International Journal of Electrical and Computer Engineering (IJECE)
Vol. 14, No. 6, December 2024, pp. 6195~6203
ISSN: 2088-8708, DOI: 10.11591/ijece.v14i6.pp6195-6203

Effective ethernet controller protocol architecture verification strategy using system Verilog

Shubha Parameshwara, Ganapathi Vithoba Sagar, Hamsa Rekha Sorekunte Dasappa, Sheetal Nagaraj
Department of Electronics and Instrumentation Engineering, Dr. Ambedkar Institute of Technology, Bangalore, India

Article Info

Article history:

Received Feb 29, 2024
Revised Aug 5, 2024
Accepted Aug 14, 2024

Keywords:

Design under test
Design under test
Ethernet controller
Layered verification architecture
Pre-silicon
Verilog language

ABSTRACT

The pre-silicon verification is typically more significant than post-silicon verification, which produces an algorithm with the correct functionality and timing parameters. In this paper we propose innovative pre-silicon verification methodology focused on the Ethernet controller architecture as the design under test (DUT). The methodology employs a layered verification architecture implemented using the system Verilog language, aiming to streamline the testing process. A novel test pattern test generator, interfaces and blocks are used to perform the verification. The test patterns are generated based on the operational principles of the ethernet controller block, ensuring comprehensive verification coverage. Additionally, the paper combines different verification parameters with existing approaches to demonstrate the effectiveness of the proposed methodology. It is observed that the performance of the proposed method is better compared to existing methods.

Corresponding Author:

Shubha Parameshwara
Department of Electronics and Instrumentation Engineering, Dr. Ambedkar Institute of Technology
Bangalore-560056, Karnataka, India
Email: shubhap.ei@drait.edu.in

1. INTRODUCTION

In every newly established architecture or algorithm, design verification is crucial since it verifies many design factors that results in the right operation of that block, hence ensuring the algorithm’s correctness. These ideas also apply to any very large-scale integration (VLSI) design, where the verification techniques fall primarily into two groups: post-silicon verification and pre-silicon verification. Code coverage, functional coverage, and other techniques are used to verify the software level design, or soft core, in pre-silicon verification. Post-silicon, on the other hand, carries out the same function on the manufactured chip (hard core).

Pre-silicon verification is typically more significant than post-silicon verification, which produces an architecture or algorithm with the correct functionality and timing parameters. In this paper, we suggested a practical method for confirming the ethernet controller’s VLSI architecture. The ethernet controller is verified using the standard system Verilog verification approach, and the standard controller model is utilized to construct the test pattern and calculate the functional coverage. It is therefore feasible to achieve 100% coverage. The methodology makes use of a comprehensive strategy to produce test patterns that cover every aspect of the design under test (DUT) block’s functionality, greatly increasing the scope of verification. This method provides robustness in identifying possible design defects and corner cases by capturing the subtleties of DUT functioning, thereby increasing the verification process’s reliability.

A technique for field programmable gate array (FPGA) testing of enhanced edge detection employing buffer lines and separation was presented by Peng *et al.* [1]. The authors looked into how well an improved edge detection method worked on an FPGA platform. Prewitt filter, separation and buffer line, and grey scale and Gaussian filter were the three components that made up the algorithm. The authors used buffer line and separation techniques to improve edge detection speed. They tested the picture pre-processing stage on an FPGA platform, which produced a better image as the output of the image processing technique.

For logic verification, Zou *et al.* [2] presented a system-level FPGA routing approach based on time division multiplexing. The authors sought to reduce runtime as well as the maximum time-division multiplexing (TDM) ratio (signal multiplexing). They structured the net routing problem as a Steiner minimum tree (SMT) problem and took into account significant ratio constraints. They used an approximation approach with a performance bound to solve it. Additionally, they proposed a novel strategy for reassigning ratios in unbalanced net groups and a refinement approach that takes into account the ratios to improve routing performance.

Deverajegowda *et al.* [3] presented a formal verification approach in an industrial context. The authors provided a useful method for applying formal verification techniques to industrial designs while taking productivity, efficiency, and quality into account. The suggested methodology tackles several facets of formal verification in a methodical manner. Before undergoing formal analysis, the designs are evaluated for their compatibility with formal methods. Automation techniques are also employed to enhance the productivity of formal verification, particularly in the development of properties.

A model checking technique for confirming reconfigurable FPGA modules provided by Petri nets is presented in the publication by Grobelna [4]. A unique formal verification system for reconfigurable modules built on FPGA hardware is presented by the author. Within an operating system, these modules-specified by Petri nets can be dynamically modified. Because of this, the multi-phase formal verification system highlighted how crucial it is to precisely describe and modify conditions during different stages of the design process.

A suggestion was made by Mehzoon *et al.* [5] for the formal verification of artificial and optimized multipliers. A revolutionary symbolic computer algebra (SCA)-system that can formally validate a large range of optimal multipliers was introduced by the authors. They provide a technique that prevents monomial explosion during backward rewriting by using dynamic negotiation ordering. The approach's usefulness in verifying different optimized multipliers, including false benchmarks, was proven by experimental data.

