高性能处理器层次化可重用模拟验证环境

胡向东*, 巨鹏锦, 朱英, 孙路

国家高性能集成电路设计中心, 上海 201204

* 通信作者. E-mail: huxdisme@vip.sina.com

收稿日期: 2014-10-09; 接受日期: 2014-12-06; 网络出版日期: 2015-03-06

国家科技重大专项“核高基”(批准号: 2009ZX01028-002-001) 资助

摘要 高性能处理器的多核化和 SoC(system on chip) 集成化发展趋势, 使得处理器芯片设计复杂度和规模成倍增长, 平面式的模拟验证环境在消耗大量机器资源的同时模拟速度急剧下降, 严重影响模拟验证效率, 而当前模拟验证仍是处理器验证中最广泛的验证手段. 为提高模拟验证效率, 必须建立高效的处理器模拟环境. 根据高性能处理器架构特点, 基于可重用验证组件开发思想, 建立层次化可重用的处理器模拟验证环境, 可有效解决高性能处理器模拟验证中内存消耗大、模拟速度慢、验证效率低等问题. 这种环境创建方法可以缩短环境创建和调试周期, 减少环境自身错误, 有利于不同处理器验证环境的可重用.

关键词 处理器 多核 模拟验证 层次化 可重用 验证组件

1 引言

随着半导体工艺和处理器架构的发展, 多核 (multi-core processor) 成为当今处理器主流架构^[1]. 高性能多核处理器为了追求更高的性能和能效比, 一是发展处理器核心架构, 二是发展处理器芯片架构, 使得高性能处理器结构越来越复杂, 在单芯片中已集成 16 个甚至更多的通用高性能核心, 并集成各种高速 I/O 接口, 如 DDR3, PCI-E2.0/3.0 接口等, 导致设计规模越来越大, 给处理器功能验证提出了新的挑战. 有研究表明, 功能验证已成为处理器研发的主要瓶颈, 在整个开发过程中需要占用 70% 以上的时间和资源^[2].

面对高性能处理器发展的挑战, 处理器功能验证也在不断改进和创新方法和技术, 不断提升验证效率. 当今高性能处理器的验证通常综合采用模拟验证、硬件仿真加速验证、FPGA 验证、形式验证等方法, 这些验证方法各自的特点如下:

(1) 模拟验证仍然是应用最广泛的验证方法^[3,4], 通过创建模拟验证环境, 施加焦点测试激励 (directed tests) 或者伪随机测试激励 (pseudo-random tests), 通过参考模型比较等方法验证设计的正确性. 模拟验证的效果可以通过各种覆盖率进行衡量, 常用的覆盖率分析方法包括两大类: 代码覆盖率和功能覆盖率. 代码覆盖率是一种客观的分析方法, 包含多种类型: 如状态机覆盖率、条件表达覆盖率、分支覆盖率等等; 功能覆盖率则是由验证或设计人员使用 PSL(property specification language)¹⁾

1) IEEE. IEEE Standard for Property Specification Language Manual. <http://www.standards.ieee.org>, 2010.

引用格式: 胡向东, 巨鹏锦, 朱英, 等. 高性能处理器层次化可重用模拟验证环境. 中国科学: 信息科学, 2015, 45: 535–547, doi: 10.1360/N112014-00296

或 SVA(systemverilog assertions)²⁾等语言描述的功能点, 具有一定的主观性, 描述的功能点是否足够依赖于功能点描述者. 采用双覆盖率驱动的模拟验证方法通常会贯穿整个处理器研发的全过程, 但随着设计规模的增大, 验证速度也急剧下降.

(2) 硬件仿真加速验证, 验证速度可以达到兆级, 速度介于模拟验证和 FPGA 之间, 具有支持设计规模大、扩展性好、状态可观性强, 以及外部设备接口丰富等特点^[5], 主要用于验证中期, 处理器设计代码的正确性有了一定的基础以后, 可以基于硬件仿真加速验证平台, 快速建立软硬件协同验证环境, 以尽早发现软硬件接口上的问题, 同时在该验证环境上可以支持软件的早期开发调试, 但硬件加速器价格昂贵, 而且运行速度仍然不够快, 仍不能满足大规模系统及应用程序验证需要.

