza nowe możliwości funkcjonalne, które z powodów ograniczeń współcześnie stosowanego oprogramowania projektowego nie mogą być w pełni wykorzystanie. W referacie opisano moduł graficznego edytora systemu SMCAD – wspomagającego projektowanie oraz przeprowadzenie procesu częściowej rekonfiguracji sterowników logicznych. W artykule opisano moduł graficzny systemu SMCAD, który umożliwia modelowanie behawioralne sterowników logicznych z wykorzystaniem diagramów maszyny stanów UML.

Słowa kluczowe: UML, maszyna stanów, rekonfigurowalny sterownik logicznych, SMCAD, częściowa rekonfiguracja.

SMCAD system supporting design of partial reconfigurable logic controllers

Abstract

Evolution of silicon technology, dynamic growth of FPGA device capacity and functionality requires introducing new techniques and developing new design tools. The application of a Petri net as a form of specification is a common solution used in the field of discrete systems. The application of the UML language, especially the state machine diagrams, is a perfect solution. These diagrams enable the hierarchical modelling of programs with concurrent elements. The UML language makes it possible to shorten the time of designing such a system and to optimise the use of hardware resources of the controller. There is no editor of the UML state machine diagrams dedicated to logic controller developing. In this paper the graphical editor module of an SMCAD system is presented. The SMCAD system is driven on reprogrammable logic controller partial reconfiguration. The new graphical editor enables behavioural modelling based on the UML state machine diagrams. Section 2 describes the graphical editor advantages compared to the existing, classical, software engineering driven UML tools like: Sybase Power Designer, ArgoUML, StarUML. It also gives the reasons for implementing the new editor. The supported subset of the UML state machine diagrams is presented in Section 3. In Section 4 there is shown an example of the manufacturing process outline (Fig. 2). The process of developing logic controller specification for the exemplary schema is also contained in this section. The behavioural specification in form of the UML state machine diagrams (Figs. 3 and 5) shows the partial reconfiguration process. The graphic specification was exported in the SCXML format (Fig. 8). Lack of possibility of *do* actions specification justifies the proposal of SCXML standard extension (Fig. 8).

Keywords: UML, state machine, reprogrammable logic controller.

1. Wstęp

Produkowane obecnie układy FPGA cechuje zarówno rozbudowana funkcjonalność – w postaci możliwości częściowej rekonfiguracji [4, 7, 9], wbudowanych mikrokontrolerów – jak i pojemności układowej (CLB). Niestety, rozwojowi technologii matrycowych układów reprogramowalnych nie towarzyszy rozwój narzędzi projektowych pozwalających w pełni wykorzystać możliwości sprzętowe. Powszechnie stosowanymi formami opisu funkcjonowania sterownika są sieci Petriego, maszyna stanów (FSM) oraz diagramy maszyny stanów UML [2, 3, 5, 6]. Należy tutaj zaznaczyć, iż badania w zakresie wykorzystania języka UML, w tym diagramów maszyny stanów UML, w procesie projektowania sterowników są w fazie początkowej. Projekt autorskiego systemu SMCAD zakłada wsparcie w zakresie projektowania sterownika logicznego, implementowanego w strukturach FPGA [1].

Projekt autorskiego systemu SMCAD obejmuje następujące moduły:

- Graficzny edytor diagramów maszyny stanów UML
- Menedżer konfiguracji specyfikacji behawioralnej
- Konwerter PN-SMUML (sieci Petriego i maszyny stanów UML)
- Generator konfiguracji modułowej
- Kreator konfiguracji różnicowej

Jednym z celów opracowania system SMCAD jest wsparcie procesu wytworzenia sterownika logicznego opartego na dualnej specyfikacji w postaci ekwiwalentnych modeli sieci Petriego i maszyny stanów UML. Ważnym priorytetem jest umożliwienie interakcji z istniejącym oprogramowaniem do specyfikacji, syntezy i implementacji poprzez zastosowanie standardu XML. W referacie przedstawiono zrealizowany (w wersji eksperymentalnej) moduł graficznego edytora maszyny stanów UML umożliwiający behawioralne modelowanie sterowników logicznych [8].