Ahmed *et al.* [6] study on dynamic partial reconfiguration included a technique for confirming the functionality. The authors presented an assertion-based verification (ABV) method for confirming the RMs' connections. System Verilog assertions (SVAs) were initially used to model the connections of the RMs. These generated assertions were then used to instrument the design, and formal verification methods were used to confirm that the connections were accurate.

A software verification procedure and approach for creating FPGA-grounded engineered safety feature systems were presented by Maerani *et al.* [7]. First, condition analysis and labor verification via software testing were part of the suggested verification method to make sure the design supporting the FPGA-grounded engineered safety feature–component control system (ESF-CCS) was reliable. In order to guarantee successful testing and compliance with customer requirements, the system satisfied quality assurance standards.

A P system-grounded method for computing polynomials with integer sections and formal verification was presented by Zhu *et al.* [8]. In order to capture polynomial values with natural number components, they created deterministic membrane systems. Later, they expanded their findings to encompass polynomials with integer figures. According to precedence, the authors presented deterministic transition P systems for the weak interpretation of same kind polynomials.

A formal verification of the circle was proposed by Bernardeschi *et al.* [9] in order to improve the safety-critical cyber-physical systems verification. Their methodology comprised an abstract framework based on the specification of advanced-order senses, verification through the application of theorems, and close coordination between the verification process and simulation and model-driven development. The architecture covered both digital and analogue systems, and examples from a variety of industries, including driverless cars, electric stopcock actuation, and implantable biomedical systems, proved how useful it was.

An AntiFact architecture and computer-aided design (CAD) inflow were presented by Nath *et al.* [10] for efficient formal verification of system-on-chip (SoC) security applications. Through the use of a flexible, centralized structure IP, their work established a crucial infrastructure for the application of security policies, enabling scalable and efficient verification of these programs. Built on top of currently available off-the-shelf tools, the suggested CAD inflow permitted both formal and dynamic (simulation-grounded) verification.

An FPGA-grounded symmetric re-encryption technique was presented by Al-Asli *et al.* [11] to secure data processing in pall-integrated internet-of-effects. In addition to guaranteeing FPGA authentication and creating a secure symmetric key session between the on-pall FPGA, the internet of things (IoT) device, and the client, their system supported many business models and achieved perfect forward secrecy. In addition, the suggested method satisfied typical IoT operating situations by enabling configuration integrity checks within the underlying FPGA during runtime. Formal verification with a realistic bushwhacker model showed that there are no weaknesses in the scheme.

Assertion-grounded verification was proposed by Ahmed *et al.* [12] for dynamic partial reconfiguration verification. In their work, they proposed a technique called assertion-grounded verification (ABV) for confirming recently presented flaws. The suggested method worked well for locating problems in actual designs that used dynamic partial reconfiguration (DPR). In order to guarantee proper operation, verification at the register transfer level (RTL) design level was prioritized before implementation on the FPGA. Any faults discovered during verification necessitated the inclusion of fixes into the design's implementation cycle.

The 2018 trends for FPGA functional verification were presented by Foster [13]. Given the growing complexity of FPGA designs, which is comparable to that of IC/ASIC designs, the study's conclusions provided insightful information on the present state of FPGA design and verification trends. The impact of this increasing complexity on verification efforts and efficacy was assessed in the research.

Model checking was presented by Buzhinsky and Pakonen [14] as a means of verifying fault-tolerant safety instrumentation and control (I&C) systems. They presented a method that used the NuSVM model checker to add communication delays and hardware element failures into I&C operational sense models. The authors focused on single failure tolerance and effectively validated incredibly complex system architectures.

A study of formal verification techniques for safety-critical system-on-chip (SoC) designs was carried out by Grimm *et al.* [15]. They looked at six well-known methods that were divided into formal verification and simulation. The authors determined that, after weighing the benefits and drawbacks of each, a hybrid approach provides the best balance between formal verification and simulation for complex system verification.

Universal verification methodology (UVM)-grounded verification was suggested by Stoddard *et al.* [16] for the HPSBC-FPGA of the fault-tolerant flying computer of the dream chaser. They gave a thorough explanation of the infrastructure, verification, and primary functions of the FPGA. They verified the operation of the cross-channel data link (CCDL) physical and operational layers, including the complex fault-tolerant CCDL communication exchange, as well as the HPSBC-FPGA.

The difficulties with big FPGA-grounded logic emulation systems were the main emphasis of Hung and Sum [17]. They focused on emulating commercial FPGA-grounded systems and brought the academic community's attention to a number of intricate difficulties in this field. Covered in the article were time sphere multiplexing, software flows, managing time sphere, influence on FPGA control sets, FPGA connectivity, partitioning, placement, routing, and leg assignment. For an overview of the International Verification and Security Workshop (IVSW), see Abadir and Aftabjahani [18]. Through the sharing of creative ideas and the development of new approaches for tackling the difficulties in various SoC design settings, IVSW sought to bring together industry interpreters and experimenters from the domains of security, verification, confirmation, test, and trustability.

