Second Generation Completeness Analysis of Formal Assertions on Compatibility of RISC-V Cores

Wayne Yun

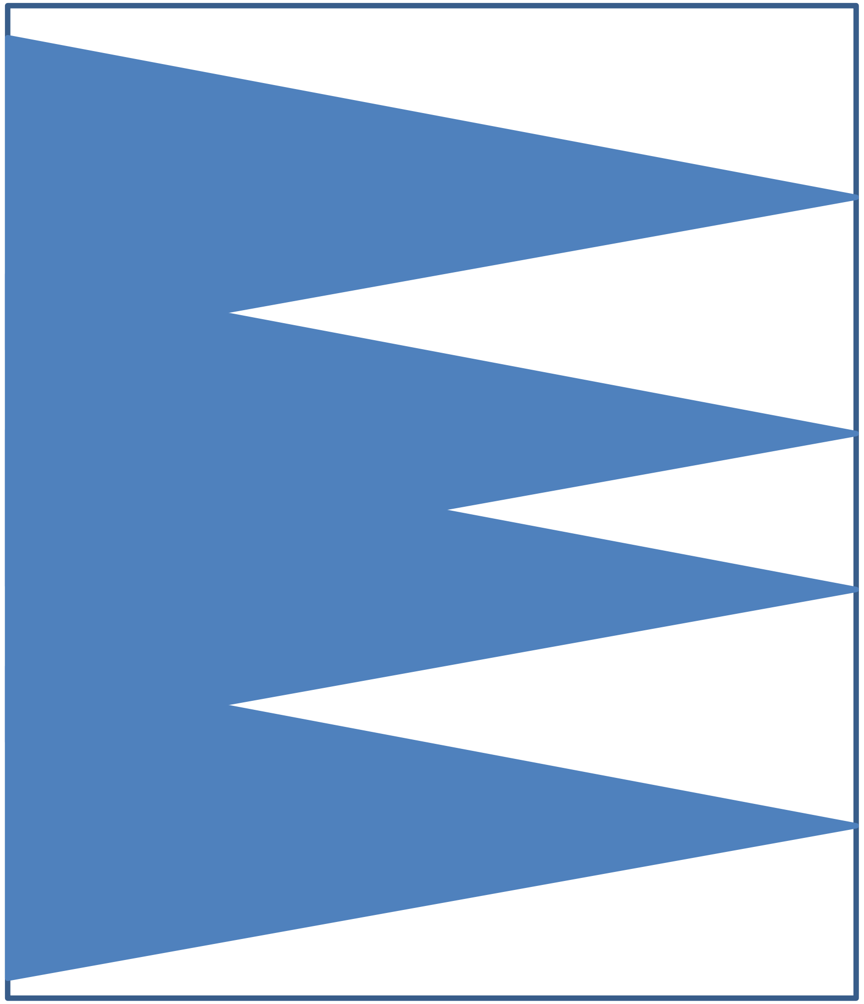
Advanced Micro Devices, Inc.

Content

- Metrics of formal assertion coverage
- Completeness Analysis of Formal Assertions (CAFA)
  - Output Enforcement Analysis (OEA, CAFA-I)
  - Output Uncertainty Analysis (OUA, CAFA-II)
- Examples and results

