Enhancing Teaching Effectiveness and Learning Experience of Digital Circuit Design using Multiple Tools

Dr. Guodong Wang, Massachusetts College of Liberal Arts

Dr. Guodong Wang is an Associate Professor in the Computer Science Department, Massachusetts College of Liberal Arts (MCLA). His research topics include: big data transfer in high-speed networks, Future Network Architecture, Software Defined Networking (SDN), Wireless Sensor Network, and Internet of Things (IoT). He has published over 60 papers in prestigious journals and international conferences.

Dr. Yanxiao . Zhao, Virginia Commonwealth University

Dr. Yanxiao Zhao is an Associate Professor in the Electrical and Computer Engineering Department, Virginia Commonwealth University. Dr. Zhao's research interests include, but not limited to: Internet of things (IoT), 5/6G communications, machine learning, cyber security, wireless energy harvesting, power management and communications in smart grid. Dr. Zhao's research has been supported by NSF, NASA, Air Force and Virginia Commonwealth Cyber Initiative (CCI). Dr. Zhao has published over 80 papers in prestigious journals and international conferences. She was the recipient of the Best Paper Award for three international conferences WASA2009, ChinaCom2016 and ICMIC2019. She has been actively organizing international conferences by serving as TPC chairs, publicity chairs and TPC members. She is an IEEE Senior Member.

Enhancing Teaching Effectiveness and Learning Experience of Digital Circuit Design using Multiple Tools

Abstract

Electronic devices, such as smart home devices, drones, robots, and autonomous cars, have been penetrating various aspects of our daily lives. To meet this growing demand, engineering programs often include at least one electrical engineering course in their curriculum. Digital circuit design is a critical component of computer engineering and a fundamental class for computer science students. The class covers the basics of digital circuits, including Boolean algebra, logic minimization, and binary arithmetic, to provide students with the skills necessary to design, build, and test digital systems. Although digital circuit design is of paramount importance, it is commonly known as a challenging subject for some students, especially those with limited proficiency in mathematical concepts. In this paper, we introduce tools (PyEDA and Logisim) as well as the integration of those tools to enhance the teaching effectiveness and learning experience of digital circuit design. The impact of these tools on student performance is evaluated by analyzing student results from a course where these tools were adopted. The results show that students can significantly benefit from those tools and those tools also make the learning of digital circuit design more enjoyable.

1. Introduction

Digital circuit design is a fundamental course for many engineering majors including computer engineering, electrical engineering and computer science. This class covers the fundamental knowledge of digital circuits, including Boolean algebra, logic minimization, binary arithmetic, circuit analysis and design. Digital circuit design also provides a foundation for understanding the inner workings of computers, which is essential for students who want to pursue careers in computer engineering or related fields after graduation. From taking this class, students will learn how to build digital circuits using basic components such as transistors, gates, and flip-flops, which form the foundation for all digital systems. Students will acquire the necessary skills to design, construct, and evaluate digital systems. They will also gain a deep understanding of how these circuits can be used to solve complex and practical problems, such as performing arithmetic operations or controlling systems. Despite the significance of digital circuit design, the class can be challenging for some students, especially those who have relatively weak mathematical background. Boolean algebra, logic minimization, and binary arithmetic are abstract concepts that require a different way of thinking. Debugging digital circuits can also be difficult, as issues are not always immediately apparent and may require specialized tools and techniques to diagnose. Additionally, understanding the components of digital circuits, such as flip-flops and memory elements may be unfamiliar to some computer science students.

Most universities teach the digital circuit design class in a traditional lecture-based style. Recently there are some efforts to improve its teaching effectiveness. [1] proposed a "flipped" classroom approach, which reverses the traditional teaching method by having students watch video lectures and complete readings and assignments before class, and then using class time for hands-on activities, interactive discussions, and problem-solving sessions. The authors of [2] recognized the challenges that students face in learning complex concepts in digital circuit design and the need for additional support outside of the traditional classroom. They developed a self-study platform that includes a variety of multimedia resources, such as video lectures, interactive simulations, and quizzes, to support students' learning and to provide additional opportunities for practice and assessment. The research of [3] investigated the feasibility of developing the Digital Electronics Practicum Guidance Module with Logisim applications. The study concluded that the module with Logisim was a suitable tool for digital electronics practical exercises. Those methods significantly changed the traditional teaching and learning style in digital circuit design, which sometimes makes it difficult for both teachers and students to adopt those methods. [4] introduced how to use Logisim and the Nexus 2 FPGA platform to facilitate the teaching of digital circuit design class. They found that those tools are beneficial for improving self-efficacy, providing remote hands-on learning opportunities, and honing debugging and troubleshooting skills. Similarly, [5] explored the ongoing development of a digital electronics course that has been remodeled to fulfill the learning objectives of the course involving the use of Logisim circuit simulation software and Basys 3 FPGA. Those two attempts have indicated that students responded well to Logisim, while obtaining FPGA trainer board may be a burden for some colleges or students.

