E-LEARNING TOOLS FOR TEACHING THE DECOMPOSITION BASED DIGITAL SYNTHESIS

ALEXANDER SUDNITSON AND MARGUS KRUUS
Department of Computer Engineering, Tallinn University of Technology, Raja 15, 12618 Tallinn, Estonia.

1. MOTIVATION

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. Besides researchers need rather often a possibility to modify an algorithm in the synthesis process, or even to modify the execution order of algorithms. Our teaching concept includes some problem based parts as well as possibilities for “Learning by doing”. By using these tools it is much easer to teach with a problem based approach. We give the students an interesting, practical task and show them how to solve it with the taught methods and the use of the tools. Because the tools are tailored to the learning process they are easy to use and have a self explaining user interface. The functionality is limited to the discussed design step and only a few variables so that different view can be placed at the same time at one screen. Every student’s input is followed by an immediate reaction of the tool in all views. Thus the students can concentrate at the solution - not at the features of a complex design tool. The tools help to avoid boring calculations which are normally necessary to solve complex task. All these teaching tools follow the concept of “Living pictures” [1] and are implemented as JAVA-Applets. This way they run platform independent on any browser platform and are accessible via the Internet free of charge. Typical “Living pictures” are highly interactive and have “The Big Picture” (a kind of summary at the end of a chapter) as user interface. Learners can interact with this interface by direct manipulation of formulas or values and make own experiments.

2. FSM CONCEPT AND DECOMPOSITION

Over the years, many important problems in sequential circuit systems design and optimization have been approached using concepts from automata theory. The concept of Finite State Machine (FSM) is a widely recognized model that is used for representing the work of sequential digital devices. Because of their finite nature, FSMs yield better to describe, to analyze and to synthesize of intricate digital systems than any other alternative model.
Traditionally, FSM is represented by algebraic model consisting of a set of states, an input and output alphabets, a transition function that maps inputs symbols and current states to a next state and an output function that produces an outputs. The FSMs are either specified directly by the designer using a hardware description language or realized via high level synthesis tools from a more abstract

description. Given an algorithmic specification and a library of functional units, high level synthesis constructs a data path by scheduling operations in the algorithm onto specific time steps and binds them onto appropriate functional units. A FSM that controls data path is then automatically synthesized.

It is not an overstatement that machine decomposition is one of the actual problems of complex discrete device synthesis in order to achieve different optimizations, such as area, performance, power consuming and testability. Commonly, the FSM decomposition problem is a task of representing prototype FSM by its network realization [2]. In other words, by decomposing source FSM we try to construct such network of interconnecting and interacting component sub-FSMs that will give us the same terminal behavior as the prototype FSM realizes. An FSM can be decomposed into smaller interacting machines in order to achieve different optimizations, such as area, performance, power consuming and testability.

Various decomposition techniques can broadly fall into two categories: multiplicative decomposition and additive decomposition. In case of multiplicative decomposition the graph of FSM is embedded into a product of smaller graphs [3]. Additive decomposition splits graph of the FSM into node disjoint subsets [4].

Multiplicative decomposition corresponds each of sub-FSMs to a partition on the set of states. A partition on S is a collection of disjoint nonempty subsets of S whose set union is S. All the sets that belongs to a single block of partition are given the same code in the corresponded sub-machine. Therefore, to distinguish two states in a single block of sub-machine the information from other sub-machines is required. The functionality of source FSM can be realized by the decomposed machine if and only if the product of partitions associated with decomposition is zero-partition (every block contains exact one state).

The main idea of additive decomposition is that the special “idle” state is added to each of sub-FSMs. Only one of sub-FSMs in the decomposed network is working in a time while all the others are suspended (stay in their “idle” states). The network of interacting sub-FSMs corresponds to a given partition on the set of states of prototype FSM. The number of blocks in the partition defines number of sub-FSMs in the network. The number of states of each sub-FSM is equal to the number of states in the corresponded block of partition plus one “idle” state.

The Generalized Additive Decomposition proceeds from a given cover on the set of the states of the source FSM. Each component machine corresponds to a subset of the set of the states of the source FSM. This method is a more generalized approach based upon the previous method (more than one machine could be active executing a computation at any given time while the other component machines are "idle").