(3) FPGA 原型验证速度比模拟验证和硬件仿真加速验证快很多, 用于处理器开发的中后期, 可进行大量的软硬件协同验证, 也可采用标准测试集等程序来进行快速软件验证. 但随着芯片规模的不断增长和 SoC 化趋势, FPGA 平台实现的复杂度和难度也日益加大, 高性能处理器的 FPGA 平台通常要采用多个 FPGA 芯片才能实现, 片间传输的速度和可靠性成为最大的挑战^[6].

(4) 功能形式验证 FFV(functional formal verification)^[7] 是基于数学证明的验证方法, 通过遍历所有可变因素的所有状态空间, 来找出是否存在违背属性描述的操作序列, 从而证明设计模型的正确性, 因受限设计类型和规模, 一般只能用于局部逻辑的验证. 经过近三十年的发展, 形式化验证方法和 EDA 工具已经成熟, 并在越来越多的处理器验证中得到应用, 相比传统的验证方法, 形式验证往往能够发现一些传统验证手段非常难以发现的设计错误, 在流片前验证中至关重要^[8], 其在处理器验证中所占比重也呈逐年增加趋势.

上述几种验证方法各有所长, 通常采用多种方法进行综合验证, 但模拟验证作为一种传统的验证方法, 在高性能处理器研发过程中发现的设计错误往往达 80%~90% 以上, 因此如何提高模拟验证效率在高性能处理器的功能验证中至关重要.

2 高性能处理器模拟验证面临的挑战和策略

2.1 面临的挑战

当高性能处理器采用多核和 SoC 集成技术, 使得规模和复杂性快速增长, 而可重用设计方法学^[9]的发展加快了处理器的研发速度, 使用基于已验证的 IP 快速构建大规模复杂设计成为当今设计的发展趋势^[10], 但给处理器模拟验证带来新的挑战:

(1) 多核处理器由于集成越来越多的核心, 模拟验证平台的计算能力和内存容量跟不上验证规模的增长, 模拟速度随着处理器规模增大而急剧下降, 如何提高多核处理器模拟效率是验证面临的一大挑战.

(2) 高性能处理器的 SoC 集成化趋势使得片上集成多种标准 I/O 接口, 这些接口通常基于 IP 开发, 必须更好地利用 IP 的可重用性来提升验证环境的可重用性^[11].

(3) 多核处理器验证要研究如何继承单个核心的验证成果, 以加快验证进度.

2.2 模拟验证策略

模拟验证首先必须解决高性能处理器多核化导致规模快速增加而带来的模拟效率问题. 有两种策略, 一种是开发并行模拟验证环境, 充分发挥并行计算资源优势; 另一种是采用层次化的验证策略

2) Cadence. System Verilog Reference Manual. <http://www.cadence.com>, 2014.

(hierarchical verification approach)^[7], 由底向上、由点到面进行验证。

并行模拟验证环境是利用处理器对称多核的结构特征, 利用集群服务器的并行计算和大内存资源, 基于 HDL 模拟器的 VPI 接口 (verilog procedural interface) 和进程间通信 IPC (inter-process communication) 机制, 来开发并行 HDL 模拟环境。优点是对模拟平台的硬件要求不高, 能够支持规模很大的多核处理器模拟验证; 缺点是进程间的通信会影响模拟速度, 通信开销与处理器结构和测试激励类型都有密切关系。实际应用中, 通过并行模拟获得的验证加速效果并不理想。

层次化的验证策略是通过合理划分处理器的功能区域, 适当控制功能区域的规模, 对验证重点、难点的功能区域建立独立的模拟环境进行验证, 并在更上一层针对接口协议和部分功能进行补充验证。优点是不同层次的验证环境能实现对处理器的交叉互补验证, 在覆盖率和速度之间取得良好平衡; 缺点是只能针对部分设计建立部件级环境, 而处理器功能复杂, 如多 Cache 一致性、多核间通信机制, 只有在处理器全芯片规模下才能验证充分。复杂处理器验证即使采用层次化策略, 也无法规避全芯片级模拟速度慢、验证强度低的问题。

