6-23-2018

# Flipping a Hardware Design Class: An Encouragement of Active Learning. Should It Continue?

Nader Rafla
*Boise State University*

H. Shelton Jacinto
*Boise State University*

# Flipping a Hardware Design Class: An Encouragement of Active Learning. Should it Continue?

**Dr. Nader Rafla, Boise State University**

Dr. Nader Rafla, P.E., received his MSEE and PhD. in Electrical Engineering from Case Western Reserve University, Cleveland, Ohio in 1984 and 1991 respectively. His Doctoral research concentrated on object recognition and localization from multi sensor data: range image, force-torque, and touch. From 1991 to 1996, he was an Associate Professor at the Department of Manufacturing Engineering at Central State University. Where he taught courses was involved in collaborative research with Wright-Patterson Air Force in applied image processing. In January 1997, he joined the newly developed electrical and computer engineering program at Boise State University where he is currently is the chair and an Associate professor. He led the development and starting of the BS and MS programs. He taught several courses and supervised numerous M.S. thesis and Senior Design Project. He contributed to the start of the PhD program and is currently advising three Ph.D. students and two MS students. He also has been conducting research and consultation in R&D for Micron Technology, Hewlett Packard and others. Dr. Rafla's areas of expertise are: security of systems on programmable chips and embedded systems; advanced methods for improving hardware and physical network security; evolvable hardware; and evolutionary and reconfigurable computing. He is a senior member of the IEEE organization and several societies, a member of the ASEE and ACM organizations.

**H. Shelton Jacinto, Boise State University**

H S. Jacinto received his BS degree in electrical and computer engineering from Boise State University, Boise, Idaho, USA, in 2017, and is currently pursuing a PhD in electrical and computer engineering from Boise State University. From 2015 to 2017 he worked with Idaho National Labs conducting research on self-powered wireless sensor networks and their security. From 2016 he now works in the High Performance Reconfigurable Systems (HiPeRS) lab on hardware security. His main research focuses on quantum network hardware cybersecurity, quantum informatics, and adaptive hardware anti-tamper and encryption technologies for use in the field of hardware security.

# Flipping a Hardware Design Class - An Encouragement of Active Learning: Should it Continue?


## Abstract

In this paper we aim to present the lessons learned from flipping the classroom of an entry-level graduate course on digital hardware design. This digital hardware design course uses hardware description languages (HDLs) for programming and requires students to learn relevant concepts and methodologies to successfully design, simulate, synthesize, and verify digital circuits created using hands-on projects and in-class activities. In addition, students in the digital hardware design class get exposure and gain familiarity with an industrial design suite to enhance their knowledge of real-life design cycles.

Typically, students struggle with provided in-class activities, assignments, and projects in any digital hardware design class; even those with prior experience in the field may still struggle due to the complex nature of HDLs. The effort required for large digital designs is often overwhelming and leads to time commitments of several hours outside of the classroom to debug design issues without help from the instructor or teaching assistant(s). This extra time spent shows that help through office hours and recitation sessions is definitely needed to help prepare students.

This work summarizes our explored implications of taking a previously all-lecture-based classroom environment, with traditional teaching methods, and changing to an active-learning environment using multi-modal teaching techniques. In this new teaching environment, students were directed to watch and follow along with short videos produced with the goal of providing instructional and pre-programmed digital hardware design exercises before coming into class. The pre-programmed exercises help change the classroom environment into a center for active-learning through means of creative activities. Students in the active-learning environment are further urged to work in teams on the provided activities to reinforce key concepts and operational ideologies.

From the students' perspectives, our preliminary results show that less time was spent working alone on assignments and projects due to the new active-learning environment, pre-class preparation, and in-class group-work activities. The new active-learning environment included online preview of lecture notes, constant interaction between the instructor, teaching assistant(s), and students, and an increased number of in-class hands-on activities. From an instructional perspective, regardless of drawbacks, the new active-learning environment and teaching techniques allowed for the instructor to reinforce and delve deeper into course content while allowing students to work efficiently with new material. The results from the change to an active learning environment on students' work on assignments and projects during non class-times is: adequate preparation, easy reference to related materials, and an overall wealth of knowledge in the field of digital hardware design.

## Introduction

The ability to design and build large digital hardware circuits requires the use of hardware description languages (HDLs) [1] such as Verilog and VHDL. Both Verilog and VHDL are specialized high-level programming-like languages which are used to describe the structure and behavior of digital circuits and larger digital systems. The major difference between HDLs and other high-level programming languages (i.e. Java) is the concept of concurrency [2], or programmed code running in 'parallel'; a concept that is difficult for even the most experienced hardware developers.

