2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION CHINA

accellera SYSTEMS INITIATIVE

ENGIANT

基于PSS中间层的验证平台架构设计

王磊、王锋、陈瑞

背景介绍 - PSS标准带来的机遇和挑战

- 验证场景抽象

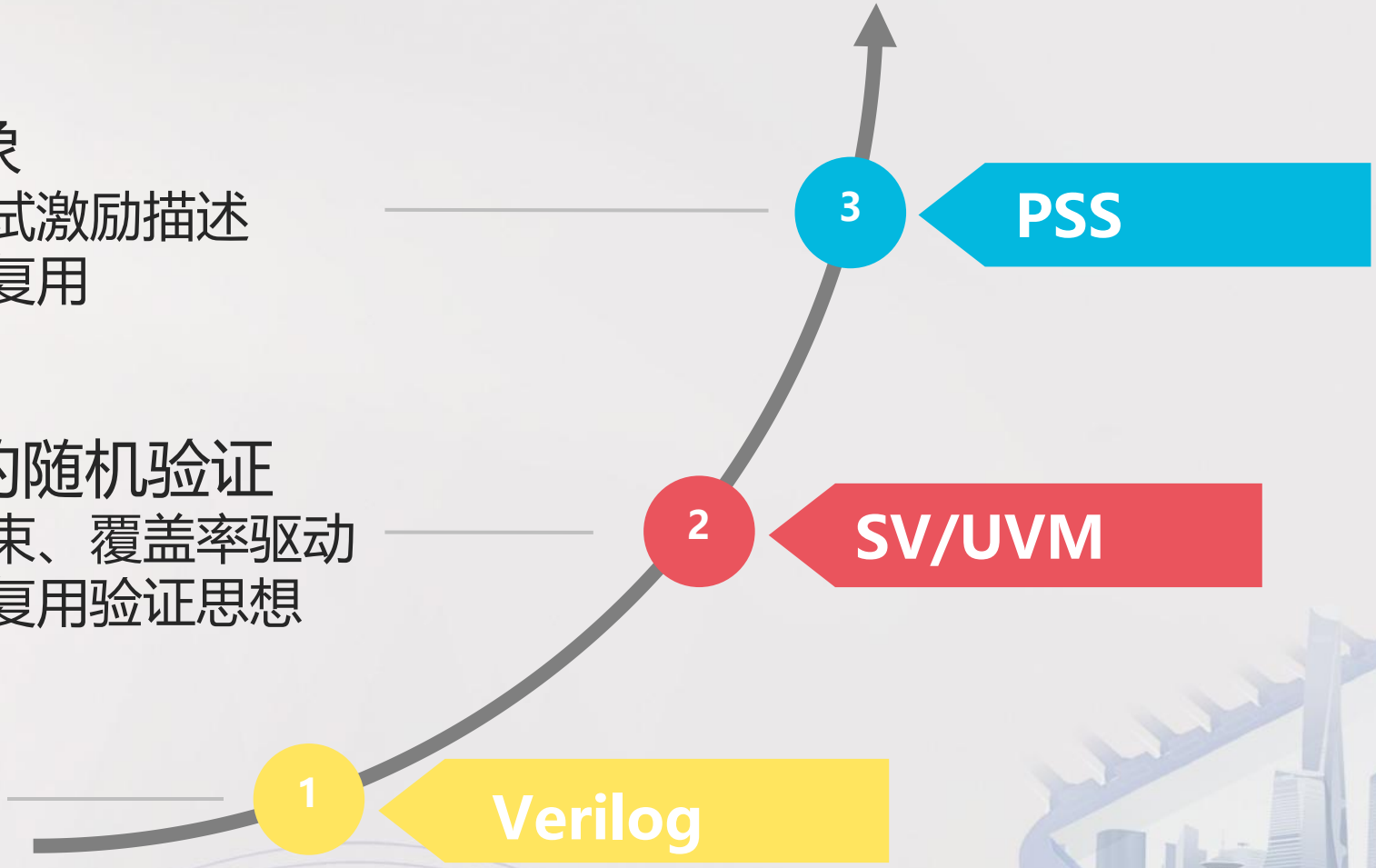
更高抽象级别的测试激励描述
更广泛的验证场景复用

- 覆盖率驱动的随机验证

面向对象、随机约束、覆盖率驱动
统一框架下的分层复用验证思想

- 行为级描述

定向测试



背景介绍 - PSS标准带来的机遇和挑战

➢ 目标

定义一个统一的可复用的高度抽象的测试场景和测试激励的模型

➢ 优势

- 场景模型可应用在不同的测试平台

  垂直复用：可在子模块、子系统和系统级验证平台上进行垂直复用

  水平复用：可在EDA、EMU、FPGA和post-silicon验证平台上进行水平复用
- 场景模型可映射到不同的目标测试语言

