自适应跨平台PSS中间件架构及开发

王　锋，王　磊，张栗榕
（新华三半导体技术有限公司 西安研究所，陕西 西安 710075）

摘　要： 芯片工艺、规模不断在提升，所包含的功能越来越复杂。多核、多线程中央处理器（Central Processing Unit，CPU），多维度片上网络（Network on Chip，NoC），高速、高密度接口，各类外设等IP（Intellectual Property）集成在芯片上系统（System on Chip，SoC），使芯片开发阶段的仿真验证场景极其复杂，对芯片特别是SoC开发和验证完备性带来巨大挑战。当前在芯片开发领域，便携式测试和激励标准（Portable Test and Stimulus，PSS）是在UVM（Universal Verification Methodology）验证方法学基础上进一步解决随机化和跨平台的复杂组合场景定义和代码生成难题。但目前的PSS标准有一定局限，例如还不支持汇编语言，也无法自适应地调用不同型号、不同平台的验证IP（Verification IP，VIP）等，影响在芯片验证中全面部署PSS。提出一种新的验证平台（Verification Platform）架构，即在PSS场景模型和测试台（Testbench，TB）层之间实现一层中间件（Midware），支持自动生成汇编语言测试代码以及自适应地调用VIP和AVIP（Accelerated VIP）等，以充分发挥PSS高层场景建模的优势，实现芯片验证灵活、高效和完备性的统一。

关键词： 芯片；PSS；中间件；验证；VIP
中图分类号： TN402　　**文献标志码：** A　　**DOI：** 10.16157/j.issn.0258-7998.222962

中文引用格式： 王锋，王磊，张栗榕. 自适应跨平台PSS中间件架构及开发[J]. 电子技术应用，2023，49(1)：20-25.
英文引用格式： Wang Feng，Wang Lei，Zhang Lirong. Self-adapting midware architecture & development for cross-platform PSS [J]. Application of Electronic Technique，2023，49(1)：20-25.

Self-adapting midware architecture & development for cross-platform PSS

Wang Feng，Wang Lei，Zhang Lirong
(Xi′an R&D Institute，New H3C Semiconductor，Xi′an 710075，China)

Abstract： With continuous evolution of semiconductor process technologies and IC (Integrated Chip) scales, more and more complex functions are integrated. Multi-core multi-thread CPU (Central Processing Unit), multi-dimension NoC (Network on Chip), high speed interfaces, kinds of peripherals and so on IP (Intellectual Property) are integrated into SoC (System on Chip). As a result, verification scenarios during IC development become extremely complicated, which leads to great challenges to the SoC development and corresponding verification completeness. Currently PSS (Portable Test Stimulus Standard) has been introduced along with the UVM (Universal Verification Methodology) for generating extensive randomized stimulus with more complicated scenarios. However, the PSS standard, as of today, doesn′t support generating assembly test code and invoking different VIP (Verification IP) flexibly. In order to solve these problems mentioned above, we introduce a verification platform architecture by implementing a new layer as midware between the PSS model and the testbench. The Midware layer can give full play to the PSS advantages of high-level scenes modeling and achieve the flexibility, efficiency and completeness of chip verification.

Key words： IC；PSS；midware；verification；VIP

0 引言

随着半导体行业的高速发展，集成电路的规模和设计的复杂性在不断地增大，使得芯片设计的正确性很难保证，与此同时，芯片验证也越来越困难，成为了现代芯片开发周期的瓶颈[1]。随着芯片验证方法学的发展，传统的电子设计自动化（Electronic Design Automation，EDA）验证发展到与硬件加速（Emulator，EMU）平台和FPGA（Field Programmable Gate Array）原型验证平台混合的验证手段。而如何在模块级、子系统级、系统级等不同层级和EDA、EMU、FPGA不同类型测试台（Test-

bench，TB）上进行测试激励的复用，确保不同平台验证的一致性，成为了新的挑战[2]。

为了实现测试激励的有效复用，继UVM（Universal Verification Methodology）之后，Accellera标准组织推出了便携式测试和激励标准（Portable Stimulus Standard，PSS），其目标是提供一个独立的测试激励来源，并在更高的抽象级别上定义激励和场景，从而实现跨层级和平台的场景描述和测试激励复用。其主要的特点如下：