Teaching digital design using HDLs requires the use of modern industrial design suites [3], available for free to students, but require a steep learning curve. In order to master the design tools, design methodologies, and concept application to real-world designs requires a large time investment on both the student's and teacher's part; not only from direct interaction but also from hands-on experience. The necessity of time-based interaction with the hardware and software was lacking when the course materials were taught using traditional teaching methods.

In prior years, digital hardware design was taught using the traditional teaching methods, which posed many basic obstacles such as time constraint and general implementation and debugging issues. In addition, there was a distinct lack of effective oral feedback on best coding practices for certain designs and general ways of improvement. Coming with the new era of active learning, the digital hardware design classroom environment has been flipped [4] by using blended instructional methods, described further in the following sections. The students in the class were surveyed after being taught in an active-learning classroom environment; their responses gave enhanced insight on teaching programming languages using said techniques. An exam was also given at the conclusion of each individual language being taught in order to assess the students' comprehension of concepts and understanding of their application to design.

## Core Structure in the Active-Learning Mode

Two different languages, VHDL and Verilog, are taught to entry-level graduate students in one 16-week semester. The objective of the digital hardware design course is to teach students the methods and techniques necessary to facilitate design, simulation, and synthesis of complex digital hardware using HDLs. The design, implementation, and testing of digital circuits on a re-configurable hardware platform is discussed along with the functional verification of the design to the typical enrollment of around 20 students, once per year, during the fall semester.

In the new active-learning mode posed to the classroom, and for each language [5, 6], students were provided with lecture notes, short recorded videos, in-class activities, and a final project. The students were expected to review the upcoming class' lecture notes and watch the short video(s) prior to coming to class. In class, the students were expected to brainstorm, ask questions, and perform specific activities to reinforce the key concepts presented in the posted online notes and videos. To further aide in student understanding and learning during class meetings, the instructor and two teacher-assistants were constantly present to help students debug in-class activities, resolve tool issues, and help students with difficulties faced during

implementation on the chosen Zybo board platform [7]. Following class time, the students should be able to complete the prescribed activity with all knowledge necessary.

In order to maximize student-learning benefits and efficiently use classroom time, students had to perform the following two activities prior to each class period:

1. *Watch instructional videos* – The first set of videos published to the students were designed to be a tutorial covering the usage of the software tools through simple worked examples. The videos explained the design suite setup, features, and commands frequently used for class activities and design projects. There were a total of 5 videos, in the range of 15 minutes each, that the students were instructed to complete watching before the beginning of the second week of classes. This "startup" time was necessary to reduce the total time and overall difficulty involved with using the tool-set provided. The remaining videos of the semester were generally in the range of 10 minutes each and contained instructions and examples for designing, simulating, and synthesizing basic components common to many digital circuits; useful for in-class activities and projects by providing a solid foundation of conceptual ideologies. The videos additionally served to explain, in general, the best practices and techniques of hardware design, and educate the students on potential pitfalls one might encounter.
2. *Read lecture notes* – The intent of providing lecture notes to students is to explain syntactical nuances inherent in hardware design and to further reinforce the concept of concurrent design. From the student's point of view, traditional lecturing on issues like syntax can be underwhelming and not useful. The notes also provide insight into how written code translates into hardware circuitry. The lecture notes are an easy and quick reference which include best practices for programming HDLs; developed as a short-hand version of the traditional in-class lectures.

During class-time, students often worked in groups of two on simple activities designed for a twofold purpose: first, to expand on what concepts are shown and explained in the videos, and second, to target creation of complete designs of sub-components to be used in future projects. The purpose of designing sub-components for use in future designs is to reinforce the common hardware design technique of "design for re-use". For example, the students may watch a video showing the design and test of a simple flip-flop which is then used in an in-class activity focusing on the design and implementation of a universal shift register composed of flip-flops. As another example, a video may show the best practices of design of a finite state machine (FSM) which is then followed by an in-class activity detailing the use of FSMs to control an OLED display [8] connected to a hardware FPGA board. After students finished with activities, their work was submitted onto BlackBoard for grading; work submitted was graded and returned with feedback in a timely manner. When work was reviewed, common mistakes were addressed at the beginning of the following class period and/or discussed individually with students. To reinforce solutions to any common problems or for further support on students' designs, recitation sessions were held once a week by teaching assistants. The recitation sessions served to allow for individual student questions, resolution of hands-on design issues, and to check-off their hardware implementation of designs.

