Teaching Cognitive-Inspired Design of Sequential Circuits

Binyamin Abramov
School of Education
Tel Aviv University
Tel Aviv 69978, Israel
avramov@post.tau.ac.il

Osnat Keren
Engineering School
Bar-Ilan University
Ramat Gan 52900, Israel
kereno@eng.biu.ac.il

Ilya Levin
School of Education
Tel Aviv University
Tel Aviv 69978, Israel
ilia1@post.tau.ac.il

Abstract— **The paper deals with a cognitive-inspired design of digital systems. Specifically we discuses a new approach for teaching of sequential circuits (SC) design. The approach is based on using Split Algorithmic State Machines (ASM) for initial specification of SC. We analyze both the complexity of the Split ASMs and the quality the corresponding hardware solutions. The proposed approach was implemented and studied in an undergraduate Computer Engineering class. The results of our study demonstrate that the proposed approach improves both the students' success in solving design tasks and the debagability and testability of resulting schemes.**

Keywords- sequential circuit, algorithmic state machine, cognitive template.

I. INTRODUCTION

The present paper belongs to the field of digital design education. Specifically we deal with teaching sequential circuits (SC) design. A process of design/synthesis of SCs comprises a sequence of initial specification transformations for obtaining an optimized circuit according to design constraints (speed, area, power dissipation etc.). In the paper we demonstrate on the deep interrelations between the initial specification of the SC and its hardware implementation. Our approach is based on two hypotheses:

1) an initial specification of any SC is strongly affected by the cognitive style of the designer;

2) a logic circuit inheriting a structure of its initial specification is more understandable and debugable than a regular circuit that doesn't inherit the initial structure.

Various representations of SCs, as well as various hardware description languages are well studied. Usually, a SC is implemented in optimized form that is quite far from the initial specification. This gap between the initial specification and implementation is well grounded if the main design concern is the implementation cost. When debugability and understandability factors become dominant, the gap becomes questionable. Moreover, the human-machine interaction on the system level of design becomes more "user-oriented", while the implementation level of the design becomes more distinct from the system level, and thus, complex to be managed. In general, this gap characterizes the technology progress in the modern society. However, the common sense and the basic technological orientation being successful in the every-day life, is not always applicable for developing digital systems and, especially, for teaching logic design.

From our teaching experience, we came to conclusion that standard automation tools are not suitable for teaching logic design due to their user-orientation and not the implementation orientation.

The majority of SC initial specifications is developed by humans and, as a result, inherits some humans' thinking templates. Actually, the thinking templates usually have a tree-like structure and, fortunately, may be formalized. We consider a specific cognitive style − dichotomic networks − as a class of representations, that correspond to a particular specification of SCs. Dichotomic networks comprise so-called dichotomic fragments [2]. We refer to dichotomic fragments as binary trees; dichotomic networks are referred to as systems of interconnected fragments. An important example of a dichotomic representation of SC is an Algorithmic State Machine (ASM) chart. Development of the ASM chart requires thinking in terms of ASM paths, each corresponding to a specific Boolean cube. The entire ASM chart forms a single dichotomic fragment. The ASM specification is natural, logical, and testable, but it requires bearing in mind all possible logical paths (situations) and referring to them within the system. This strict requirement renders the ASM specification logical and testable, however, it puts designers in a position where they have to define very sophisticated and even artificial/unexpected states of the system.

When the number of variables grows, the process of defining an ASM specification becomes difficult and even unnatural. Whereas Hardware Description Languages (HDLs) allow a system specification in a natural way (split representations), the majority of the existing VLSI CAD synthesis tools are still based on the conventional SC representation. As a result, the solutions tend to be non-understandable and non-debugable. The main subject of our study is a newly introduced Split ASM having all the advantages of a conventional ASM but free of the above drawbacks. A SC can be defined both as a single ASM and by a network of ASMs of low complexity. The trade-off between these two extreme cases is of a special interest. This trade-off as well as the size of the ASM fragments and the structure of

