Ain Shams University
Faculty of Engineering
Computer and Systems Department
Spring 2024

CSE 313s
Selected Topics in Computer Engineering
Sheet 2

1. Define hardware functional verification and explain its importance in the development process of electronic devices.

2. Discuss the role of simulation in hardware functional verification and provide examples of commonly used simulation tools.

3. Explain the concept of constrained random testing in hardware functional verification and how it helps in improving test coverage.

4. Compare and contrast hardware emulation and simulation in the context of functional verification.

5. What are the challenges associated with hardware functional verification, and how can they be addressed?

6. Discuss the significance of formal verification techniques in ensuring the correctness of hardware designs.

7. Describe the role of assertion-based verification in hardware functional verification.

8. Explain how code coverage metrics are used to assess the effectiveness of hardware functional verification tests.

9. What is the primary goal of functional verification in hardware design?
    a. Ensuring timing constraints are met
    b. Verifying the correctness of the design's functionality
    c. Optimizing power consumption
    d. Minimizing area utilization

10. Which of the following is NOT a commonly used technique in functional verification?
    a. Simulation
    b. Formal Verification
    c. Static Timing Analysis
    d. Emulation

11. Which stage of the design process typically involves functional verification?
    a. RTL Design
    b. Synthesis
    c. Place and Route
    d. Physical Verification

12. What is the purpose of testbenches in functional verification?
    a. To generate random stimuli for the design
    b. To provide a reference model for comparison
    c. To automate the verification process
    d. All of the above

13. What is the fundamental difference between static and dynamic verification in hardware systems?
    a. Static verification checks for correctness at runtime, while dynamic verification examines the design properties statically.
    b. Static verification analyzes the design without executing it, while dynamic verification involves running tests on the design.
    c. Static verification focuses on timing analysis, while dynamic verification focuses on functionality.
    d. Static verification requires formal methods, while dynamic verification relies on simulation.

14. Which of the following statements best describes static verification?
    a. Static verification ensures that the design functions correctly under all possible input conditions.
    b. Static verification involves running test cases on the design to detect bugs.
    c. Static verification analyzes the design properties without actually executing it.
    d. Static verification verifies the design's timing constraints dynamically.

15. In dynamic verification, what is typically observed?
    a. Design properties are analyzed without running test cases.
    b. Timing constraints are verified without executing the design.
    c. The design is simulated or emulated with test stimuli to observe its behavior.
    d. Formal methods are used to verify the design properties.

16. In directed testing, test cases are designed based on:
    a. Random inputs
    b. Specific requirements or scenarios
    c. Historical data
    d. None of the above

17. Which testing technique focuses on uncovering defects by selecting inputs randomly from the input domain?
    a. Directed testing
    b. Random testing
    c. Constrained random testing
    d. Exhaustive testing

18. In constrained random testing, constraints are typically defined based on:
    a. User preferences
    b. Input domain boundaries
    c. Random number generators
    d. Historical test results

19. Random testing is also known as:
    a. Black-box testing
    b. Ad hoc testing
    c. Fuzz testing
    d. White-box testing

20. What is the primary advantage of using constrained random testing over random testing?
    a. Higher defect detection rate
    b. Faster test case generation
    c. Greater test coverage
    d. More intuitive test design process

21. The effectiveness of random testing can be improved by:
    a. Increasing the number of test cases
    b. Incorporating code coverage analysis
    c. Adding more constraints to the test inputs
    d. Reusing test cases from previous projects

22. What is the primary purpose of using MITRE circuits in equivalence checking for hardware systems?
    a. To perform formal verification of designs
    b. To generate random test vectors for simulation
    c. To detect functional discrepancies between designs
    d. To optimize power consumption in digital circuits

23. What advantage do MITRE circuits offer over traditional equivalence checking methods?
    a. MITRE circuits provide exhaustive analysis of all possible design states.
    b. MITRE circuits are faster and more efficient for large-scale designs.
    c. MITRE circuits do not require any inputs and can automatically verify designs.
    d. MITRE circuits can detect subtle functional differences between designs.