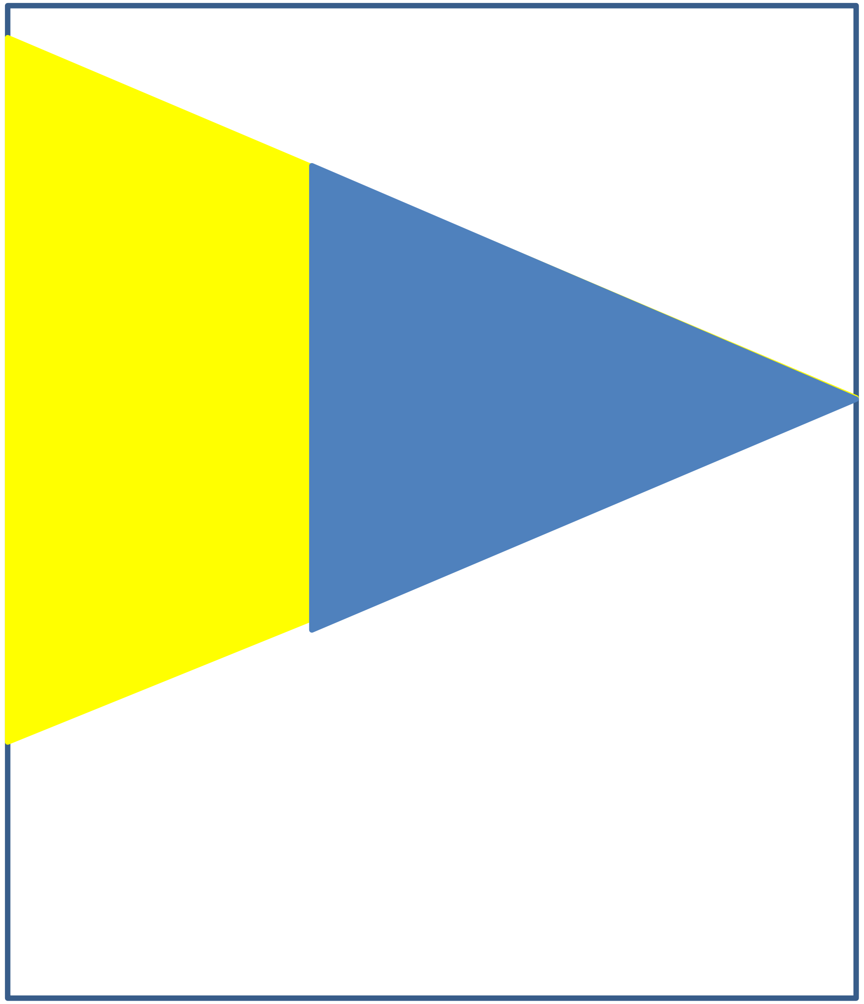
Formal Assertion Coverage

- Formal Property Verification (FPV)
- Measuring corner cases checked
- Finding gaps between assertions
- Improving quality of verification, sign-off
- Not measured can become not verified



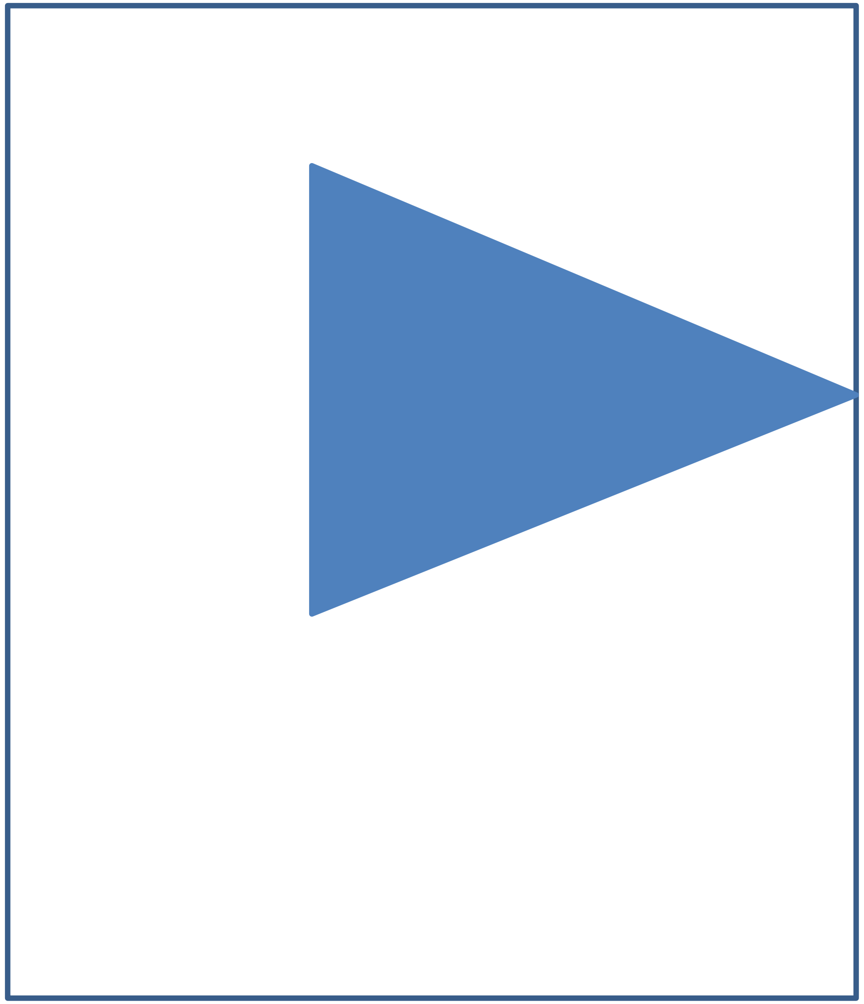
Example - Cone of Influence

- Logic and inputs leading to an assertion
- Part of it may be not checked
  - False positive
  - May leak design bug



Example - Formal Proof Core

- Logic and inputs needed to prove an assertion
- It can be a weak measurement for assertions even if the score is perfect
- A piece of logic is marked as covered once one of its use cases is checked
- Other cases may be not checked
  - False positive
  - May leak design bug



Example - Mutation Coverage

- Faults are injected into design
- A fault can be involved in multiple scenarios
  - The fault is covered if it is detected in any scenario
- Other scenarios may be not checked
  - False positive
  - May leak design bug

Completeness Analysis of Formal Assertions

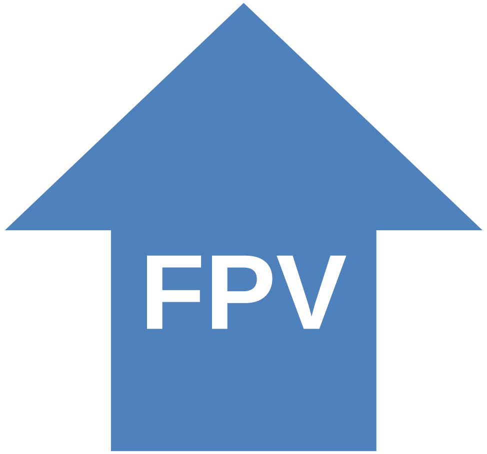
- Newly invented metrics of assertion coverage
  - Innovative, focusing on outputs rather than internal logic
  - Pure generic formal technologies
  - Distinguishing theoretical completeness

