Marek WĘGRZYN, Andrei KARATKEVICH
UNIVERSITY OF ZIELONA GORA, INSTITUTE OF COMPUTER ENGINEERING AND ELECTRONICS,
ul. Podgórna 50, 65-246 Zielona Góra, Poland

Experimental comparison of synthesis tools Altera Quartus II and Synthagate

Marek WĘGRZYN, Ph.D.

He is a Head of Computer Engineering Division at University of Zielona Góra. His research interests are focused on Digital System Design, especially Programmable Logic and Hardware Description Languages (VHDL and Verilog). He is a member of PTI (Polish Information Processing Society), POLSPAR, IEEE and IFAC (Chair of Technical Committee 3.1: *Computers for Control*).

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

Andrei KARATKEVICH, Ph.D., D.Sc.

Professor of the University of Zielona Góra, deputy dean of the Faculty of Electrical Engineering, Computer Science and Telecommunications. Research interests concentrate mainly on the applications and theory of Petri nets and problems of analysis and formal verification of the parallel control systems. A member of PTI (Polish Information Processing Society).

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

Abstract

The paper presents comparison between efficiency of an industrial FPGA design software tool Altera Quartus II and a design software tool of an academic origin - Synthagate by Syntezza company. The experiments were performed using a series of examples describing the finite state machines; one-hot state encoding was used in all cases. Area was the main parameter used for the comparison. Influence of the way of FSM description 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.

Keywords: logical design, state machines, logical devices, FPGA, VHDL.

1. Introduction

Authors of the CAD systems of an academic origin often claim that their tools are much more efficient than the industrial CAD tools [2,4]. Some of the experiments described in the publications show that area of the devices constructed by the academic and industrial tools differs several times in average.

The authors decided to perform some experiments for checking such claims. Two systems have been compared: a popular industrial tool Altera Quartus II and tool Synthagate by Syntezza company, which uses the algorithms developed in Bar Ilan University (Israel) by the team lead by S. Baranov [1]. The sequential FPGA devices were synthesized, specified by the finite state machine descriptions.

2. Notes on the Way of Specification

The first experiments demonstrated that the two compared tools interpret the VHDL descriptions in different ways. Synthagate converts an FSM state transition table into a VHDL description. Fragment of such description is shown in Fig. 1. The problem is that Altera Quartus interprets such description in such a way that there is a flip-flop added for every output signal in a synthesized device; but Synthagate synthesizes the devices with purely combinational outputs. That means that different VHDL descriptions should be used for two tools to ensure purity of the experiments.

```
begin
   process (clk, rst)
   procedure proc_Girl10 is
   begin
     y1   <= '0';  y2   <= '0';
     y3   <= '0';  y4   <= '0';
     y6   <= '0';  y7   <= '0';
     y8   <= '0';  y9   <= '0';
     y10  <= '0';

   case current_Girl10 is
   when s1 =>
      if ( x6 ) = '1' then
         y8 <= '1';
         y9 <= '1';
         current_Girl10 <= s2;
      elsif (not x6 and x7) = '1' then
         y6 <= '1' ;
         current_Girl10 <= s5;
      else
         y3 <= '1';
         y6 <= '1';
         y10 <= '1';
         current_Girl10 <= s5;
      end if;
   ...
   end process;
```

Fig. 1. A fragment of an FSM description in VHDL (Synthagate)

For this reason specifications used for Quartus were changed in such a way that the values were assigned to the output signals outside the process which has the clock signal on its sensitivity list. Fragment of such description is shown in Fig. 2.

```
FSM_machine_1: process (clk, reset)
  begin
    if reset='1' then
       stan <= st1;
    elsif clk'event and clk = '1' then
     case stan is
       when st1 =>
          if x(6) = '1' then
               stan <= st2;
          elsif x(6) = '0' and x(7) = '1' then
               stan <= st5;
          elsif x(6) = '0' and x(7) = '0' then
               stan <= st5;
          end if;
       when st2 =>
   ...
  end process;

Y_assignment:
  y <=    "0000000110"
          when (stan=st1 and (x(6)='1'))
     else "0000010000"
          when (stan=st1 and (x(6)='0' and x(7)='1'))
     else "0010010001"
          when (stan=st1 and (x(6)='0' and x(7)='0'))
     else
     ...
```

Fig. 2. A fragment of an alternative FSM description (variant 1)

