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

# CSE 313s
# Selected Topics in Computer Engineering
# Sheet 7

1. What is SVA?

2. What are the benefits of using assertions??

3. Write a System Verilog assertion to check the following:
   *a* and *b* are two signals, which can be active at any time, but should never be active together.

4. Write a System Verilog assertion to check the following:
   Every time the request *req* goes high, *ack* arrives exactly 3 clocks later

5. Write a System Verilog assertion to check the following:
   If *a* is high at a clock edge, followed by 3 consecutive cycles in which *b* is high, then in each of the 3 cycles the data output *DO* is equal to the data input *DI.*

6. Write a System Verilog assertion to check the following:
   Every time the valid signal *vld* is high, the *cnt* is incremented

7. Write a System Verilog assertion to check the following:
   If *b* is high at a clock edge, then 2 cycles before that, *a* was high.

8. Write a System Verilog assertion to check the following:
   If there are two occurrences of "*a"* rising while state = ACTIVE, and no "*b"* occurs between them, then within 3 cycles of the second rise of "*a"*, START must occur.

9. Write a System Verilog assertion to check the following:
   Every "*a"* must eventually be acknowledged by "*b*", unless "*c*" appears any time before "*b*" appears.

10. Write a System Verilog assertion to check the following:
    Every time the request *req* goes high, *gnt* arrives exactly 3 clocks later. If this is not achieved an error is reported with the message: "no grant after request".
    But this assertion should only be checked if the reset signal, *rst*, is not active.

11. Write a System Verilog assertion to check the following:
    If a signal "*a*" is high on a given posedge of the clock, the signal "*b*" should be high for 3 clock cycles followed by "*c*" that should be high after "*b*" is high for the third time. During this entire sequence, if reset is detected at any point, the checker will stop.

12. Write a System Verilog assertion to check the following:
    A request "*req*" is high for one or more cycles, then returning to zero, is followed after one or more cycles, by an acknowledge, "*ack*" for one or more cycles before "*ack*" returns to zero. "*ack*" must be zero in the cycle in which "*req*" returns to zero. During this entire sequence, if reset is detected at any point, the checker will stop.