Celem opracowania nowego edytora graficznego (rys. 1) jest ułatwienie inżynierowi projektującemu system informatyczny wprowadzanie poprawek a także w celu późniejszego przeprowadzenia procesu częściowej rekonfiguracji. Przedstawiono tutaj scenariusz opisujący sytuację wykrycia i wprowadzenia poprawek w systemie. Analogicznie przebiega proces częściowej rekonfiguracji sterownika logicznego opisanego maszyną stanów – pewien wyodrębniony obszar ulega wymianie, pozostała część zostaje pozostawiona bez zmian.

2. Edytor graficzny maszyny stanów UML

Zarówno moduł graficznego edytora maszyn stanów UML jak i cały system SMCAD został zaimplementowany w języku JAVA. Wybór tej platformy nie był przypadkowy: kierowano się przede wszystkim możliwością powszechnego dostępu do systemu, a to właśnie zapewnia wieloplatformowość języka. Do uruchomienia systemu wystarczy jedynie system operacyjny z zainstalowaną wirtualną maszyną JAVY. Zastosowanie języka obiektowego silnie wspieranego przez nowoczesne techniki inżynierii oprogramowania skraca czas implementacji systemu jak również umożliwia w przyszłości prostą implementację kolejnych modułów. Przykładem może być moduł konwertera PN-SMUML umożliwiającego konwersję modeli sterownika opisanych siecią Petriego i maszyną stanów UML w ramach dualnej specyfikacji.

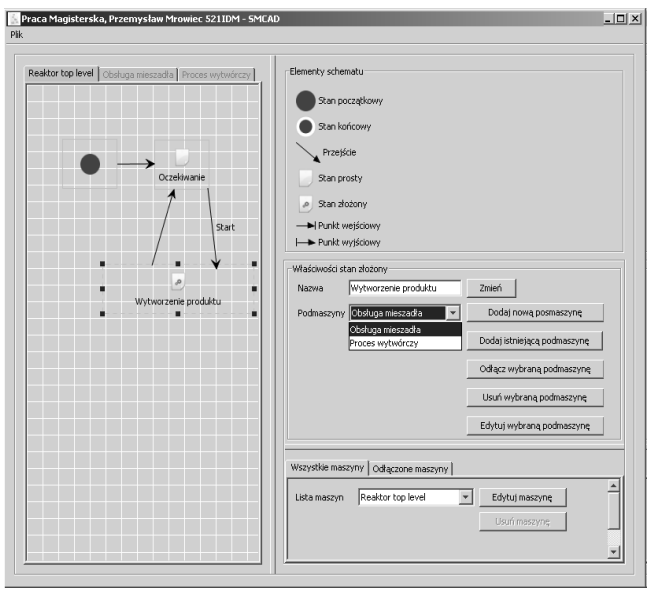
Dodatkowym atutem języka jest duża ilość gotowych szablonów oraz mechanizmów, dzięki którym czas potrzebny na implementację oprogramowania znacząco się skraca. Jednym z wymogów stawianych aplikacji była możliwość wymiany danych z wykorzystaniem standardu XML, język JAVA oferuje duże wsparcie standardu poprzez dostępność licznych parserów na licencji open source.

Na rynku dostępnych jest wiele narzędzi umożliwiających modelowanie z wykorzystaniem języka UML. Różnią się one pomiędzy sobą przede wszystkim zgodnością z obowiązującym standardem języka UML. Dostępne są zarówno narzędzia płatne (*Sybase Power Designer, Rational Rose*) jak i na licencji *open source (ArgoUML, StarUML)*. Różnią się one również funkcjonalnością – w szczególności jest to widoczne w grupie narzędzi *open source*. Narzędzia te zostały utworzone w celu wsparcia procesu inżynierii oprogramowani. Chociaż cel ten spełniają z mniejszym lub większym powodzeniem, to jednak zastosowanie wymienionych narzędzi w procesie projektowania sterowników cyfrowych jest utrudnione.

Przesłanką do podjęcia prac nad systemem SMCAD był między innymi brak odpowiedniego narzędzia do behawioralnej specyfikacji sterownika logicznego opartej na diagramach maszyny stanów UML.

