DIGITAL SYNTHESIS TOOLS FOR EDUCATION AND RESEARCH

Elena Fomina. Peeter Ellervee, Margus Kruus, Alexander Sudnitson, Kalle Tammemae
Raja 15, 12618 Tallinn, Dept. of Computer Engineering, Tallinn Technical University
Estonia

Abstract: The growing sophistication of applications continually pushes the design and manufacturing of digital systems to new grades of complexity. Computer aided design education deals with the computer as a tool to help perform the various tasks that comprise the design process of digital systems. In order to understand the capabilities of that tool, one has to study how the tool is used in the different steps of the design process. This paper describes approaches and computer-aided design software developed at the Department of Computer Engineering of Tallinn Technical University for research and teaching purposes.

Keywords: High-level synthesis, decomposition, educational computer-aided synthesis tools.

1. INTRODUCTION

Computer-aided design technology has been developing rapidly during the last decades. This has been caused both by the growing size of task, i.e., chips to be produced, and by the availability of increasingly powerful computers to solve these tasks. The advantages in general, and especially the use of higher level of abstractions for specification, are being pushed by the following factors:

- the need to decrease the time to design a VLSI circuit and to get it to the market;
- the increased complexity of circuits, amplified by the improvements in the processing technology;
- the increased cost of design iteration – from specification to fabrication – requires less error prone design methods;
- the availability of fast prototyping approaches have somewhat lessened effects of the increased iteration cost, but they add more iterations in effect;
- it is often necessary to retarget an existing design to a new technology. The design process can be simplified also by reusing parts (components) of existing older designs;
- related to the reuse is the cost of maintenance, i.e., it is essential that the specification is easy to read and is well documented.

In this paper, we present computer aided synthesis tools at different abstraction levels. Our work is motivated by the fact that the existing commercial tools are not always good for academic activities because it is very hard, if not impossible, to get into details of the synthesis process:

- there must exist a way to show to the students how a synthesis tool works step-by-step; and
- researchers need rather often a possibility to modify an algorithm in the synthesis process, or even to modify the execution order of algorithms.

The use of Java applets can encourage asynchronous distance learning and thus overcome the limitations inherent in traditional instructional techniques. Java applets can help create an interactive environment of “leaning by doing”. Beyond their ability to better convey certain concepts, the applets can increase motivation and instill greater interest among students.

The rest of the paper is organized as follows. In section 2, we will present a synthesis-related research at the Department of Computer Engineering of Tallinn Technical University. Section 3 describes computer-aided synthesis tools at high-level, and register-transfer level (including sequential machine decomposition) of design hierarchy. Section 4 gives some concluding remarks.

2. SYNTHESIS RELATED RESEARCH

Researchers at the Department of Computer Engineering [1] have worked on synthesis problems ranging from logic level synthesis to system level synthesis. Today, the research is targeting higher abstraction levels to cope with the complexity. At the same time, design experience is transferred to students using modern design tools. A typical design flow starts with a hardware description language, mostly VHDL, model at higher abstraction levels. Describing models, for instance, at behavioral level, allows to find specification errors at earlier design phases without the need to build actual hardware. Step-by-step specification refinement together with validation allows to ensure that all transformation give expected results. That is, making sure that simulations of a high-level algorithm and register-transfer level structure give the same results. Automation of most of the steps allows not only faster design times but also additional evaluations of different solutions – design space exploration.

Both commercial and academic tools are used at the department for modeling and synthesis. Commercially available tools are used also for teaching to make students familiar with them. In laboratory experiments, tools from different vendors are used in a combined manner. For instance, modeling can be done by simulators either from Synopsys [2] or from Model Technology [3]. A free student version from the latter is included in WebPACK synthesis tool from Xilinx [4], thus making it possible to perform simple tasks even at home. Synthesis tools are from Synopsys and Xilinx, the first targeting ASICs and the second targeting FPGAs. The use of FPGAs as target technology allows fast prototyping to ensure the correctness of architectural solutions. Currently the main prototyping platforms are Xilinx Spartan-2 boards from XESS [5], allowing to build systems containing up to 100,000 equivalent gates.

3. COMPUTER-AIDED SYNTHESIS TOOLS

In this section, the three main research and teaching topics and corresponding computer-aided synthesis tools developed at the Department of Computer Engineering are described.