(1) 通过PSS建模在更高抽象级别上指定激励和测试，可定义面向CPU（Central Processing Unit）和各类接口协议的复杂组合场景。

(2) 可以方便地生成随机组合场景的C/C++或者SV（SystemVerilog）代码，通过编译并加载C/C++如案卷程序实现CPU的验证场景，通过调用验证IP（Verification Intellectual Property, VIP）或硬件加速VIP（Accelerated VIP，AVIP）实现对特定协议接口的激励。

(3) PSS场景模型不仅可应用于模块级、子系统级和系统级EDA测试台，还可以用在EMU、FPGA平台，为不同级别的平台产生相同的激励，实现了测试激励复用，确保了验证的一致性。

目前最新的PSS 2.0标准协议保留了“C”“CPP”和“SV”这三个目标编程语言标志符，用于指定代码块的预期编程语言为C、C++和SystemVerilog，同时提供这三种语言的数据类型绑定（binding）方法，这样可以方便地生成C、C++和SystemVerilog随机化代码[3]。

对于相同的总线或协议接口，通过开发PSS模型并在指定的目标SV（SystemVerilog）代码中调用特定的VIP接口任务，可实现通过特定VIP接口随机化的激励。但是，不同的测试台可能会集成不同的VIP/AVIP，它们的调用流程也有区别，如果切换VIP/AVIP，则需要在PSS模型中定义不同的具体调用VIP/AVIP的细节，这大大增加了PSS建模人员的工作量，而且导致加大了PSS模型与测试台具体实现的耦合度，无法完全实现PSS模型只关注高层抽象场景和易于移植的需求。

此外，汇编语言是CPU指令机器码的别名，指令机器码和汇编指令通常都会同时发布在CPU指令手册中，对于CPU指令集的验证，与C/C++语言相比，汇编语言具有更简洁、更高效、更全面的优势。但是，当前的PSS标准并没有为汇编语言预留相应的标志符，特别是没有提供标准的数据类型绑定方法，这不便于生成CPU指令验证所需的各类随机汇编代码。

为了解决以上两类问题，本文提出了添加中间件（Midware）层的验证平台架构，在PSS高层级抽象建模的基础上，可实现VIP/AVIP的灵活选择和自适应切换，并支持不同种类的CPU汇编指令随机验证，实现了PSS建模与具体测试台实现方法的解耦，能够更充分发挥PSS标准的优势。

1 基于中间件层的PSS验证平台架构

1.1 验证平台架构介绍

如图1所示，本文提出由场景描述层、中间件层和测试台（Testbench，TB）层三个层次组成的基于PSS的验证平台架构，充分发挥PSS场景抽象定义和测试具体实现。

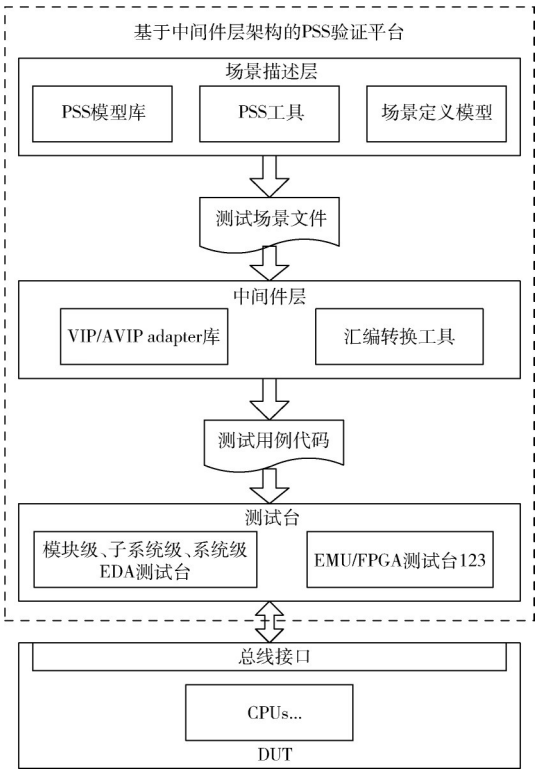

图1 基于PSS中间件层的验证平台架构示意图

1.2 场景描述层

场景描述层由PSS相关组件和工具构成，充分发挥PSS对验证场景抽象化的能力，对验证目标场景进行较高层次的定义，使得场景描述层的组件可复用，与模块级、子系统级、系统级EDA环境，以及EMU硬件加速和FPGA原型验证等多个平台实现一致的验证场景。场景描述层定义的场景与接口协议的具体硬件实现无关。