  通过PSS定义的执行块，可将场景模型映射到C/Verilog/SV/UVM等不同验证平台语言
- 静态覆盖率分析

  可以在场景生成阶段就能够计算出功能覆盖率，工具可通过它来优化随机求解过程

背景介绍 - PSS标准带来的机遇和挑战

➢ 挑战

- 谁来开发PSS场景模型

  系统工程师？验证工程师？软件工程师？

  DSL语言的学习曲线。
- 如何协调PSS模型和底层验证平台之间的协同工作

  PSS模型映射到底层验证平台的具体形式？
- 如何具体实现场景复用

  子模块、子系统、系统级平台如何复用？

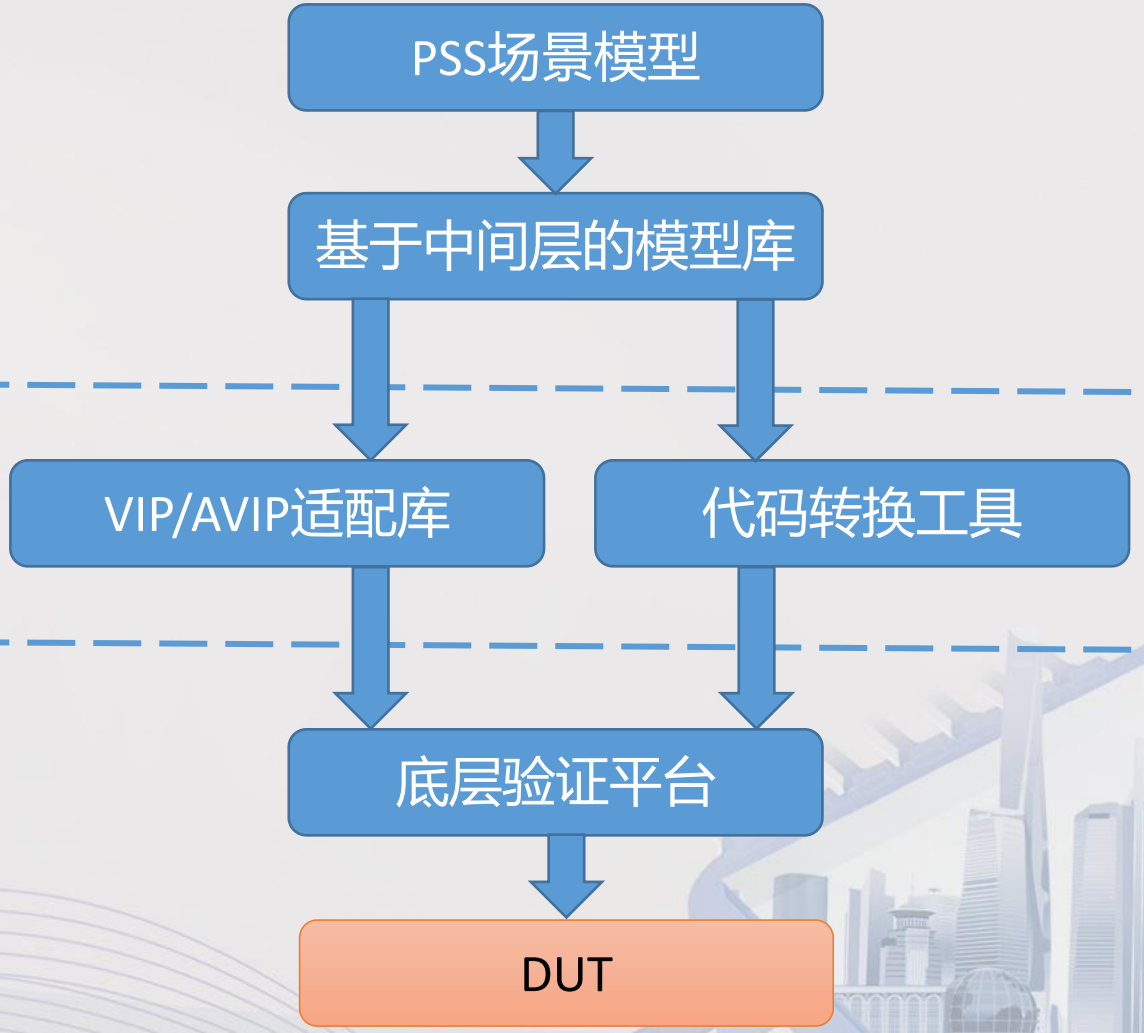
  EDA、EMU、FPGA、post-silicon平台如何复用？

基于PSS中间层的验证架构

场景抽象层

PSS中间层

事务层、驱动层



基于PSS中间层的验证架构
- VIP/AVIP适配库

➢ 挑战

- 芯片会集成多个不同种类的IP，对应不同的总线接口；
- 每个总线协议可能会有多种不同的VIP/AVIP供选择；
- 不同的VIP/AVIP的接口调用流程不相同；