Uniwersalność języka UML, będąca niewątpliwie jedną z jego największych zalet, wprowadza pewną nadmiarowość w elementach dostępnych dla projektanta. Pozwala to na lepsze specyfikowanie systemu, ukazywanie pewnych aspektów z różnych perspektyw jednak może powodować nieścisłości w interpretacji modelów.

Wiele elementów, wchodzących w skład diagramów maszyny stanów UML jest ściśle związana z programowaniem obiektowym. Przykładem może być punkt zniszczenia – w modelu określającym zachowanie oprogramowania odpowiada on sytuacji w której wywoływany jest destruktor – obiekt zostaje zniszczony. Punkt zniszczenia nie jest wykorzystywany w procesie projektowania sterownika logicznego, ponieważ w sprzętowej implementacji nie istnieje sytuacja którą mógłby opisywać. Istnieją również elementy nadmiarowe, jak np. punkt wyboru, które da się co prawda zinterpretować w kontekście sterowników logicznych jednak wprowadzają one pewną niepożądaną nadmiarowość – identyczną funkcjonalność da się przedstawić z wykorzystaniem mniejszego podzbioru elementów diagramu maszyny stanów UML. Dostępność elementów niepożądanych i nadmiarowych – jaką wprowadza istniejące oprogramowanie – może doprowadzić do sytuacji gdzie na podstawie specyfikacji nie da się przeprowadzić procesu implementacji układowej bądź będzie ona utrudniona, wprowadzone będą dwuznaczności w opisie, nie będzie on transparentny.



Rys. 1. Okno główne edytora graficznego
Fig. 1. The graphics editor main window

Z pośród dostępnych elementów diagramu maszyny stanów UML wybrano niezbędne i wystarczające do specyfikacji behawioralnej sterownika logicznego (rozdział 4) a następnie na tej podstawie przeprowadzono proces implementacji edytora graficznego. Moduł graficznego edytora jest również ukierunkowany na proces częściowej modułowej rekonfiguracji – specyfikacja jest w pełni hierarchiczna. Modułami niezmiennymi bądź podlegającymi rekonfiguracji są stany złożone w modelu na najwyższym poziomie hierarchii. W przypadku odwzorowania na sieć Petriego są to makromiejsca z sieci głównej.

Wyboru elementów maszyny stanów wspieranej przez system dokonano mając również na względzie możliwość bezpośredniego odwzorowania w hierarchicznej sieci Petriego.

3. Podzbiór elementów maszyny stanów UML

W obecnej wersji moduł edytora wspiera następujące elementy diagramów maszyny stanów:

- stany
  - proste
  - złożone (współbieżne, sekwencyjne)
- pseudo-stany
  - stan początkowy
  - stan końcowy
  - wznowienie (płytkie, głębokie)
- punk wejściowy
- punkt wyjściowy
- przejście

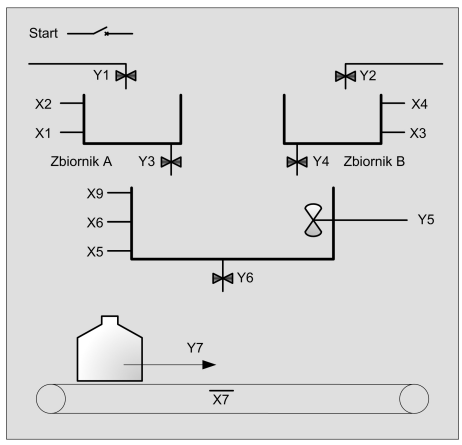
Stany proste zawierają w sobie akcje: entry, do oraz exit, do których to można przypisać akcję generowaną gdy dany stan jest aktywny. Stany złożone, składają się z co najmniej jednej podmaszyny. W ten sposób realizowana jest hierarchia, zastosowanie dwóch lub większej liczby podmaszyn przypisanych dla danego stanu złożonego odpowiada za reprezentację współbieżności. Przyjęto iż hierarchia i współbieżność będzie przedstawiana graficznie z wykorzystaniem notacji ukrytej. Kolejne podmaszyny mogą być narysowane na osobnych arkuszach (zakładkach), a następnie przypisane do danego stanu złożonego – ułatwi to w znacznej mierze realizację złożonych systemów sterowania binarnego. Można również tworzyć stany złożone, które w swojej budowie zawierać będą kilka pod-maszyn. Projektant może je modyfikować, odłączać lub przenosić do innego stanu złożonego. Rozwiązanie to pozwala tworzyć najbardziej optymalne pod względem odwzorowania potrzeb użytkownika systemy sterowania.