their interconnection within the Split ASM is subjects of our study. It is known that when people are given a problem for which there exists a correct solution and an initial estimate of that solution, they tend to provide a final estimate close to the initial one. This is termed anchoring. Anchoring heuristics help humans simplify problem solving in complex situations without conscious effort. In our work, we apply the anchoring heuristic principle to teaching SC design. We consider the initial specification of SC as an initial estimate of the design solution. Since the entire design process is a sequence of equivalent transformations between various representations of the same SC, each of the representations may consider a certain implementation of the system. Thus, the initial specification is also a kind of implementation. If the initial specification is considered an anchor, then the design problem can be solved by using the anchoring and adjustment heuristics algorithm. In contrast to conventional teaching methods, we synthesize a SC directly from its initial specification. Circuits produced by the anchoring synthesis may have an additional overhead in comparison to circuits produced by the conventional optimization method. Nevertheless, we consider this a reasonable price for such an important property as debug-ability and readability of the SC. Moreover, we show that a relatively small overhead is required for the proposed approach.

The present work presents:

- A study of dichotomic networks as cognitive basis for specification of SCs.
- A study of the efficiency of teaching SCs defined in Split ASM form.

We show that use of Split ASM notation and dichotomic cognitive templates allows design of SCs in a very efficient way. Circuits designed by students according to the proposed approach are understandable and debugable. The proposed approach allows increasing a size and complexity of SCs that can be handled by students.

II. SPLIT ASM

A design process transforms a set of specifications into an implementation of the specifications. Both the specification and the implementation are forms of description of the system functionality, but they have different levels of abstraction. Given the behavioral description of the system's (observed or desired) functionality, from the formal model stage onwards, we distinguish between two main paradigms: the procedural (algorithmic) paradigm and the declarative paradigm [1,3].

According to the declarative paradigm, the person (e.g., student, user, designer) defines the structure of the system by focusing on the logical scheme of the SC. In contrast, according to the procedural paradigm the user focuses on the behavior of the system.

In our study, we deal with the most popular paradigm - the procedural paradigm. The main formal construct for the procedural paradigm is the ASM chart. The ASM chart is a directed connected graph comprising an initial vertex, a final vertex, a finite set of operator vertices, and conditional vertices [4]. Each conditional vertex contains a single logical condition from a set of input variables (x's) of the SC. Each operator vertex contains a specific output vector (Y) from the set of output signals of the SC.

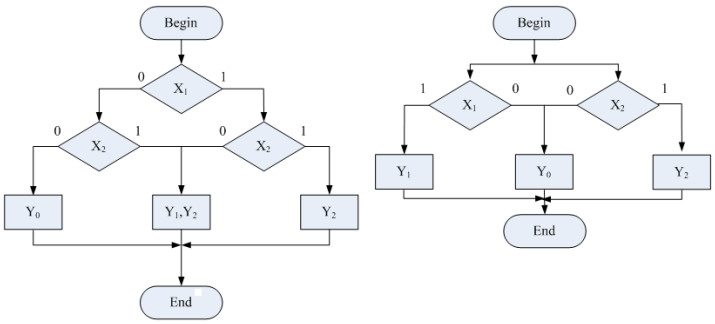
We define the Split ASM by connecting a number of conventional ASMs (component ASMs) as follows:

1) Parallel connection: connecting roots of two or more components ASMs.

2) Sequential connection: replacing one terminal node of an ASM with another ASM.

These component ASMs (sub-graphs in to Split ASM) correspond to dichotomic fragments. The output vectors of each of the fragments are logically summed.

Fig. 1 illustrates the ASM and Split-ASM descriptions of the same SC that controls two-axis manipulator having two inputs and two outputs variables, which form 3 types of output patterns.



a) b)

Figure 1. Example of a SC representation by ASM (a) and Split ASM (b).

Clearly, an ASM is very suitable form of the initial specification [2]. Nevertheless, it has a significant cognitive contradiction. On one hand, the ASM corresponds to a classical von Neumann's architecture that is based on the sequential, step-by-step execution of the algorithm. On the other hand, on the hardware level, the behavior of the ASM is concurrent. Furthermore, the ASM description includes a significant redundancy, associated with its tree-like nature. One of the main requirements for a modern systems' specification is its ability to support the concurrency.