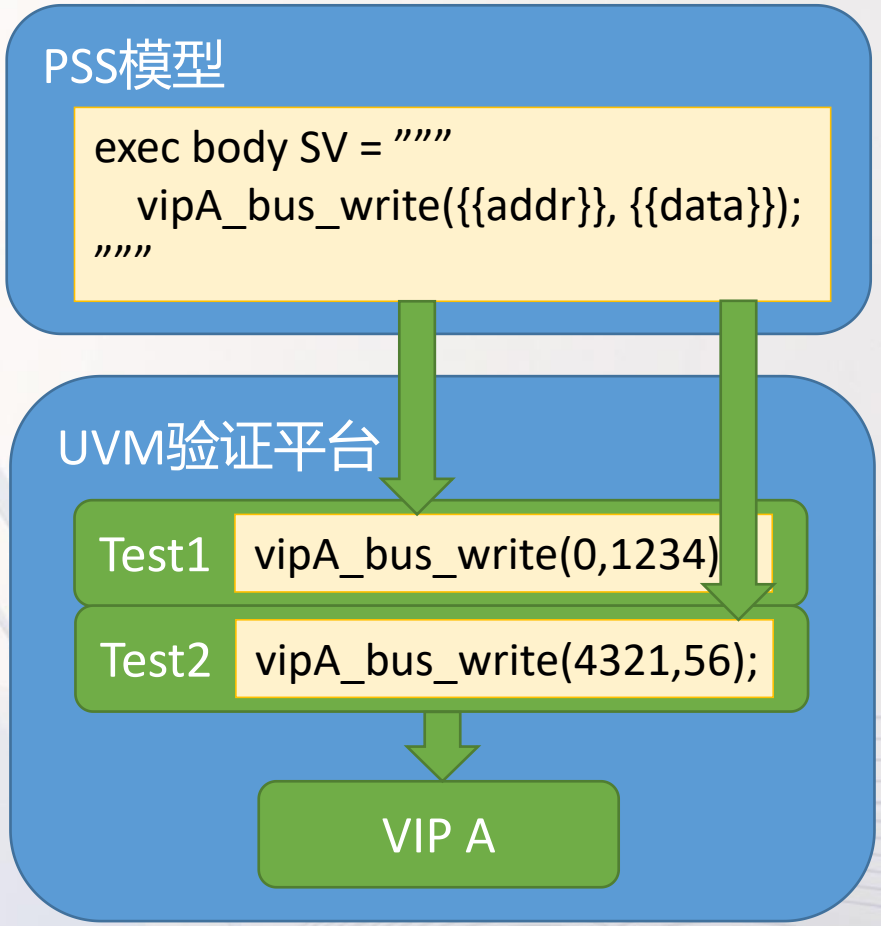
如何实现PSS模型与多种不同VIP/AVIP具体调用之间的对接？

用以下VIP A的接口为例：