CoreIR symbolic analyzer (CoSA), a model testing tool for CoreIR designs, was introduced by Mattarei *et al.* [19]. Model-checking problems were encoded by CoSA into first-order formulas that could be solved by solvers of satisfiability modulo theories (SMT). The ability of CoSA to extract useful information from CoreIR, including lemmas, to speed up the verification process was highlighted by the authors as they illustrated how to integrate it into the verification pipeline for nimble tackle design.

A theorem-proof-grounded gate position information inflow shadowing for tackling security verification was proposed by Qin *et al.* [20]. Their work concentrated on an Rivest-Shamir-Adleman (RSA) perpetration, an OpenRISC development interface core, and other Trojan marks. The suggested solution tackled Trojans while figuring out their detection conditions and successfully identified security flaws in timing channels.

This study builds on prior developments and discusses the difficulties in obtaining thorough verification coverage, especially in complicated FPGA designs and safety-critical systems. The approach seeks to improve verification robustness and reliability by proposing a novel verification methodology specifically designed for ethernet controller systems. This methodology advances the state-of-the-art in FPGA verification practices by ensuring comprehensive validation of design functions. In particular, the work offers a methodical approach that has not been previously discussed in the literature today, explaining how the suggested methodology closes current gaps in FPGA-based ethernet controller verification techniques. Extensive case studies and outcomes illustrate the efficacy of this methodology. The paper is organized as follows: section 2 describes the proposed verification methodology, section 3 describes the results and discussions, and conclusion is discussed in section 4.

2. PROPOSED VERIFICATION METHODOLOGY

In this section, we present a new approach for verification methodology based on ethernet protocol controller block. The proposed method is shown in Figure 1, experimented results using finite state machine (FSM) technique compared with the standard ethernet protocol controller architecture [21] is used to verify the ethernet protocol block. The standard verification method [22] is used to perform the verification, using individual test cases functional coverage are calculated.

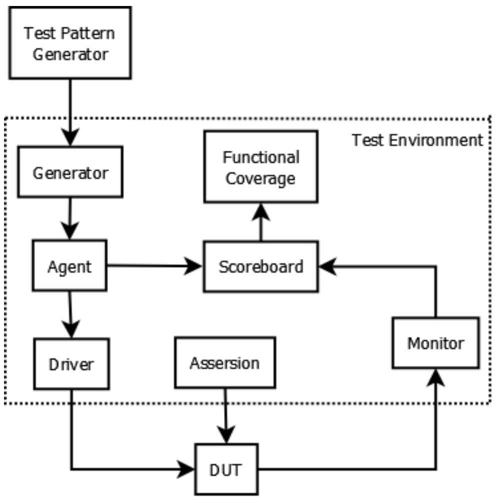


Figure 1. Verification methodology

2.1. Design under test

Figure 2 depicts the design under test module, which is a straightforward ethernet controller block made up of a controller [21], FIFO, and dual port RAM [23]. The entire process is managed by the controller architecture. In this instance, the cyclic coded data packet is first placed into the FIFO block and then momentarily stored in the RAM block.

When the controller is activated, the FSM model that was created for it, as seen in Figure 3, starts out in an ideal state. It reacts to different addresses appended to the supplied data in this case. Depending on whether the following ethernet block in the sequence is available, the system dynamically creates cyclic codes and addresses. By coordinating data transmission and reception effectively, this procedure maximizes the controller’s performance within the larger ethernet framework.

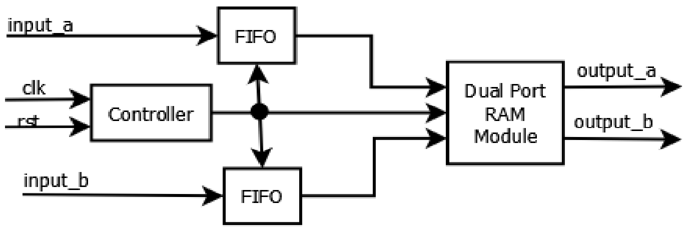


Figure 2. Architectural diagram of ethernet controller

2.2. Test pattern generator

Using random source and destination addresses, the FSM model is used to generate the test patterns. An example of how to generate a test pattern is provided below. Take into account a situation where an ethernet controller design needs to send a data packet, for instance. In this case, the data is initially combined with some cyclic code that the system generates, storing the packet in a FIFO block before sending a request to the closest ethernet block. The RAM block is currently awaiting the acknowledgement, and the controller will deliver the packet to the designated block once it is received. The system Verilog language [22] generates the data packet, cyclic code, and closest dummy ethernet blocks to verify this. This aids in accurate verification.

