1. Write an assertion check to make sure that a signal is high for a minimum of 2 cycles and a maximum of 6 cycles.
2. Are following assertions equivalent:

@(posedge clk) req |=> ##2 $rose(ack);

@(posedge clk) req |-> ##3 $rose(ack);

1. For a synchronous FIFO of depth = 16, write an assertion for the following scenarios. - Assume a clock signal(clk), write and read enable signals, full flag and a word counter signal.

a. If the word count is >15, FIFO full flag set.

b. If the word count is 15 and a new write operation happens without a simultaneous read, then the FIFO full flag is set.

1. Write an assertion checker to make sure that an output signal never goes X?
2. Write an assertion to make sure that a 5-bit grant signal only has one bit set at any time?
3. Write an assertion which checks that once a valid request is asserted by the master, the arbiter provides a grant within 2 to 5 clock cycles.
4. As long as signal\_a is up, signal\_b should not be asserted. Write an assertion.
5. The signal\_a is a pulse; it can only be asserted for one cycle, and must be deasserted in the next cycle.
6. Signal\_a and signal\_b can only be asserted together for one cycle; in the next cycle, at least one of them must be deasserted.
7. . When signal\_a is asserted, signal\_b must be asserted, and must remain up until one of the signals signal\_c or signal\_d is asserted.
8. After signal\_a is asserted, signal\_b must be deasserted, and must stay down until the next signal\_a.
9. If signal\_a is received while signal\_b is inactive, then on the next cycle signal\_c must be inactive, and signal\_b must be asserted.
10. signal\_a must not be asserted together with signal\_b or with signal\_c.
11. In a RESP operation, request must be true immediately, grant must be true 3 clock cycles later, followed by request being false, and then grant being false.
12. Request must true at the current cycle; grant must become true sometime between 1 cycle after request and the end of time.
13. Req must eventually be followed by ack, which must be followed 1 cycle later by done.
14. The active-low reset must be low for at least 6 clock cycles.
15. Enable must remain true throughout the entire ack to done sequence.
16. Write an assertion for glitch detection.
17. If signal\_a is active, then signal\_b was active 3 cycles ago.
18. If the state machine reaches active1 state, it will eventually reach active2 state.
19. Write an assertion: A high for 5 cycles and B high after 4 continuous highs of A and finally both A and B are high?
20. Write an assertion: On rose of a, wait for rose of b or c. If b comes first, then d should be 1. If c comes first d should be zero.