Małgorzata KOŁOPIEŃCZYK
UNIWERSYTET ZIELONOGÓRSKI

System wspomagający projektowanie mikroprogramowanych układów sterujących

Dr inż. Małgorzata KOŁOPIEŃCZYK

Absolwentka Wydziału Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego (1999 r.). Obecnie adiunkt w Instytucie Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Zajmuje się następującymi zagadnieniami: projektowaniem i syntezą sterowników logicznych oraz inżynierią oprogramowania.

e-mail: m.kolopienczyk@iie.uz.zgora.pl.

Streszczenie

W artykule zaprezentowano system wspomagający proces projektowania mikroprogramowanych układów sterujących (ang. Compositional Microprogram Control Unit). Głównym zadaniem systemu jest usprawnienie procesu opisu algorytmu sterowania mikroprogramowanych układów sterujących za pomocą sieci działań jak również translacja graficznej reprezentacji sieci działań do specyfikacji tekstowej. Aplikacja umożliwiała tworzenie sieci działań a następnie zapisanie utworzonego modelu do jednego z dostępnych w systemie formatów: pliku tekstowego o zadanej strukturze, dokumentu XML, pliku graficznego JPEG. W artykule omówiono główne funkcje programu jak również struktury danych umożliwiające tworzenie, przechowywanie oraz prezentację sieci działań.

Słowa kluczowe: sieć działań, mikroprogramowany układ sterujący, reprezentacja tekstowa i graficzna sieci działań.

System aiding design of compositional microprogram control unit

Abstract

This paper presents a system which supports the design process of compositional microprogram control units (3, 6, 8). The control algorithm of the compositional control unit is very often described by flow-charts. The proposed approach improves the design process of a microprogram control unit algorithm defined by flow-charts. It allows translating the graphical flow-charts representation into a textual flow-chart specification form as well [3, 5]. The software FCADiagramsEditor has been implemented using Java language and is independent of the operating system and processor. FCADiagramsEditor allows saving the flow-chart in XML structure (Fig. 4) and in JPEG file (Fig. 2). In this paper the data structures for creation, storage and presentation of flow-charts are also proposed. The paper is divided into four parts. The first section is a brief introduction to the issues of compositional microprogram control unit design [1, 7]. In the second section the graphical and textual flow-chart specification is presented .The structure of the data formats and programmable environment are described in the third section, while the last – fourth – section contains a summary.

Keywords: flow-chart, compositional microprogram control unit, textual and graphical flow-charts representation.

1. Wstęp

W ostatnich latach, mikroprogramowane układy sterujące znalazły zastosowanie niemalże w każdej dziedzinie naszego życia. Możemy je znaleźć zarówno w fabryce samochodów gdzie sterują skomplikowanymi procesami technologicznym, jak również w sprzęcie RTV/AGD czy w zabawkach dla dzieci.

Za tak powszechne wykorzystanie układów cyfrowych odpowiedzialne jest między innymi, coraz doskonalsze, „przyjazne" dla użytkownika oprogramowanie, względnie niska cena oraz łatwość ich programowania.

Jednym ze sposobów opisywania algorytmu sterowania układów mikroprogramowanych jest wykorzystanie sieci działań (ang. flow-chart). Sieć działań reprezentuje, przez figury geometryczne połączone strzałkami, kolejne czynności w projektowanym algorytmie. Powstało wiele aplikacji, które w sposób mniej lub bardziej skomplikowany pozwalają opisywać za pomocą sieci działań algorytm sterowania układów cyfrowych. Można tutaj wymienić choćby takie programy jak ISaGRAF firmy ICS Triplex, który umożliwia nie tylko tworzenie algorytmu sterowania opisanego siecią działań, ale również jego implementację na wybranym sterowniku, czy choćby Visio firmy Microsoft za pomocą którego użytkownik tworzy jedynie graficzną reprezentację algorytmu.

W chwili obecnej na Uniwersytecie Zielonogórskim prowadzonych jest szereg badań dotyczących implementacji sterowników logicznych w strukturach programowalnych.