In this paper, we explored the main topics in digital circuit design and their associated challenges in learning this class. We proposed a feasible and effective method which can facilitate teaching effectiveness and students' learning experience. In this method, we adopted two useful tools to help students in their digital circuit design: PyEDA [6] and Logisim [7]. Teachers can use those tools to facilitate students' understanding of abstract concepts in Boolean algebra and circuit design. Students can also use those tools to verify their answers, build circuits, and simulate the functionalities of their circuits. Those tools are especially useful for students who struggle with understanding the abstract concepts involved in Boolean algebra, logic minimization, and binary arithmetic, especially if they do not have a strong background in mathematics.

We used three examples to demonstrate how those tools can help students understand abstract concepts in Boolean algebra and facilitate their circuit design for both combinational circuits and sequential circuits. According to a comparison of students' learning outcomes and their responses, the adoption of these tools greatly speeds up their studies of digital circuit design and enhance their learning experience. Boolean algebra is a complex topic and the design of circuits can present difficulties, however, the utilization of these helpful tools made the learning of digital circuit design more manageable and enjoyable.

2. Three Main Topics in Digital Circuit Design Class.

As an instructor of a digital circuit design course, several key topics are commonly taught. They are: Boolean Algebra and Logic Gate, Combinational Logic Circuits, and Sequential Logic Circuits. Those three topics form the foundation of digital circuit design and provide students with a comprehensive understanding of the principles and practices involved in the design and analysis of digital circuits.

2.1 Boolean Algebra and Logic Gate: This topic focuses on the basic building blocks of digital circuits, including AND, OR, NOT, NAND, and NOR gates, as well as the principles of Boolean algebra and its use in the design of digital circuits. Students will learn how to simplify Boolean expressions, use truth tables to analyze digital circuits, and design digital circuits using Boolean algebra.

Logic gates and Boolean algebra are fundamental concepts in the study of digital circuit design. They present difficulties for students who are learning these concepts for the first time. Boolean algebra and logic gates operate on abstract concepts, such as true and false values, which is difficult for students to grasp initially. Boolean algebra and logic gates are mathematical in nature and require a strong foundation in mathematics to fully understand. Digital circuits can become quite complex, especially when multiple logic gates are used in combination. Understanding the behavior of digital circuits is also challenging for students who are new to the subject. Boolean algebra and logic gates are abstract concepts and how to apply those abstract concepts of logic topics to practical problems is also a challenge for students who are just starting to learn the subject.

2.2 Combinational Logic Circuits: In this topic, students will learn about the analysis and design of combinational circuits, which perform simple logical operations such as addition, subtraction, and comparison. Students will also learn about the design of multiplexers, decoders, and encoders, as well as the use of Karnaugh maps to simplify Boolean expressions and minimize the number of gates in a digital circuit.

It can be a challenge for students to analyze and design combinational circuits. Understanding the behavior of combinational logic circuits is not easy to students. Analyzing the behavior of combinational logic circuits can be a complex task, especially when dealing with large circuits. Students may struggle with understanding how to approach the analysis of combinational logic circuits. Designing combinational logic circuits is also challenging as it involves making decisions about the number of inputs, the types of logic gates to use and the arrangement of the gates.

2.3 Sequential Logic Circuits: This topic focuses on the analysis and design of sequential circuits, which are digital circuits that store and manipulate information over time. Students will learn about the topics of flip-flops, state machines, state tables, and they will learn how to use timing diagrams to analyze and design digital circuits. Additionally, they will also learn about the design of synchronous and asynchronous circuits and the trade-offs between the two.

Sequential logic circuits are digital circuits that store and process information, as opposed to combinational logic circuits that only perform logic operations, which presents more difficulties for students. Understanding the state transition of sequential logic circuits is not an easy task for students. Students may have troubles in understanding how input signals change the state of sequential logic circuits. Debugging sequential logic circuits is also complex, especially when there are multiple inputs, outputs, and states involved. Students are often confused with identifying the root cause of problems in sequential logic circuits.

