Marek WĘGRZYN, Andrzej KARATKIEWICZ
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI, ul. Podgórna 50, 65-246 Zielona Góra

Analiza porównawcza narzędzi syntezy Altera Quartus II i Synthagate

Dr inż. Marek WĘGRZYN

Kierownik Zakładu Inżynierii Komputerowej w Instytucie Informatyki i Elektroniki Uniwersytetu Zielonogórskiego. Zainteresowania badawcze obejmują zagadnienia projektowania systemów cyfrowych, ze szczególnym uwzględnieniem logiki programowalnej i języków opisu sprzętu (VHDL i Verilog). Członek PTI, POLSPAR, IEEE oraz IFAC (przewodniczący komitetu TC 3.1: *Computers for Control*).

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

Dr hab. inż. Andrzej KARATKIEWICZ

Profesor Uniwersytetu Zielonogórskiego, prodziekan Wydziału Elektrotechniki, Informatyki i Telekomunikacji UZ. Ukończył Miński Instytut Radiotechniczny (1993), obronił pracę doktorską w 1998, uzyskał stopień doktora habilitowanego w 2009. Zainteresowania naukowe dotyczą głównie teorii sieci Petriego oraz zagadnień analizy i formalnej weryfikacji współbieżnych systemów sterowania. Członek PTI.

e-mail: A.Karatkiewicz@iie.uz.zgora.pl

Streszczenie

W artykule przedstawiono analizę porównawczą skuteczności działania przemysłowego narzędzia syntezy układów cyfrowych FPGA (na przykładzie systemu Altera Quartus II), a narzędzia syntezy o pochodzeniu akademickim (Synthagate). Eksperymenty przeprowadzono z wykorzystaniem szeregu przykładów opisujących automaty skończone. Przedyskutowano wpływ sposobu opisu automatów na wyniki syntezy. Stwierdzono, że system Synthagate daje na ogół lepsze wyniki pod względem wykorzystania zasobów układów programowalnych oraz działa znacznie szybciej, niż narzędzie przemysłowe.

Słowa kluczowe: synteza logiczna, automaty skończone, układy cyfrowe, FPGA, projektowanie wspomagane komputerowo.

Comparative analysis of design software tools Altera Quartus II and Synthagate

Abstract

The paper presents comparison between efficiency of an industrial FPGA design software tool Altera Quartus II and similar design software tool Synthagate by Syntezza company of an academic origin. The experiments were performed using a series of examples describing the Moore finite state machines; one-hot state encoding was used in all cases. Area (number of used logical blocks) was the main parameter used for the comparison. Influence of the way of FSM description (in VHDL language) on the quality of synthesis was studied. The obtained results show that Synthagate in almost all cases performs synthesis more efficiently and essentially quicker than Altera Quartus. Section 1 presents motivation of the research; Section 2 describes problems which had to be solved to provide correctness of experimental comparison. In Section 3, the experimental results are presented. Section 4 describes still existing problems related to the comparison, which have to be solved. Section 5 presents the conclusions.

Keywords: logical design, finite-state machines, logical devices, FPGA, computer-aided design.

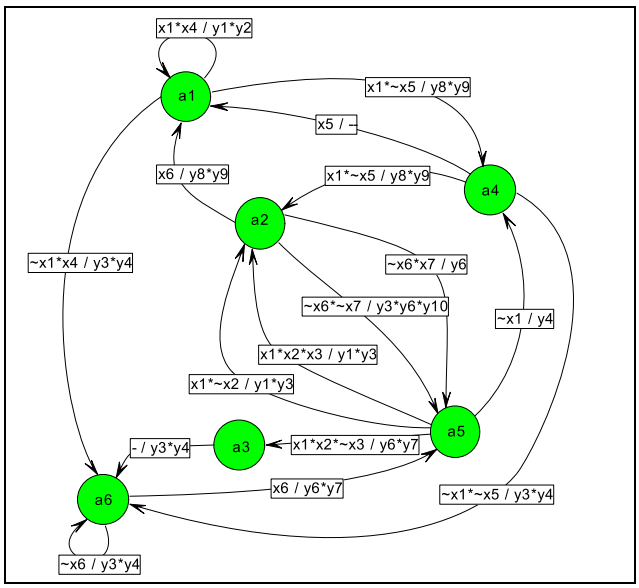
1. Motywacja

Autorzy akademickich (lub powstałych jako akademickie) narzędzi syntezy cyfrowej często twierdzą (przykłady z prac [2,3,4,7,8]), że opracowane przez nich narzędzia demonstrują znacznie większą skuteczność (pod względem minimalizacji rozmiaru projektowanych układów, szybkości działania tych układów oraz szybkości działania samych narzędzi syntezy), niż odpowiednie systemy przemysłowe. Na przykład, w [7] przedstawiono wyniki eksperymentów porównujących skuteczność działania szeregu przemysłowych narzędzi syntezy (w tym Altera MAX+Plus II, Quartus II oraz Leonardo Spectrum) z programem akademickim DEMAIN.