Istnieje wiele formalnych metod opisu algorytmów sterowania. Przedmiotem prezentowanych badań jest wykorzystanie sieci działań w postaci tekstowej oraz graficznej [3, 6].

Forma tekstowa sieci działań ma charakter uniwersalny i pozwala na łatwe przekształcanie opisu algorytmu, np. do opisu w VHLD. Komercyjne systemy nie mają możliwości przekonwertowania sieci działań do postaci tekstowej. Z tego powodu na potrzeby badań został zaprojektowany i zaimplementowany system FCADiagramsEditor, umożliwiający tworzenie algorytmu sterowania układów cyfrowych z wykorzystaniem sieci działań oraz zapisanie jej w formie tekstowej.

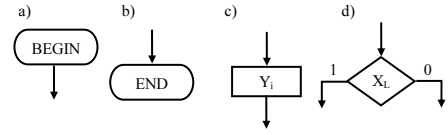
Opracowane w systemie struktury danych umożliwiają tworzenie, przechowywanie oraz edycję graficznej postaci sieci działań. System współpracuje z translatorem formy graficznej na formę tabelaryczną i specjalizowany język tekstowy umożliwiający współdziałanie z wcześniej opracowanymi podsystemami [1]. Nowym elementem jest translacja na format KISS2 [9] oraz ustalony szablon w języku VHDL. Symulacja behawioralna mikrosterownika może być przeprowadzona w komfortowym środowisku programistycznym np. firmy Xilinx.

Aplikacja wspomagająca projektowanie mikroprogramowanych układów sterujących została zaimplementowana i nadal jest rozwijana w środowisku Java. Jest ona łatwo przenaszalna na inne systemy operacyjne wspierające Javę, takie jak Windows, SUN, Linux czy BSD. Powstały kod jest niezależny od systemu operacyjnego i procesora.

Nowy system jest ukierunkowany w stronę implementacji sprzętowych w reprogramowalnych (rekonfigurowalnych) układach matrycowych za pośrednictwem języków HDL takich jak VHDL i Verilog.

2. Sieć działań

Sieć działań jest spójnym i zorientowanym grafem [4, 7, 8], który zawiera cztery rodzaje składników: blok początkowy, blok końcowy oraz bloki operacyjne i warunkowe (rys. 1).



Rys. 1. Elementy sieci działań
Fig. 1. Components of the flow-chart

W sieci działań pierwszym elementem algorytmu jest blok początkowy, natomiast blok końcowy jest jego ostatnim elementem. Bloki operacyjne posiadają tylko jedno wejście i jedno wyjście i opisują stan wyjść układu sterującego. Bloki warunkowe posiadają jedno wejście i dwa wyjścia, którym przyporządkowano odpowiednio stan „1" lub „0". Symbole sygnałów wyjściowych Y_i

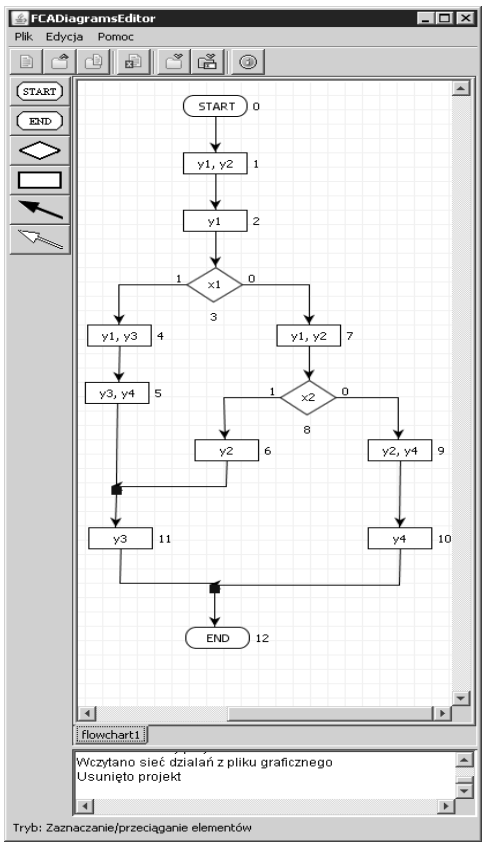
w blokach operacyjnych nazywane są mikroinstrukcjami, a symbol X_L w blokach warunkowych jest sprawdzeniem aktualnego warunku [2].