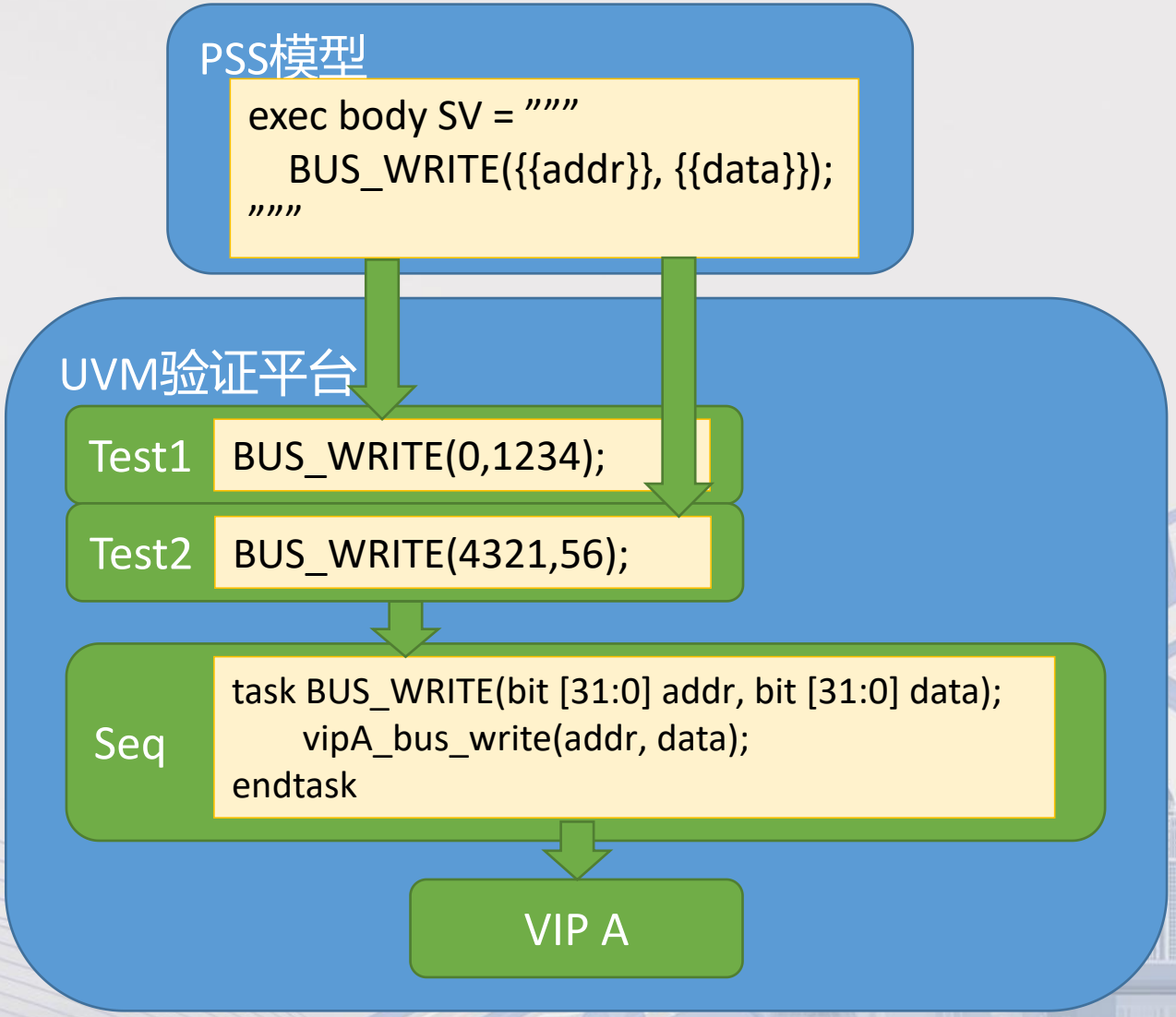
```
vipA_bus_write(bit [31:0] addr, bit [31:0] data)
```

