Work in Progress - Developing a Cognitively Based Digital Logic Calculator

Joseph Handzik and Geoffrey L. Herman
University of Illinois at Urbana-Champaign, jhandz2@illinois.edu, glherman@illinois.edu

***Abstract* – Cognitive research in digital logic has shown that students struggle to organize their knowledge of concepts and tools into meaningful categories that facilitate problem solving or design. When students learn about digital logic design tools, they must practice using these tools independently and often come to think of these tools as "problems-unto-themselves," rather than as tools that can be used to solve more complicated problems. To help students properly organize their knowledge of digital logic design tools, we propose the creation of a digital logic calculator that helps students learn the purposes and functionality of the different digital logic design tools.**

Index Terms – Digital logic, misconceptions, cognition, learning tools

INTRODUCTION AND BACKGROUND

"Expertise requires well-organized knowledge of concepts, principles, and procedures of inquiry [1]."

Cognitive research in digital logic has shown that students develop poor intuitions about the concepts and procedures they learn in computer architecture and that they struggle to apply their knowledge from one type of problem to another [2],[3],[4]. These results are not surprising as cognitive research has generally shown that students tend to poorly organize concepts and procedures into categories [5],[6]. These categories limit students' ability to access their knowledge or transfer it to new domains. Students can develop more useful cognitive structures through practice, but their attempts to learn these structures are often time consuming and fraught with mistakes. Fortunately, this learning process can be expedited by constraining students to organize their knowledge like an expert [7]. To help digital logic students organize their knowledge like digital logic experts, we propose the development of a digital logic "calculator." We believe that this calculator will enable better conceptual learning and transfer.

When students learn about digital logic, they learn about numerous tools that can aid in the design process. For example, when students learn about combinational circuits, they are taught about Boolean expressions, truth tables, Karnaugh maps, circuit diagrams, and timing diagrams (to name a few). Each tool presents alternate representations of information about a combinational circuit, and each tool facilitates different aspects of the design process. For example, a Karnaugh map primarily facilitates the optimization of the number of gates in a circuit, and the circuit diagram can facilitate analysis of the delays and timing considerations of a circuit. Despite their different roles in the design process, each of these tools is simply a different application of the same core concepts.

Cognitive research in digital logic has shown that students struggle to understand that these tools are simply different applications of the same concepts [2],[8]. Students, rather, classify each of these tools as unrelated problems to solve. Despite being able to solve problems about each of these tools in isolation, students rarely choose the appropriate tool for different phases of the design process, and they fail to transfer their understanding of digital logic concepts from one tool to the next.

DESIGN AND CREATION

We are developing the calculator as a Flash application written in Actionscript 3.0. The calculator presently allows the user to manipulate four digital logic design tools: Boolean expressions, truth tables, Karnaugh maps, and circuit diagrams (See Figure 1 for a screenshot of the tool).

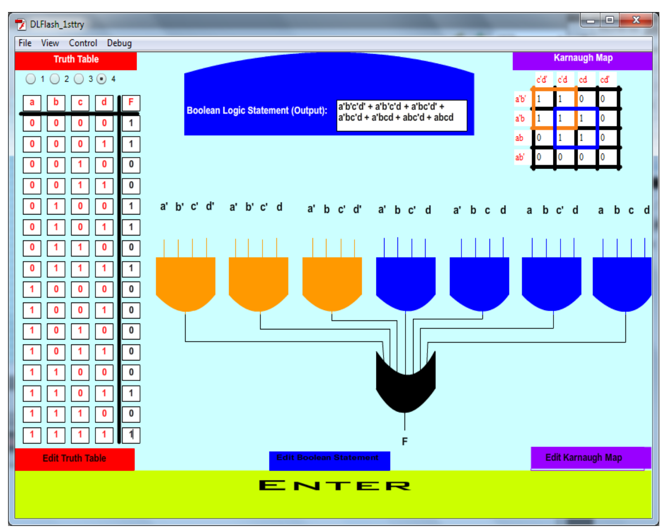

FIGURE 1
SCREENSHOT OF THE DIGITAL LOGIC CALCULATOR

Each tool in the digital logic calculator can be used as an input or output device. The user toggles the input device by pressing the button that corresponds to each tool.

We are designing the calculator to help students learn the purpose of each design tool. We are using color coding and tool-specific outputs to introduce students to the relevant features of each tool.

The functionality of the Boolean expression output box changes to emphasize the purpose of the input tool. For example, if the truth table is used as an input device, the Boolean expression output box will show the user a canonical Sum-of-Products (SOP) Boolean expression (Figure 1 shows one canonical SOP output). However, when the Karnaugh map is used as the input device, the Boolean expression output box will show a minimal SOP Boolean expression. The circuit layout tool reinforces this information.