The proposed paradigm allows combining advantages of both considered paradigms: the declarative and the procedural. Each sub-graph of Split ASM is the usual ASM, allowing the use of the existing design methodology. At the same time, communication sub-graphs allow to trace the structure of the described system. Our assumption of necessity of the initial specification and the final description conformity, finds the acknowledgement in the proposed paradigm. In additional, the Split ASM specification is therefore much more natural. It

corresponds to human cognitive patterns to present a complex entity as an assembly of simpler components.

Our study combines methods both from cognitive science and from computer science. Namely, we reveal the mental representations people have for a given domain and the visual devices they use to convey it, and at the same time, we create effective hardware, which inherits the cognitive style of the designer.

The cognitive theory presented in [6] formulates a number of principles and understandings about human learning. Human memory has two channels for processing information: visual and auditory.

- Human memory has a limited capacity for processing information.
- Learning occurs by active processing in the memory system.

The proposed Split ASM fulfils the above principles; it has a graphical representation, each separated component ASM includes a smaller number of variables (i.e. reduction of complexity). Moreover, since the working memory has a limited capacity [5], the task of designing a SC becomes more difficult and complicated as the number of input-variables increase. The Split ASM allows to reduce the number of variables and hence to use efficiently the short-term memory.

III. EXPERIMENTS

The experiments included eight SC design tasks (Berger counter design, 1-hot code controller and standards benchmarks designs [7]). A group of 43 undergraduate Computer Engineering students were asked to define appropriate ASMs, both in the conventional and in the split form. The resulting designs were compared in terms of their quality and the design time. Additionally, in order to compare the understandability of the two schemes, the students were asked to perform reverse engineering and to find out the functionality that a given ASM represents.

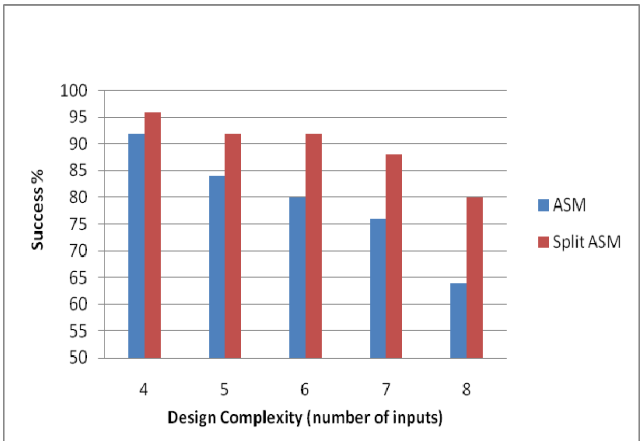


Figure 2. Average success in completion of design tasks.

The comparison between the two representations indicates that:

1) the Split ASM has a shorter average path length than the corresponding conventional ASM and a smaller number of vertices;

2) the Split-ASM is preferable from of the point of the hardware overhead – the Split ASM provides about 20% reduction in the number of gates.

Let a success in solving a given task be the completion of the design in a given time. Fig. 2 shows the relation between the success and the number of input variables for both ASM forms.

A review of the designs and interviews of the students indicate that the majority of students preferred to work with a Split ASM due to its simplicity and efficiency when the number of input variables has increased. Fig. 3 demonstrates this observation.

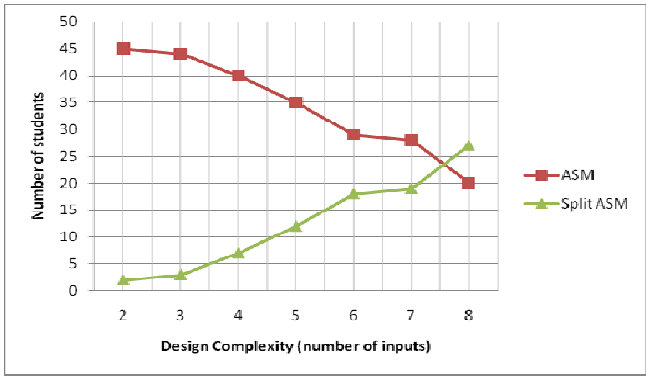


Figure 3. The distribution of the design paradigms as a function of the number of input variables.