3. Digital Circuit Design Class with Aids of Useful Tools

In this section, we will analyze the challenges of learning digital circuit design and introduce useful tools to address those challenges and help students learn digital circuit design quickly.

Based on the discussion in Section 2, we find our students have the following difficulties in learning digital circuit design.

1) It is a challenge for students to understand abstract concepts such as Boolean algebra and logic gates, and apply those abstract concepts to practical problems for the first time. Our solution is to use actual circuits to help them learn Boolean algebra and provide a concrete, hands-on approach to the abstract concepts. In particular, we introduce Logisim, an open-source software that can be used to design and simulate digital circuits. It is an effective way for students to learn about digital circuit design, as it provides a visual and interactive interface for designing, simulating, and testing digital circuits.

2) When analyzing and designing complex combinational circuits, students struggle with complex Boolean expression simplification, as well as making decisions about the number of inputs, types of logic gates to use, and arrangement of the gates. For example, they need to simplify Boolean expressions before drawing the circuit. If they have typos in this step, the designing of the circuit will be definitely wrong and they will lose the majority of the points. We address this challenge by introducing PyEDA, which is a Python library for electronic design automation. PyEDA focuses on the automation of the design, analysis, and testing of electronic circuits and systems. It also provides a set of tools for designing and simulating digital circuits, including support for Boolean algebra, truth tables, and Boolean function minimization.

3) Students have difficulties of understanding sequential logic circuits. Sequential logic circuits present more difficulties for students compared to combinational logic circuits. Students have difficulties of understanding the state transition of sequential logic circuits. Debugging sequential logic circuits is also complex, especially when there are multiple inputs, outputs, and states involved. We use the simulation tool in Logisim to help students understand the state transition. Students can achieve precise debugging after designing their circuit using Logisim. They can manipulate the clock frequency of Logisim and even set it to tick only once in order to investigate the status changes that occur at each step.

In the rest of this section, we will use three examples to demonstrate how to integrate those tools into the process of teaching and learning.

3.1 Use Logisim and PyEDA to help students learn Boolean algebra and logic gates.

The AND, OR, and NOT gates are basic building blocks in digital circuit design and play a significant role in the design of various digital systems. Those gates operate based on the principles of Boolean algebra, and a beginner may find it difficult to understand the concepts of Boolean algebra and how they apply to digital logic. Therefore, we use actual circuits to help students learn Boolean algebra and provide a concrete, hands-on approach to the abstract concepts introduced in logic gates and Boolean algebra.

At the beginning, when introducing students the concepts of AND, OR, and NOT, we use circuits to show the associated operations in Boolean algebra as depicted in Fig. 1.

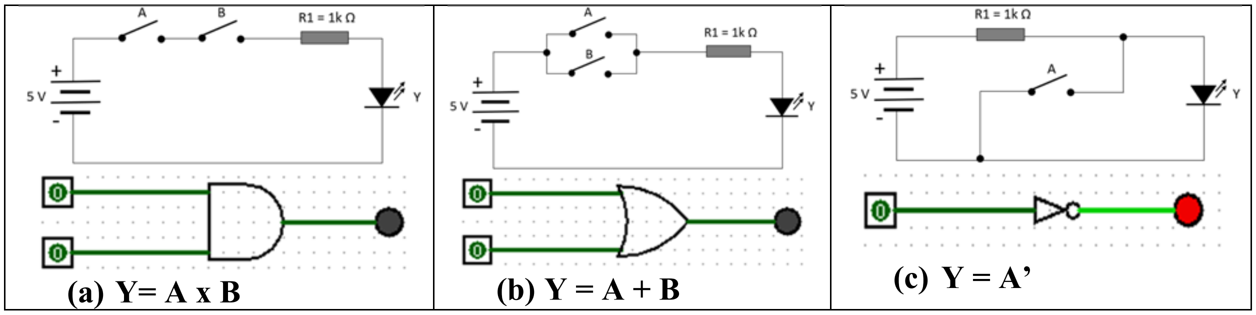
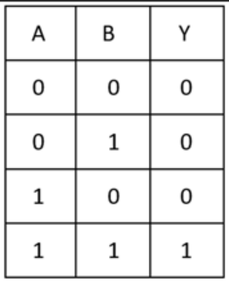


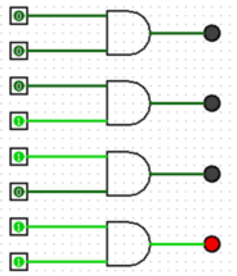
Fig. 1 Circuits showing three basic operations in Boolean algebra.