场景描述层的构成如图2所示，主要包括：

(1) PSS工具

PSS工具可使用基于PSS 2.0标准版本的商用工具，包括PSS语法编译器compiler和解算器solver，负责编译、随机化并生成基于中间件层的SystemVerilog和C代码。

(2) PSS核心库

PSS标准定义了符合PSS 2.0标准的PSS核心库（PSS core library），包括组件类型、数据类型、函数和属性，为常见的PSS应用程序（如内存和寄存器读写访问

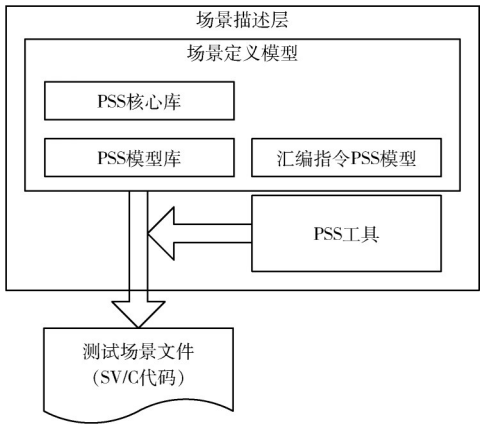

图2　场景描述层示意图

等)提供标准的可移植功能和使用程序。

(3) PSS模型库

针对常用标准协议,例如PCIe、以太网接口、AMBA总线等,本文提出新的架构,开发PSS模型组件构成PSS模型库,实现了对标准协议基本操作、资源管理、目标语言生成等功能。这里生成的PSS模型库目标语言也高度抽象化,直接与中间件层的接口对接,隐藏了测试台的具体实现方式。

(4) 汇编指令PSS模型

利用PSS 2.0标准保留的C目标编程语言标志符,本系统也实现了平台无关的汇编指令PSS模型,该指令模型对汇编指令抽象化,可生成C目标语言格式的特定指令代码,该代码可通过中间件转化为面向特定CPU的可编译可执行汇编指令,在一定程度上抽象化CPU指令,隐藏了具体CPU的汇编命令格式。

(5) 场景定义PSS模型

基于PSS核心库、PSS模型库和PSS汇编指令模型,验证场景PSS模型的开发人员可根据验证目标芯片的实际需求,通过使用PSS模型库中预定义的协议模型组件,定义各种复杂交叉场景,包括并不限于接口协议的组合场景、汇编指令的组合场景等。PSS场景模型开发人员不需要了解测试台的具体实现方式和具体CPU类型,可将工作内容聚焦于场景定义。

1.3 中间件层

不同类型或不同层级的测试台可能使用不同的VIP,因此场景描述层定义的验证场景需要映射到不同的测试台层以实现其具体功能。本文提出的中间件层实现PSS抽象场景定义到不同测试台的映射。

中间件层的基本架构如图3所示,主要包含两大类中间件:一类是进行VIP/AVIP适配的adapter工具,支持多种接口和总线协议;另一类是代码转换工具,实现汇编代码生成。