- Two members
  - Output Enforcement Analysis (OEA, CAFA-I) - DAC 2019 Las Vegas
  - **Output Uncertainty Analysis (OUA, CAFA-II) - this presentation**

A Simple FPV Example

DUT

```
module dut (input clk, input rst, input a, output o);
  reg q, qq; assign o = qq;
  always @(posedge clk) q <= rst ? 1'b0 : a;
  always @(posedge clk) qq<= rst ? 1'b0 : q;
endmodule
```



ASSERT

```
module fpv (input clk, input rst, input q, input a, input o);
  wire q; reg qq;
  always @(posedge clk) qq <= rst ? 1'b0 : q;
  m_r: assert property (@(posedge clk) $fell(rst) |-> !q);
  m_a: assert property (@(posedge clk) ##1 q == $past(a));
  m_o: assert property (@(posedge clk) o==qq);
endmodule
```

Example of OEA


```
module dut (input clk, input rst, input a, output o);
  reg q, qq; assign o = qq;
  always @(posedge clk) q <= rst ? 1'b0 : a;
  always @(posedge clk) qq<= rst ? 1'b0 : q;
endmodule
```

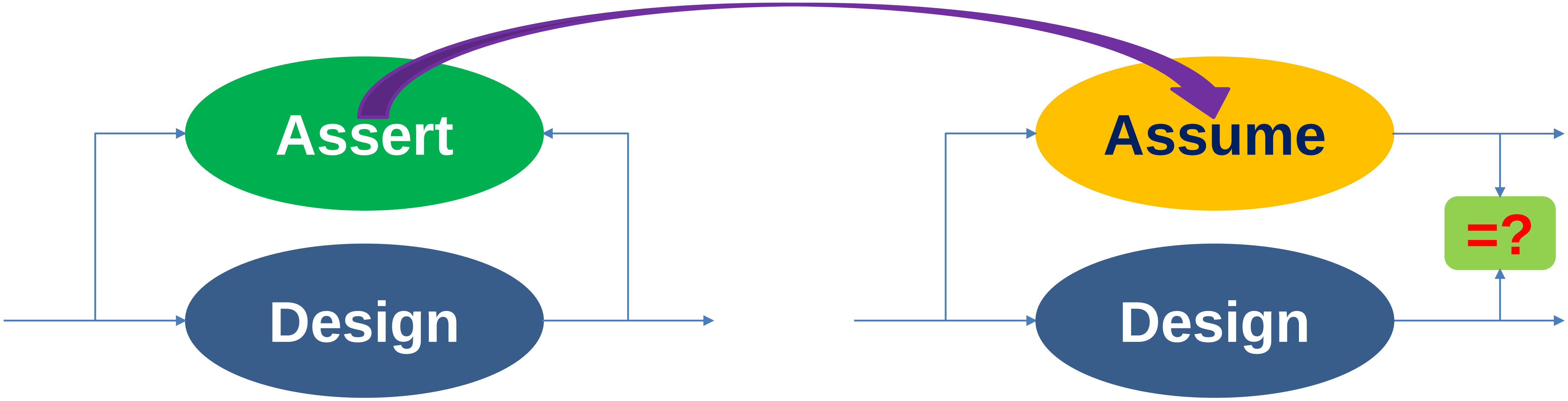
```
module cafa (input clk, input rst, input a, output o);
  wire q; reg qq; assign o = qq;
  always @(posedge clk) qq <= rst ? 1'b0 : q;
  m_r: assume property (@(posedge clk) $fell(rst) |-> !q);
  m_a: assume property (@(posedge clk) ##1 q == $past(a));
endmodule
```

```
module fpv (input clk, input rst, input q, input a, input o);
  wire q; reg qq;
  always @(posedge clk) qq <= rst ? 1'b0 : q;
  m_r: assert property (@(posedge clk) $fell(rst) |-> !q);
  m_a: assert property (@(posedge clk) ##1 q == $past(a));
endmodule
```

Output Enforcement Analysis

1. Formal Property Verification

2. Sequential Equivalence Check



OEA metric is the percentage of outputs found equivalent

Example of OUA


```
module dut (input clk, input rst, input a, output o);
  reg q, qq; assign o = qq;
  always @(posedge clk) q <= rst ? 1'b0 : a;
  always @(posedge clk) qq<= rst ? 1'b0 : q;
endmodule
```