Take Fig. 1 (a), the AND operation as an example. With the help of the circuit, students can easily understand that the two switches are controlling the status of Y. When and only when both A and B are closed (status 1), the status of Y is on (status 1). After understanding this logic, we introduce the concept of truth table, which is a table used in digital circuit design to represent the output of a logic circuit for all possible combinations of inputs. A truth table lists all possible inputs to a logic circuit and the corresponding output, showing the relationship between the inputs and the output. Then, the truth table as well as the functionality of a two-input AND gate can be simulated and verified through Logisim, as depicted in Fig. 2. By constructing and testing circuits, students are able to gain a deeper understanding of how the logical operations of Boolean algebra are implemented in systems.



A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

(a) Truth table



(b) Verification

Fig. 2 Using Logisim to verify the add operation in Boolean algebra.

Simplifying Boolean expressions is a great challenge for new students, especially as the number of variables and logical operations increase, which makes it hard in finding the most optimal or simplest expression. PyEDA is very helpful for students to verify whether their answer is the most simplified Boolean expression or not. Take the following question as an example: "*simplify the following Boolean expression: (x + y)(x' + z)(y + z)*". The correct answer is xz + x'y as shown below.

(x + y)(x' + z)(y + z)

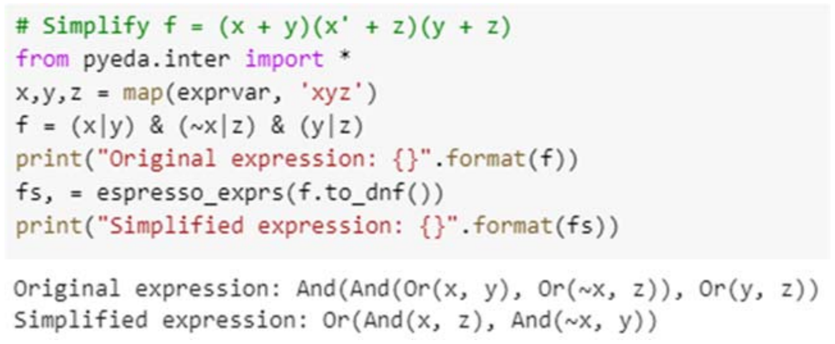
= (x + y)(z + x'y) since (x' + z)(y + z) = (z + x')(z + y) = z + x'y

= xz + yz + x'y (Most students stop here and think this is the most simplified expression)

= xz + x'y + yz (x + x') = xz + x'y + xyz +x'yz = xz + xyz + x'y + x'yz

= xz (1 + y) + x'y(1 + z) = xz + x'y

However, most students just derive: xz + yz + x’y, and stop here. With the help of PyEDA, students can immediately know that their answer is not the most simplified expression yet, which can inspire them to further simplify the express and find the correct one. The example of using PyEDA is depicted in Fig. 3.



```
# Simplify f = (x + y)(x' + z)(y + z)
from pyeda.inter import *
x,y,z = map(exprvar, 'xyz')
f = (x|y) & (~x|z) & (y|z)
print("Original expression: {}".format(f))
fs, = espresso_exprs(f.to_dnf())
print("Simplified expression: {}".format(fs))
```

```
Original expression: And(And(Or(x, y), Or(~x, z)), Or(y, z))
Simplified expression: Or(And(x, z), And(~x, y))
```

Fig. 3 Demonstration of Boolean expression simplification using PyEDA

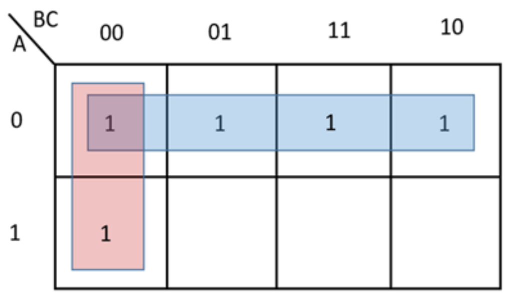
PyEDA can also help students mitigate incorrect simplifications, which may result in a different or incorrect expression, and lead to errors in the final design. This highlights the importance of thoroughly verifying the simplification process and ensuring that the simplified expression is equivalent to the original expression. PyEDA is a powerful tool and can easily conduct those verifications.