Reprezentacja graficzna sieci działań jest schematem blokowym składającym się z wymienionych wyżej elementów.

Reprezentacja tekstowa sieci działań musi zawierać informację o strukturze sieci, jak również o wszystkich sygnałach wejściowych i wyjściowych.

Wykorzystany w omawianym systemie tekstowy format sieci działań spełnia wymienione wyżej wymagania. Szczegółowy opis formatu można znaleźć w pracach [2, 5, 6],

Na rysunku 2 zaprezentowano przykładową, graficzną reprezentację sieci działań utworzoną w programie FCADiagramsEditor.



Rys. 2. Sieć działań: reprezentacja graficzna
Fig. 2. Flow-chart: graphical representation

Rysunek 3 przedstawia opis tekstowy sieci działań z rysunku 2.

```
0S: 1.
1O: Y1, 2.
2O: Y2, 3.
3X: x1, 4, 7.
4O: Y3, 5.
5O: Y4, 6.
6O: Y5, 12.
7O: Y1, 8.
8X: x2, 11, 9.
9O: Y6, 10.
10O: Y7, 12.
11O: Y8, 6.
12E.

Y1: y1, y2.
Y2: y1.
Y3: y1, y3.
Y4: y3, y4.
Y5: y3.
Y6: y2, y4.
Y7: y4.
Y8: y2.
```

Rys. 3. Sieć działań: reprezentacja tekstowa
Fig. 3. Flow-chart: textual representation

3. Aplikacja FCADiagramsEditor

Program FCADiagramsEditor został napisany w języku Java. Język ten jest jednym z nowszych języków programowania, który dzięki swoim licznym zaletom ciągle zyskuje nowych użytkowników. Programy napisane przy użyciu tego języka są również łatwo przenaszalne na inne systemy operacyjne wspierające Javę.

System wspomagający projektowanie mikroprogramowanych układów sterujących obsługuje dwa formaty danych wejściowych. Jednym z nich jest format XML, który umożliwia odczyt i zapis plików zawierających sieci działań.

Na rysunku 4 pokazano fragment sieci działań z rysunku 2 zapisany w postaci dokumentu XML.

```
- <FCADiagram>
- <start id="0">
  <position x="300" y="20" />
  <nextComponent id="1" />
      </start>
- <process id="1">
  <position x="300" y="80" />
  <microOperation>y1, y2</microOperation>
  <nextComponent id="2" />
  <presentConditional id="-1" />
      </process>
- <process id="2">
  <position x="300" y="140" />
  <microOperation>y1</microOperation>
  <nextComponent id="3" />
  <presentConditional id="-1" />
      </process>
- <conditional id="3">
  <position x="303" y="200" />
  <name>x1</name>
  <trueComponent id="4" />
  <falseComponent id="7" />
  <presentConditional id="3" />
      </conditional>
- <process id="4">
  <position x="217" y="260" />
  <microOperation>y1, y3</microOperation>
  <nextComponent id="5" />
  <presentConditional id="-1" />
      </process>
- <process id="5">
  <position x="215" y="320" />
  <microOperation>y3, y4</microOperation>
  <nextComponent id="11" />
  <presentConditional id="0" />
      </process>
- <process id="6">
  <position x="310" y="380" />
  <microOperation>y2</microOperation>
  <nextComponent id="11" />
  <presentConditional id="0" />
      </process>
- <process id="7">
  <position x="381" y="260" />
  <microOperation>y1, y2</microOperation>
  <nextComponent id="8" />
  <presentConditional id="-1" />
      </process>
```