(1) VIP/AVIP adapter

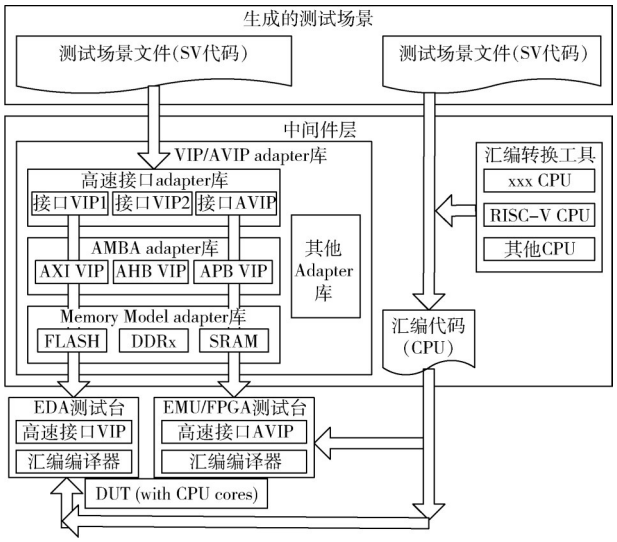

图3　中间件层示意图

VIP/AVIP adapter是一组SV和C/C++的API接口库。高层抽象的PSS模型所生成的SV/C++代码被集成到测试台前,通过该adapter调用具体的底层VIP测试组件,实现验证的一致性。

(2) 汇编转换工具

PSS 2.0标准协议保留了“C”“CPP”和“SV”这三个目标编程语言标志符,用于指定代码块的预期编程语言为C、C++和SystemVerilog,同时提供这三种语言的数据类型绑定,这样可方便地生成C/C++和SystemVerilog随机化代码,但PSS标准并没有为汇编语言预留相应的标志符,特别是没有提供标准的数据类型绑定,这不便于生成CPU指令验证所需的随机汇编代码。为了更好地生成随机汇编代码,本系统利用C目标编程语言标志符,生成的C语言函数对应不同的CPU汇编指令,通过函数参数扩展实现汇编指令参数随机化。PSS compiler/solver工具将PSS模型转换为随机参数的C代码之后,代码转换工具对其参数进行识别,转换为随机汇编指令。这样,当使用不同的CPU选项时,得到的汇编代码会有所不同,对CPU指令组合场景进行建模时,可直接使用PSS模型库中的汇编指令模型,在一定程度上隐藏了不同CPU类型之间的区别。

1.4 测试台层

测试台包含除test外的验证框架和组件实现,包括接口、驱动、监测、参考模型、结果判定等,可以是模块级、子系统级或系统级验证的UVM平台,也可以是EMU硬件加速、FPGA原型验证平台。不同的测试台可能挂接不同的VIP/AVIP实现对协议接口的驱动。对由中间件层生成的具体的汇编代码进行编译、加载、运行和检查功能。

测试台层编译或运行时需要加载中间件生成的SV、

C和汇编代码，架构如图4所示。

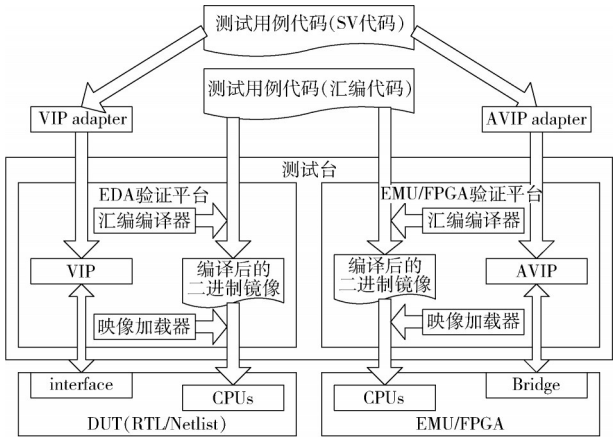

图4　测试台示意图

2 中间件层的实现

2.1 VIP/AVIP adapter库

VIP/AVIP adapter库作为场景模型与底层测试台不同种类VIP/AVIP之间的桥梁，实现了各类标准协议的封装。

以PCIe高速接口数据传输为例，VIP/AVIP adapter库为PSS场景描述层提供类似于以下通用SystemVerilog调用接口：

```
task vip_pcie_send_data(input pcie_send_cfg cfg);
```

PSS场景描述层中的PSS模型库中针对PCIe高速接口协议，定义一组抽象模型用以描述PCIe功能场景。如PCIe数据传输模型有如下PSS模型库实现的参考伪码：

```
component pcie_modeling {
  ......
  action send_data {
    exec body SV """
      vip_pcie_send_data({{pcie_send_cfg}});
    """
  };
};
```