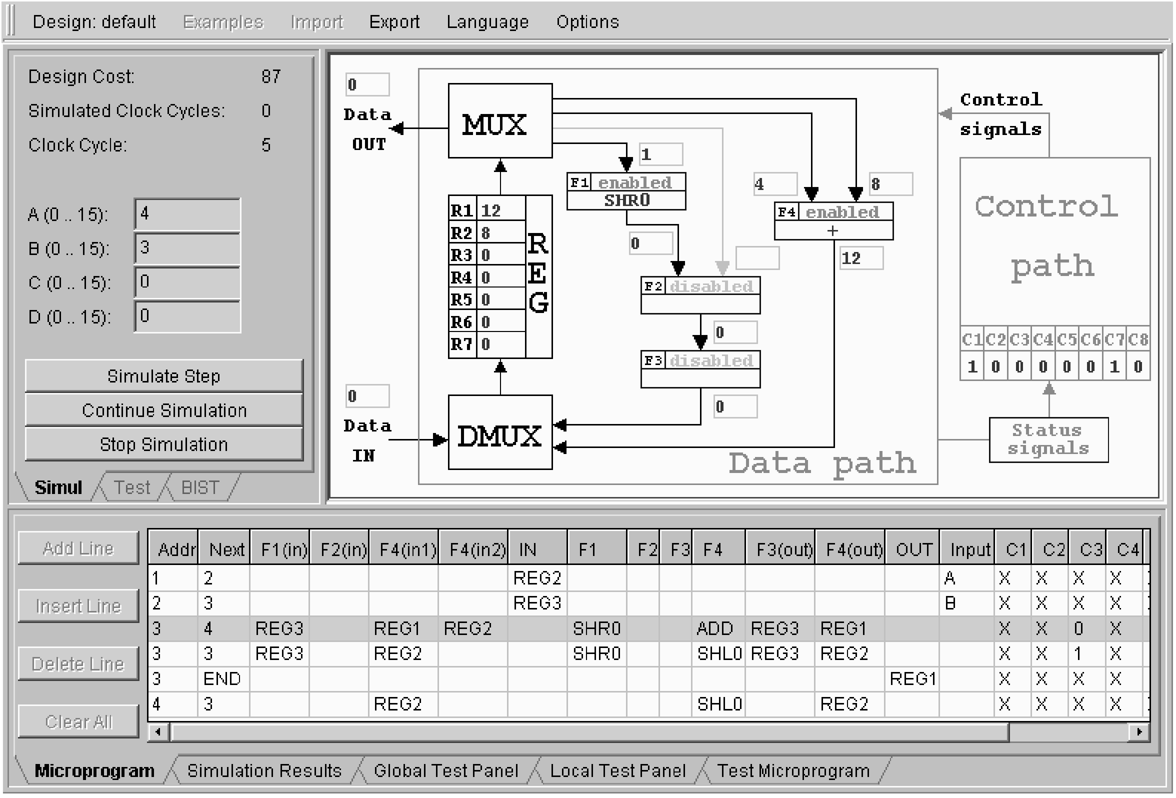
3.1. High-Level and System Synthesis

Although High Level Synthesis (HLS) has been successfully used in many cases and there exist rather many HLS tools, it is still not as indispensable today as layout or logic synthesis. Experiments with designs that are dominated not by the data flow but control flow and data transfers have pointed out that the traditional (and so far the most successful) data flow oriented synthesis strategy does not work well. A different approach is needed that would take into account the main characteristics of control and memory intensive systems (CMIST).

xTractor has been developed to test HLS methodology of CMIST applications. The nature of applications, i.e., control and data transfer dominance, defined the overall synthesis flow and transformations needed to convert a behavioral description of a design into Register Transfer Level (RTL) description. The tool consists of several smaller component programs to solve certain synthesis steps, and of an interactive shell around the component programs.

The overall synthesis flow, used in xTractor, is similar to the synthesis flow of any HLS. The three main steps can be outlined as follows:

- in the partitioning phase memories are extracted from the initial behavioral description as separate components;
- operations are assigned to states (control steps) during the scheduling phase; and
- unified allocation and binding assigns operations to specific functional units.