3.2 Design a combinational circuit with the help of PyEDA and Logisim.

In this example, we are going to design a simple combinational circuit and show how PyEDA and Logisim help with the circuit design. Suppose we are designing the following circuit. “*Design a combinational circuit with three inputs and one output. The output is 1 when the binary value of the inputs is less than or equal to 4. The output is 0 otherwise.”* There are three main steps in designing this circuit: (a) determine inputs and outputs and derive a truth table; (2) use K-map to find the most simplified expression of Y from the truth table; (3) draw the circuit based on the Boolean expression of Y. The detailed steps are depicted in Fig. 4.

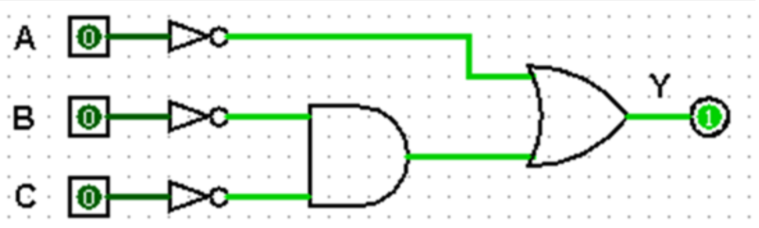
A	B	C	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

(a) Determine inputs and outputs and derive truth table



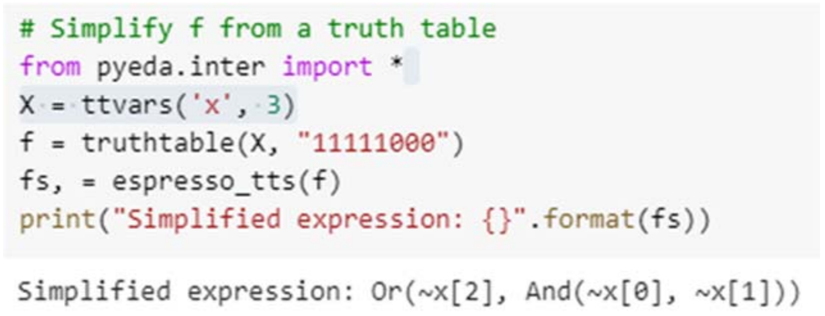
A \ BC	00	01	11	10
0	1	1	1	1
1	1			

(b) Using K-map to find the most simplified expression Y = A’ + B’ C’



(3) Draw and verify the circuit

Fig. 4 The three steps of designing a combinational circuit



```
# Simplify f from a truth table
from pyeda.inter import *
X = ttvars('x', 3)
f = truthtable(X, "11111000")
fs, = espresso_tts(f)
print("Simplified expression: {}".format(fs))
```

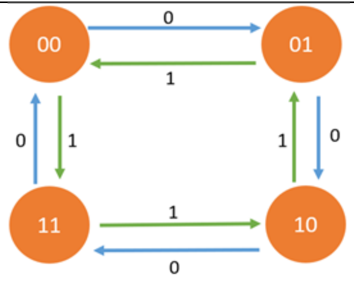
```
Simplified expression: Or(~x[2], And(~x[0], ~x[1]))
```

Fig. 5 PyEDA can directly generate the simplified Boolean expression from truth table

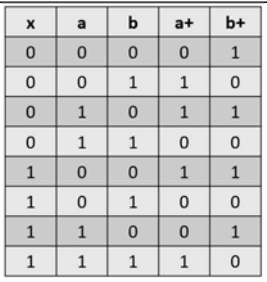
Since this is a relatively easy combinational circuit, a majority of students are able to find out the answer. Even though, students can also benefit from PyEDA and Logisim. First, students can directly use PyEDA to obtain the simplified Boolean expression from the truth table as depicted in Fig. 5. They can compare their result with that from PyEDA and check if the answer is correct before drawing the circuit. Second, they can use the visual and interactive interface of Logisim to draw the circuit, which is more efficient and easier to draw a well-organized circuit, as depicted in Fig. 4 (3). Most importantly, students can conveniently simulate their circuits and check if the designed circuit functions as required. They can also conduct debugging and fix potential issues of their circuit using Logisim.