当抽象场景中需要进行PCIe数据传输时，只需要调用对应的action即可实现。如PCIe重复数据发送的PSS场景模型参考伪码如下：

```
component pss_top {
  ......
  action master {
    activity {
      pcie.link;
      repeat (50) {
        [100]: do pcie. send_data with (pcie. mode == pcie_mode_inbound);
      }
    }
  };
};
```

由此可见，编写PSS场景模型时，只需调用PCIe模型库的send_data action，不需要关心验证平台类型、具体底层实现以及集成了哪个VIP/AVIP，有效实现了高层抽象特性。

另一方面，adapter库针对不同的PCIe VIP和AVIP预置了对应的task或function实现，如以下针对某特定商用PCIe VIP，有如下adapter参考伪码：

```
task vip_pcie_send_data(input vip_pcie_send_cfg cfg);
  UvmUserTransaction pkt;
  pkt = new();
   pkt. mode = (cfg. mode==vip_pcie_mode_inbound)? UvmUserTransaction:: INBOUND: UvmUserTransaction:: OUTBOUND;
  assert(pkt.randomize() with pkt.size = cfg.length; ...);
  ......
  `uvm_send(pkt)
endtask
```

当验证平台自动化工具生成测试台代码时，通过选择VIP选项指定特定VIP/AVIP，其对应的adapter package代码会被工具自动附加到测试台文件列表中，对测试台进行编译和运行仿真时会自动加载并使用该VIP adapter代码。

通过以上方法，VIP/AVIP adapter库有效地将场景抽象层与测试台VIP/AVIP连接到一起。

2.2 汇编代码转换工具

汇编代码转换工具实现了指令模型到特定CPU类型汇编代码的实现。

以load指令为例，场景描述层中的PSS模型库中定义了以下抽象模型，用以定义load汇编指令：

```
component cpu_asm_moding {
  ......
  action asm_load {
    exec body C """
      asm_load({{mem_addr}}, {{cpu_r_reg}});
    """
  };
  };
```

当抽象场景中需要调用汇编load指令时，只需要调用对应的action即可实现。如DMA传输场景模型可按如下方式编写：

```
component pss_top {
  ......
  action dma_single {
    activity {
```

```
        do cpu0.asm_load with (mem_addr == src.addr);
        do cpu0.asm_load with (mem_addr == dst.addr);
        do cpu0.asm_dma;
        ......
    }
  };
};
```

以上PSS场景模型代码经PSS工具进行compile/solve之后，生成以下测试场景C伪代码：

```
asm_load(0xf8000000, r3);
asm_load(0xf8300000, r5);
```

这里0xf8xxxxxx是PSS工具随机化之后得到的地址，r3/r5是随机化得到的CPU通用寄存器名。上述C伪代码无法使用C语言编译器直接进行编译，需要通过汇编代码转换工具进行转换。

汇编代码转换工具可对应多种不同的CPU类型，当验证平台自动化工具生成测试台代码时，通过选择CPU选项指定特定CPU类型，其对应的转换工具会自动添加到测试台工具链中，当对测试台的CPU软件进行编译时会自动转换为对应的汇编代码。转换结果示例如下：

如果选择某RISC-V CPU类型时，生成的汇编代码如下：

```
li x3, 0xf8000000
ld x3,0 (x3)
li x5, 0xf8300000
ld x5,0 (x5)
```

如果选择另一种CPU类型时，生成的汇编代码如下：

```
mov_s r3, [0xf8000000]
ld.di r3,[r3]
mov_s r5,[0xf8300000]
ld.di r5,[r5]
```

可以看到，当使用不同的CPU选项时，得到完全不同的汇编代码。因此，通过使用PSS模型库中的指令模型，在一定程度上隐藏了不同CPU类型之间的区别，实现了CPU指令的高层抽象特性。

