AC 2011-2475: TEACHING DIGITAL SYSTEMS VERIFICATION METHODOLOGIES USING SYSTEMVERILOG

Nader Rafla, Boise State University

Dr. Nader Rafla, P.E. received his MSEE and Ph.D. 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 range image data, force-torque, and touch sensors data.

From 1991 to 1996, he was an Associate Professor in the department of Manufacturing Engineering at the Central State University. Where he taught courses related to the electrical engineering component of the program. In the mean time, he developed and was involved in a research program 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 an Associate professor and chair of the Electrical Engineering Department. He led the development and starting of the MS of Computer Engineering; He taught several courses and supervised numerous MS thesis and Senior Design Projects. He also has conducted research and consulted in R&D for Micron Technology, Hewlett Packard and others.

Dr. Rafla's area of expertise is systems on a programmable chip and embedded & microprocessor-based system design; Evolvable and self-reconfigurable systems; and implementation and hardware architectures of digital image and signal processing algorithms applied to recognition, identification, inspection, automation and control. He is a senior member of the IEEE organization and several societies and a member of the ASEE organization.

Teaching Digital Systems Verification Methodologies using SystemVerilog

Abstract

With the growing complexity of modern digital systems and embedded system designs, the task of verification has become the key to achieving the faster time-to-market requirement for such designs. This paper describes a graduate level, Verification of Digital Systems using SystemVerilog, offered at Boise State University as a part of the Master of Science program in Computer Engineering,. This course does not only teach syntax and semantics but also coverage-driven, constrained-random, and assertion-based verification methodologies employing the advanced features of SystemVerilog to ensure that designs meet the required specifications. The course also emphasizes the practical aspects of verification methodologies through providing students with hands-on experience on commercial verification tools such as QuestaSim, the Advanced Functional Verification suite from Mentor Graphics. Course goals are explained along with course content, format, and benefits to students. The course is designed around small practical exercises to illustrate the main concepts, tools, and language usage. A mid-term and a final project are also offered that require an automated verification environment to be designed and tested on given designs.

Introduction

Complex and large-sized modern System-on-Chip (SoC) designs need to be comprehensively verified to ensure their correct functionality and to reduce design cycle time. The process of verification consumes from 60% to 80% of the design cycle which is expected to continually increase [1]. Traditional verification methods struggle to keep pace with the ever-increasing size and complexity of designs. The only way to address this problem is to adopt a reuse-oriented, coverage-driven verification methodology. Such a method should be built on the rich semantic support of a standard language. The SystemVerilog hardware design and verification language standard [2] offers high-level programming and object-oriented capabilities with which a more efficient and realistic verification environment can be developed. SystemVerilog is not only a new standard language for describing complex structures, but also provides a platform for developing automated and advanced verification environments. In addition to the standard testbench development structure, the language includes built-in verification functionality such as constrained randomization, assertions, and coverage monitoring. These additional eatures allows engineers to develop sophisticated test scenarios with very few lines of code. Using the Open Verification Methodology (OVM)[3] and others described in the Verification Methodology

Manual (VMM)[4] enable the creation of reusable modules and IPs that communicate with each other via transaction-level interfaces.

This paper describes the contents of the course “Digital Design Verification” taught at Boise State University. It explains the course goals, infrastructure and projects, along with the verification tool that is widely used by industry.

Course Objectives

The purpose of this course is to provide students with the knowledge necessary for understanding what needs to be verified, how to design a verification environment, and how to analyze and interpret the results. It provides a strong foundation for verification techniques through lectures and practical exercises covering the many aspects of the verification process, tools, and methodologies. Therefore, the objectives of the course are

- To provide students with a good understanding of verification methodologies and tools
- To prepare students for research in the area of digital systems verification
- To encourage students explore this area o research
- To help students develop research topics or their theses and dissertations

Course Outline

This course is taught over a period of fifteen weeks (three 50 minute periods per week). Each week consists of a lecture (one period) and one practical exercise (two periods). Students work individually on these projects during the first lab period and continue at home then compare and discuss the results of their verification techniques during the second lab period. In addition, there is a midterm project and a major final project. To achieve a passing grade, students must complete all practical exercises (50%) and finish and demo their projects (20% for the mid-term and 30% for the final). Two textbooks [5,6] are used in this course along with lecture notes developed by the author. The language reference manual is also heavily used by the students. Students also are required to select and present a recently published technical paper about application of verification methodologies or other verification environments. The course is divided into five parts as explained next.

The first part of the course provides an introduction to the art of verification and testbench architectures. A simple calculator design [7] is given and students are asked to design and develop a verification plan and testbench to verify the design functionality.

The second part introduces the main semantics of SystemVerilog such as structs and unions, data types and arrays, modules and processes, and interfaces. Three practical exercises are given to enforce the understanding of these language features.

The third part discusses assertions. Two practical exercises are given. In the first one, students are asked to write and practice with simple assertions for a given design and explore the assertion features of the QuestaSim simulator [8]. The objective of the second exercise is to allow students practice writing properties and sequences to verify a given design that is more complex than the previous one. In the mean time, students are instructed to write and use simulation scripts.