Stany początkowe i końcowe (zwane również pseudo stanami), określają, gdzie zaczyna się proces sterowania i w którym momencie się kończy. W obecnej wersji wymiana danych odbywa się zgodnie ze standardem XML a dokładniej SCXML (ang. State Chart XML) w przyszłości system będzie wspierał standard XMI (ang. XML Metadata Interchange) w celu zapewnienia wymiany diagramów pozostałymi edytorami. System zostanie rozbudowany o filtr elementów wejściowych diagramu – wymienione wcześniej niewspierane elementy diagramów maszyny stanów będą pomijane. Projektant zostanie o tym fakcie powiadomiony dzięki czemu będzie mógł wprowadzić poprawki konieczne dla zapewnienia pełnej funkcjonalności sterownika.

4. Przykładowy projekt systemu sterownia binarnego w systemie SMCAD

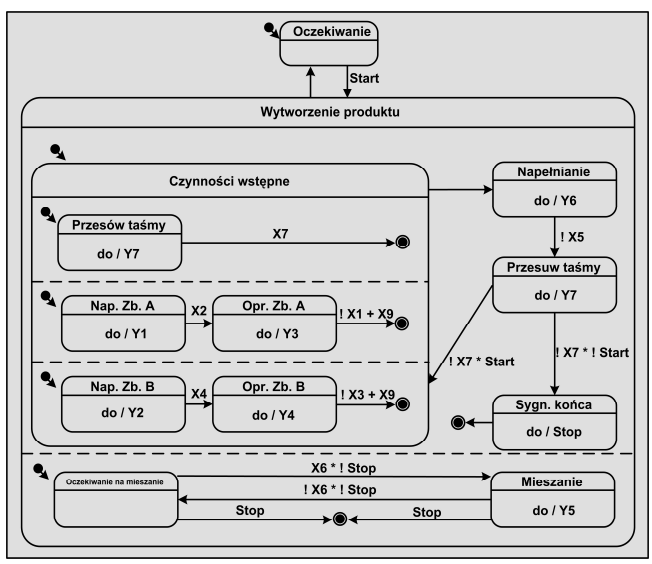
Zadaniem modelu jest przedstawienie wymagań funkcjonalnych sterownika istotnych na danym etapie projektowym. Pierwszym krokiem projektanta jest określenie ogólnego zarysu systemu – maszyny stanów na najwyższym poziomie w hierarchii. W kolejnych krokach następuje doprecyzowanie wymagań poprzez uzupełnienie podmaszyn na niższym poziomie hierarchii. W kolejnych cyklach projektowych uwzględnić dodatkowe wymagania funkcjonalne.

Działanie modułu edytora graficznego zostanie zaprezentowane na przykładzie modelu systemu sterowania binarnego dla procesu przemysłowego przedstawionego na rysunku 2. Proces przemysłowy polega na odmierzeniu w zbiornikach A i B pożądanej ilości substancji płynnej. Po otwarciu zaworów Y3 i Y4 zawartość zbiorników pomocniczych zostaje opróżniona do reaktora. Po osiągnięciu poziomu X6 przez substancje w reaktorze, następuje uruchomienie mieszadła, odpowiedzialnego za przyspieszenie procesu mieszania. Gotowy produkt jest konfekcjonowany do pojemników transportowanych na podajniku taśmowym Y7. Jeśli po zakończeniu pełnego cyklu produkcyjnego aktywny jest sygnał Start, następuje przygotowanie kolejnego produktu. Proces przemysłowy ma charakter poglądowy i pominięto w nim takie aspekty jak chociażby obsługa sytuacji wyjątkowych.