3.3 Design a sequential circuit with the help of PyEDA and Logisim.

In this subsection, we are using another example to show how PyEDA and Logisim can help design sequential circuits. Suppose we are designing the following circuit: "*Using D latch, design a counter with one input x. If x is 0, this counter will count from 0, 1, 2, to 3 and continues. If x is 1, this counter will count reversely from 3, 2, 1, to 0 and continues.*"

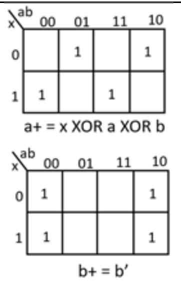


(a) State Diagram

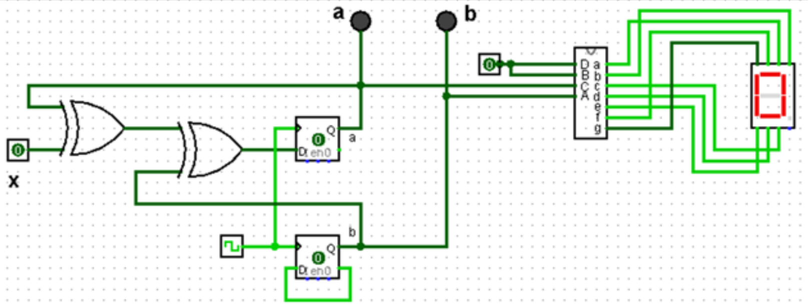


x	a	b	a+	b+
0	0	0	0	1
0	0	1	1	0
0	1	0	1	1
0	1	1	0	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	1
1	1	1	1	0

(b) State Table



(c) State Equations



(d) Circuit Diagram and Simulation

Fig. 6 Steps of designing a sequential circuit

Fig. 6 depicts the steps of designing this sequential circuit: (a) get state diagram based on the descriptions of the question; (b) derive state table from the state diagram; (c) obtain state equations from the state table; (d) draw the circuit from the state table. In this process, we can gain all the benefits of using those two tools when designing a combinational circuit as discussed in Section 3.2. In addition, students can learn more through Logisim. For example, students can check how the input value x controls the counter going forwardly and reversely; students can learn the seven-segment display as well as the driver; students can control the frequency of the clock; students can make the clock just tick once to explore the status changes for each step, etc.

4. Teaching Effectiveness

In this section, we compare students' learning effectiveness in two classes. One adopted with those tools while the other did not. The criteria for student learning effectiveness are summarized in Table I and each criterion has a scale from 0 to 10. The student's learning effectiveness is calculated based on students' self-assessment, homework, quiz and exams. Students' self-assessment weights 50%, and homework, quiz and exams together count the rest 50% when calculating the learning effectiveness.

Fig. 7 compares students' learning effectiveness obtained from these two classes. It can be seen that students with the help of those tools achieved higher points in all criteria than those who did not use those tools. In particular, their capability of simplifying Boolean expressions (criterion 1) had increased significantly because they could use PyEDA to verify their answers. With the help of Logisim, they could implement their circuits by dragging/dropping gates and quickly built the circuit. Logisim facilitated them to draw circuits and enabled them to perform debugging and testing associated functionalities conveniently. Therefore, the students gained higher scores in implementing the design of a combinational circuit using digital logic gates (criterion 6). They could change tick frequencies and even made the clock just tick once to do precision debugging through Logisim, so they gained high points in understanding the importance of clocking in sequential circuit design (criterion 10).

Table I. Selected Student Learning Outcome Criteria

Criteria	Scale (1-10)
1. Be able to simplify and manipulate Boolean expressions using the rules of Boolean algebra.	
2. Be able to simplify Boolean expressions using Karnaugh maps.	
3. Understand the relationship between Boolean algebra and digital circuit design.	
4. Be able to analyze combinational logic circuits using Boolean algebra.	
5. Be able to design simple combinational digital circuits using Boolean expressions.	
6. Be able to implement the design of a combinational circuit using digital logic gates.	
7. Be able to use state diagrams and state transition tables to model and design finite state machines.	
8. Be able to analyze sequential circuits, including flip-flops, counters, and state machines.	
9. Understand the behavior of sequential circuits in response to different input signals.	
10. Understand the importance of clocking in sequential circuit design.	
11. Be able to design simple clocked sequential circuits.	
12. Be able to apply digital circuit design principles to solve practical problems, such as encoding, decoding, arithmetic operations, and digital systems with memory elements.	