24. What is the primary objective of equivalence checking using MITRE circuits?
    a. To verify the correctness of the design's RTL code
    b. To ensure compliance with industry standards for hardware design
    c. To detect functional discrepancies between two versions of a design
    d. To optimize the performance of the design's timing constraints

25. Static Analysis involves simulating a model.
    a. True
    b. False

26. An escaped bug is a design bug that is not detected during pre-silicon verification, and it is only caught during post-silicon verification.
    a. True
    b. False

27. Formal verification encompasses all techniques that leverage mathematical reasoning and proofs to determine the correctness of a silicon design.
    a. True
    b. False

28. Name two of the coverage metrics that we discussed in lectures.

29. You are given a Verilog code snippet for a simple adder module, along with a set of input values. Your task is to compute statement coverage for the provided Verilog code by executing the test cases and determining which statements are covered.

```
1  module adder (
2      input [3:0] a,
3      input [3:0] b,
4      output reg [4:0] sum
5  );
6
7  always @(*) begin
8      sum = a + b;
9  end
10
11 endmodule
```

Input Values:
- Test Case 1: a = 2, b = 3
- Test Case 2: a = 0, b = 7
- Test Case 3: a = 8, b = 8

Compute the statement coverage percentage for the Verilog code based on the executed statements.

30. Below is a Verilog code for a simple 2-to-1 multiplexer:

```
module mux_2to1(input wire A, B, S, output reg Y);
    always @(A or B or S)
    begin
        if (S == 0)
            Y = A;
        else
            Y = B;
    end
endmodule
```

Input values:
- A = 1
- B = 0
- S = 0

Compute statement coverage for this Verilog module with the given input values.

31. For the given code and input values compute the toggle coverage

```
module toggle_module(input wire clk, input wire rst, output reg q);

    reg prev_clk_state;

    always @(posedge clk or posedge rst)
    begin
        if (rst)
            q <= 1'b0;
        else if (clk && !prev_clk_state)
            q <= ~q;
        prev_clk_state <= clk;
    end

endmodule
```

Input values at consecutive clock cycles:
- clk = 1, rst = 0 (initial values)
- clk = 0, rst = 0
- clk = 0, rst = 0
- clk = 0, rst = 0

32. Define static verification explain its importance in the design and development process.

33. Describe three common techniques used for static verification in hardware design and provide an example for each.

34. Discuss the advantages and limitations of static verification compared to dynamic verification in the context of hardware systems.

35. Explain what dynamic verification entails in the context of hardware systems and how it contributes to ensuring the reliability and functionality of the hardware.

36. Differentiate between simulation-based dynamic verification and emulation-based dynamic verification, providing examples of each.

37. Compare and contrast static and dynamic verification approaches in hardware design, highlighting their respective strengths and weaknesses.

38. Discuss how static and dynamic verification techniques complement each other in ensuring the correctness and reliability of hardware systems.

39. Explain the challenges associated with verifying complex hardware designs and how a combination of static and dynamic verification methodologies can address these challenges.

40. Identify and explain three popular tools used for static verification in hardware design. Discuss their key features and how they aid in ensuring the correctness of hardware designs.

41. Explore the role of hardware simulation platforms in dynamic verification. Provide examples of widely-used simulation tools and their advantages in the verification process of hardware systems.

42. Design a directed test case to verify the functionality of a hardware adder circuit that adds two 8-bit binary numbers.

43. Explain the advantages and limitations of directed testing in the context of hardware verification. Provide examples of scenarios where directed testing is particularly effective and where it may fall short.

44. Develop a random test generation algorithm to verify the functionality of a hardware state machine controller with four states and multiple transition conditions. Describe how the algorithm generates random input sequences and evaluates the correctness of the controller's behavior.

45. Discuss the benefits and challenges of random testing in hardware verification. Explain how randomness helps uncover corner cases and hidden defects, and how test coverage metrics can be used to assess the effectiveness of random test generation.

46. Define constraints for a constrained random test environment targeting a hardware memory controller.

47. Compare and contrast constrained random testing with directed and random testing approaches, and provide examples of scenarios where constrained random testing is advantageous.

48. What are some common verification metrics used to measure the quality and completeness of hardware designs?

49. How is code coverage used as a verification metric in hardware systems development?

50. What is the significance of functional coverage as a verification metric in hardware design?