The fourth part of the course deals with different verification methodologies and the use of clocking blocks, programs, and the process of automation. Exercises of this part concentrate on developing bus functional models including calling tasks and functions. They also introduce the idea of transaction-based testbenches.

The fifth and final part introduces the object-oriented features, constraints, coverage and other advanced features. Exercises ask for developing a test harness environment that contains the instantiation of the device under test and two bus functional models, including interfaces, for inputs and outputs. This environment is a class-based simulation with stream generation and input data randomization using given constraints.

Course Projects

The two given projects are used to assess students' understanding of the design methodology used in the development of their verification environment. They challenge them in two aspects: 1) the given designs under verification are complex enough to prevent students from testing all possible combinations of inputs. This forces them to develop a test plan, use constraint randomization, and perform coverage analysis. 2) Intentional bugs are purposely inserted in the designs to ensure that students would use specific verification methodology. The design code is made transparent to the students and the only way to find bugs is to develop a meaningful and functional verification environment.

The mid-term design project is an ALU design based upon the exercises of part one. This ALU perform the basic logic functions (AND, OR, NOT, and XOR) and the arithmetic operations (ADD, SUB, Shift right, and Shift left). Selection of the performed operation is selected by a control word given to the ALU. The design is restricted to the execution of one logic function and one arithmetic operation at any specific instant in time. Timing specifications for inputs, control, and outputs are given in form of a waveform. Students are to develop a test plan and test cases for all operations to determine where the bugs may be at. After finding each bug, a fixed design is given to students and they are asked to develop a strategy for verifying interesting corner cases and to provide the reasoning behind selecting each case.

The final project asks the students to develop a sophisticated class-based verification environment employing all the features discussed in parts four and five of the course. This environment is requested for a design of a CPU of an 8-bit MIPS processor with memory interface. There are no bugs given for this project but rather a test plan is provided to guide the students through the verification environment development. Assertions, constraint randomization, coverage analysis, and transaction streaming techniques are also required to be

employed. Each student presents his/her verification environment to the class for discussions and criticism. A rubric is used for the assessment of different aspects of the project.

Course Benefits and Assessment

This course is an important course to graduate students interested in computer engineering and play a vital role in stimulating their interest to perform research in the area of hardware verification techniques and environment. Through lectures, readings, and working with practical designs, students learn the pros and cons of different verification methodologies. Each time the course is offered, its contents change to reflect the new trends in industry including any new simulation tools or features.

After successful completion of the course, students are expected to get a well understanding of several verification methodologies and the techniques used in developing portable and reusable modules common to different verification environments. The practical exercises and projects teach the best-practices used for verifications using state-of-the-art commercial tools.

An additional and major benefit of this course is to encourage students to explore the research area of hardware verification and to help them develop research ideas in this area. Some students who took this course came to me asking for Thesis topics while others develop enthusiasm to start their doctorate studies.

The course was offered During the Spring Semester of odd years. It had been offered twice so far. A total of ten students successfully completed the course, two of them (20%) are working on their Master's thesis in the area of digital systems verification and one other MS student (10%) is using the knowledge gained from this course in his thesis. A student working on his Ph.D. dissertation under my supervision (10%) is developing a new methodology to verify the functionality of a new memory architecture as a part of his dissertation. Four graduate students (40%) took the course out of their own interest to gain knowledge of the field as a part of their non-theirs Master degrees. The rest of the students (20%) are undecided.

In conclusion, the course contents and structure (lectures, practical exercises, and projects) has met its goals. However, there exist a steep learning curve and it is a continuing struggle to achieve high level of quality while covering such variety of topics. It is rewarding to see students going over many stumbling blocks and reach the point of being able to develop a meaningful automated verification environment for SoC. This course represents a good example for connecting teaching to research.

References

[1] Abraham, J.A.; Saad, D.G.; *"Tutorial T4A: Formal Verification Techniques and Tools for Complex Designs,"* International Conference on VLSI Design (2007).

[2] *"SystemVerilog 31a. Language Reference Manual (LRM),"* Accellera (2004).

[3] *"Open Verification Methodology (OVM) Reference Manual,"* Ver. 2.1, Mentor Graphics (2008).

[4] Bergeron, J.; Cerny, E.; Hunter, A.; Nightingale, A; *"Verification Methodology Manual for SystemVerilog,"* Springer (2006).

[5] Spear, Chris*; "SystemVerilog for Verification: A Guide to Learning the Tetstbench Language Features,"* Second Edition, Springer Publishin (2008).

[6] Vijayaraghavan, Srikanth; Ramanthan, Meyyappan *; "SystemVerilog Assertions: A Practical Guide,"* Springer Publishing (2005).

[7] Wile ,Bruce; Goss, John; and Roesner , Wolfgang; "*Comprehensive Functional Verification: The Complete Industry Cycle*," Morgan Kaufmann Publishers (2005).

[8] "*QuestaSim Reference Manual,"* Mentor Graphics Corporation, Version 6.3C (2009).