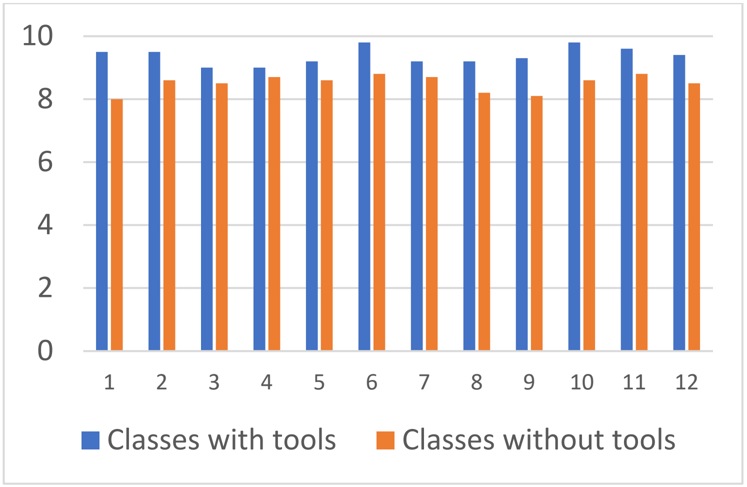


Fig. 7 Comparison of students' learning effectiveness

We have also received many positive comments from our students. Some of them are listed below:

"I really enjoyed the hands-on part of this class. The PyEDA helped me a lot to find out simplified expressions."

"I enjoyed building circuits using Logisim. My favorite project is the forward/reverse counter."

"I have learned a lot from this class. I liked the professor's teaching style. This is one of my favorite classes."

"It was quite an enjoyable experience and I can't wait for more classes with you."

Those comments show that those tools as well as the digital circuit design class not only increased my teaching effectiveness, but also made students be willing to take more classes from me. As of the time spent on learning those tools, we found that it took students about one to two classes for them to get familiar with the PyEDA and Logisim. In addition, the Logisim (logisim-win-2.7.1) does not include a TTL-7447 like seven-segment display driver. Therefore, we need to download associated libraries manually. [8] is one of the options to implement the seven-segment display driver in Logisim.

5. Conclusion

Digital circuit design is a fundamental and important class for computer engineering and an essential class for computer science majors as well. Given the importance of digital circuit design, we have adopted two representative tools and integrated them into our teaching and students' learning process. In this paper, we took three examples to show how those tools help students in learning the three main topics of digital circuit design, including a) Boolean algebra and logic gates; b) design combinational circuits; c) design sequential circuits. Based on the comparison of students' learning effectiveness and their feedback, those tools helped them significantly in the study of digital circuit design. Boolean algebra is an abstract concept and designing circuits is a challenge, while those useful tools make digital circuit design easier and enjoyable to learn.

Reference

[1] K. Yelamarthi, and E. Drake. "A Flipped First-Year Digital Circuits Course for Engineering and Technology Students." IEEE Transaction on Education, pp 1-8, 2014.

[2] D. Baneres, R. Claris, J. Jorba, and M. Serra. "Experiences in digital circuit design courses: A self-study platform for learning support". IEEE Transactions on Learning Technologies, (4):360–374, 2014.

[3] Dewi Dewantara, et al. "Development of Digital Electronics Practicum Guidance Module with Logisim Applications", Journal of Physics: Conference Series, vol 1796, pp 12-19, 2021.

[4] Luković V, et al. "Comparison of the effectiveness of Logisim software tool and remote experiments based on Nexys 2 FPGA platform in learning digital circuits design". The 4th Experiment at International Conference, 2017.

[5] Jeritt Williams and Mohammed Jaby. "Work-in-Progress: Right out of the Gate: Supporting Applied Technology and Engineering Students in Inroductory Digital Logic Courses Using Logisim-Evolution and Basys 3". 2022 ASEE Annual Conference and Exposition, 2022.

[6] "PyEDA" [Online]. Available: https://pyeda.readthedocs.io/en/latest/boolalg.html [Accessed: Feb-12-2023]

[7] "Logisim" [Online]. Available: http://www.cburch.com/logisim/index.html [Accessed: Feb-12-2023]

[8] "Logisim-7-segment-display-driver" [Online]. Available: https://github.com/marceloboeira/logisim-7-segment-display-driver [Accessed: Feb-12-2023]