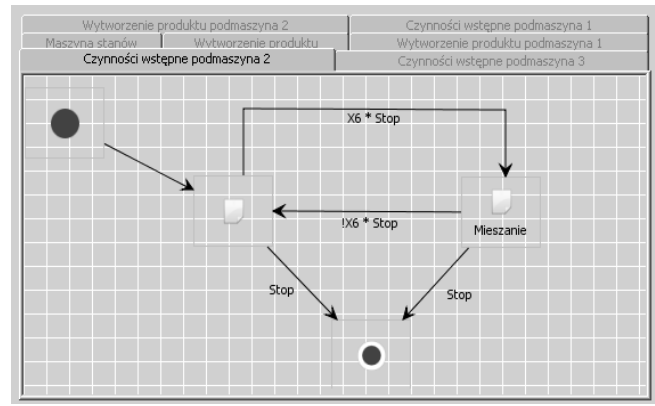
Rys. 2. Przykład procesu przemysłowego
Fig. 2. Example of manufacturing process

Dla modelowego procesu przemysłowego (linii produkcyjnej) przedstawionego na Rysunku został naszkicowany diagram maszyny stanów UML (rys. 3). Stany złożone zostały ukazane w notacji jawnej. Oznacza to iż poszczególne podmaszyny i obszary współbieżne rysowane są wewnątrz stanu złożonego.



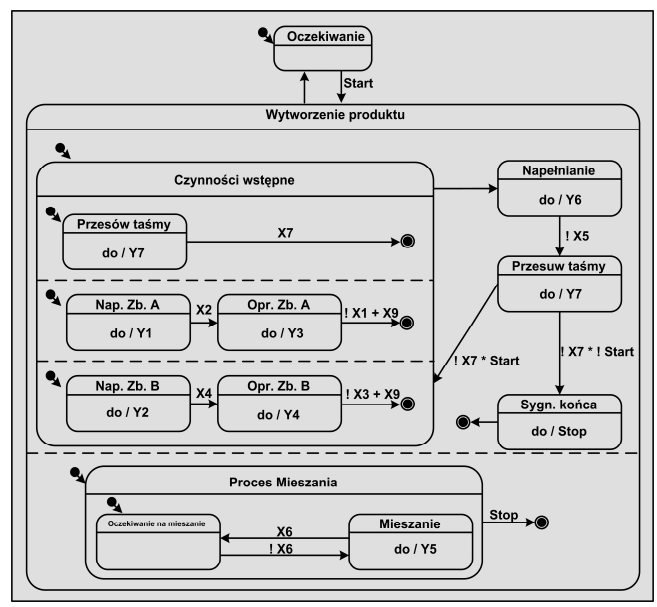
Rys. 3. Wstępny diagram maszyny stanów UML
Fig. 3. Initial state machine diagram

W celu przedstawienia możliwości modułu graficznego edytora SMCAD, przedstawiono wybrany element systemu sterowania (rys. 4) – podmaszynę odpowiedzialną za obsługę podprocesu mieszania substancji w reaktorze. Edytor graficzny umożliwia reprezentacje stanów złożonych jedynie w notacji ukrytej. Jest to zabieg celowy gdyż dla złożonych systemów prezentacja a następnie analiza diagramu w którym stany złożone przedstawiono w notacji jawnej była by znacznie utrudniona – w niektórych przypadkach wręcz niemożliwa. Edytor graficzny pozwala na hierarchiczne modelowanie z zastosowaniem notacji ukrytej – każda z podmaszyn przedstawiona jest na osobnym diagramie. Umieszczenie diagramów na osobnych zakładkach ułatwia znacząco proces projektowania i weryfikacji specyfikacji.



Rys. 4. Specyfikacja sterowania mieszadłem
Fig. 4. Mixer control specification

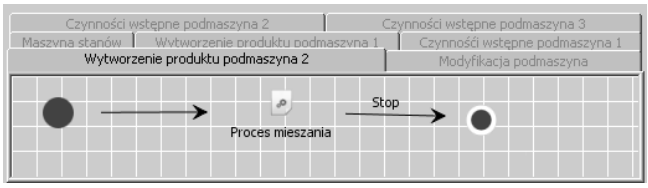
Aby zobrazować proces wprowadzania poprawek oraz proces częściowej rekonfiguracji wprowadzono modyfikację w podmaszynie stanu złożonego Wytworzenie produktu.