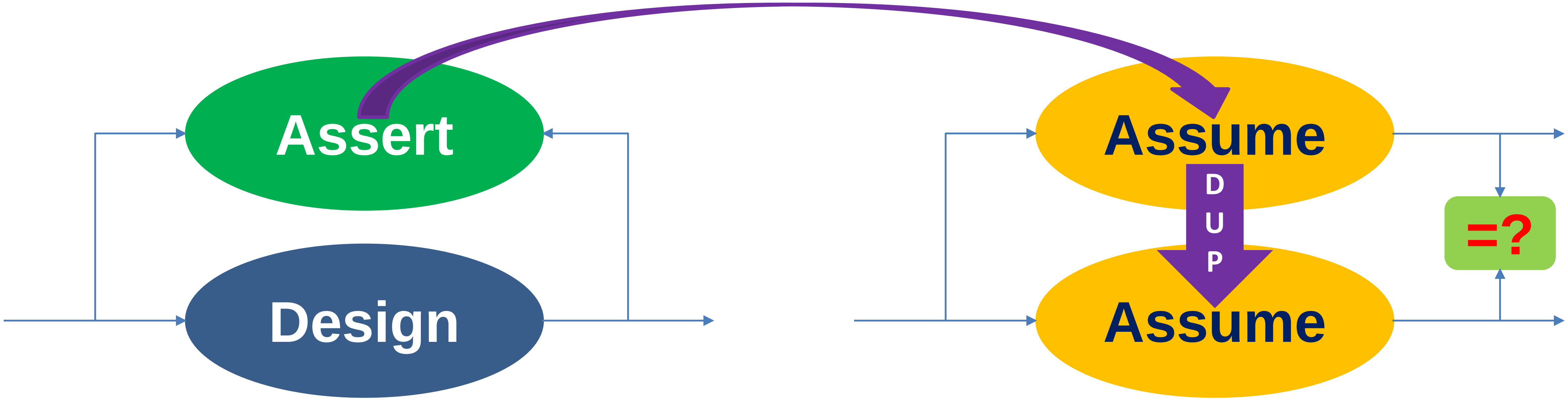
```
module cafa (input clk, input rst, input a, output o);
  wire q; reg qq; assign o = qq;
  always @(posedge clk) qq <= rst ? 1'b0 : q;
  m_r: assume property (@(posedge clk) $fell(rst) |-> !q);
  m_a: assume property (@(posedge clk) ##1 q == $past(a));
endmodule
```

```
module fpv (input clk, input rst, input q, input a, input o);
  wire q; reg qq;
  always @(posedge clk) qq <= rst ? 1'b0 : q;
  m_r: assert property (@(posedge clk) $fell(rst) |-> !q);
  m_a: assert property (@(posedge clk) ##1 q == $past(a));
endmodule
```