本文研究并提出的层次化可重用模拟验证环境, 可有效解决处理器芯片级模拟速度慢的问题, 也能有效继承原有的处理器验证成果, 加快新研发的处理器验证进度。该环境创建方法在一款国产多核处理器芯片验证中取得了非常好的应用效果, 也为后续的多款处理器芯片模拟验证发挥了良好的作用。

3 层次化可重用模拟验证环境

3.1 可重用方法学

可重用验证方法学 (reuse verification methodology) 是开发可重用验证组件和验证环境的理论指导, 近些年 EDA 厂商推出了多种可重用验证方法学, 如 ERM (e reuse methodology), OVM (open verification methodology), RVM (reference verification methodology) 等等。由于这些方法学往往仅支持各自厂家自己的工具, 当验证环境中同时集成多家采用不同方法学开发的 VIP (verification IP) 时存在不兼容问题。

2009 年 12 月, Accellera 组织推出的 UVM 标准 (universal verification methodology)³⁾ 得到了业界共识, 目前已有多家 EDA 厂商支持该标准。UVM 方法学同时支持面向对象的程序设计 (OOP) 和面向结构的程序设计 (AOP) 方法, 支持 3 种当前流行的验证语言 System Verilog, Specman E 和 SystemC。UVM 为解决验证环境的可重用性、可扩展性和兼容性问题, 提供了跨语言的标准接口和可重用开发规范, 还提供了一些通用标准验证组件 UVC (universal verification component), 如 agent, driver, sequence, score-board 等等, 另外各 EDA 厂商也提供基于 UVM 的 VIP, 如 USB, AMBA, PCI-E 等。这些 UVC 是建立处理器验证组件库的基础, 也为开发面向处理器结构的验证组件提供了非常好的范本。

3.2 环境创建方法

处理器层次化可重用模拟环境创建方法就是采用层次化的验证策略, 依据处理器结构特点, 采用统一的可重用开发规范, 建立可重用的验证组件, 并基于这些验证组件构建层次化验证环境。

多核处理器验证环境一般可分为 3 个层次, 即部件级、单核级和芯片级。部件级环境具有模拟速度快、伪随机强度大、错误发现率高的优点, 但是环境的创建和维护需要投入的人力和时间较多, 跨芯片重用比较困难; 单核级环境是针对处理器核心而专门创建的环境, 模拟速度较快, 核心对外接

3) Accellera. Accellera Standard UVM1.2. <http://www.accellera.org>, 2014.

图 1 层次化模拟环境特点

Figure 1 The characteristics of hierarchical simulation environment

口一般变动不大, 因此该环境跨芯片继承性较好; 芯片级验证环境创建和维护工作量最小, 跨芯片重用时, 难点在于芯片级测试激励的重用. 3 个层次的模拟环境特点如图 1 所示.

层次化可重用验证环境的创建原则是

- (1) 面向处理器架构规划各层次验证环境, 保证验证全面性和关键部件的验证强度;
- (2) 部件级环境和验证组件开发应遵循统一的可重用开发规范, 保证代码和文档的质量, 保证验证环境和验证组件的可靠性、可维性和可重用性;
- (3) 建立面向处理器结构特点的验证组件库, 便于不同层次、不同项目的资源共享, 减少验证环境的开发工作量, 提高验证环境的正确性收敛速度.

按照上述原则实现的验证组件, 不仅能加快部件级验证环境的创建速度, 而且因组件是面向处理器结构且遵循可重用规范开发的, 可用于“重构”芯片级验证环境, 降低设计规模, 解决模拟效率问题.

可重构处理器芯片级验证环境的特点是:

- (1) 支持对设计结构进行可配置的裁剪;
- (2) 支持虚拟设计模型类的验证组件, 对特定设计模块进行可配置的替换;
- (3) 验证环境结构可配置, 能实现与芯片重构的同步配置, 例如核心 Monitor、多核参考模型、实时结果比较等.