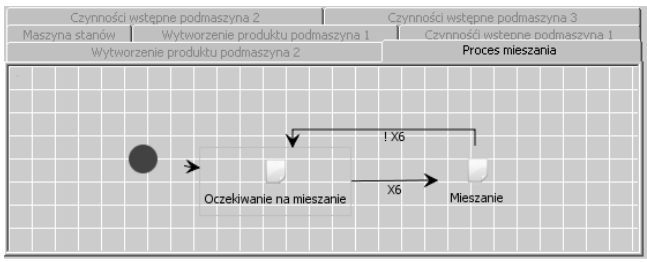
Rys. 5. Zmodyfikowany diagram maszyny stanów UML
Fig. 5. Modified UML state machine diagram

Poprawka ma na celu wprowadzenie dodatkowego poziomu hierarchii i wywłaszczenia powodującego opuszczenie stanu *Proces mieszania* w sytuacji, gdy proces wytworzenia produktu zostanie zakończony (rys. 5). Modyfikacja ma na celu wprowadzenie dodatkowego poziomu hierarchii, a tym samym uporządkowanie diagramu. Dwie tranzycje realizujące zakończenie pracy (rys. 3) zostały zastąpione jedną (rys. 5), która wywłaszcza podmaszynę w momencie pojawienia się sygnału *Stop*.

Poprawki odnoszą się jedynie do obszaru współbieżnego stanu wytworzenie produktu. W module edytora została utworzona dodatkowa podmaszyna (rys. 7). Podmaszyna stanu złożonego wytworzenie produktu po zmodyfikowaniu została załączona do stanu złożonego Proces mieszania (rys. 6).



Rys. 6. Specyfikacja procesu mieszania
Fig. 6. Mixing process specification



Rys. 7. Specyfikacja procesu mieszania – sterowanie mieszadłem
Fig. 7. Mixing process specification – mixer control

```
<?xml version="1.0"?>
<scxml initial="Oczekiwanie">
 <state id="Oczekiwanie">
  <transition event="Start" target="Wytworzenie produktu"/>
 </state>
 <state id="Wytworzenie produktu">
  <parallel id="WP_parallel">
   <state id="SM1" initial="Czynności wstępne">
     <state id="Czynności wstępne" src="Czynności
wstępne.scxml">
      <transition target="Napełnianie"/>
      </state>
      <state id="Napełnianie">
       <ondo>
       <send event="Y6"/>
       </ondo>
      <transition cond="! X5"  target="Przesuw taśmy"/>
      </state>
      <state id="Przesuw taśmy">
       <ondo>
       <send event="Y7"/>
       </ondo>
      <transition cond="! X7 & Start"  target="Czynności
wstępne"/>
      <transition cond="! X7 & ! Start"  target="Sygn.
Końca"/>
      </state>
      <state id="Sygn. Końca">
       <ondo>
       <send event="Stop" target="Self"/>
       </ondo>
      <transition target="SM1_Final"/>
      <final id="SM1_Final"/>
      </state>
   </state>
   <state id="SM2" initial="Proces Mieszania">
     <state id="Proces Mieszania" initial="Oczekiwanie na
mieszanie">
     <state id="Oczekiwanie na mieszanie">
       <transition cond="X6" target="Mieszanie"/>
      </state>
      <state id="Mieszanie">
       <ondo>
       <send event="Y5" target="Self"/>
       </ondo>
       <transition cond="X6" target="Oczekiwanie na
mieszanie"/>
      </state>
     <transition cond="Stop" target="SM2_Final"/>
     </state>
     <final id="SM2_Final"/>
   </state>
  </parallel>
  <transition target="Oczekiwanie"/>
 </state>
</scxml>
```

Rys. 8. Specyfikacja w formacie SCXML
Fig. 8. Specification in form of SCXML format

Poszczególne podmaszyny opisywane są w osobnych arkuszach roboczych (osobnych zakładkach). Podmaszyny identyfikowane są po unikalnych nazwach – umożliwia to wymianę podmaszyn i tworzenie różnych konfiguracji. Zadanie to zostanie w pełni osiągnięte w momencie rozszerzenie systemu SMCAD o moduł Menadżera konfiguracji specyfikacji.

