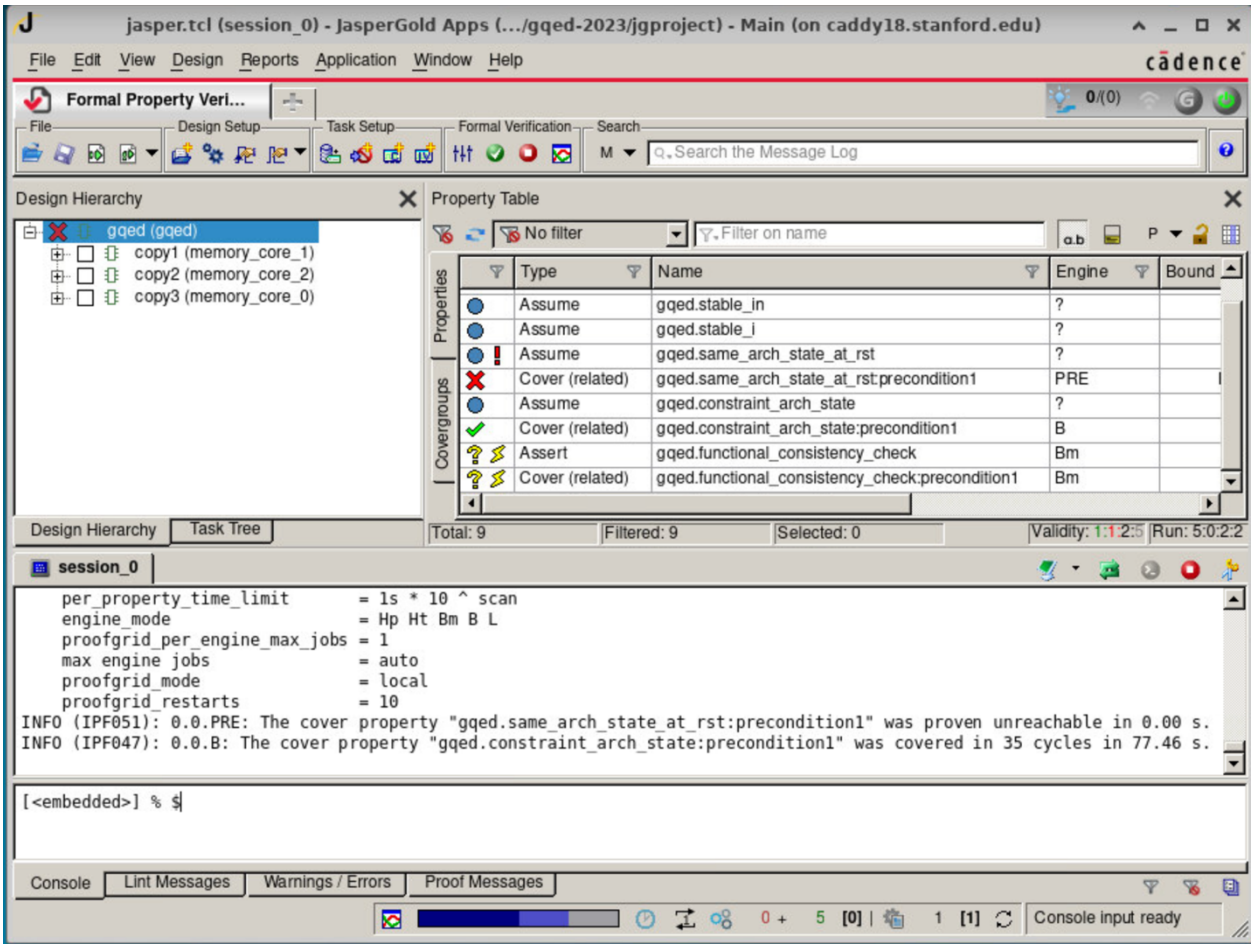
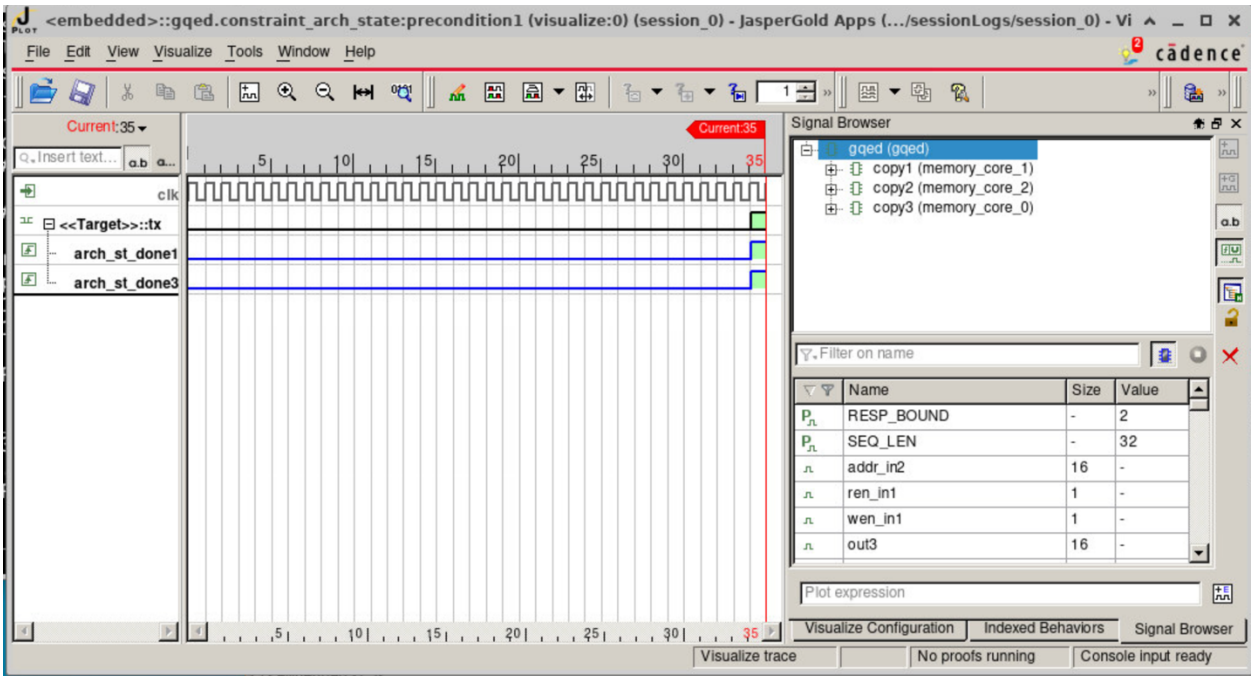
EE 271 Final Project Part 4: G-QED