基于PSS中间层的验证架构
- VIP/AVIP适配库

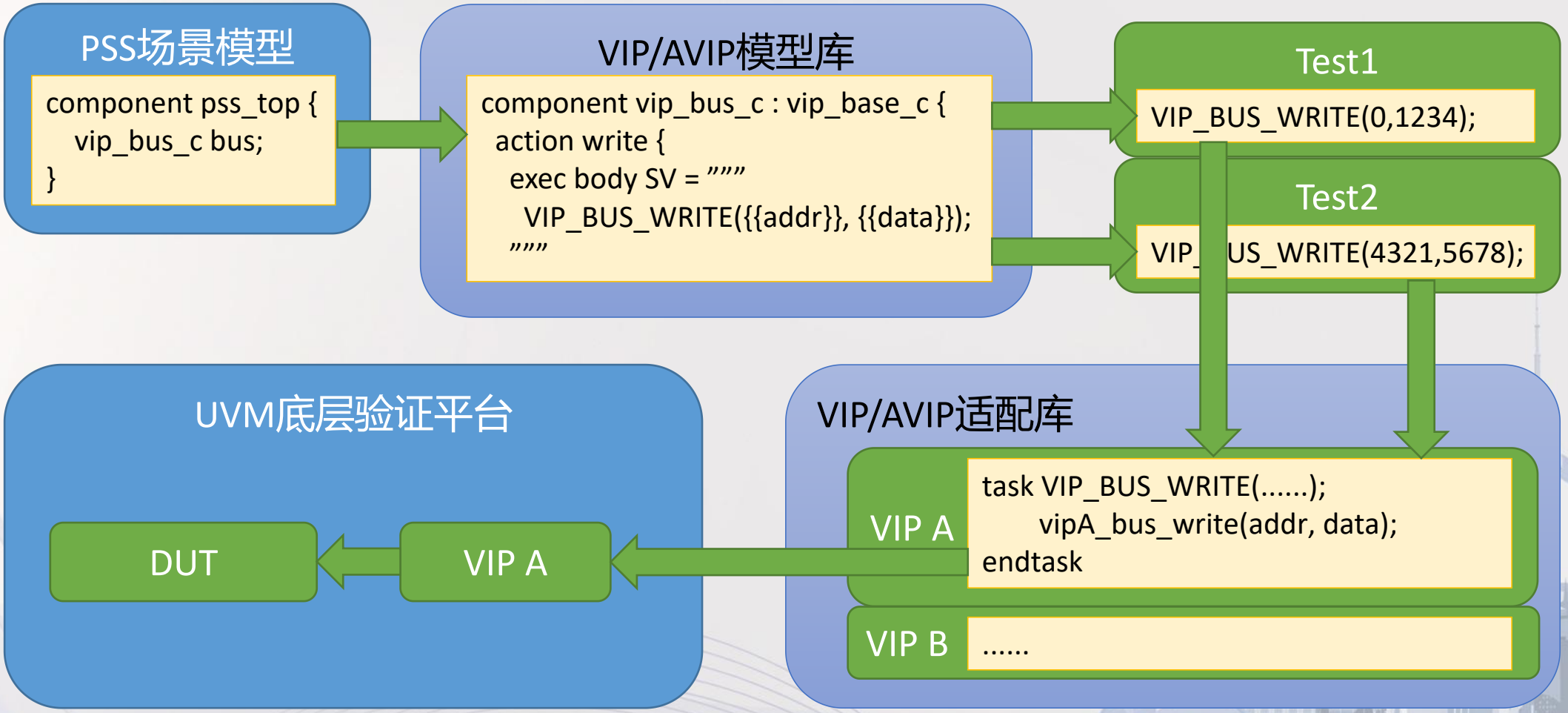
VIP接口调用在PSS模型中实现



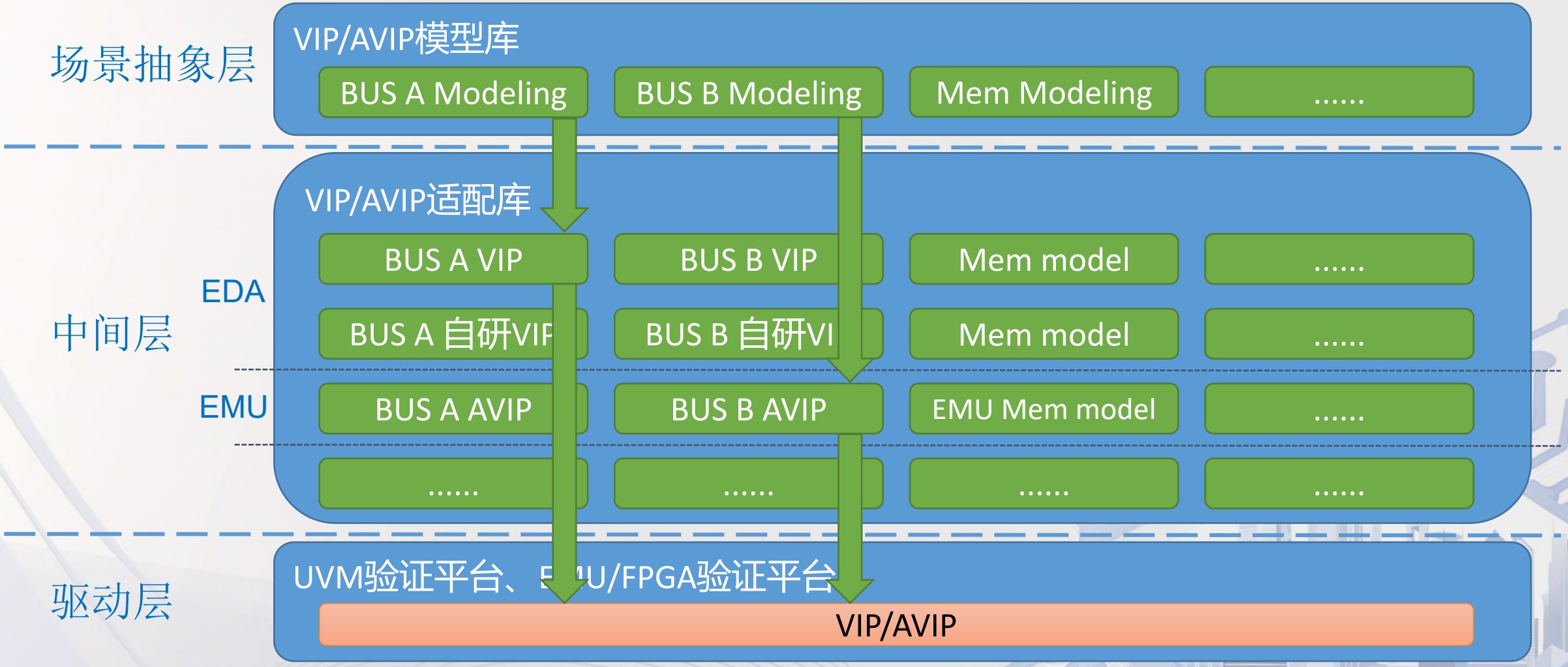
VIP接口调用在底层验证平台中实现



基于PSS中间层的验证架构
- VIP/AVIP适配库