Output Uncertainty Analysis

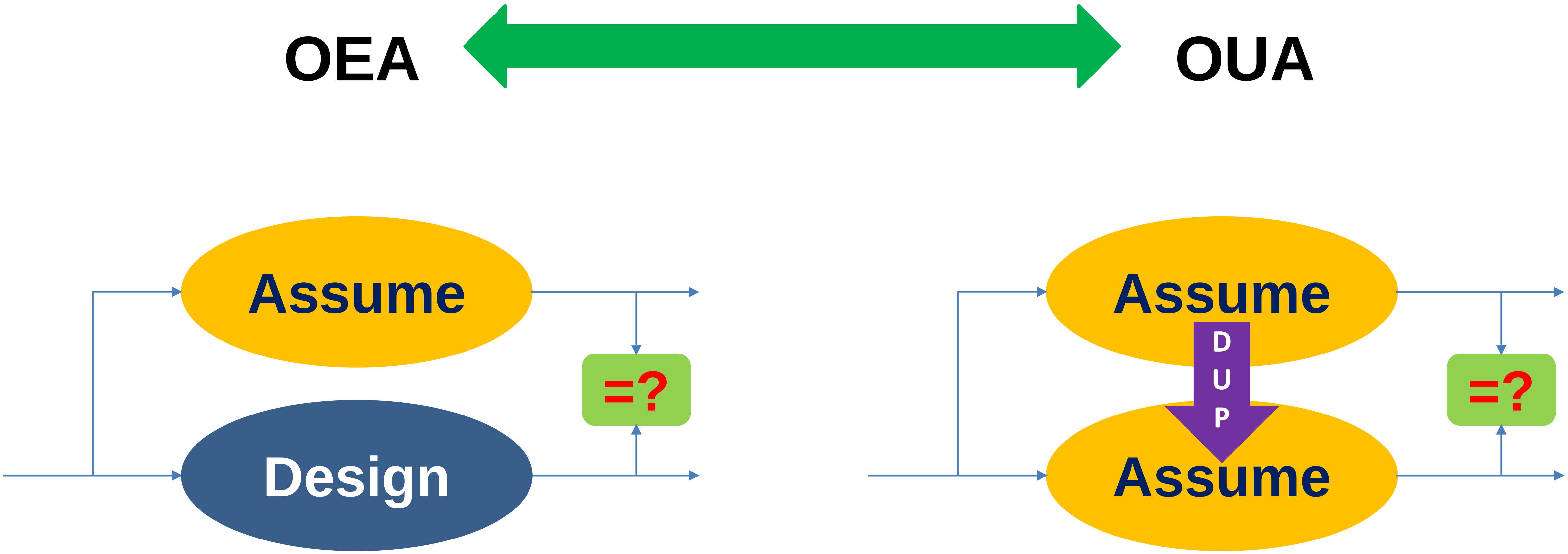
1. Formal Property Verification

2. Sequential Equivalence Check



OUA metric is the percentage of outputs found equivalent

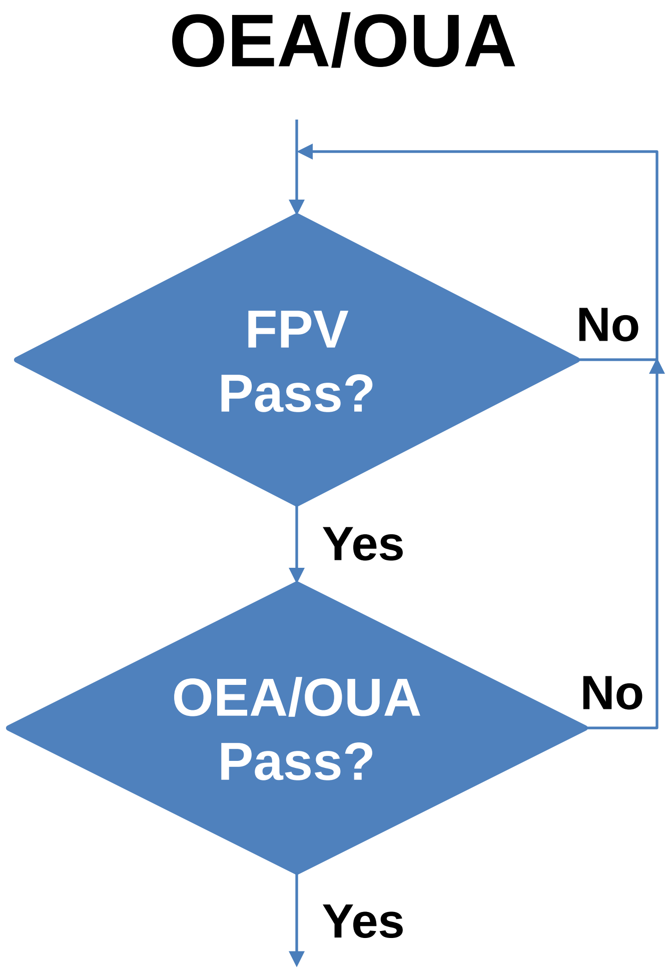
Identical Coverage



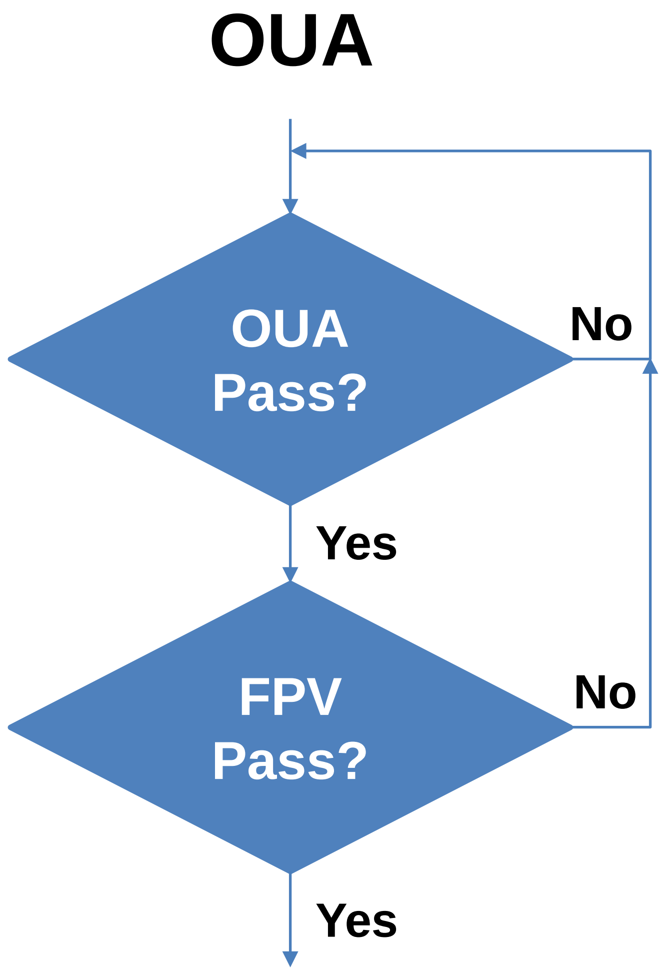
Pros and Cons

Aspect	OEA	OUA
Coverage	Identical	Identical
Waveform	Having Design	No Design
Dependency	Design	Not Design
Structure	Heterogenous	Symmetric
Performance	Baseline	Improved
Formal Engine	Generic	Generic

Workflow



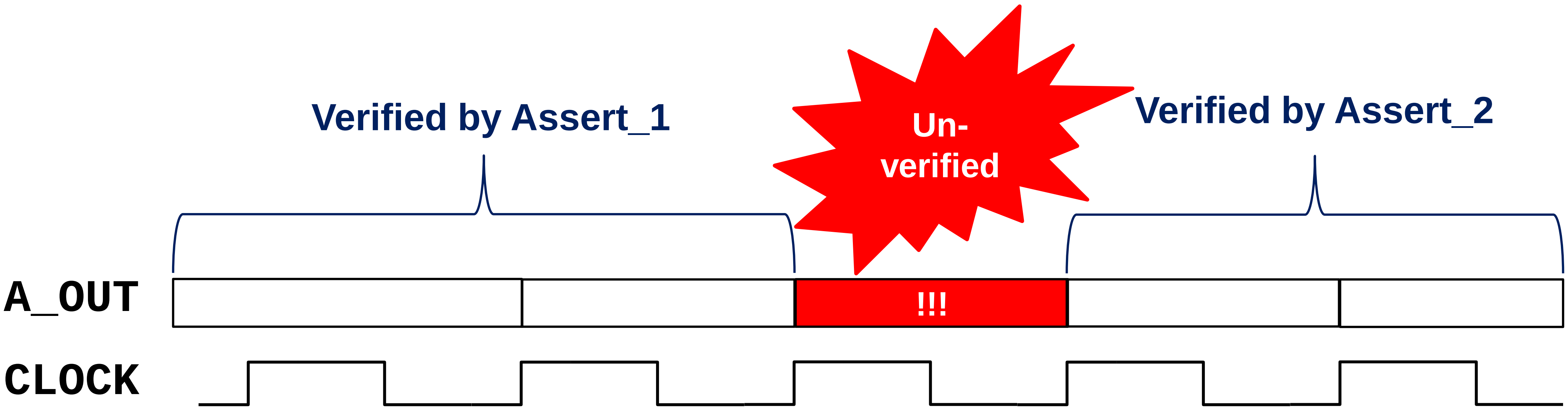
OEA/OUA
FPV Pass?
No
Yes
OEA/OUA Pass?
No
Yes