Hugo Chen, Qi Jiang





A brief description of the bug found.

We have the bug found in the gqed constraint_arch_state:precondition1. It could cause by the functional error in our G-QED code.

One or more points on how to improve the G-QED module coverage.

Improving the coverage of the G-QED module involves enhancing the verification strategy to ensure a more thorough exploration of the design space.

1. Variable Sequence Lengths (SEQ_LEN):

Instead of fixing the sequence length (i) during BMC tool initialization, consider exploring various sequence lengths to cover a broader range of scenarios. Vary the sequence length from short to long to ensure the verification process is not biased towards specific input lengths.

2. Randomization of Inputs:

Introduce randomization in the generation of input sequences to explore a wider input space.
Use randomization to cover edge cases and scenarios that may not be captured by deterministic sequences.

3. Dynamic Architecture State Initialization:

Dynamically initialize the architecture state of the third copy in the G-QED module based on the saved state from the first copy. Consider varying the initialization point within the saved architecture state to ensure comprehensive coverage.

4. Multiple Functional Consistency Checks:

Perform functional consistency checks for multiple sets of copies, introducing variations in the architecture state tracking or output saving methods. Create additional copies or variations to handle different aspects of the design, such as different memory configurations or operational modes.

5. Sequential and Concurrent Input Execution:

Explore scenarios where inputs are executed sequentially and concurrently across the three copies. Test how the design behaves when inputs are applied in parallel or in specific sequential orders.

6. Explore Design Idling Scenarios:

Investigate different conditions for design idling, such as variations in the duration of idle periods or the criteria for transitioning to idle states. Test how the design handles transitions between active and idle states.