3.3 层次化可重用验证环境

3.3.1 部件级验证环境

部件级环境的一般规模不大, 激励通常采用在线生成方式, 具有模拟速度快、测试强度大、便于测试回归等特点. 在验证进度安排上, 部件级模拟验证环境与设计同步开发, 适合验证人员在开发早期介入, 缩短关键模块的验证时间. 建立部件级环境的目的是加大验证强度, 弥补其他层次验证环境的不足. 部件级验证环境开发的关键是遵循统一的可重用环境开发规范, 开发可重用的验证组件, 逐步构建面向处理器的验证组件库, 最终实现验证环境的快速创建.

处理器部件级验证环境主要用于功能复杂的模块或相对独立的模块进行验证. 如功能复杂的访存部件验证, 由于访存流水线的访问延迟不固定、Cache 一致性设计实现复杂, 而且访存指令从取指到执行的流水线很长, 因此在单核或全片级验证环境下, 访存验证要达到很高的测试强度比较困难. 建立访存部件级验证环境则可弥补单核和芯片级验证环境的不足. 在部件级环境下, 可以抽象指令流水

图 2 单核心验证环境

Figure 2 Core level verification environment

线设计, 访存请求由环境中的激励生成器产生并直接作用于访存部件接口, 从而大大提高访存请求的密集度. 在访存部件的接口上施加测试激励, 也使得验证者更容易对访存流水线的行为进行定向随机控制, 实现对各种访存行为的覆盖.

处理器中功能独立的设计也非常适合采用部件级验证, 如浮点部件、I/O 接口相关设计、片上互连网络等. 这些设计的特点是功能比较独立, 与其他设计模块间的接口协议清晰, 基于标准协议或自定义协议传输, 非常适合面向对象的建模方法, 便于生成可重用性强的验证组件, 一般这些设计通常以部件级验证为主, 其他层次验证环境为辅.

部件级验证环境虽然高效、快速, 但通常存在一些局限性, 因此部件级验证环境一般侧重于设计的主要功能, 而不会对一些次要功能做验证, 比如访存部件级验证环境不会对 Cache 阵列做 BIST 验证. 因此处理器功能验证的完整性需要各层次验证环境协同实现.

3.3.2 单核心验证环境

处理器单核心验证环境也可以认为是部件级验证环境的一种, 不同点在于一般部件级环境激励都是在线生成测试激励, 而单核心环境的测试激励则与全片环境的测试激励相同, 都是离线生成的处理器指令码序列 (如汇编程序).

单核心验证环境主要由处理器核心和一系列验证组件构成, 包括系统接口 UVC 及内存模型、参考模型、核心 Monitor、实时比较环境等. 典型的单核心验证环境如图 2 所示.

系统接口 UVC 是单核心验证环境的关键组成部分, 主要功能是模拟核外产生的各种请求和响应. 另外系统接口还应实现请求和响应的随机类型控制、随机延迟控制、核心接口的协议检查等功能.

单核心验证环境承担了核心大多数功能验证, 如指令集验证、操作系统接口验证, 以及指令流水线、运算部件、访存流水线等核心结构和微结构设计的功能验证. 单核心环境运行速度快、随机性和可控性强, 对核心的验证覆盖率可达到 90% 以上, 是核心验证的主要环境.

3.3.3 芯片级验证环境

处理器芯片级验证环境通常是一个完整的处理器芯片级验证环境, 支持对处理器核心、片上系统

图3 “裁剪”的多核处理器芯片级模拟环境

Figure 3 Simplified chip level environment of multi-core processor

以及芯片接口的各项功能验证。在层次化验证策略下, 部件级和核心级验证环境实现了大多数关键部件的功能验证, 而芯片级环境则承担了处理器大多数的全局功能验证。对多核处理器而言, 具体包括初始化复位、核间通信、多核 Cache 一致性以及芯片的可靠性、可测性和低功耗设计的验证等。

处理器结构从单核发展到多核、从单一 CPU 到 SoC 集成, 大量采用了设计重用、IP 集成的设计方法, 这种可重用的设计思想自然也成为多核处理器芯片级验证环境的构建方法。

多核处理器可重构芯片级环境实现方法, 重用了在部件级和单核级开发的验证组件:

(1) 基于单核参考模型快速实现多核参考模型。本文基于文献 [12] 所述的基于处理器最小并行状态集的多核参考模型创建方法, 在单核参考模型基础上创建了多核参考模型, 采用这种方法开发的指令级精确多核参考模型简单而高效, 16 核规模的参考模型运行速度可达 2M 条指令/秒, 很好地满足了验证需要。

(2) 重用核心 Monitor 和核心实时比较 UVC, 实现了多核处理器验证结果的实时比较。

(3) 重用标准接口的 VIP 实现芯片的外部 I/O 接口验证环境。

(4) 重用访存部件级环境的伪核 UVC, 可部分或全部替换芯片级环境中的核心, 实现真、伪核心混合验证。

为了解决芯片级环境规模大、模拟速度慢、机器资源要求高的难题, 提高 RTL 模拟运行效率, 通过可重构的芯片级环境创建方法, 可根据验证需要, 实现对特定设计模块的裁剪, 如图 3 所示, 其中“C”表示处理器核心, 阴影部分为裁剪掉的设计模块。

可重构芯片级环境还可以利用伪核 UVC 对真核进行替换, 在保持芯片的完整结构下重新构造芯片级环境。与真核执行的指令来自主存不同, 伪核 UVC 是通过伪随机生成器在线生成激励, 具有更好的随机性和可控性, 从而加大了芯片级验证的强度。由于伪核 UVC 采用验证语言抽象实现, 因此重构过的芯片级环境运行速度有很大的提升。图 4 为“重构”的多核处理器模拟环境, 环境中一部分核心被裁剪掉, 还有一些核心被替换为伪核 UVC。

可重构的芯片级环境可根据功能验证的需要, 通过环境配置自动重构验证环境。例如对多核 Cache 一致性的验证, 可以指定使用 N 个伪核和 M 个真核构造验证环境, 以加大验证测试强度; 再如验证芯片的可靠性、可测性等功能时, 可以只用一个真核进行验证, 以加快模拟速度。

可重构的全片级验证环境的验证方法有效化解了处理器多核化、SoC 集成化给处理器模拟验证带来的挑战, 保证功能验证覆盖率的同时, 提升了验证速度。

图 4 “重构”的多核处理器芯片级模拟环境
Figure 4 Reconstructed chip level environment of multi-core processor

图 5 多种验证手段发现错误比例
Figure 5 Ratio of errors found by different methods

4 验证效果

本文以一款 16 核通用处理器的功能验证为例, 说明层次化可重用验证环境在模拟验证中发挥的作用. 该处理器包含 4 个核组, 每个核组包含 4 个通用核心和 Cache 一致性处理部件, 4 个核组通过片上网络互连, 此外处理器还集成了 DDR3, PCIE2.0, 以太网等多个标准接口, 芯片设计规模达到了 6 亿多个晶体管.

4.1 模拟验证效果

该处理器同时采用了模拟验证、硬件仿真加速验证和 FPGA 实物验证以及代码风格检查 (Linting) 等验证手段, 从图 5 可以看出模拟验证仍然是处理器功能验证的主要手段, 模拟验证发现的设计错误占总错误的 85%.

表 1 层次化验证环境及可重用验证组件

Table 1 Hierarchical verification environments and reusable UVCs

Verification environment	Level	UVC composition
Memory unit environment	Component level	VCORE, eSYS
PCIE environment	Component level	PCIE VIP
Single core environment	Core level	eSYS
Chip environment	Chip level	VCORE, PCIE VIP, Ethernet VIP

图 6 层次化验证环境中的 UVC 重用

Figure 6 Reusable UVCs in hierarchical environments

4.2 多核处理器模拟环境实现

在该处理器的模拟验证中, 建立了一系列层次化可重用的部件级、单核级和芯片级模拟验证环境, 部分环境如表 1 所示。

从表 1 可以看出, 面向处理器结构实现的验证组件 VCORE 和 eSYS 在访存部件级环境、单核心验证环境和芯片级验证环境中都实现了重用, 标准接口 VIP 也可以在部件级和芯片级环境中重用。图 6 展示了该处理器验证中面向处理器结构实现的验证组件 VCORE 和 eSYS 的重用方法。这种层次化可重用的验证方法缩短了各种验证环境的开发时间, 提高了验证效率。