3 基于中间件层的PSS验证平台开发

开发的中间件及整个PSS验证平台集成在验证平台管理系统中，相关的系统架构、工具链及流程如图5所示。

开发和运行基于中间件的PSS验证平台流程：

(1) 中间件开发

生成adapter库和汇编代码转换工具。

(2) PSS验证场景模型开发

使用PSS模型库，开发各种验证场景PSS模型。

(3) 测试台开发

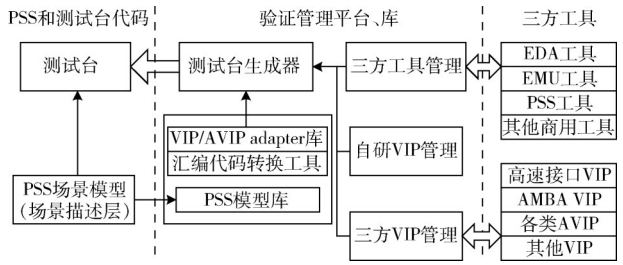

图5 基于中间件的PSS验证平台管理系统

使用验证平台自动化工具，选择正确的测试台类型、VIP选项和CPU选项，自动生成相应的验证平台框架，支持自动集成适配的VIP/AVIP、VIP/AVIP adapter和汇编代码转换工具。

(4) 启动仿真或回归测试

主要流程如下：

① 调用PSS compiler/solver对验证场景PSS模型进行处理，生成随机化之后的测试场景代码（SV和C代码）；

② 调用汇编代码转换工具对特定CPU上运行的C代码进行转换，生成对应的汇编代码；

③ 对验证平台进行编译；

④ 启动仿真或者回归测试。

步骤(3)通过选择不同选项，可以生成集成了不同VIP、面向不同CPU类型的EDA或EMU验证平台架构，这些不同的验证平台都可以使用相同的PSS场景模型（步骤(2)开发得到的），从而实现验证场景和激励的跨平台一致性。

4 结论

综上所述，通过开发基于中间件架构的PSS验证平台，本文实现了以下目标：

(1) 验证场景和激励在跨平台时保持一致性，适用于CPU指令验证和使用通用VIP/AVIP的验证场景；

(2) 验证场景模型开发和测试台开发的完全解耦，通过中间件层将底层实现与场景定义彻底隔离开，充分发挥了PSS高层建模特性，便于跨项目、跨平台扩展移植。

(3) PSS定义汇编指令场景，与C语言相比，提升了对CPU指令验证的效率和完备性。

(4) 完善PSS模型库、VIP/AVIP adapter库和汇编语言转换工具，可对应各类VIP和CPU类型。当芯片升级换代时，可灵活替换自适应验证环境中CPU和VIP类型，大大提高了验证环境的继承性。

(5) 基于PSS验证中间件的验证平台开发和使用，由验证管理平台进行统一管理，实现了对第三方PSS工具的灵活调用以及对标准的验证平台自动化构建流程，大大提高了基于PSS验证方法学的模型和验证平台搭建

和扩展效率。

(6) 测试台只需集成VIP/AVIP对应的adapter，其代码量增长有限，对测试台编译和仿真效率影响可以忽略。

(7) 目前本系统主要应用于芯片硅前（pre-Silicon）验证（包含EDA verification、EMU和FPGA validation），将来可扩展、统一硅后（post-Silicon）validation及其他验证、测试平台。

参考文献

[1] 叶甜春.“十三五”期间中国集成电路制造产业链发展的思考[J].集成电路应用,2016(1):6-7.

[2] 吕毓达, 谢雪松, 张小玲. 基于UVM的可重用SoC功能验证环境[J]. 半导体技术, 2015(3):234-238.

[3] Accellera Systems Initiative. Portable Test and Stimulus 2.0[DB/OL]. (2021-4-14) [2022-05-11]. https://www.accellera.org/downloads/standards/portable-stimulus2021.

[4] BHATNAGAR G, BROWNELL D. Portable stimulus vs. formal vs. UVM: a comparative analysis of verification methodologies throughout the life of an IP block[C]// DVCON, 2018.

（收稿日期：2022-05-11）

作者简介：

王锋（1977-），通信作者，男，硕士，工程师，主要研究方向：芯片设计验证，E-mail：wangfengdec@163.com。

王磊（1976-），男，本科，工程师，主要研究方向：CPU和片上总线NoC。

张栗榕（1983-），男，硕士，工程师，主要研究方向：高速接口和数字射频。

扫码下载电子文档

版权声明