OUA
OUA Pass?
No
Yes
FPV Pass?
No
Yes

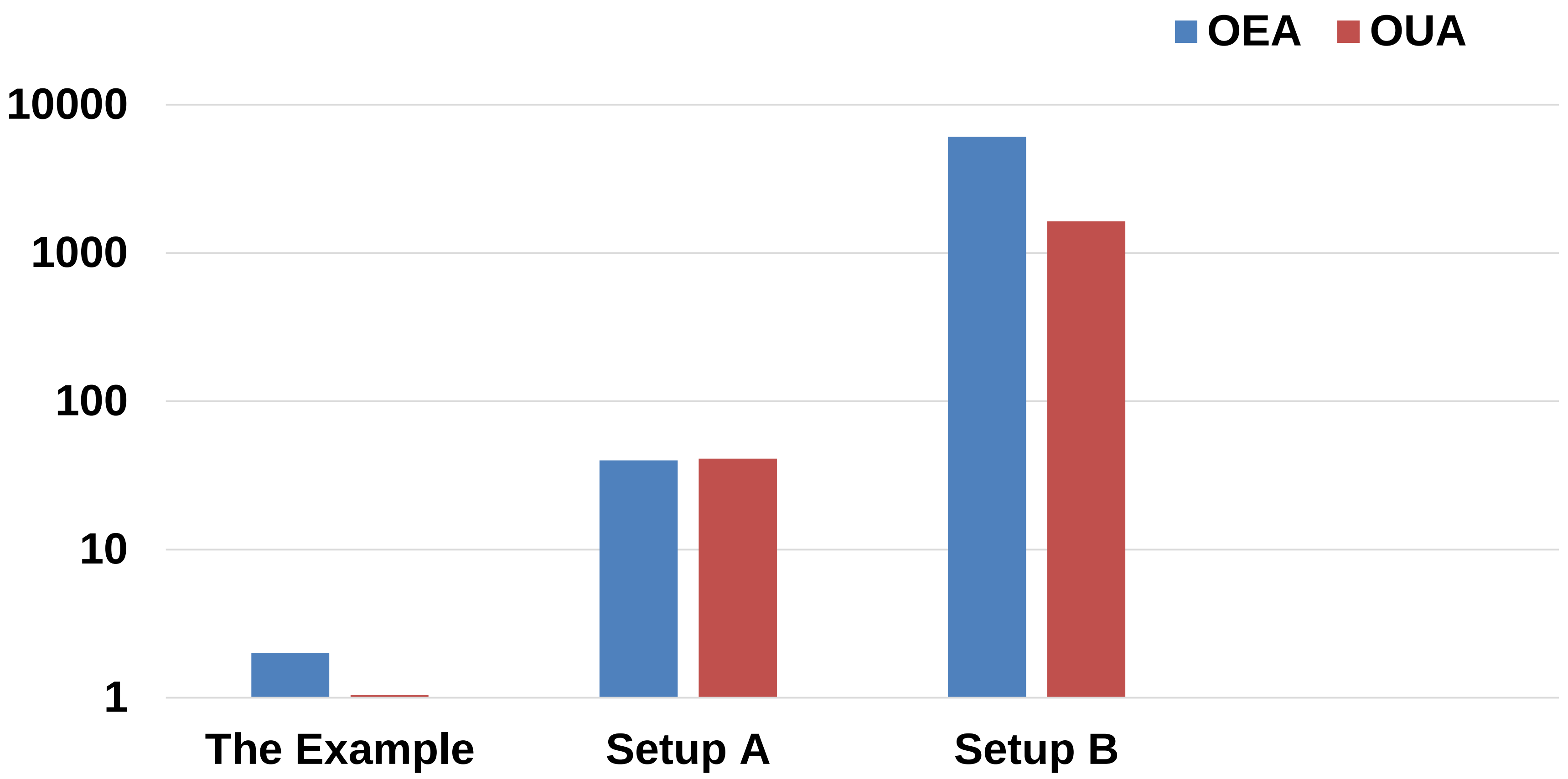
A Gap Caught by CAFA

- In a mature FPV setup
- COI, formal proof core and mutation coverage were positive