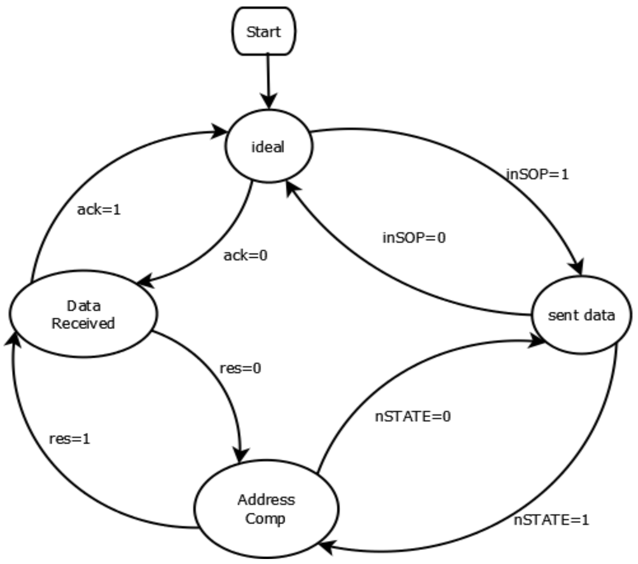


Figure 3. FSM model of ethernet controller

2.3. Test environment design

In order to increase coverage, the test environment is developed utilizing a tiered verification architecture with the help of the suggested test cases. Data structures, scoreboards, cover points, and transactor components are all made with classes in system Verilog. They enable the manipulation of stimuli provided by pattern generators using techniques akin to those found in an environment designed in C++. In order to confirm that certain aspects of the design must always be true, assertions are essential. They aid in identifying design defects in the DUT and are described in the interface file description. For thorough reporting, assertions can be integrated using the Synopsys unified report generation (URG) tool.

To monitor the verification plan and validate the elements of a design that have been validated, functional coverage is evaluated. Through the use of cover groups, which watch the stimuli provided to the DUT and keep an eye on the functionality employed in the replies, code is inserted during this process. The verification strategy should contain cover groups, and the information acquired during simulation is used to decide whether or not a cover group would be useful in a test case situation.

Cover points, when used to designate higher levels of abstraction in a design, enhance the effectiveness of simulation. They serve as a powerful tool for identifying functional code coverage. Similar to assertions, cover points can be assembled into a comprehensive reporting structure using the URG tool. Scoreboards, created using classes, are used to compare the data from the DUT to the expected data provided by the reference model. In a verification methodology manual (VMM)-style verification environment, data is often transmitted to the scoreboard through channels or mailboxes as a reliable method within a test bench.

Transaction-level models provide a higher-level abstraction of the design and enable efficient modeling and verification of complex systems. They capture the interactions and transactions between components, allowing for more accurate and scalable verification. Regression capability refers to the ability to rerun a set of tests or verification processes to ensure that modifications or additions to the design have not introduced any new issues. It allows for comprehensive validation of the design across multiple test cases and iterations.

Overall, these components in system Verilog contribute to effective verification and validation of digital designs, ensuring their correctness and adherence to design specifications. Scoreboards can be dynamically operated during a simulation and offer the deciding factor as to whether a test case has succeeded or failed. When executing regression suites of test cases, this information is located in the pass/fail log files. Stimuli are generated using random pattern generators to match the data members provided in the base class of a transactor.

In order to apply complete random data to a bus functional model (BFM) that interfaces with the DUT, the random pattern generator provides the data. In certain instances, the data is restricted to reduce the quantity of useless data that is produced. A reference model that implements the necessary functionality at a very high degree of abstraction is known as a “transaction level model,” or TLM. TLMs are often written more rapidly and efficiently because the desired function of the device is captured at an acceptable level of abstraction, unlike RTL code, which is written at a low level of abstraction. Regression capability is a way for the verification environment to emphasize and support specific data that may subsequently be extrapolated to establish alternative test case results. In order to compile various code coverage findings and determine the overall level of verification given to the DUT, regression runs are combined.

3. RESULTS AND DISCUSSION

The proposed verification methodology, implemented using system Verilog, includes both the DUT and the verification environment designed in the same language. The simulation waveform, as shown in Figure 4, illustrates a random value being fed to the ethernet module, which is then correctly encoded and routed by the ethernet block. The test report snapshot generated by the ModelSim [24] tool, depicted in Figure 5, explains how data with Ethernet packet specifications is applied to the DUT. The system checks all possible routing paths, compared with the paths generated by the test generator module. Based on these test comparisons, the system decides the coverage.