Design Cost: 87

Simulated Clock Cycles: 0

Clock Cycle: 5

A (0 .. 15): 4

B (0 .. 15): 3

C (0 .. 15): 0

D (0 .. 15): 0

R1	12
R2	8
R3	0
R4	0
R5	0
R6	0
R7	0

C1	C2	C3	C4	C5	C6	C7	C8
1	0	0	0	0	0	1	0

Addr	Next	F1(in)	F2(in)	F4(in1)	F4(in2)	IN	F1	F2	F3	F4	F3(out)	F4(out)	OUT	Input	C1	C2	C3	C4
1	2					REG2								A	X	X	X	X
2	3					REG3								B	X	X	X	X
3	4	REG3		REG1	REG2		SHR0			ADD	REG3	REG1			X	X	0	X
3	3	REG3		REG2			SHR0			SHL0	REG3	REG2			X	X	1	X
3	END												REG1		X	X	X	X
4	3			REG2						SHL0		REG2			X	X	X	X

Fig. 1. RTL design applet window

The separate steps of the flow are executed by component programs. The component programs manipulate Control and Data Flow Graph (CDFG) translated from a VHDL subset. The modular structure of the shell makes it easy to add such additional tools.

3.2. Register Transfer Level Synthesis

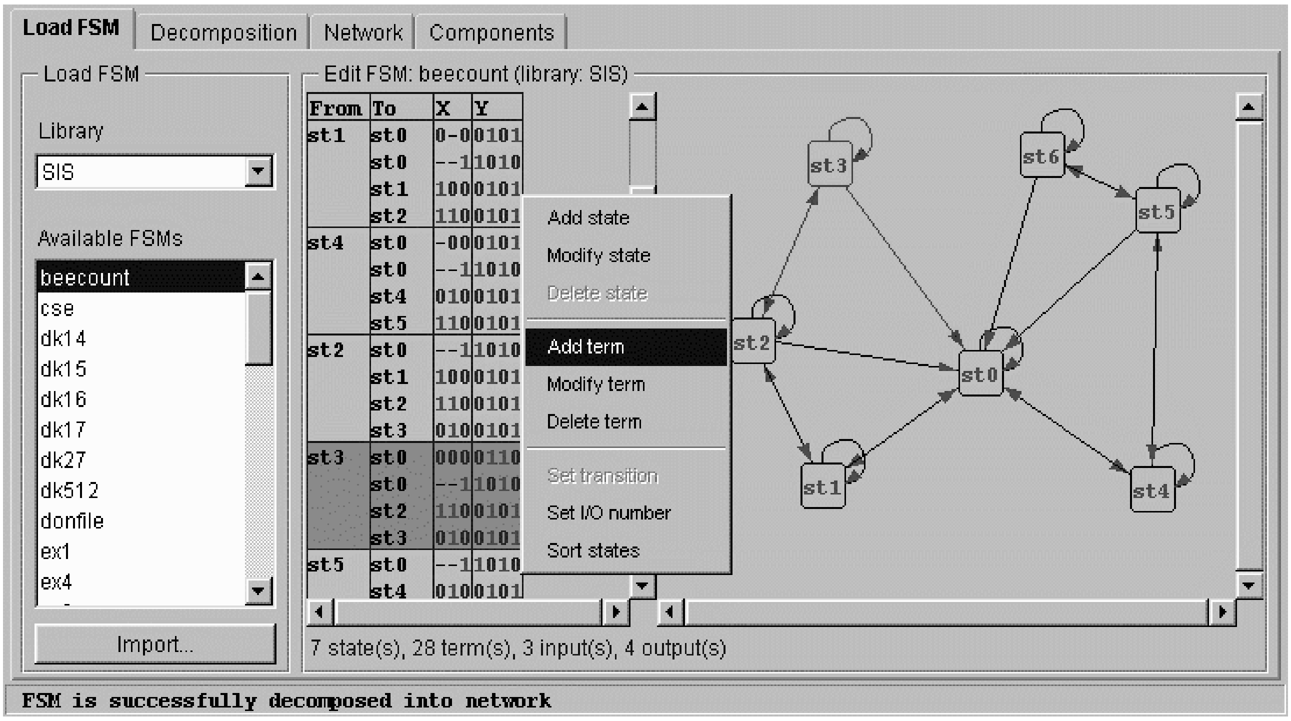
A digital system at register-transfer-level is characterized as follows [6]:

- the system is divided into a data subsystem (datapath) and control subsystem (control-path);
- the state of the data-path is defined by the contents of a set of registers;
- the function of the system is performed as a sequence of register transfers (in one or more clock cycles);
- a register transfer is a transformation performed on a datum while the datum is transferred from one register to another; and
- the sequence of register transfers is controlled by the control-path.