One project was given, with ample time allotted, at the conclusion of each of the two languages

taught. Students were tasked to conduct group brainstorming exercises related to the design projects and were expected to complete their work individually. A final project was given at the end of the course related to the implementation of the cryptographic hash function MD-5 [9]. The MD-5 project was extremely time consuming and relied heavily on concepts learned throughout the course but proved useful to showcase a "large" design in comparison to the simplistic in-class activities.

## Assessment

Literature discusses several methods of assessment for flipped classrooms [10, 11, 12] however, the assessment of programming classes in literature is based majorly on survey results [13, 14, 15, 16], whereas flipping programming classes is only done to help facilitate introductory-level undergraduate courses.

Due to the nature of this class being graduate level and the complication of working with two separate languages, VHDL and Verilog taught consecutively in a single semester, a specific non-numeric survey was developed and distributed to students after finishing each language. At the conclusion of the first language taught, all survey comments were reviewed and any necessary changes were put in place. Changes after the first survey included modification of contents and delivery methods of the material required for the second language. A formal evaluation of the class was conducted at the end of the semester using numeric scales. The formal evaluation focused on the overall effectiveness of the course, the instructor's performance, and other comments related to future course improvement. Survey questions are shown in Table 1.

Table 1: Non-numerical survey questions asked to students about flipping the class.

| # | Question |
|---|---|
| 1 | How did the videos, lecture notes, and collaboration during the class meetings help you learn the material? |
| 2 | What additional teaching aid materials or teaching style were you expecting to see that can improve the pace of your learning? |
| 3 | Were the in-class activities (ICAs) helpful in learning the material? If so, explain in what way? If not, how would you prefer this to be changed? |
| 4 | Did you work in a team during ICAs? If so, how did that affect your learning? If not, why not? |
| 5 | Were the recitation sessions helpful? How did you feel about the frequency and the length of time allocated for them? What kind of changes would you like to see? |
| 6 | Was the presence of TAs helpful? What were they helpful for? |
| 7 | Are there other things would you like to see the TAs do during class? |
| 8 | Are there other things would you like the instructor be doing during the flipped class? |
| 9 | What improvements would you suggest for future offering of this flipped class? |
| 10 | Describe the amount of effort you have been putting in at home? What aspect was most time consuming? |

In order to assess the effectiveness of the flipped classroom environment on student learning, a formative assessment was able to provide us with frequent and immediate feedback. Accordingly, we varied the proportioning of traditional in-class lectures, videos, and in-class activities between teaching the first and second design languages. To control the many variables affecting the study, and to assess student performance in the course, a summative assessment was conducted[1]. To accomplish our assessment, two similar exams were given after teaching each language. The two exams targeted the same key concepts, present in both languages, and were graded using the same rubric. The exams contained questions relevant to the digital hardware design course: hierarchical components, design concepts, concurrency and sequential behaviors, simulation, synthesis, and implementation. The letter grade distribution for each of these exams is shown in Table 2.

Table 2: Grade distribution on exams given after the first and second HDLs were taught.

| Grade | After First Language | After Second Language |
|---|---|---|
| A | 5% | 30% |
| B | 80% | 65% |
| C | 15% | 5% |
| D | 0% | 0% |
| F | 0% | 0% |

As seen from Table 2, a formative assessment and it's feedback had a tremendous impact on grade distribution and student performance. The students scored significantly higher after learning hardware design skills using HDLs in a blended learning environment.

**Analysis of Survey Results and Lessons Learned**

All students enrolled in the hardware design class completed an anonymous survey in exchange for an offered incentive. The results of the survey were collected, itemized, and analyzed, leading to the following conclusions listed in Table 3, where the percentage indicates students in agreeance with the item.

From the results in Table 3, some conclusions can be drawn about the change in methodology between languages taught. The change from 63% to 72% agreeance for item 1 shows that as the difficulty of complex design problems was increased and prior knowledge of hardware increased, the sudden change to another programming language threw off many students with regards to understanding newly seen syntactical errors. However, in relation to item 2, the agreeance rate of 87% increased to 95%, showing that students were able to learn better from the videos on the new language. By watching the videos, students were able to follow along better; presumably due to students settling in with the flow of the new flipped classroom environment.

