Module: SV for Verification
Section: Randomization **Task:** Immediate Assertion

Task 3

Immediate Assertions

➢ **Part 1: Pre_randomize vs Post_randomize (EDA Project)**

➢ **Part 2:**

1. **Severity Tasks:**
   In SystemVerilog, assertion messages are categorized based on severity levels, and there are four system tasks used for this purpose: $fatal, $error, $warning, and $info. These system tasks help classify the importance of each message. If an action block is specified, users can define custom severity levels using these tasks.

   - **$fatal**: This generates a fatal run-time error and immediately terminates the simulation with an error code. The first argument passed to $fatal must follow the same conventions as the first argument to $finish. Calling $fatal implicitly triggers a call to $finish.
   - **$error**: This indicates a run-time error but does not terminate the simulation.
   - **$warning**: This issues a run-time warning, signaling that something may be wrong, but it's not critical enough to halt the simulation.
   - **$info**: This is used for informational messages, indicating that the message has no particular severity associated with it.

2. **Niall's Comment:**
   Niall suggests that while severity tasks were originally constrained to assertions in earlier standards (IEEE 1800-2005), they were later expanded for broader use in procedural code (starting from IEEE 1800-2009). However, some simulators, like Cadence, still follow the older restriction.

3. **Cadence:**
   Cadence reports compilation errors when severity tasks ($info, $warning, $error, and $fatal) are called outside of SystemVerilog assertions because it follows the older **IEEE 1800-2005** standard, which restricted these tasks to assertion blocks.

   Although the restriction was lifted in **IEEE 1800-2009**, allowing severity tasks to be used in any procedural code, Cadence has not fully adapted to the newer

standard. As a result, it flags errors when severity tasks are used outside assertions, unlike other simulators that follow the updated standards.

4. **Solution:**
   The suggested solution to Cadence not supporting severity tasks outside of SystemVerilog assertions is to wrap the severity tasks inside trivial assertions.

   For example, you can use assert(1'b1) to ensure the severity task is placed in an assertion block, which satisfies the older **IEEE 1800-2005** standard that Cadence follows. This workaround prevents compilation errors while maintaining functionality.

   Ideally, Cadence should update its simulator to comply with the newer **IEEE 1800-2009** standard, which allows severity tasks to be used in any procedural code.