Pomijając pytanie, dlaczego duże firmy z branży elektronicznej wydają się nie wykazywać zainteresowania najbardziej skutecznymi algorytmami syntezy układów logicznych, postanowiono sprawdzić, czy systemy przemysłowe faktycznie przegrywają z systemami akademickimi. W tym celu sprawdzono działanie dwóch takich narzędzi na przykładach syntezy układów sekwencyjnych FPGA, opisanych jako automaty skończone. Jako narzędzia wykorzystano system Quartus II firmy Altera (wersja 12) i system Synthagate firmy Syntezza, implementujący algorytmy, opracowane w Bar Ilan University (Izrael) przez zespół pod kierownictwem profesora S. Baranowa [1].

2. Wpływ sposobu opisu automatów na wyniki syntezy

Pierwsze eksperymenty pokazały przytłaczającą przewagę narzędzia Synthagate nad narzędziem Quartus, jednak uważniejsza analiza procesu syntezy wykazała, że dwa systemy w istotnie różny sposób interpretują opisy automatów. Formatem wejściowym dla systemu Synthagate jest, w przypadku syntezy automatów, format tekstowy opisujący przejścia automatu skończonego (Rys. 1). Specyfikacja w takim formacie jest konwertowana na specyfikację w formacie VHDL (lub Verilog).



Rys. 1. Przykładowy automatu
Fig. 1. An example of FSM

System Altera Quartus interpretuje taki zapis w ten sposób, że każdemu z sygnałów wyjściowych przydzielany jest dodatkowy przerzutnik (wyjścia rejestrowe), ponieważ wartości są im przypisywane wewnątrz procesu, który posiada sygnał zegarowy CLK w liście czułości [10]. Natomiast system Synthagate w takiej sytuacji nie przydziela dodatkowych przerzutników sygnałom wyjściowym (wyjścia kombinacyjne), wskutek czego liczba wykorzystanych przerzutników, a co za tym często idzie, bloków logicznych okazuje się mniejsza.

Oznacza to, że wykorzystanie specyfikacji w formacie VHDL, wygenerowanych przez Synthagate, dla eksperymentów z narzędziem Altery nie pozwoli na dokonanie rzetelnego porów-

nania działania dwóch narzędzi. Zaszła potrzeba generowania specyfikacji automatów w języku VHDL w inny sposób (dwa różne sposoby). Rys. 2 przedstawia fragment modelu. Taka specyfikacja powoduje istotnie lepsze wyniki syntezy, dokonywanej przez Quartus II i pozwala na dokonanie dokładniejszego porównania systemów Quartus i Synthagate

```
Outputs_States: process (stan, x)
begin
    case stan is
       when st1 =>
            if x(6) = '1' then
               stan_temp <= st2; y <= "0000000110";
            elsif x(6) = '0' and x(7) = '1' then
               stan_temp <= st5; y <= "0000010000";
            elsif x(6) = '0' and x(7) = '0' then
               stan_temp <= st5; y <= "0010010001";
            else stan_temp <= st1; y <= "0000000000";
            end if;
       ...
end process;

FSM_machine_2: process (clk, reset)
begin
    if reset='1' then
        stan <= st1;
     elsif clk'event and clk = '1' then
        stan <= stan_temp;
     end if;
end process;
```

Rys. 2. Fragment alternatywnej specyfikacji automatu
Fig. 2. A fragment of an alternative FSM description

3. Wyniki eksperymentów

W badaniach wykorzystano kilkadziesiąt przykładów automatów Mealy'ego z binarnymi wejściami i wyjściami, po części pochodzących z projektów studenckich, a częściowo z przemysłu. Przykłady zostały podzielone na 5 grup, w zależności od ich rozmiaru: *Small*, *Medium*, *Large*, *Great*, *SuperGreat*. Dla wszystkich przypadków wykorzystano metodę kodowania *one-hot*, która jest zalecana dla „bogatych w przerzutniki" struktur FPGA, co pozwala zaoszczędzić liczbę wykorzystanych bloków logicznych przez uzyskanie prostszych funkcji wzbudzeń przerzutników [9], oraz opcję optymalizacji „area" (rozmiar) dla obu badanych narzędzi.

Wyniki testów wyraźnie wskazują, że algorytmy zastosowane w systemie Synthagate są skuteczniejsze, niż stosowane w środowisku firmy Altera; Synthagate pozwala syntezować układy, które dla realizacji takiej samej funkcjonalności potrzebują znacznie mniej bloków logicznych, niż układy syntezowane przy pomocy systemu Altera Quartus II. W tabeli 1 przedstawiono uśrednione wyniki dla różnych grup przykładów. Pokazują one, że wraz ze wzrostem złożoności automatów wzrasta również różnica pomiędzy analizowanymi systemami wyrażona rozmiarami syntezowanych układów.