As a result of students not fully complying with pre-watching the video materials for the second language taught in the class, many students stopped utilizing the lecture notes available online during in-class assignments. The change in lecture note usage is shown by the results for item 3

[1]The assessment itself is beyond the reach and focus of this work.

Table 3: Numerical results of changes in teaching method between first and second language.

| Item | After first language | After second language |
|---|---|---|
| (1) Students had difficulty adapting to the flipped classroom environment when trying to fix syntax errors. | 63% | 72% |
| (2) Students enjoyed the short videos and the coherence of material presented to them. Videos helped students successfully complete the in-class activities. | 87% | 95% |
| (3) Students referred to class notes while doing the in-class activities rather than reading them prior to class. | 85% | 45% |
| (4) Students preferred to have some traditional lectures in addition to the flipped classroom. | 78% | 83% |
| (5) Working in teams sped up students' learning process – Finding a common time is a major factor. | 87% | 92% |
| (6) Students utilized recitation sessions – Time conflict was a factor. | 50% | 50% |
| (7) Students thought the presence of the teacher assistants were helpful – This was adequate resources. | 73% | 86% |

with an 85% agreeance rate for the first language taught, then a sharp decrease to a 45% agreeance rate for the second language taught. Another reason for the drop in agreeance for item 3 could be the inclusion of some intensive in-class lectures given for the second language to help get students up-to-speed quicker than for the beginning of the class. Item 4 reflects the agreeance rate for students preference of traditional lectures. The rise of preference for traditional lectures increased from 78% to 83%, not very dramatic but, a preference towards in-class lecturing; helping students learn new materials quickly once they have learned the basics of digital hardware design and the nuances of discovering and pointing out key issues and aspects of digital hardware design.

The following items (5 through 7) relate to teamwork, recitation session timing, and presence of teaching assistants in addition to the instructor, respectively. Students did find that teamwork helped them complete in-class activities more efficiently and further reinforced learned material for the second language with an increase of 5%. Most likely the small increase in agreeance for teamwork activities providing a positive impact was due to the quicker learning that can be accomplished for the new second language taught. Recitation session timing appeared to be a constant throughout the semester with a 0% change. Employing two recitation sessions at differing days and/or times could potentially give more flexibility to students but, the students' responses to question 5 in Table 1 indicated that those that have utilized recitation sessions were happy with the feedback and results while others could not attend due to time conflicts. Students found the availability of the teaching assistant(s) in class a better resource once the new language was taught and assignments were becoming more difficult, with a 13% change. The change in percentage for availability of the teaching assistant(s) increased especially once new methods of time distribution between teams was established and common problems were fully understood.

**Should a Flipped Hardware Programming Classroom Continue?**

The early insights gained into flipping a digital hardware design class indicate that it is better to not flip the classroom environment for all parts of course teaching, according to student responses to survey question 2 in Table 1. It is of paramount importance that a way should be devised such that the instructor is able to ensure that students have fully watched and understood videos and/or lecture notes posted prior to the start of class. In the implemented efforts to flip the digital hardware design class, observations were made that some students would attempt to watch videos while simultaneously completing the in-class activities instead of watching videos prior to class, as indicated by student responses to question 1 in Table 1. The problem of not watching videos prior to class-time could be remedied by administering a quick, 5 minute or less, online quiz covering the basic topic which each video covered. Another possible solution, utilizing BlackBoard's tracking feature, is to monitor student activities online with the course content. Monitoring content access could be implemented where each student's portion of their grade would depend on them being active with the videos and/or lecture notes.

Learning the concept of concurrency and concurrent methods for hardware design has several aspects: *a*) Learning language syntax and semantics, *b*) understanding development of algorithmic strategies for design prior to coding, *c*) learning the ability to debug and fix syntactical errors, *d*) ensuring that desired, synthesized, hardware matches code programmed, and *e*) perception of steps necessary to finally implement the design into hardware. Unfortunately, not all of these aspects can be taught using videos and/or class notes due to the hands-on nature of digital hardware design. The time that students spend in the classroom should be used for items *(d)* and *(e)* above, instead of items *(a)* through *(c)* which can be easily accomplished by following simple instructions given in videos and/or lecture notes. A change is required in the methods that the instructor and teaching assistant(s) follow to help students during in-class activities. Many students were observed asking very similar questions which, when answered on an individual basis, proved to be quite time consuming. Answering questions may prove to be more efficient if the questions were grouped into similar categories and answered to the entire class at once.