On the Karnaugh map, all prime implicants will be circled in order to show the user how the expression can be optimized. The circles are coded with distinct colors to allow for them to be uniquely related to parts of a digital circuit or a Boolean expression. Note how the orange and blue circles correspond to the orange and blue logic gates in Figure 1.

We also show the user a 2-level AND-OR circuit, which helps the user develop intuitions about the physical implications of expression minimization. In the future, we plan to allow users to draw their own AND-OR circuits.

Because the calculator is designed to reinforce the relative strengths of the different design tools, Instructors who use the calculator should focus on helping students select the correct tools rather than letting students think that these tools are problems unto themselves.

Instructors should be aware that the calculator will not help students properly organize their knowledge of digital logic tools if the calculator is simply used to solve Boolean logic problems. The calculator is not intended to be a full-scale simulation or design environment and it is not intended to remove a student's need to think. The calculator should be used to introduce the different strengths of the different digital logic tools until students are familiar with each tool and know how each tool can help solve specific types of problems.

Below is a current list of features that are implemented or are in development for the digital logic calculator.

- Input/Output tools
  - Truth table
  - Karnaugh map
  - Boolean Algebra
  - 2-level AND-OR circuit
- Tool-specific functionality
  - Calculator will display prime implicant circling and minimized Boolean expressions when the Karnaugh map is the input tool.
  - Calculator will display canonical SOP Boolean expressions when the truth table is the input tool.
  - Circuit drawing tools.
  - Color coordination between circuit diagram and Boolean expressions when the circuit diagram is the input method.

Future Plans and Conclusions

To improve the calculator, we will use interviews to determine if the layout of the calculator is appealing and intuitive. To maximize learning, the tool must be intuitive so that students can focus on learning about the design tools rather than on learning how to use the calculator. During interviews, students will be asked to carry out common tasks without significant instruction from the interviewer. For example, an interviewer may ask a student to fill out a Karnaugh map with the calculator. These task-oriented interviews will help us identify weaknesses in our interface design and make subsequent targeted improvements. We also hope to learn if students find the tool helpful and how they expect the tool to be used.

In the future, we will attempt to make the tool modular to accomplish two goals. First, a modular version of the tool will give course instructors greater control over how the calculator will be used, because they can disable any piece of the calculator that they do not want students to use. Second, modularity allows for pieces of the tool to be integrated into homework assignments without enabling students to forego valuable practice of manipulating the different design tools.

Ultimately, we believe that the calculator can enable instructors to create assignments that help students properly organize their knowledge of the various digital logic design tools. When students properly organize their knowledge, they will develop better intuitions about how to apply these tools to different design and problem solving situations.

ACKNOWLEDGMENTS

This work was supported by Grants for the Advancement of Teaching Engineering (GATE) from the College of Engineering at the University of Illinois at Urbana-Champaign. This project was developed through the Promoting Undergraduate Research in Engineering (PURE) program at the University of Illinois at Urbana-Champaign.

REFERENCES

[1] J. D. Bransford, A. L. Brown, and R. R. Cocking. *How People Learn: Brain, Mind, Experience, and School*. National Academy Press, Washington, DC, 1999.

[2] G. L. Herman, M. C. Loui, L. Kaczmarczyk, and C. Zilles, "Discovering students' misconceptions in Boolean logic," (in review).

[3] G. L. Herman, C. Zilles, and M. C. Loui, "Flip-flops in` students' conceptions of state," *IEEE Transactions on Education*, (in press).

[4] G. L. Herman, M. C. Loui, and C. Zilles, "Students' misconceptions about medium-scale integrated circuits," *IEEE Transactions on Education*, (in press).

[5] M. T. H. Chi, P. J. Feltovich, and R. Glaser, "Categorization and representation of physics problems by experts and novices," *Cognitive Science*, vol. 5, pp. 121–152, 1981.

[6] P. T. Hardiman, R. J. Dufresne, and J. P. Mestre, "The relation between problem categorization and problem solving among novices and experts," *Memory & Cognition*, vol. 17, pp. 627–638, 1989.

[7] R. Dufresne, W. Gerace, P. Hardiman, and J. Mestre, "Constraining novices to perform expert-like problem analyses: Effects on schema acquisition," *Journal of the Learning Sciences*, vol. 2, pp. 307–311, 1992.

[8] J. T. Longino, M. C. Loui, and C. Zilles, "Student misconceptions in an introductory digital logic design course," in *Proceedings of the 2006 American Society for Engineering Education Annual Conference and Exposition*, June 2006.

AUTHOR INFORMATION

Joseph Handzik Senior in Computer Engineering, University of Illinois at Urbana-Champaign, jhandz2@illinois.edu.

Geoffrey Herman Ph.D. Candidate in Electrical and Computer Engineering, University of Illinois at Urbana-Champaign, glherman@illinois.edu.