Współpraca między systemem SMCAD a innym oprogramowaniem odbywa się za pomocą plików w formacie XML. Tyczy się to zarówno specyfikacji w postaci sieci Petriego (PNML – ang. *Petri Net Markup Language*) jak i diagramów maszyny stanów UML (XMI, SCXML). W odróżnieniu od konkurencyjnych systemów w których autorski format zapisu diagramów maszyny stanów (SSF – ang. *Statechart Specification Format*) uniemożliwia zastosowanie innych edytorów graficznych. Stan złożony *Czynności wstępne* został zdefiniowany w osobnym pliku SCXML (rys. 8).

Problematycznym okazało się generowanie tekstowego opisu maszyny stanów w postaci tekstowej w formacie SCXML. Standard nie pozwala na specyfikowanie akcji etykietą *do* – stosowaną powszechnie w procesie specyfikacji sterowników cyfrowych. Związane jest to z faktem iż w procesie inżynierii oprogramowanie przyjęto iż akcje typu *entry*, *do* oraz *exit* traktowane są jako niepodzielne (atomowe). Specyfikowane są za pomocą znaczników <*onentry*> oraz <*onexit*>. Działanie maszyny stanów zatrzymywane jest na czas realizacji poszczególnych zdarzeń. Proponuje się rozszerzenie standardu poprzez wprowadzenie znacznika <*ondo*> odpowiadającemu etykiecie *do* diagramów maszyny stanów.

5. Podsumowanie

Powszechnie stosowanym rozwiązaniem w procesie projektowania sterowników logicznych jest zastosowanie sieci Petriego. Posiadają one dobrze rozwinięty aparat matematyczny weryfikacji formalnej. Diagramy maszyny stanów UML są co prawda uboższe w narzędzia do weryfikacji formalnej, jednak są dobrze znane i akceptowane w środowisku inżynierskim. Ograniczenia w zastosowaniu praktycznym diagramów maszyny stanów wynikają z braku edytorów dedykowanych w procesie projektowania sterowników cyfrowych.

Artykuł prezentuje możliwość modelowania behawioralnego rekonfigurowalnych sterowników logicznych z wykorzystaniem diagramów maszyny stanów UML. W szczególności artykuł przedstawia moduł graficznego edytora maszyny stanów UML – wykorzystywany w procesie specyfikacji sterowników logicznych.

Celem dalszych badać jest opracowanie algorytmów wzajemnej konwersji pomiędzy siecią Petriego a maszyną Stanów UML. Konieczne jest również rozwinięcie systemu SMCAD o możliwość automatycznej wzajemnej konwersji modeli.

6. Literatura

[1] M. Adamski, A. Karatkevich, M. Węgrzyn (red): Design of embedded control systems, Springer, New York 2005.
[2] F. Basile, P. Chiacchio, D. Del Grosso: Modelling sutomation systems by UML and Petri Nets, Proceedings of the 9th International Workshop on Discrete Event Systems Gooteborg, IEEE, 2008.
[3] M. Doligalski, Specyfikacja programów sterowania oparta na hierarchicznym modelu maszyny stanów UML, Michał Doligalski, IX International PHD Workshop - OWD 2007. Wisła, Polska, 2007.- Warszawa, 2007. - Conference Archives PTETiS, Vol. 23, s. 299-304.
[4] M. Doligalski, M. Węgrzyn, Metody częściowej rekonfiguracja układów FPGA, Przegląd Telekomunikacyjny i Wiadomości Telokomunikacyjne .- 2008, nr 6, s. 773-775.
[5] D. D. Gajski, F. Vahid, S. Narayan, J. Gong, Specification and Design of Embedded Systems, P T R Prentice Hall, New Jersey 1994.
[6] G. Łabiak: Wykorzystanie hierarchicznego modelu współbieżnego automatu w projektowaniu sterowników cyfrowych, Oficyna Wydawnicza Uniwersytetu Zielonogórskiego, Zielona góra 2005.
[7] C. Kao: Benefits of Partial Reconfiguration, Xcell Journal, Xilinx Inc. 2005.
[8] P. Mrowiec: Projekt i implementacja aplikacji do modelowania systemów sterowania binarnego (w opracowaniu).
[9] M. Węgrzyn: Częściowa rekonfiguracja sterowników binarnych opisanych sieciami Petriego, Pomiary Automatyka Kontrola, 2006, nr 6, s 26-28.

Artykul recenzowany