The RTL design and test applet (see Figure 1) allows to solve and illustrate many problems related to RTL synthesis. The range of problems includes:

- design of a data-path and controller;
- investigation of trade-offs between speed and hardware cost;
- RTL simulation;
- fault simulation;
- test generation; and
- design for testability and BIST (Built-In Self-Test).

The applet has a flexible design. The RTL system model, shown in Figure 1 is not mandatory. Any other model, specified in a configuration file, can be used as easily as the original one.



Load FSM

Library: SIS

Available FSMs: beecount, cse, dk14, dk15, dk16, dk17, dk27, dk512, donfile, ex1, ex4

Edit FSM: beecount (library: SIS)

From	To	X	Y
st1	st0	0-0	0101
	st0	--1	1010
	st1	100	0101
	st2	110	0101
st4	st0	-00	0101
	st0	--1	1010
	st4	010	0101
	st5	110	0101
st2	st0	--1	1010
	st1	100	0101
	st2	110	0101
	st3	010	0101
st3	st0	000	0110
	st0	--1	1010
	st2	110	0101
	st3	010	0101
st5	st0	--1	1010
	st4	010	0101

7 state(s), 28 term(s), 3 input(s), 4 output(s)

FSM is successfully decomposed into network

Fig. 2. The decomposition applet window

3.3. Finite State Machine Decomposition

FSM is the oldest way of describing and modeling autonomous systems. Although design complexities have grown continuously, this classical description method is still used by several synthesis tools. FSMs have been widely used also to express algorithms, communication protocols, digital systems, sequential logic circuits, and sequential logic cells.

Decomposition has been a classic problem of discrete system theory for many years. A large hardware behavioral description is decomposed into several smaller ones. One goal is to make the synthesis problem more tractable by providing smaller sub-problems that can be solved efficiently. Another goal is to create descriptions that can be synthesized into a structure that meets the design constraints. In the past, synthesis focused on quality measures based on area and performance. The continuing decrease in feature size and increase in chip density in recent years have given rise to considering decomposition theory for low power as new dimension of the design process [7].

The theoretical background of our approach is the automata decomposition theory that uses partition pair algebra [8]. The importance of this theory lies in the fact that it provides a direct link between algebraic relationships and physical implementations of finite state machines.

For teaching decomposition research purposes, a set of applets for studying the basics of the decomposition theory of FSMs have been developed. The applet for FSM network construction (see Figure 2) allows to experiment with decomposition of the prototype machine. Different partitions can be chosen to decompose the given FSM to meet different design restrictions.

The developed applet can be considered as a research tool that we use to carry out experiments intended to further develop decomposition synthesis. Experiments can be carried out on a set of well-known FSM benchmarks.

4. CONCLUDING REMARKS

In this paper, we present a conception of how to improve the skills of the students studying digital design related topics. Our work is motivated by the fact that the existing commercial tools are not always good for academic activities because it is very hard, if not impossible, to get into details of the synthesis process. We apply a learning method based on using the interactive teaching modules. Our goal is to use Web-based tools so that they can be used in asynchronous-mode learning of digital design.

The presented design system can be considered as a research tool that we use to carry out experiments guided to further development of the synthesis strategies.

ACNOLEDGEMENTS

This work has been supported by the Estonian Science Foundation (under Grant 5643 and Grant 5601).

REFERENCES

1. Department of Computer Engineering – http://ati.ttu.ee/english.php

2. Synopsys, Inc. – http://www.synopsys.com/

3. Model Technology – http://www.model.com/

4. Xilinx, Inc. – http://www.xilinx.com/

5. XESS Corp. – http://www.xess.com/

6. M. Ercegovac, T. Lang, and J. H. Moreno. *Introduction to Digital Systems*, John Wiley & Sons, NJ: 1999.

7. Hwang, E., Vahid, F., and Y.-C.Hsu. FSMD Functional Partitioning for Low Power. *Proceedings of the D A T E International Conference*, 1999, pp.22-28.

8. Hartmanis, J., and R. E. Stearns. *Algebraic Structure Theory of Sequential Machines*. Englewood Cliffs, N.J.: Prentice-Hall, 1966.