Another problem was detected related to such variant of specification. The experiments show that the devices synthesized by Altera Quartus using such description have separate combinational parts generating output signals and input signals for the flip-flops; however Syntezza performs common minimization of both output functions and transition functions, which allows reducing the device area. For this reason one more variant of the specification

was developed, shown in Fig. 3. This variant was used for the comparison.

```
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;
```

Fig. 3. A fragment of an alternative FSM description (variant 2)

3. Experimental results

For the experiments a series of state machine descriptions with binary inputs and outputs was used. It was divided into 5 groups differing in their size: *Small*, *Medium*, *Large*, *Great*, *SuperGreat*, with average number of states about 30, 100, 200, 700 and 1000, respectively. For all examples and both tools the *one-hot* state encoding method was used (which is common for FPGA devices [5]) and area as the main optimization criterion was selected. The input specifications for Quartus were generated using variant 2 presented above (Fig. 3).

Results of experiments for several examples from different groups are presented in Table 1. It also demonstrates comparison between numbers of FPGA logic blocks in the devices designed by two tools. Table 2 presents average results for all groups of examples (95 FSM descriptions were used for the experiments).

Tab. 1. Selected result of the experiments

Example	Group	Number of states	Number of blocks (Quartus)	Number of blocks (Synthagate)	Area (%)
girl10	Small	6	23	19	83%
lift2	Small	13	61	48	79%
cpu	Small	16	80	38	48%
sara	Medium	36	233	114	49%
bs	Medium	17	368	170	46%
gol	Medium	58	449	141	31%
yaron	Medium	99	848	231	27%
mars2M	Medium	218	803	422	53%
bulln	Large	111	2305	946	41%
exxm	Large	79	3151	990	31%
mars2	Large	274	1312	566	43%
bigm2r	Great	174	12622	3948	31%
zoom	Great	319	17326	4481	26%
group15m	Great	422	28468	5933	21%
otherm	Great	1275	26481	6772	26%

Tab. 2. Average results of the experiments for the groups of different size

Group	Average number of blocks (Quartus)	Average number of blocks (Synthagate)	Area (%)
Small	55	35	64%
Medium	540	216	40%
Large	2253	834	37%
Great	21224	5284	25%
SuperGreat	--	18313	--

The obtained results show, that Synthagate is a more efficient and deeply optimizing tool for designing sequential FPGA devices than Altera Quartus II. It is worth noting that Synthagate works essentially quicker. Quartus was unable to perform synthesis for the examples of the group *SuperGreat*, and Synthagate needed only few minutes for each of them.

4. Further Research

The following topics should be investigated for better understanding of differences between two kinds of the design tools.

1. The performed experiments were intended to compare area of the synthesized devices, but comparison of speed was not performed yet. Such comparison would be especially interesting for the case when speed is selected as the main optimization criterion in both tools. Synthagate allows calculating for the synthesized devices the critical path length, which limits maximal possible frequency of clock signal and hence maximal speed of the device. Calculating of critical path for the Quartus synthesis results would allow comparison of speed. According to the obtained results, Synthagate with selected option of speed optimization generates the devises consisting of less number of logic blocks, then Altera Quartus with the option of area optimization.
2. Interesting and important seems to be comparison between synthesis time for both tools. The results obtained up to now demonstrate that synthesis times may have the orders of magnitude difference, especially for the biggest test examples. It is hoped that detailed analysis will allow understanding where is the bottleneck of the industrial tool, however the difficulty is that we usually do not know how long the separate steps of synthesis process are executed.

5. Concluding Remarks

The research demonstrates that the CAD tool of an academic origin solves the tasks of synthesis of sequential FPGA devices more efficiently and quicker than one of the most popular industrial synthesis tools. The experiments also show that the difference is especially remarkable for the bigger examples.

An intriguing question is: why big corporations produce relatively inefficient design tools, when it is possible to use the algorithms that are quicker and provide better optimization? Obviously, the industrial systems have richer and more user-friendly interfaces, huge libraries and many other features, which lack the tools created by small companies or university teams. On the other hand, tools like Synthagate allow their user to have more information about structure of the designed devices. Both kinds of tools have their good and bad sides. But it still does not explain why the corporations, in spite of competition between them, do not use the efficient algorithms of synthesis.

6. References

[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] 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, pp.63470J1-13.

[4] 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, pp.193-204, DOI: 10.1007/0-387-28327-7_17.

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