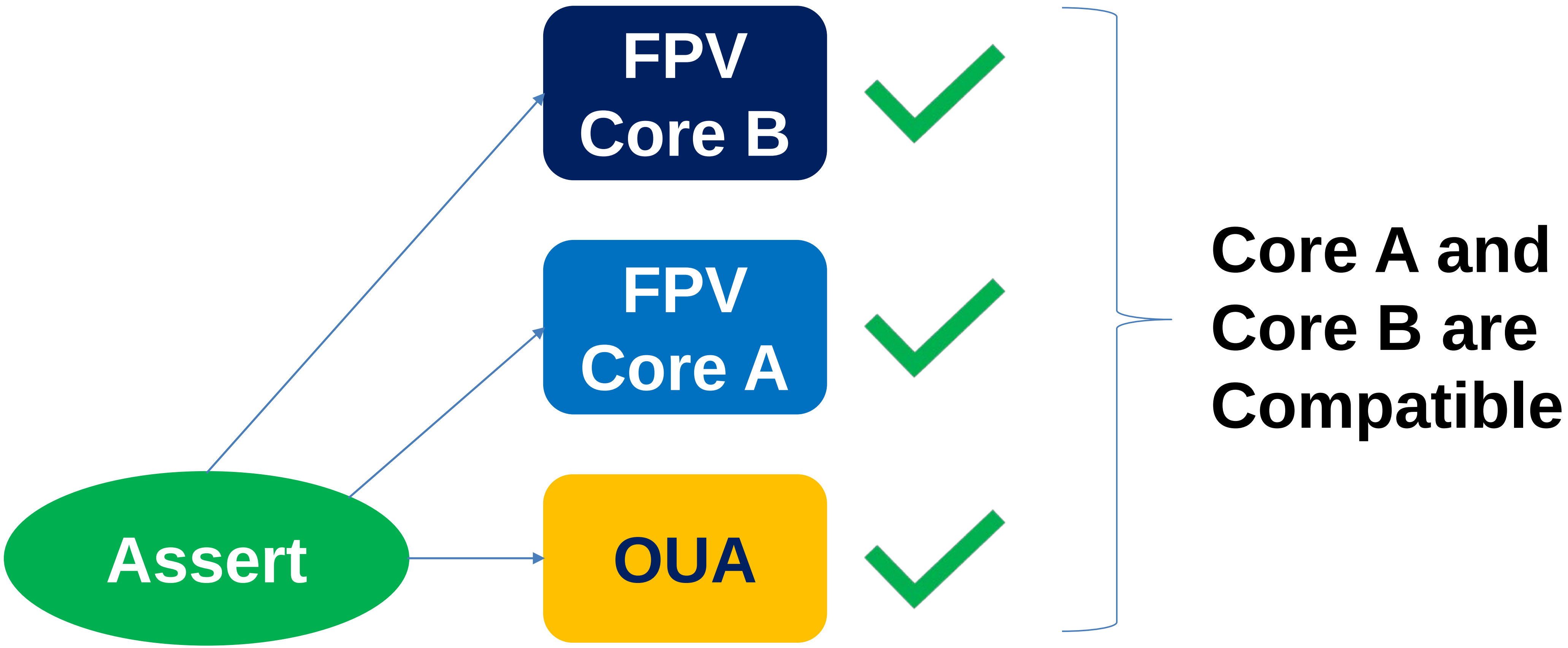


Performance

- Measured in wall-clock time (unit seconds)

