

IC Lab Formal Verification

Bonus Report 2025 Spring

Name: 吴挺宇

Student ID: 311651052

Account: iclab139

(a) What is Formal verification?

What's the difference between **Formal** and **Pattern** based verification?

And list the pros and cons for each.

The formal verification:

Formal verification refers to applying mathematical formal methods to rigorously prove or refute whether the intended algorithms within a system adhere to a specific formal specification or satisfy certain defined properties.

Pros:

1. Less testbench effort required.
2. Using systematic method.
3. Improves productivity and schedule.
4. Leads to higher quality

Cons: It takes more time to verify.

Pattern verification:

1. depth-first search
2. Test only one state in one cycle. It will repeat the same combination, because it is random test.

Pros: Free to use(often). Easy to use. We can test the tasks what we think immediately.

Cons:

1. We probably miss some case.
2. It takes more time to write testbench.

(b) Explain SVA (SystemVerilog Assertions) and the roles of Assertion, Cover, and Assumption.

What is glue logic?

Why will we use **glue logic** to simplify our SVA expression?

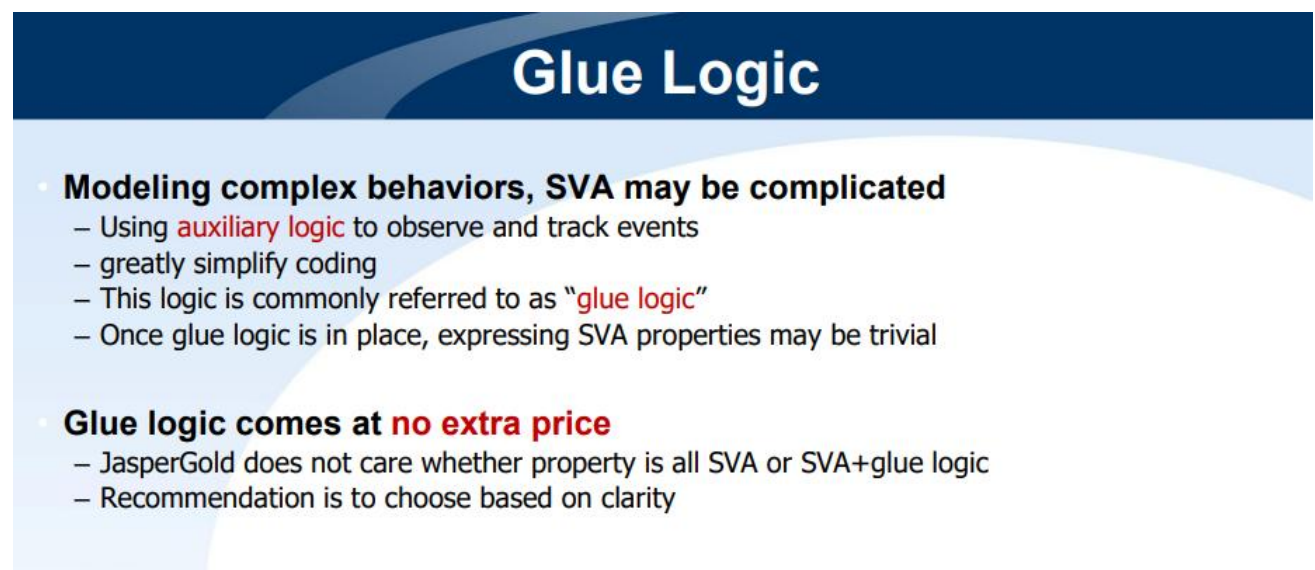
SVA is a language for expressing properties

- Not only assertions, but covers and assumptions too!
- Can be mixed with Verilog, SystemVerilog, and VHDL

Assertion : To verify whether the design behaves correctly.

Cover : To check whether a certain behavior ever occurs.

Assumption : To constrain the environment with assumed conditions that the design relies on.



Glue Logic

- **Modeling complex behaviors, SVA may be complicated**
 - Using **auxiliary logic** to observe and track events
 - greatly simplify coding
 - This logic is commonly referred to as “**glue logic**”
 - Once glue logic is in place, expressing SVA properties may be trivial
- **Glue logic comes at no extra price**
 - JasperGold does not care whether property is all SVA or SVA+glue logic
 - Recommendation is to choose based on clarity

1. glue logic is the custom logic circuitry used to interface a number of off-the-shelf integrated circuits.

2. Modeling complex behaviors with glue logic, while SVA may be complex, and if glue logic is in place, expressing SVA properties may become trivial. Writing complex properties using glue logic also greatly simplifies coding. Using simple Verilog logic to track events/state could result in better coding style.

(c) What is the difference between **Functional coverage** and **Code coverage**?

What's the meaning of 100% code coverage, could we claim that our assertion is well enough for verification? Why?

Functional coverage:

1. Describe specific states, conditions, or sequences to be verified.
2. Requires planning, coding, and debug
3. Noise-free – nothing is don't-care
4. Possible to represent all meaningful design functionality
5. Implements the verification plan – what needs to be verified
6. May be incomplete due to human error

Code coverage:

1. easy to generate automatically.
2. Guaranteed to be structurally complete – not prone to human error
3. Can be noisy – “don't-care” or duplicate covers
4. It's may not able to capture all meaningful design functionality
5. Standard models capture much of the meaningful behavior of the design

100% code coverage means every line of your RTL (Register Transfer Level) code is being executed. Yet, this doesn't assure the correctness of your RTL code by itself. Essential functionalities or computations in the design may be missing from your RTL implementation. Hence, even with full code coverage, there remains a possibility of incorrect functionality

- (d) What is the difference between **COI coverage** and **proof coverage** for realizing checker's completeness? Try to explain from the meaning, relationship, and tool effort perspective.

Meaning:

COI coverage: Each assertion affected by some cover items.

proof coverage: Find the region cannot truly influence assertion status.

Relationship :

COI coverage: COI represents the maximum potential of proof coverage.

proof coverage: Subset of the COI.

Find the region **cannot truly influence assertion status**

– Example:

– assign a = b+c; } COI
– assign m = a; } coverage } Proof coverage
– **assert** property @(posedge clk) a == m;

Represents the portion of the design verified by formal engines

– **Subset of the COI** – COI represents the maximum potential of proof coverage

(e) What are the roles of **ABVIP** and **scoreboard** separately?

Try to explain the definition, objective, and the benefit.

ABVIP :

Definition: The Assertion Based Verification Intellectual Properties (ABVIPs) are a comprehensive set of checkers and RTL that check for protocol compliance.

Objective: Verifying a protocol and analyzing its completeness is a key challenge for engineers these days Benefit: We can use it without writing properties by ourselves.

Scoreboard:

Definition: Scoreboard behaves like a monitor

Objective: Observe input data and output data of DUV Benefit: We can use it to check the simple cases and reduce barrier for adoption.

- (f) Among the JasperGold tools (Formal Verification, SuperLint, Jasper CDC, IMC Coverage), which one do you think is the most effective based on its functionality and typical application scenarios? Please explain your reasoning by describing a hypothetical scenario where this tool would be particularly beneficial, and discuss any potential challenges or limitations that might arise when using it.

In the verification process, I think IMC Coverage is the most effective tool because it can quickly determine whether the issue lies in my checker or the pattern itself based on the number of cover hits.

In Lab 10, when the coverage test initially didn't reach 100%, I checked which items in the IMC were not hit. Then, I simply needed to cover those missed items to achieve the target.