Tab. 1. Średnie wyniki eksperymentów dla grup przykładów o różnych rozmiarach
Tab. 1. Average results of the experiments for the groups of different size

Grupa	*Średnia liczba bloków (Quartus)*	*Średnia liczba bloków (Synthagate)*	*Rozmiar (%)*
Small	55	35	64%
Medium	540	216	40%
Large	2253	834	37%
Great	21224	5284	25%
SuperGreat	--	18313	--

4. Kierunki dalszych badań

Przeprowadzone eksperymenty nie dają odpowiedzi na wszystkie postawione pytania, dotyczące porównania narzędzi, ani też nie pozwalają na porównanie wszystkich istotnych aspektów ich działania. W artykule przedstawiono szereg zagadnień, które określają perspektywę dalszych badań omawianego tematu.

5. Wnioski

Badania pokazały, że system akademicki pozwala uzyskać lepsze wyniki syntezy układów sekwencyjnych i w znacznie krótszym czasie, niż jedno z czołowych narzędzi przemysłowych. Choć niektóre aspekty tego zagadnienia wymagają dalszych badań, wyżej wymieniony wniosek jest wyraźnie potwierdzony przez przeprowadzone eksperymenty. Ponadto eksperymenty wyraźnie pokazują, że wraz ze wzrostem złożoności automatów wzrasta również różnica między rozmiarami syntezowanych układów, na niekorzyść badanego systemu przemysłowego. Dalsze badania powinny wykazać, które kroki optymalizacji są realizowane przez przemysłowe systemy we względnie nieskuteczny sposób.

Inne, istotne pytanie, na które nie uzyskano na razie odpowiedzi (które jednak leży w innym obszarze) – dlaczego duże firmy produkują mniej skuteczne narzędzia syntezy, mimo możliwości optymalizacji zarówno wyników, jak i prędkości ich działania. Oczywiście, narzędzia przemysłowe, tworzone przez duże zespoły, mają szereg niewątpliwych zalet w porównaniu z „minimalistycznymi" narzędziami, powstającymi na uniwersytetach lub w małych firmach. Systemy przemysłowe są bardziej rozbudowane, zintegrowane, posiadają wygodniejszy interfejs, obszerne biblioteki oraz możliwości bardziej szczegółowej analizy właściwości projektowanych układów, na przykład, ich oczekiwanej maksymalnej szybkości działania. Z drugiej strony narzędzia, takie jak Synthagate, dają możliwość bardziej szczegółowego wglądu w strukturę syntezowanych układów, niż systemy przemysłowe, ponieważ dla inżyniera-użytkownika przemysłowych narzędzi taki wgląd nie jest na tyle istotny, jak dla specjalisty, który się zajmuje doskonaleniem algorytmów syntezy. Czyli każdy z tych rodzajów narzędzi ma swoje wady i zalety, ale to nie jest jeszcze odpowiedzią na pytanie, dlaczego duże firmy, które przecież konkurują ze sobą, zaniedbują wykorzystanie algorytmów, pozwalających na głęboką optymalizację projektowanych układów.

6. Literatura

[1] Baranov S.: Logic and System Design of Digital Systems. TUT Press, Tallinn, 2008.

[2] Baranov S.: ASMs in High Level Synthesis of EDA tool Synthagate. Proceedings of the 4th IFAC Workshop on Discrete-Event System Design, Gandia Beach, Spain, 2009.

[3] Barkalov A., Titarenko L., Hebda O.: Optimization of Moore finite-state-machine matrix circuit. Pomiary, Automatyka, Kontrola. Vol. 57, nr 8, 2011, s. 939-941.

[4] Borowik G.: Improved State Encoding for FSM Implementation in FPGA Structures with Embedded Memory Blocks. Electronics and Telecommunications Quarterly, Vol. 54, no 1, 2008.

[5] Bukowiec A.: Synthesis of Finite State machines for FPGA Devices Based on Architectural Decomposition. University of Zielona Góra Press, Zielona Góra, 2009.

[6] Chmielewski S., Węgrzyn M.: Modelling and synthesis of automata in HDLs. Proceedings of SPIE, Vol. 6347, Photonics Applications in Astronomy, Communications, Industry, and High-Energy Physics Experiments 2006; Ryszard S. Romaniuk (Ed.), 2006, s.63470J1-13.

[7] Rawski M., Łuba T., Jachna Z., Tomaszewicz P.: The Influence of Functional Decomposition on Modern Digital Design Process, w: Design of Embedded Control Systems, Springer-Verlag, 2005, s. 193-204, DOI: 10.1007/0-387-28327-7_17.

[8] Rawski M., Tomaszewicz P., Borowik G., Łuba T.: Logic Synthesis Method of Digital Circuits Designed for Implementation with Embedded Memory Blocks of FPGAs, Design of Digital Systems and Devices (Lecture Notes In Electrical Engineering, Vol. 79), pp. 121-144, Springer-Verlag, Berlin Heidelberg, 2011.

[9] Xilinx. HDL Synthesis for FPGAs Design Guide, 1995.

[10] Zwoliński M.: Projektowanie układów cyfrowych z wykorzystaniem języka VHDL. Wyd. 2, Wydaw. Komunikacji i Łączności, Warszawa, 2007.