基于PSS中间层的验证架构
- VIP/AVIP适配库



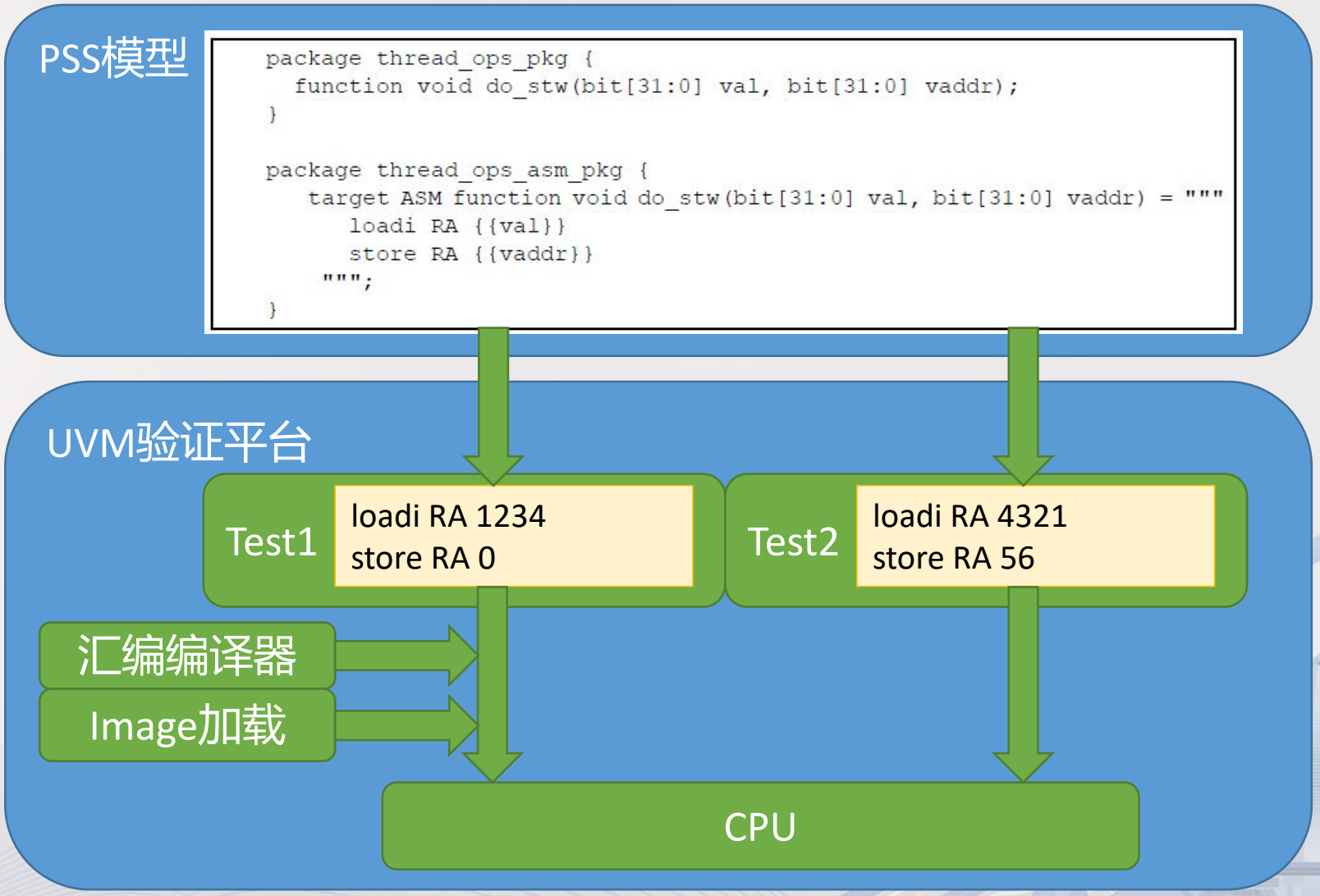
基于PSS中间层的验证架构
- CPU汇编指令转换工具

PSS 2.0标准保留了”C”、”CPP”和”SV”这三个目标编程语言标志符，用于指定目标代码块的预期编程语言为C、C++和SystemVerilog，并提供了数据类型绑定。