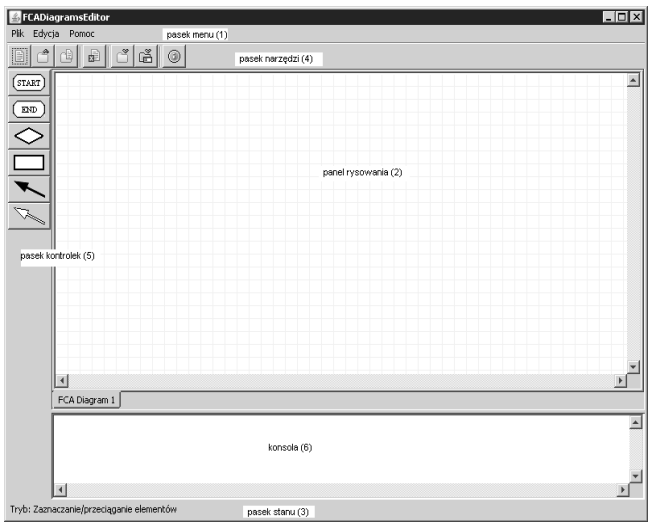
Rys. 4. Sieć działań: reprezentacja w postaci dokumentu XML
Fig. 4. Flow-chart: XML representation

Działanie i budowę aplikacji oparto na strukturze pliku tekstowego. Ponieważ rozwiązanie takie nie oferuje możliwości zapisu oraz odczytu pozycji poszczególnych elementów sieci działań, to w celu rozwiązania tego problemu, zdecydowano się na adaptację, na potrzeby tworzonej aplikacji, algorytmu automatycznego rozmieszczania elementów sieci działań. Algorytm ten stara się rozmieścić powiązane ze sobą elementy blisko siebie w danym obszarze roboczym. Umożliwiło to wizualizację modeli sieci działań zapisanych jedynie w pliku tekstowym.

Głównym zadaniem aplikacji jest możliwość łatwego tworzenia sieci działań opisującej algorytm sterowania mikroprogramowanego układu sterującego, a następnie zapisanie utworzonego modelu do:

- pliku tekstowego,
- dokumentu XML,
- pliku graficznego JPEG.

W aplikacji przewidziano również możliwość przekonwertowania sieci działań zapisanej w specyfikacji tekstowej do formatu graficznego. Na rysunku 4 zaprezentowano graficzny interfejs użytkownika omawianego programu.

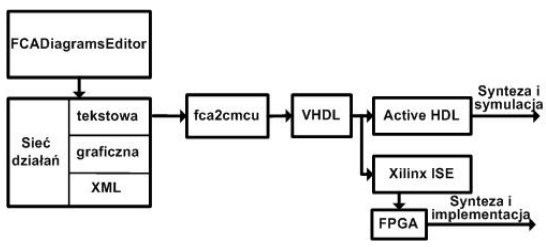


Rys. 5. Interfejs graficzny
Fig. 5. Graphical interface

Pasek menu (1) dzieli się na trzy części, z których każda rozwija jedną lub więcej opcji umożliwiających tworzenie i przetwarzanie sieci działań, np.:
- otwarcie istniejącego projektu z dokumentu XML,
- otwarcie istniejącego projektu z pliku tekstowego,
- jednoczesne zapisanie sieci do wszystkich formatów obsługiwanych przez aplikację.

Panel rysowania (2) stanowi obszar roboczy, na którym można jedynie tworzyć i modyfikować sieć działań. Pasek stanu (3) informuje użytkownika, która z kontrolek jest obecnie używana. Na pasku narzędzi (4) znajdują się podstawowe przyciski z paska menu. Natomiast pasek narzędzi (5) zawiera kontrolki potrzebne do stworzenia sieci działań. Ostatnim elementem okna głównego jest konsola, na której wyświetlane są informacje o operacjach wykonanych przez użytkownika.

Na rysunku 6 przedstawiono typową ścieżkę projektową wykorzystującą oprogramowanie FCADiagramsEditor.



Rys. 6. Ścieżka projektowa z wykorzystaniem oprogramowania FCADiagramsEditor
Fig. 6. Design path using the FCADiagramsEditor software

W wyniku działania programu tworzone są trzy pliki wynikowe zawierające opisane w artykule struktury sieci działań.