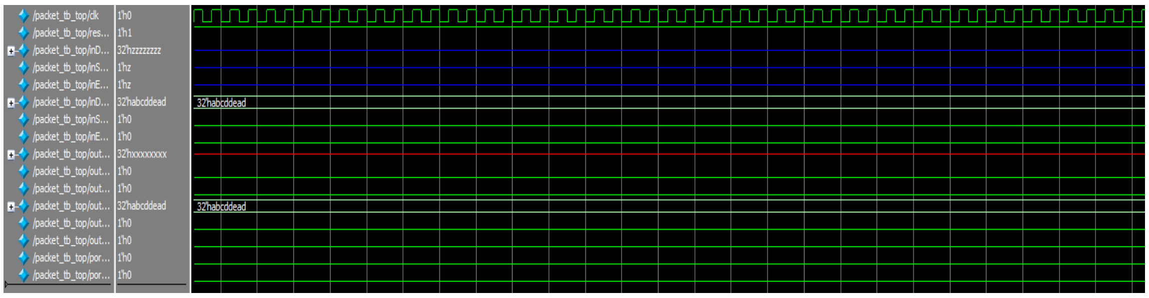
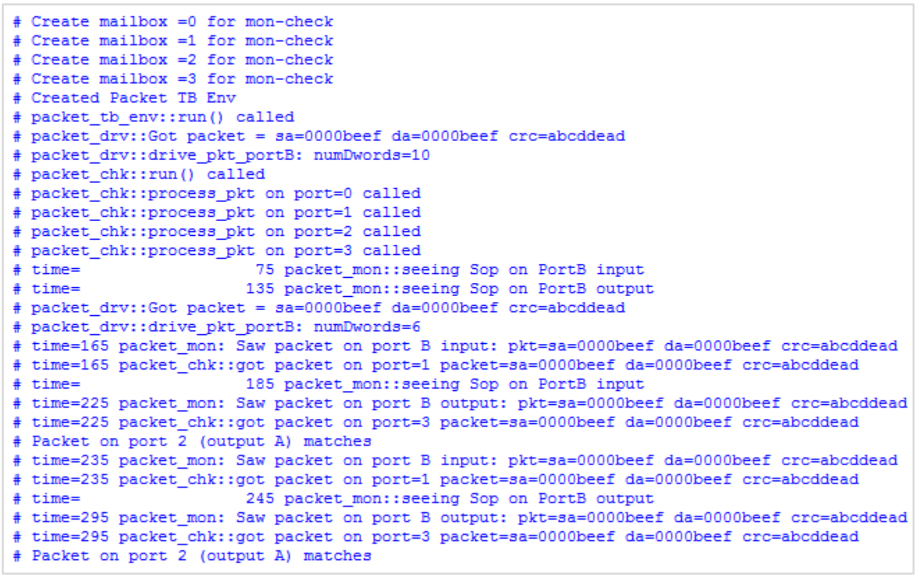


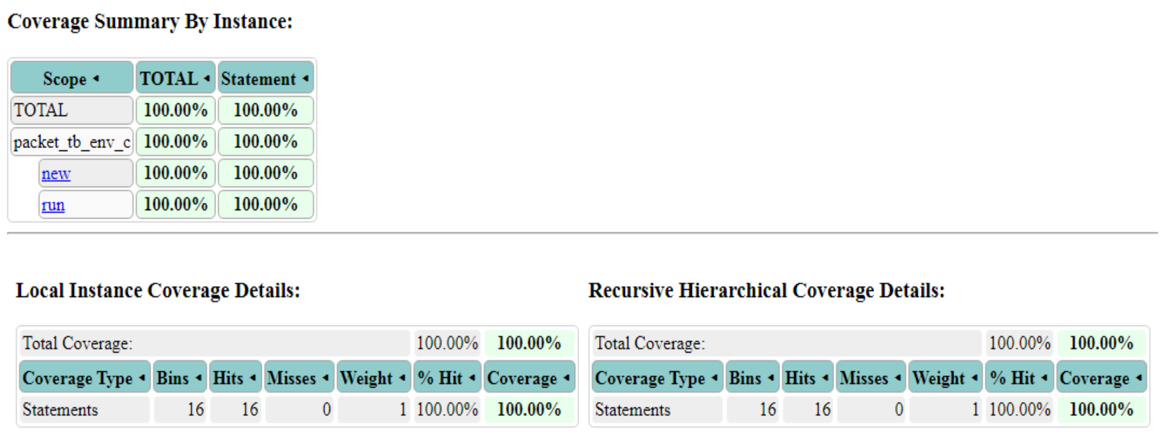
Figure 4. Simulation waveform



```
# Create mailbox =0 for mon-check
# Create mailbox =1 for mon-check
# Create mailbox =2 for mon-check
# Create mailbox =3 for mon-check
# Created Packet TB Env
# packet_tb_env::run() called
# packet_drv::Got packet = sa=0000beef da=0000beef crc=abcddead
# packet_drv::drive_pkt_portB: numDwords=10
# packet_chk::run() called
# packet_chk::process_pkt on port=0 called
# packet_chk::process_pkt on port=1 called
# packet_chk::process_pkt on port=2 called
# packet_chk::process_pkt on port=3 called
# time=                   75 packet_mon::seeing Sop on PortB input
# time=                  135 packet_mon::seeing Sop on PortB output
# packet_drv::Got packet = sa=0000beef da=0000beef crc=abcddead
# packet_drv::drive_pkt_portB: numDwords=6
# time=165 packet_mon: Saw packet on port B input: pkt=sa=0000beef da=0000beef crc=abcddead
# time=165 packet_chk::got packet on port=1 packet=sa=0000beef da=0000beef crc=abcddead
# time=                  185 packet_mon::seeing Sop on PortB input
# time=225 packet_mon: Saw packet on port B output: pkt=sa=0000beef da=0000beef crc=abcddead
# time=225 packet_chk::got packet on port=3 packet=sa=0000beef da=0000beef crc=abcddead
# Packet on port 2 (output A) matches
# time=235 packet_mon: Saw packet on port B input: pkt=sa=0000beef da=0000beef crc=abcddead
# time=235 packet_chk::got packet on port=1 packet=sa=0000beef da=0000beef crc=abcddead
# time=                  245 packet_mon::seeing Sop on PortB output
# time=295 packet_mon: Saw packet on port B output: pkt=sa=0000beef da=0000beef crc=abcddead
# time=295 packet_chk::got packet on port=3 packet=sa=0000beef da=0000beef crc=abcddead
# Packet on port 2 (output A) matches
```