As can be seen from Fig. 3, the number of input variables - 7, is the threshold, after which the migration from ASM to Split ASM is inescapable. Our result is well correlated with the known fact that the depth of short-term memory is 7±2 items input (variables input variables in our case) [9].

Fig. 4 shows the relationship between the number of components in Split ASM and the number of input variables.

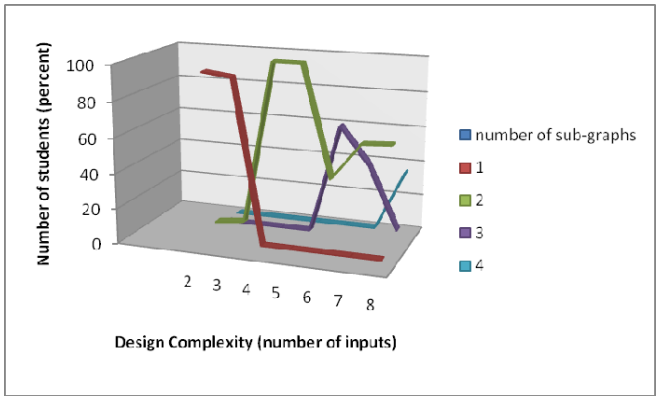


Figure 4. Number of components versus design complexity.

For evaluation of design mental complexity [10] and comparison between ASM and Split ASM, we introduce a

new complexity criterion so called a rectangle of complexity: a product of the Split ASM components number and the average path length. Experiments show that for each initial specification this new criterion in most cases remains constant and does not depend on number of components.

To demonstrate the understandability property of the Split-ASM, we performed an additional experiment in a group of 14 students. The students were asked to find the proper specification of a given implementation, i.e., to perform a reverse engineering. The success rate in performing a correct reverse engineering of a Split-ASM was 70%, while for conventional ASM the success rate was 50%.

IV. CONCLUSIONS

We presented a new approach for teaching SC design. The approach is based on the cognitive-inspired Split-ASM notation. Circuits designed by students according to this approach were found to be more understandable and debugable than the conventional designs. Moreover, the proposed approach allows increasing a size and complexity of SCs that can be handled by students. The effectiveness of the approach was examined on a number of design tasks. The new approach gives a more adequate idea about the digital system's behavior. We believe that the Split ASM approach has good future perspectives in digital design teaching.

REFERENCES

[1] I. Levin, D. Mioduser, A Multiple-Constructs Framework for Teaching Control Concepts, IEEE Transactions of Education, 39(4), 1996, pp. 488-496.

[2] I. Levin, O. Keren, Split Multi-terminal Binary Decision Diagrams. 8th International Workshop on Boolean Problems, Freiberg, 2008, pp. 161-167.

[3] D. Mioduser, I. Levin, Cognitive-conceptual Model for Integration Robotics and Control into the Curriculum. Computer Science Education, Special Issue: Robotics in Computer Science & Engineering Education, 7(2), 1996, pp. 199-210.

[4] S. Baranov, Synthesis for Control Automata, Kluwer Academic Publisher, 1994.

[5] R. Gagne, L. Briggs and W. Wager, Principles of Instructional Design 4th Ed., 1992.

[6] Alan H. Schoenfeld (1987) COGNITIVE SCIENCE and MATHEMATICS EDUCATION, LAWRENCE ERLBAUM ASSOCIATES, PUBLISHERS.

[7] Logic synthesis and optimization benchmarks. – University of California, Berkley, December 1988. – Report NC 27709.

[8] J. Feldman, Minimization of Boolean complexity in human concept learning. Nature, 2000, pp. 630–633.

[9] N. Cowan, The magical number 4 in short-term memory: A reconsideration of mental storage capacity. Behavioral and Brain Sciences, 24, 2001, pp. 97-185.

[10] J. Feldman, The simplicity principle in human concept learning. Current Directions in Psychological Science, 12(6), 2003, pp. 227–233.