Następnie za pomocą oprogramowania fca2cmcu reprezentacja tekstowa sieci działań może być przetworzona do struktur mikroprogramowanych układów sterujących [6].

Wynikiem działania programu fca2cmcu jest plik zawierający opis struktury mikroprogramowanego układu wykonany w języku VHDL. Kolejnym krokiem może być synteza i symulacja działania mikroprogramowanych układów sterujących w środowisku Active-HDL oraz implementacja na przykład z wykorzystaniem pakietu Xilinx ISE.

4. Wnioski

W chwili obecnej, na Uniwersytecie Zielonogórskim, prowadzonych jest wiele badań nad implementacją sterowników logicznych, w których do opisu algorytmu sterowania wykorzystuje się tekstową reprezentację sieci działań. Niestety żaden z komercyjnych systemów nie konwertuje sieci działań do postaci tekstowej.

Ręczne konwertowanie sieci, zwłaszcza o dużej ilości bloków operacyjnych i warunkowych, zajmuje stosunkowo dużo czasu i często zawiera wiele błędów. Zaprezentowany w artykule system w znacznym stopniu usprawnił prowadzone prace i skrócił czas tworzenie sieci testowych.

Program FCADiagramsEditor umożliwiała tworzenie sieci działań a następnie zapisanie utworzonego modelu do:
- pliku tekstowego o zadanej strukturze,
- dokumentu XML,
- pliku graficznego JPEG.

Prace nad implementacją aplikacji objęły również adaptację algorytmu do automatycznego rozmieszczenia elementów sieci działań w danym obszarze roboczym. Umożliwiło to wizualizację modeli zapisanych tylko w pliku tekstowym, który nie zawiera informacji o współrzędnych poszczególnych elementów.

W systemie wspomagającym projektowanie układów mikroprogramowanych zaimplementowano również moduł obsługi dokumentów XML. Moduł ten zapewnia możliwość bezpośredniego odczytu i zapisu na dysku lokalnym plików reprezentujących sieć działań. Ponieważ struktura dokumentu XML zawiera informacje o współrzędnych poszczególnych elementów, pozwoliło to na uniknięcie szeregu skomplikowanych obliczeń potrzebnych do prawidłowego rozmieszczenia elementów sieci, a co za tym idzie znacznie przyspieszyło wczytanie danej sieci działań.

5. Literatura

[1] S. Baranov: Logic and system design of Digital Systems, Tallinn University of Technology Press and SiB Publishers (Toronto) 2008.
[2] A.A. Barkalov, V.A. Salomatin, K.E. Starodubow: System of CAD for design of control units with PALs and ROMs. Control systems and machines, No. 5, 1991, s.22-26.
[3] A. Barkalov, L. Titarenko: Logic synthesis for compositional microprogram control units, Berlin: Springer-Verlag, 2008, 272 s.
[4] M. B. Gorzałczany: Układy cyfrowe metody syntezy. Tom II Układy sekwencyjne układy mikroprogramowane. Wydawnictwo Politechniki Świętokrzyskiej, Kielce 2000
[5] M. Kołopieńczyk: Zastosowanie konwertera adresów do zmniejszenia rozmiaru pamięci mikroprogramowanego układu sterującego ze współdzieleniem kodów, Rozprawa doktorska, Uniwersytet Zielonogórski, Zielona Góra 2007.
[6] M. Kołopieńczyk: Program konwertujący tekstową sieć działań do struktur mikroprogramowanych układów sterujących ze współdzieleniem kodów. Przegląd Telekomunikacyjny i Wiadomości Telekomunikacyjne 2008, nr 6, s. 809-810.
[7] T. Łuba: Synteza układów cyfrowych, Praca zbiorowa pod redakcją prof. Tadeusza Łuby Warszawa: WKŁ 2003, s. 296.
[8] M. Molski: Modułowe i mikroprogramowalne układy cyfrowe. WKŁ, Warszawa 1986.
[9] S. Yang: Logic Synthesis and Optimization Benchmarks User Guide. Version 3.0, Technical report, Microelectronics Center of North Carolina, North Carolina, 1991.

Artykuł recenzowany