The model of a flipped classroom provides a golden opportunity to increase instructor interaction with students, allowing for instant assistance or feedback on their work. Flipping the classroom also provides a great opportunity for nurturing an active-learning environment rather than the typical passive-listening environment seen with traditional lectures; further increasing student learning and enhancing a student's self confidence in the topic of digital hardware design. The flipped classroom therefore can provide ample learning opportunity to those students who wish to take advantage of all available resources.

In conclusion, our preliminary experience of flipping a graduate classroom for teaching digital hardware design and accompanying HDLs indicated a positive student outlook towards the approach. However, a flipped classroom environment is only good for some aspects of teaching within a hardware design cycle, from developing code to testing a working system. There are still some challenges which need resolution before teaching digital hardware design in the future, including the need to guarantee that students are well-prepared and ready for in-class activities prior to coming to class. Verifying that students are well prepared for class time is difficult at best, where generally a student will not watch videos or read lecture notes prior to class-time,

necessitating the need for a quick survey method to ensure compliance with before-class activities. Future plans for the flipped digital hardware design course include the development of a strategic method to survey students on prior knowledge of HDLs, knowledge gained from videos and/or lecture notes, and shortening in-class activities to be sure that they can be accomplished fully within the allotted class time. An idea to ensure students are prepared to complete the in-class activity, in class, is to have them contemplate design methodology prior to coming to class. In addition, time should be dedicated to address common problems that previous students have encountered with the in-class activities and provide advice on possible pitfalls at the beginning of each class. Students should continue to work individually on projects while dedicating parts of several class periods for interaction between the instructor and/or teaching assistant(s).

## References


[1] R. Boute, "Fundamentals of hardware description languages and declarative languages," in *Fundamentals and Standards in Hardware Description Languages*, pp. 3–38, Springer, 1993.

[2] S. A. Edwards, "Tutorial: Compiling concurrent languages for sequential processors," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 8, no. 2, pp. 141–187, 2003.

[3] Xilinx Inc., *Vivado Design Suite: User Guide*, January, 2017.

[4] J. L. Bishop and M. A. Verleger, "The flipped classroom: A survey of the research," in *ASEE National Conference Proceedings, Atlanta, GA*, vol. 30, pp. 1–18, 2013.

[5] L. Murphy, K. Blaha, T. VanDeGrift, S. Wolfman, and C. Zander, "Active and cooperative learning techniques for the computer science classroom," *Journal of Computing Sciences in colleges*, vol. 18, no. 2, pp. 92–94, 2002.

[6] A. Herala, E. Vanhala, A. Knutas, and J. Ikonen, "Teaching programming with flipped classroom method: a study from two programming courses," in *Proceedings of the 15th Koli Calling Conference on Computing Education Research*, pp. 165–166, ACM, 2015.

[7] "Zybo Z7: Zynq-7000 ARM/FPGA SoC Development Board," tech. rep.

[8] "Pmod OLEDrgb: 96 x 64 RGB OLED Display with 16-bit color resolution," tech. rep.

[9] R. Rivest, "The md5 message-digest algorithm," 1992.

[10] R. Talbert, "Four Assessment Strategies for the Flipped Learning Environment," *Faculty Focus | Higher Ed Teaching & Learning*, Aug. 2015.

[11] P. V. Roehling, "Assessing the flipped classroom," in *Flipping the College Classroom*, pp. 115–133, Springer, 2018.

[12] M. W. Redekopp and G. Ragusa, "Evaluating flipped classroom strategies and tools for computer engineering," in *Proceedings of the Annual Conference of the American Society of Engineering Education*, 2013.

[13] W. Puarungroj, "Inverting a computer programming class with the flipped classroom," in *Proceedings of the International Conference on eLearning for Knowledge-Based Society*, pp. 40–1, 2015.

[14] C. P. Rosiene and J. A. Rosiene, "Flipping a programming course: The good, the bad, and the ugly," in *Frontiers in Education Conference (FIE), 2015 IEEE*, pp. 1–3, IEEE, 2015.

[15] N. Sarawagi, "Flipping an introductory programming course: yes you can!," *Journal of Computing Sciences in Colleges*, vol. 28, no. 6, pp. 186–188, 2013.

[16] E. Breimer, M. Fryling, and R. Yoder, "Full flip, half flip and no flip: Evaluation of flipping an introductory programming course," *Information Systems Education Journal*, vol. 14, no. 5, p. 4, 2016.