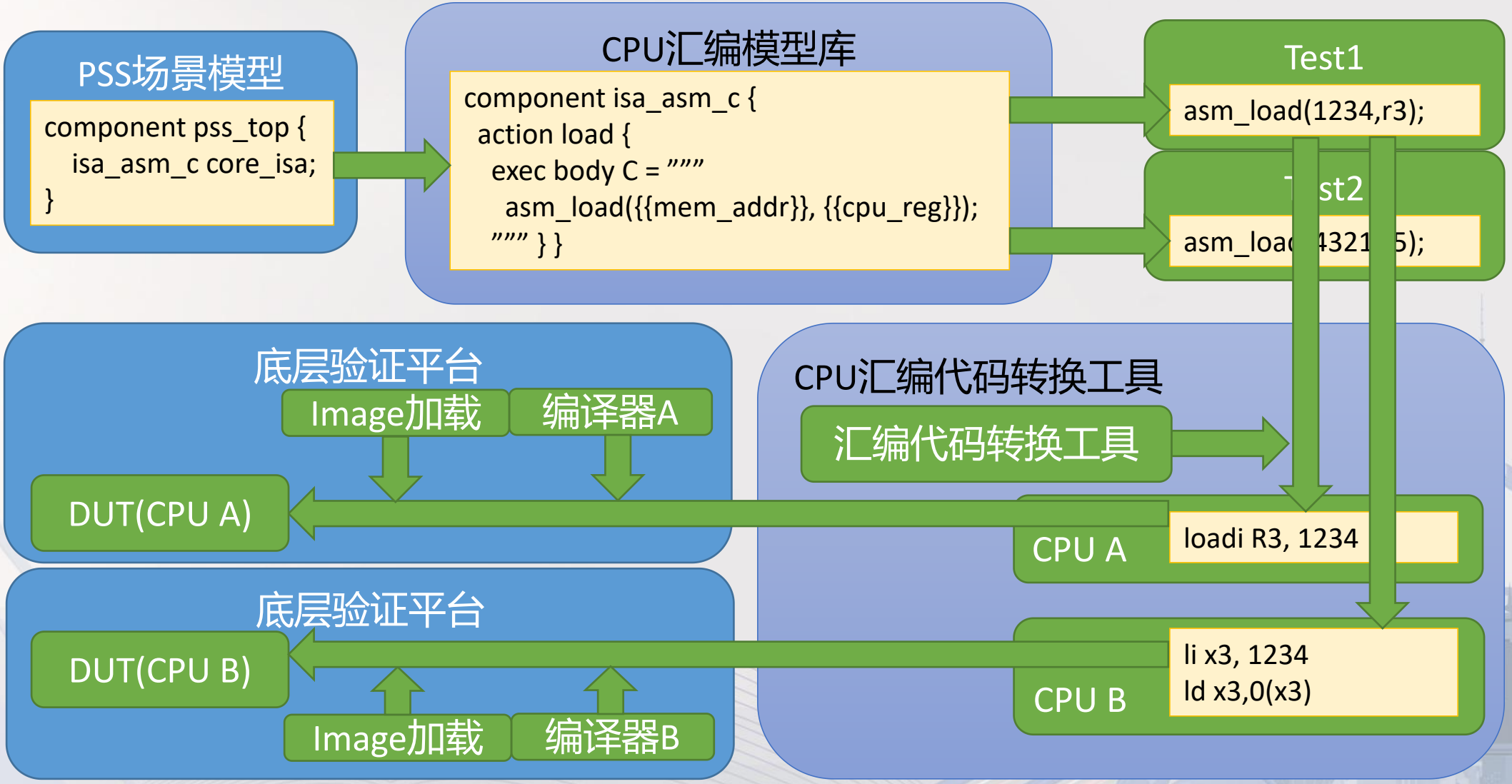
- CPU指令验证使用汇编语言更为常见；
  - 能够正确反映指令测试意图，高级语言编译优化过程中可能会导致指令流程的变化
  - 不依赖于C语言等高级语言编译器的开发
- 不同CPU类型有不同的汇编语法；
- 不同CPU微架构可能会对汇编语言有特殊约束；