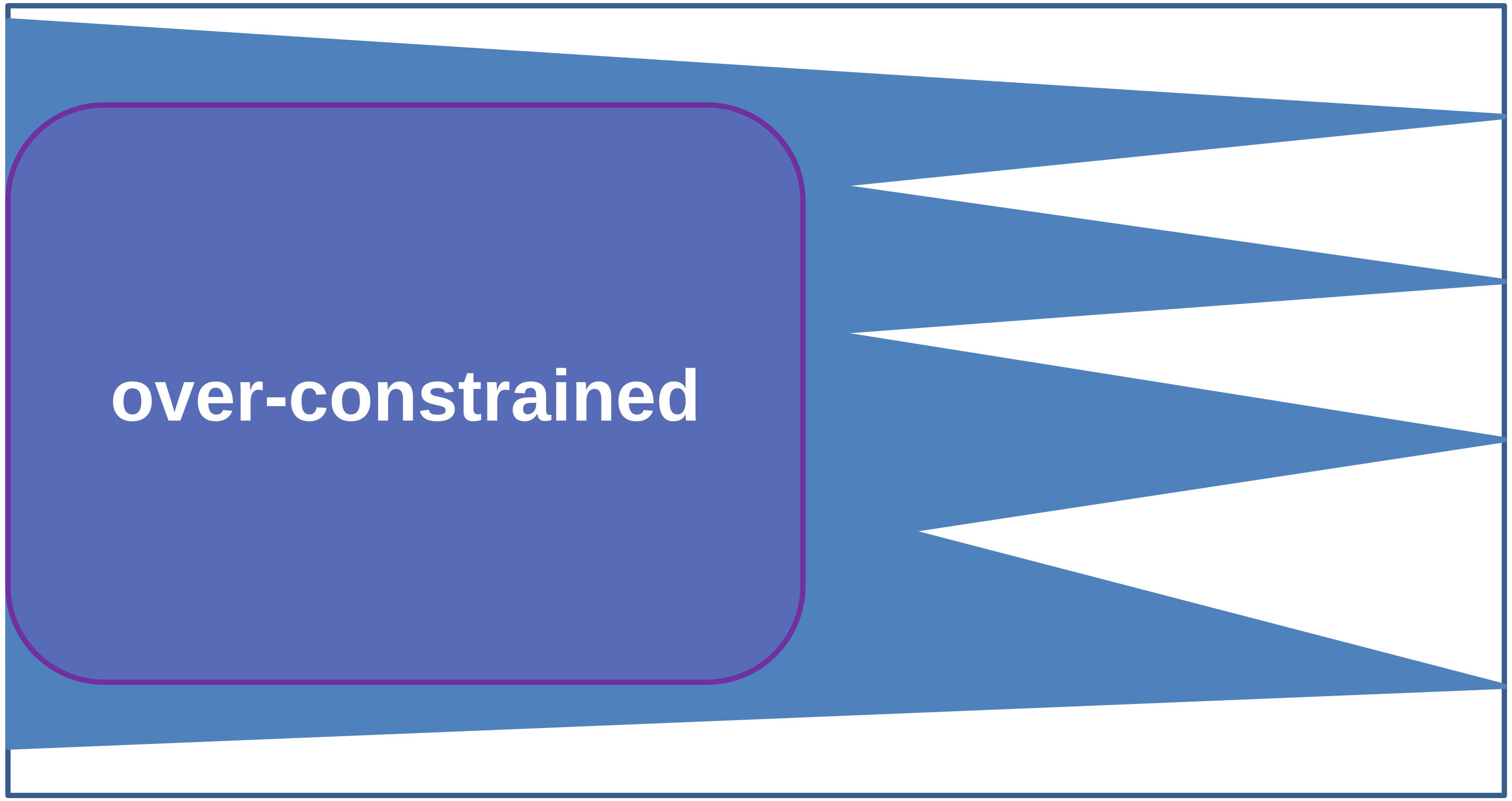


RISC-V Cores

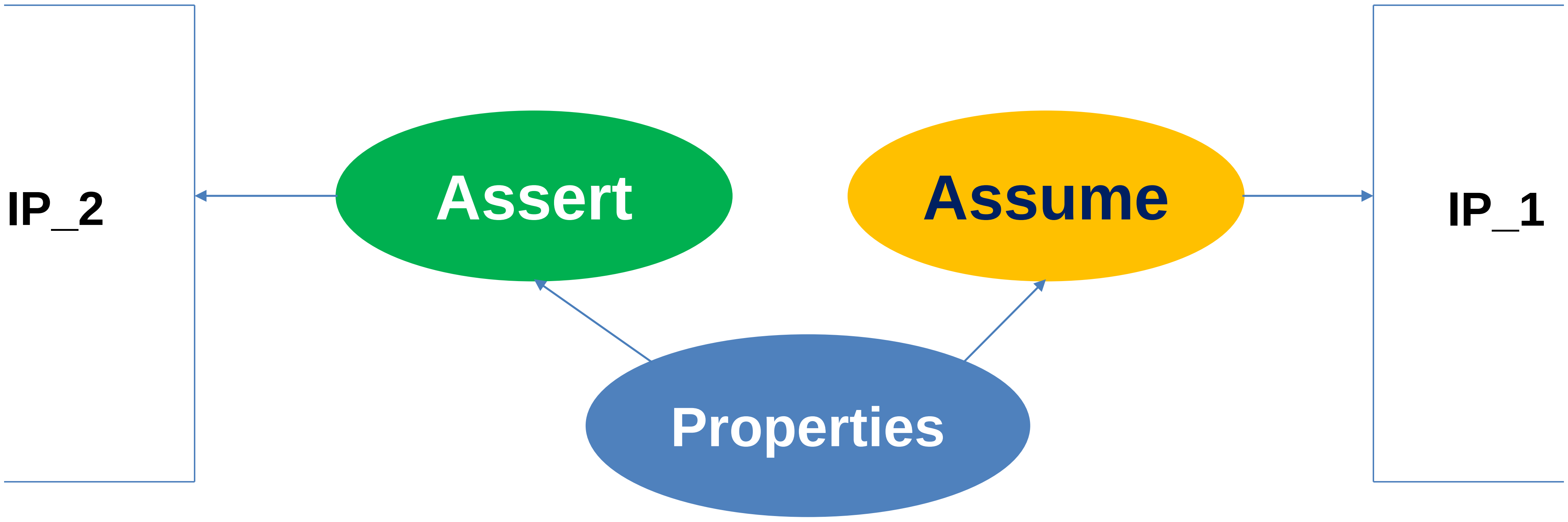

FPV
Core B
FPV
Core A
Assert
OUA
Core A and
Core B are
Compatible

Over-Constraint

- One major source of remaining issues



From Bricks to Wall



Summary

- COI, formal proof core and mutation coverage can be false positive
- CAFA - OEA and OUA - can find gaps between assertions
  - Theoretical completeness
- CAFA can solve some very challenging verification problems
- Advantages of OUA
  - Available ahead of the design – shift-left of verification tasks
  - Symmetric internal structure – opportunities of optimizing performance
- Improving quality, project schedule, and metric runtime

Questions ?

Disclaimer & Attribution

Disclaimer

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Attribution