Figure 5. Test report generated by ModelSim tool

The overall verification coverage, computed using ModelSim [24] shown in Figure 6, ensures comprehensive validation of the ethernet controller architecture [22]. ModelSim's integrated environment for simulating and debugging very high-speed integrated circuit hardware description language (VHDL), Verilog, and mixed-language designs makes it ideal for verifying complex FPGA designs. By utilizing ModelSim, the verification process can precisely track functional coverage, code coverage, and assertion coverage. The FSM model's use in designing test cases is the main reason behind achieving 100% coverage. This thorough analysis helps identify any gaps in the verification process, ensuring all aspects of the design are thoroughly tested and validated. The tool's advanced debugging capabilities facilitate the identification and resolution of design flaws, contributing to the robustness and reliability of the verification methodology.



Coverage Summary By Instance:

Scope	TOTAL	Statement
TOTAL	100.00%	100.00%
packet_tb_env_c	100.00%	100.00%
new	100.00%	100.00%
run	100.00%	100.00%

Local Instance Coverage Details:

Total Coverage:					100.00%	100.00%
Coverage Type	Bins	Hits	Misses	Weight	% Hit	Coverage
Statements	16	16	0	1	100.00%	100.00%

Recursive Hierarchical Coverage Details:

Total Coverage:					100.00%	100.00%
Coverage Type	Bins	Hits	Misses	Weight	% Hit	Coverage
Statements	16	16	0	1	100.00%	100.00%

Figure 6. The overall verification coverage computed using ModelSim tool

3.1. Key findings and limitations

The proposed methodology has demonstrated remarkable success by achieving 100% verification coverage, underscoring it is effectiveness. Utilizing FSM models to generate test patterns has been pivotal, ensuring these patterns align closely with the design’s functional states. This alignment facilitates thorough testing across all possible scenarios, thereby yielding robust verification outcomes. Both the simulation waveform and the generated test report from ModelSim provide concrete evidence of the methodology’s precision and thoroughness in validating the ethernet controller architecture.

Interpreting these results highlights the methodology’s ability to ensure comprehensive coverage of the ethernet controller architecture, surpassing existing methods. The key takeaway is the efficacy of using FSM-based test pattern generation for thorough verification. Nonetheless, the study acknowledges limitations, such as the potential complexity and time investment in developing FSM models, which may pose scalability challenges for diverse FPGA designs. Future research directions should focus on streamlining FSM model development and extending this methodology to broader applications within FPGA technology.

3.2. Comparison of proposed method with existing methods

The performance of proposed method is compared with existing methods presented by Chitti *et al.* [25], Babu and Swaroop [26], Selvakkani and Venkatesan [27]. It is validated that the proposed approach achieved superior performance compared to existing methods. The percentage optimum FSM value is high in the case of proposed method compared to existing methods. The proposed verification strategies are compared with existing [25]–[27] which is shown in Table 1. The percentage optimum FSM value is high in the case of proposed method compared to existing methods. From which it can be seen that the proposed methodology is able to produce accurate verification strategy since we are generating the test patterns depending upon the FSM models used in the design.

Table 1. Comparison of verification strategies with existing techniques

Sl. No.	Authors	Programming Language	Tool	Optimum FSM %
1	Chitti *et al.* [25]	System Verilog	QuestaSim	91.25
2	Babu and Swaroop [26]	System Verilog+UVM	ModelSim	89.23
3	Selvakkani and Venkatesan [27]	System Verilog+UVM	ModelSim	30
4	Proposed method	System Verilog	ModelSim	100

4. CONCLUSION

The Pre-silicon verification is usually more significant than post-silicon verification. In this paper, we proposed a novel verification methodology for an Ethernet controller, designed using the system Verilog language and implemented in a layered manner. The methodology is validated using the ModelSim tool, with the Ethernet controller block serving as the DUT. To ensure accurate verification strategies, the methodology leverages a state machine model to generate test patterns and verify the corresponding responses. By employing a specific FSM model during the verification process, the methodology achieves a comprehensive coverage of 100%. It is noticed that, the performance of proposed method is better compared to existing methods.