基于PSS中间层的验证架构
- CPU汇编代码转换工具

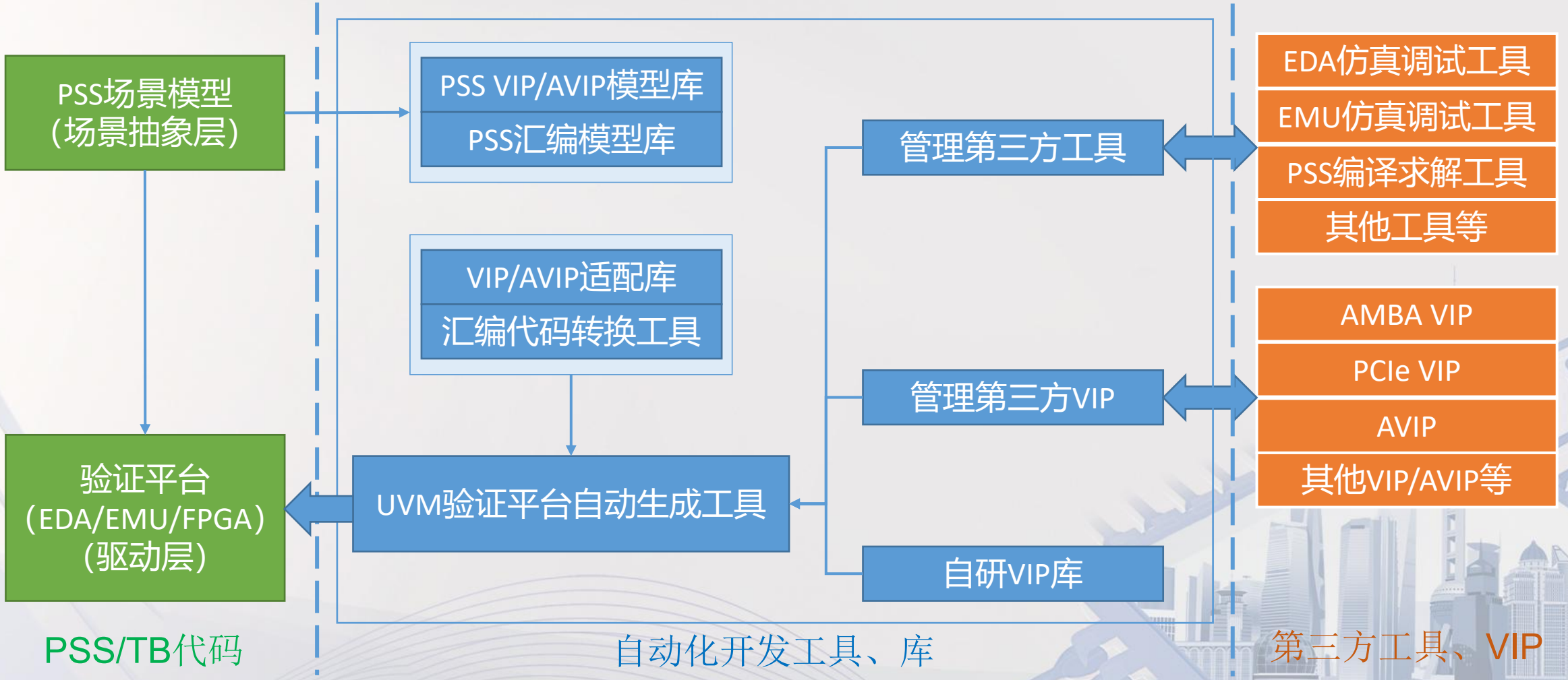
PSS模型中实现汇编语法



基于PSS中间层的验证架构
- CPU汇编代码转换工具



基于PSS中间层的验证架构
- 验证平台开发自动化



基于PSS中间层的验证架构
- 验证平台开发自动化

- 验证场景PSS模型开发
  - 使用PSS模型库，开发各种验证场景PSS模型
- 验证平台开发
  - 使用验证平台自动化生成工具，选择正确的验证平台类型，选择VIP，选择CPU类型，自动生成验证平台架构。
  - 生成的验证平台自动集成正确的VIP/AVIP、适配库和汇编代码转换工具，编写补充其他组件；
- 启动仿真或者回归测试
  - 调用PSS编译求解工具对验证场景PSS模型进行求解，生成随机化后的测试用例代码；
  - 对验证平台进行编译，将同时编译VIP及对应的适配库；
  - 调用汇编代码转换工具生成汇编代码并编译；
  - 启动仿真；

总结与展望

➢ 总结

- 更清晰有效的分层验证架构，实现了验证场景PSS模型开发和验证平台开发的完全解耦，通过中间层将底层实现与场景定义完全隔离，充分发挥PSS高层建模特性；
- 实现了验证场景和激励的跨平台一致性，方便进行各种类型的复用；
- 大大降低了PSS方法在芯片验证项目上的应用门槛；
- 更方便地实现PSS相关验证平台的自动化；

➢ 展望

- PSS模型通过中间层实现与底层验证平台之间的动态交互；
- 如何应用在post-silicon板级测试；
- 代码转换工具的扩展，考虑如何使用编译器思想进行优化和完善；

谢谢！