Ain Shams University
Faculty of Engineering
Computer and Systems Department
Spring 2026

Sheet 1 Solutions

1. Which of the following statements best describes the difference between hardware verification and testing?
   a. Hardware verification ensures that the design specifications are correctly implemented, while testing focuses on finding defects and bugs in the final product.
   b. Hardware verification and testing are interchangeable terms and refer to the same process.
   c. Hardware testing ensures that the design meets performance requirements, while verification validates the functionality.
   d. Hardware verification is conducted by users, while testing is performed by engineers during the design phase.

2. Which phase of the hardware development process primarily focuses on confirming that the design adheres to the specified requirements?
   a. Verification
   b. Testing
   c. Integration
   d. Validation

3. During hardware verification, what is the primary goal?
   a. Finding defects and bugs
   b. Ensuring design specifications are correctly implemented
   c. Assessing performance under various conditions
   d. Testing for user acceptance

4. When is hardware verification typically conducted?
   a. During the initial design phase
   b. After the hardware is manufactured
   c. Throughout the entire hardware development process
   d. Only after the hardware is deployed to end-users

5. Which activity is typically performed before hardware testing?
   a. Integration
   b. Verification
   c. Validation
   d. Debugging

6. What does Murphy's Law state?
    a. Anything that can go wrong will go wrong
    b. Every action has an equal and opposite reaction
    c. The sum of the forces acting on an object is equal to zero
    d. Objects at rest tend to stay at rest unless acted upon by an external force

7. What is the implication of Murphy's Law in engineering and project management?
    a. Projects always finish ahead of schedule
    b. Errors and setbacks are inevitable
    c. Success is guaranteed with proper planning
    d. Projects never face unexpected challenges

8. Which of the following scenarios best exemplifies Murphy's Law?
    a. A hardware project with no bugs
    b. A project completed without any delays
    c. A manufacturing process with zero defects
    d. A product failing during its first demonstration

9. What was the primary consequence of the FDIV bug?
    a. Incorrect calculation of floating-point division operations
    b. Overheating of the CPU
    c. Memory leakage
    d. Data corruption during file operations

10. In what year was the FDIV bug first discovered?
    a. 1990
    b. 1994
    c. 1998
    d. 2000

11. How did Intel finally respond to the discovery of the FDIV bug?
    a. Issuing a recall of all affected processors
    b. Offering free replacements for affected processors
    c. Denying the existence of the bug
    d. Releasing a software patch to fix the bug

12. Which term best describes the nature of Intel’s FDIV bug?
    a. Firmware glitch
    b. Software defect
    c. Hardware bug
    d. Network vulnerability

13. What was the primary cause of the failure of the NASA Polar Lander mission in 1999?
    a. Engine failure during descent
    b. Communication loss during landing
    c. Error in the descent sequence
    d. Structural damage during entry into Mars' atmosphere

14. Which component of the landing sequence was affected by the failure during the NASA Polar Lander mission?
    a. Parachute deployment
    b. Heat shield release
    c. Retro-rocket firing
    d. Landing leg deployment

15. What lesson was learned from the failure of the NASA Polar Lander mission?
    a. The importance of rigorous testing and verification
    b. The need for more advanced propulsion systems for Mars landers
    c. The significance of redundancy in critical spacecraft components
    d. The inevitability of technical challenges in space exploration

16. What was identified as the primary cause of the Patriot missile defense system failure during the Gulf War in 1991?
    a. Radar malfunction
    b. Software Error
    c. Insufficient training of operators
    d. Mechanical failure

17. Which aspect of the Patriot missile defense system's software was responsible for the failure during the Gulf War?
    a. Target acquisition
    b. Missile propulsion
    c. Trajectory calculation
    d. Guidance system

18. What specific issue with the Patriot missile defense system's software caused the failure during the Gulf War?
    a. Decimal point rounding errors
    b. Memory overflow
    c. Communication delays
    d. Sensor calibration errors

19. How did the software glitch in the Patriot missile defense system affect its ability to intercept incoming Scud missiles during the Gulf War?
    a. It caused the missiles to veer off course
    b. It failed to track the missiles accurately
    c. It triggered premature detonation of interceptors
    d. It delayed the firing of interceptors

20. Why is it important to fix bugs in hardware as soon as possible?
    a. To avoid software compatibility issues
    b. To prevent performance degradation
    c. To minimize the risk of product failures or malfunctions
    d. To improve aesthetic appeal

21. What can happen if hardware bugs are not addressed promptly?
    a. Decreased manufacturing costs
    b. Increased customer satisfaction
    c. Loss of revenue due to product recalls
    d. Enhanced product features

22. What is one consequence of delaying the resolution of hardware bugs?
    a. Improved product reliability
    b. Enhanced brand reputation
    c. Increased cost of bug fixes
    d. Decreased time-to-market

23. What is the primary purpose of Clock Domain Crossing (CDC) verification in digital design?
    a. To synchronize clocks between different domains
    b. To ensure proper signal transfer between different clock domains
    c. To optimize clock frequencies for better performance
    d. To reduce power consumption in digital circuits

24. Why do some companies neglect this aspect of product development?
    Some companies may neglect hardware verification due to various reasons, including tight project deadlines, budget constraints, lack of expertise or awareness, and a focus on short-term profitability over long-term reliability.

25. Explain the potential consequences of overlooking hardware verification in the manufacturing process. Provide examples of real-world incidents where inadequate verification led to costly failures or recalls.
    Consequences of overlooking hardware verification can include product malfunctions, safety hazards, costly recalls, damage to brand reputation, legal liabilities, and loss of customer trust. Examples include the Intel Pentium FDIV bug and the Therac-25 radiation therapy machine accidents.

26. Compare and contrast the investment required for hardware verification versus the potential costs of product recalls or customer dissatisfaction due to hardware failures. Why do some companies still prioritize cost-cutting over thorough verification processes?
    While investing in hardware verification may require upfront costs, neglecting it can lead to much higher expenses in the long run due to product failures, recalls, and damage control efforts. However, some companies prioritize cost-cutting to meet short-term financial goals, risking the quality and reliability of their products.

27. Assume that you can simulate 10^6 simulations per second. How long does it take to exhaustively simulate a design with 50 inputs?
    $2^{50} / (10^6 \times 60 \times 60 \times 24 \times 365) = 35.7$ years

28. Why is it impractical to test all possible input scenarios when simulating hardware systems?

It is impractical to test all possible input scenarios when simulating hardware systems due to the huge number of potential combinations, which can be prohibitively large and time-consuming to execute.

29. Discuss the limitations of exhaustive testing in hardware simulation. How does the number of possible input combinations impact testing feasibility?
Exhaustive testing in hardware simulation is limited by the exponential growth of input combinations, making it infeasible to cover all scenarios. The number of possible input combinations increases exponentially with the number of input variables, leading to combinatorial explosion.