REFERENCES

[1] T. Peng, T. S. Kavya, Erdenetuya, Y.-M. Jang, and S.-B. Cho, “A FPGA verification of improvement edge detection using separation and buffer line,” in *2020 International Conference on Electronics, Information, and Communication (ICEIC)*, IEEE, Jan. 2020, doi: 10.1109/iceic49074.2020.9051159.

[2] P. Zou *et al.*, “Time-division multiplexing based system-level FPGA routing for logic verification,” in *2020 57th ACM/IEEE Design Automation Conference (DAC)*, IEEE, Jul. 2020, doi: 10.1109/dac18072.2020.9218569.

[3] K. Devarajegowda, L. Servadei, Z. Han, M. Werner, and W. Ecker, “Formal verification methodology in an industrial setup,” in *2019 22nd Euromicro Conference on Digital System Design (DSD)*, IEEE, Aug. 2019, doi: 10.1109/dsd.2019.00094.

[4] I. Grobelna, “Model checking of reconfigurable FPGA modules specified by Petri nets,” *Journal of Systems Architecture*, vol. 89, pp. 1–9, Sep. 2018, doi: 10.1016/j.sysarc.2018.06.005.

[5] A. Mahzoon, D. Grose, C. Scholl, and R. Drechsler, “Towards formal verification of optimized and industrial multipliers,” in *2020 Design, Automation and Test in Europe Conference and Exhibition (DATE)*, IEEE, Mar. 2020, doi: 10.23919/date48585.2020.9116485.

[6] I. Ahmed, H. Mostafa, and A. N. Mohieldin, “On the functional verification of dynamic partial reconfiguration,” in *2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS)*, IEEE, Aug. 2018, doi: 10.1109/mwscas.2018.8624012.

[7] R. Maerani, J. K. Mayaka, and J. C. Jung, “Software verification process and methodology for the development of FPGA-based engineered safety features system,” *Nuclear Engineering and Design*, vol. 330, pp. 325–331, Apr. 2018, doi: 10.1016/j.nucengdes.2018.02.009.

[8] M. Zhu, G. Zhang, Q. Yang, H. Rong, W. Yuan, and M. J. Perez-Jimenez, “P systems-based computing polynomials with integer coefficients: design and formal verification,” *IEEE Transactions on NanoBioscience*, vol. 17, no. 3, pp. 272–280, Jul. 2018, doi: 10.1109/tnb.2018.2836147.

[9] C. Bernardeschi, A. Domenici, and S. Saponara, “Formal verification in the loop to enhance verification of safety-critical cyber-physical systems,” *Electronic Communications of the EASST*, vol. 77, pp. 1–9, 2019.

[10] A. P. D. Nath, S. Bhunia, and S. Ray, “ArtiFact: architecture and CAD flow for efficient formal verification of SoC security policies,” in *2018 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, IEEE, Jul. 2018, doi: 10.1109/isvlsi.2018.00081.

[11] M. Al-Asli, M. E. S. Elrabaa, and M. Abu-Amara, “FPGA-based symmetric re-encryption scheme to secure data processing for cloud-integrated internet of things,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 446–457, Feb. 2019, doi: 10.1109/jiot.2018.2864513.

[12] I. Ahmed, H. Mostafa, and A. N. Mohieldin, “Dynamic partial reconfiguration verification using assertion based verification,” in *2018 13th International Conference on Design and Technology of Integrated Systems In Nanoscale Era (DTIS)*, IEEE, Apr. 2018, doi: 10.1109/dtis.2018.8368552.

[13] H. D. Foster, “2018 FPGA functional verification trends,” in *2018 19th International Workshop on Microprocessor and SOC Test and Verification (MTV)*, IEEE, Dec. 2018, doi: 10.1109/mtv.2018.00018.

[14] I. Buzhinsky and A. Pakonen, “Model-checking detailed fault-tolerant nuclear power plant safety functions,” *IEEE Access*, vol. 7, pp. 162139–162156, 2019, doi: 10.1109/access.2019.2951938.

[15] T. Grimm, D. Lettnin, and M. Hübner, “A survey on formal verification techniques for safety-critical systems-on-chip,” *Electronics*, vol. 7, no. 6, May 2018, doi: 10.3390/electronics7060081.

[16] A. Stoddard, J. Frey, K. Kazi, and W. Johnson, “UVM based verification for HPSBC-FPGA of the dream chaser’s fault tolerant flight computer,” in *2019 IEEE Aerospace Conference*, IEEE, Mar. 2019, doi: 10.1109/aero.2019.8741931.

[17] W. N. N. Hung and R. Sun, “Challenges in large FPGA-based logic emulation systems,” in *Proceedings of the 2018 International Symposium on Physical Design*, in ISPD ’18. ACM, Mar. 2018. doi: 10.1145/3177540.3177542.

[18] M. Abadir and S. Aftabjahani, “An overview of the international verification and security workshop (IVSW),” in *2019 IEEE International Test Conference (ITC)*, IEEE, Nov. 2019. doi: 10.1109/itc44170.2019.9000165.