3. DECOMPOSITION AND SYNTHESIS SOFTWARE

Decomposition and Synthesis (D&S) software is aimed to organize an environment for experimenting with various methods of decomposition. The package consist of

several components that capable to perform all the three types of decomposition that were described in the previous section.
Here is a list of main features that are provided by D&S environment:

- FSM decomposition by using several techniques
- Synthesis of network of interacting sub-FSMs
- Search of partitions for decomposition by specified criteria
- Random generation of partitions
- Import and export of FSM, sub-FSMs and network in various formats (VHDL, BLIF, KISS2)
- Built-in libraries of FSMs
- Integrated module for steady-state probabilities computation

As the first, step the prototype FSM together with partitions should be defined as source data for decomposition. The FSM can be selected from the built-in library of FSMs (that already contains several collections of benchmarks [5]) or imported using Berkeley KISS2 state machine description format. The set of partitions can be either generated randomly or constructed using special heuristic algorithm of partition search. Another supplementary tool that is capable to analyze steady-state probabilities by given input probabilities can also be used for elaborating set of source partitions.
After the decomposition is performed the target network of FSMs is obtained. The network complexity in comparison with source FSM can be measured using complexity analyzer. Currently, very simple method is used to analyze complexity of prototype FSM and target network – this is a subject of future work. As the next step, the result network of interacting FSMs can be exported for further using in other CAD software. VHDL and BLIF formats are used to represent whole network while sub-FSMs can be separately described in KISS2 format.
Most part of the software is implemented in the form of easy-to-use graphical user interface (GUI) applications.. Besides this, some of GUI modules are also duplicated by their command-line versions. Command-line tools are very useful for carrying out batch experiments on the set of benchmarks. They use less system resources and can be executed on server using a terminal. Currently basic command-line tools are available for performing the following tasks: decomposition, partition search, random partition generation and checking the FSM specification.
Besides the common way of using there is the possibility to access D&S package directly over Internet. The latter case is more suitable for educational and demonstration purposes. The only drawback of this approach is that web-based part (that is implemented in a form of Java applet) imposes some limitations on use of it (for example, applets are not allowed to save the results of user's work on the local computer). The web-based part of the D&S environment is available at [6].
One of the major benefits of D&S environment is that it is developed using Sun Java platform. Since Java technology was used, the range of platforms and operating systems where the programs can be executed is limited only by availability of Sun-compatible Java Runtime Environment (JRE) for the specific system. Note that since Microsoft Internet Explorer has own Sun-compatible JRE (that is usually included with the standard installation), no additional software is needed to use the web-based part of the environment under Windows operating

system. The most distributives of Linux also have built-in support of Java platform. Another significant part of D&S environment is the decomposition API (Application Programming Interface) that was developed along with software development. The most of the algorithms and data structures of API are documented and can be reused in further work on enhancing the software. The API allows to quickly build new decomposition modules and implement supplementary tools that can be later integrated into the main environment.

4. CONCLUSION

The developed D&S system provides user with the essential software environment for education and experimenting in the area of FSM decomposition. The various components of software package implement the several decomposition approaches. The system is intended to be used for two basic purposes.
The developed system can be easily used for educational purposes, because of it interactive nature and user-friendly graphical interface. In this case, the great advantage is that the GUI modules can be easily accessed from any point of the world over Internet. Besides this, it can act as a research instrument for investigations on decomposition. In addition to GUI version of software that gives the possibility to execute experiments interactively, the included command-line tools are useful for carrying out benchmarks. Due built-in support of widespread data formats, the experiments can be carried out on the range of well-known benchmarks of FSMs as well as on used-defined FSMs. The results of experiments can be exported to other CAD-related software for further design.
The decomposition API was also developed as a part of the work on the system. This feature provides developers with the framework for extending functionality of currently developed programs and creating new software in this area.

5. REFERENCES

[1] H.-D. Wuttke, K. Henke, "Schaltsysteme- eine automatenorientierte Einführung", Verlag: Pearson-Studium Deutschland, 2003.
[2] S. Hassoun, T. Sasao, “Logic Synthesis and Verification”, Boston: Kluwer Academic Publishers, 2002.
[3] G. De Micheli, “Synthesis and Optimization of Digital Circuits”, New York: McGraw-Hill, 1994.
[4] S. Baranov, “Logic Synthesis for Control Automata”, Boston: Kluwer Academic Publishers, 1994.
[5] McElvain, K.: LGSynth’93 Benchmark Set: Version 4.0, 1993.
Available: http://www.cbl.ncsu.edu/benchmarks.
[6] Web-Based Part of Decomposition Environment Available:
http://www.pld.ttu.ee/decomposition, http://www.pld.ttu.ee/applets.