IC Lab Formal Verification  
Bonus Report 2024 Fall

**Name: 何東員 Student ID: 313580068 Account: iclab173 y**

1. What is Formal verification?

透過數學或邏輯方法來驗證設計的正確性。

它可以一次測試一個時鐘週期內的所有可能輸入。

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

And list the pros and cons for each.

主要區別：形式驗證通過數學和邏輯方法證明系統的正確性，而模式驗證是每次測試一組輸入組合。

**Formal：**

優點：

* 高精確度
* 可重用性

缺點：

* 時間耗費大

**Pattern：**

優點：

* 高效率

缺點：

* 可能錯失一些邊界情況

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

**Assertion:**是用來檢查電路設計是否符合特定條件或規則的表達式。如果不符合，會觸發報錯。

**Cover:**用於驗證測試是否達到了設計的某些關鍵點，特別是在仿真中確保某些場景被觸發。

**Assumption:**主要用於約束環境或輸入信號的行為，讓驗證工具能更有效地執行驗證。

What is glue logic?

是指連接不同模組或功能塊之間的中介邏輯電路，通常用於格式轉換、信號對應或簡化邏輯實現。這些邏輯可以是multiplexers, decoders AND, OR gates, and similar components.

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

1. 簡化**Assertion**表達式 : **glue logic**可以將複雜的條件預先處理，生成簡單的中間信號，減少SVA中直接使用的條件複雜度。這樣可以提高SVA表達式的可讀性和易維護性。
2. 提高性能 : 在驗證中，簡化後的SVA表達式能減少驗證工具的計算負擔，從而提高執行效率。
3. 易於調試  
   通過**glue logic**提取中間信號，可以幫助工程師更方便地定位問題，因為每個信號的作用清晰且單一。
4. What is the difference between **Functional coverage** and **Code coverage**?

**Functional Coverage**：檢查設計在不同條件下的功能和規格是否正確。它由設計者設計，可能因人為錯誤或考慮不足而不完整。

**Code Coverage**：測試 RTL 代碼的執行情況（如分支、語句和表達式）。由 EDA 工具生成，避免人為錯誤，但有時會檢查一些不必要的部分。

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

不能。100% 代碼覆蓋僅意味著 RTL 代碼的每一行都被執行過，但功能未必正確。

1. 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：回溯影響通過斷言檢查的邏輯的所有覆蓋項，並驗證這些項目。

proof coverage：將電路簡化為數學邏輯模型，只驗證直接影響檢查邏輯的覆蓋項目。這種驗證通過形式引擎完成，針對每個週期的所有可能情況。

**Relationship**: Proof 覆蓋是 COI 覆蓋的子集。

**Tool Effort**：

COI coverage：檢查影響輸出的覆蓋區域，資源需求低。

proof coverage：使用形式引擎，需更多資源和時間，資源需求高。

1. What are the roles of **ABVIP** and **scoreboard** separately?

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

**Definition**:

ABVIP：基於斷言的驗證 IP，是一組檢查器，用於檢查設計是否能與特定協議通信。

Scoreboard：類似監視器，用於觀察輸出數據是否符合預期，測試設計的正確性。

**Objective**:

ABVIP：提供一個設計良好的工具，幫助設計者檢查複雜的協議，設計者可以使用 ABVIP 驗證協議，而不是自己撰寫檢查器。

Scoreboard：檢查設計的功能是否正確。

**Benefit**:

ABVIP：為設計者提供更容易的協議檢查方法，且精度更高。

Scoreboard：降低了狀態空間的複雜性。

1. 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.

Formal Verification是我認為是最具效能的工具，因為它可以在不依賴test vectors的情況下，全面驗證設計是否符合一組指定的Property。

我正在設計一個 訪問控制器，該控制器管理不同用戶對共享記憶體的訪問。它有多種操作模式，包括：

1. 只讀模式：僅允許讀取操作。
2. 寫入模式：僅允許特定用戶寫入。
3. 混合模式：多用戶可讀可寫，但必須保證寫入與讀取不發生競爭。

訪問控制器的核心需求是：

* 在任何操作模式下，必須保證不發生Data Race。
* 使用者變更模式時，需確保系統在 3 個cycle time內穩定過渡到新模式。

|  |  |  |
| --- | --- | --- |
| **特性** | **傳統隨機測試** | **Formal Verification** |
| **覆蓋率** | 無法保證完整覆蓋，需手動調整測試集 | 全面覆蓋，數學分析所有狀態 |
| **稀有情況測試** | 隨機生成難以觸發 | 確保覆蓋所有可能情況 |
| **測試數量** | 需要大量測試用例 | 僅需定義屬性 |
| **問題定位** | 波形分析耗時 | 生成反例，快速找出問題 |

challenges

**Complexity Explosion**

對於大型複雜的設計，Formal Verification可能因狀態空間的爆炸性增長而遇到困難，導致計算資源需求過高或無法完成驗證。

**Tool Expertise**  
形式驗證工具具有一定的學習曲線，對於不熟悉的用戶來說可能難以有效使用。