[19] C. Mattarei, M. Mann, C. Barrett, R. G. Daly, D. Huff, and P. Hanrahan, “CoSA: integrated verification for agile hardware design,” in *2018 Formal Methods in Computer Aided Design (FMCAD)*, IEEE, Oct. 2018, doi: 10.23919/fmcad.2018.8603014.

[20] M. Qin, W. Hu, X. Wang, D. Mu, and B. Mao, “Theorem proof based gate level information flow tracking for hardware security verification,” *Computers &amp; Security*, vol. 85, pp. 225–239, Aug. 2019, doi: 10.1016/j.cose.2019.05.005.

[21] M. Qian, J. Zhu, Y. Cao, and C. Yang, “Design and FPGA verification of dual-speed adaptive ethernet controller,” in *2011 7th International Conference on Wireless Communications, Networking and Mobile Computing*, IEEE, Sep. 2011. doi: 10.1109/wicom.2011.6040465.

[22] C. Spear, “System Verilog for verification,” *System Verilog for Verification*, 2008, doi: 10.1007/978-0-387-76530-3.

[23] S. Srinivas and A. B. Gudi, “Design and implementation of dual port SRAM memory architecture using MOSFET’s,” in *2017 International Conference on Smart grids, Power and Advanced Control Engineering (ICSPACE)*, IEEE, Aug. 2017. doi: 10.1109/icspace.2017.8343457.

[24] S. Das, J.-F. Li, A. Nayak, M. Assaf, S. Biswas, and E. Petriu, “Circuit architecture test verification based on hardware software co-design with ModelSim,” *IETE Journal of Research*, vol. 59, no. 2, 2013, doi: 10.4103/0377-2063.113032.

[25] S. Chitti, P. C. Sekhar, and M. Asharani, “Ethernet MAC verification with loopback mechanism using efficient verification methodology,” *International Journal of Electronics and Communication Engineering*, vol. 3, no. 11, pp. 57–60, Nov. 2016, doi: 10.14445/23488549/ijece-v3i11p115.

[26] P. A. Babu and K. J. Swaroop, “Ethernet verification using UVM methodology,” *International Conference on Industrial Instrumentation and Control (ICIC) 2015 IEEE*, vol. 6, no. 2, 2016.

[27] K. Selvakkani and K. Venkatesan, “High speed UVM based verification IP for gigabit ethernet protocol,” *International Journal of Engineering Research and Technology (IJERT)*, vol. 2, no. 12, pp. 2278–0181, 2013.

BIOGRAPHIES OF AUTHORS

Shubha Parameshwara received her B.E. degree in instrumentation engineering from Dr. AIT, her M. Tech degree in computer science from RVCE Bangalore, and her Ph.D. in electrical and electronics engineering sciences from Visvesvaraya Technological University, Karnataka, in 2021. She has been an assistant professor in the Department of Electronics and Instrumentation Engineering at Dr. Ambedkar Institute of Technology, Bangalore. She has 9 years of research experience and 19 years of teaching experience. Her keen areas of interest include building autonomous robots, image processing, video processing, embedded systems, and the internet of things. She can be contacted at email: shubhap.ei@drait.edu.in.

Ganapathi Vithoba Sagar is working as an associate professor in the Department of Electronics and Instrumentation Engineering at Dr. Ambedkar Institute of Technology, Bangalore. He obtained his B.E. degree in instrumentation technology from BDT, College of Engineering Davangere. His specialization in master degree was bio-medical instrumentation from VTU, Belgaum and He was awarded Ph.D. in computer science and engineering from Bangalore University. He has over 30 research publications in referred International Journals and Conference Proceedings. His area of interest is in the field of signal processing, biomedical signal processing and control Engineering. He can be contacted at email: ganapathi04.ei@drait.edu.in.

Hamsa Rekha Sorekunte Dasappa is working as an assistant professor in the Department of Electronics and Instrumentation Engineering at Dr. Ambedkar Institute of Technology, Bangalore. She obtained his B.E. degree in instrumentation technology from R V College of Engineering. Her specialization in master degree was power electronics from VTU, Belgaum and She is pursuing Ph.D. in electrical and electronics engineering sciences from Visvesvaraya Technological University, Karnataka. Her area of interest is in the field of bio medical sensors, embedded systems and internet of things. She can be contacted at email: hamsarekhasd.ei@drait.edu.in.

Sheetal Nagaraj is working as an assistant professor in the Department of Electronics and Instrumentation Engineering at Dr. Ambedkar Institute of Technology, Bangalore. She obtained his B.E. degree in instrumentation technology from Bangalore Institute of Technology. Her specialization in master degree was power electronics from VTU, Belgaum and She is pursuing Ph.D. in electrical and electronics engineering sciences from Visvesvaraya Technological University, Karnataka. Her area of interest is in the field of image processing, sensors and actuators, embedded systems and internet of things. She can be contacted at email: sheetaln.ei@drait.edu.in.