4.3 典型层次化模拟环境性能对比

由于验证环境规模和实现方法的不同, 不同层次的验证环境对同一设计的验证效率也各不相同。图 7 是在部件级环境、单核心级环境和单核芯片级裁剪环境下, 对处理器核心内的访存部件验证时, 各层次模拟环境的性能对比。图 7(a) 是这 3 种层次环境的模拟速度比较, 图 7(b) 是这 3 种层次环境的内存用量比较。从图中可以看到访存部件级环境运行速度最快、内存占用最少, 单核心环境次之, 而单核芯片级裁剪环境则居前两者之后。由于部件级和单核心级环境的内存用量相对较少, 因此基本上可以不受机器内存限制, 可同时启动多个模拟进程, 使得多处理器或者多线程模拟验证资源得到充分利用, 而全片级环境由于内存用量较大, 可能会受制于内存资源限制, 造成验证资源的浪费。

图 7 不同层次环境运行速度和内存用量比较

Figure 7 Simulation speed and memory consumption of hierarchical environments. (a) Simulation speed; (b) memory consumption

由此可见, 采用层次化的验证策略对某一设计进行验证时, 应根据各层次验证环境的特点和被验证设计的功能, 合理制定各层次验证环境的测试计划, 充分发挥不同层次验证环境的优势, 才能达到最佳验证效果。

4.4 可重构芯片级环境在典型配置下的性能对比

图 8 是该处理器可重构芯片级验证环境在几种典型配置下的运行速度和内存用量的比较。从图 8(a) 中可以看到由于将真实的核心替换为采用 E 语言抽象实现的伪核心 (VCORE), 减小了验证环境规模, 提高了模拟验证速度, 在核数相同的情况下, 采用“真伪各半”的芯片级环境比“全真核”的环境速度普遍快 90% 以上, 尤其在 16 核时效果最为明显, “真伪各半”的环境模拟速度提升了 170%。除了模拟速度的提升外, 由于 VCORE 的访存指令 IPC (instruction per clock) 数十倍于真实核心, 从而大大增强了“真伪各半”的芯片级环境在全片 Cache 一致性方面的验证强度。

从图 8(b) 中可以看到这种芯片级模拟环境重构方法还可以有效减少验证环境的内存开销。随着芯片核数的增加, 相同核数下“真伪各半”的芯片级环境与“全真核”的芯片级环境相比, 内存用量减少幅度逐渐增加, 在 16 核时最多减少内存用量达到 36%。

4.5 多核处理器层次化验证效果

图 9 是该多核处理器在各层次验证环境中发现的错误数占模拟验证发现的总错误数的比例。从图中可以看出, 将近 90% 的错误都是在运行速度快的部件级和单核级环境发现的, 这说明建立层次化的处理器模拟环境可以大大提高设计错误的收敛速度。

为了说明层次化验证环境对设计的交叉验证和互补验证的效果, 选取了本文所述多核处理器中访存部件和 PCI-E 接口部件作为分析对象, 统计这两部分设计在各层次验证环境下发现的错误数量进行对比分析, 图 10 为 PCI-E 接口部件和访存部件在不同层次的验证环境中发现的错误比例。

从图 10(a) 中可以看到 PCI-E 接口部件在部件级验证发现的错误数量最多, 这是因为 PCI-E 设计的功能较为独立, 与其他设计之间接口清晰, 且 PCI-E VIP 跨芯片重用性强, 验证环境可快速投入使用, 并且验证强度远远大于芯片级验证环境。从图 10(b) 中可以看到访存部件在单核环境发现的设

图 8 芯片级重构环境运行速度和内存用量比较

Figure 8 Simulation speed and memory consumption of reconstructed environments. (a) Simulation speed; (b) memory consumption

图 9 各层次验证环境发现错误比例

Figure 9 Ratio of errors found in environments of different level

计错误最多, 而部件级发现的错误相对较少, 这主要是因为访存部件与核心内其他模块之间接口复杂, 因而跨芯片重用 UVC 修改较多, 导致投入验证的时间比单核要晚, 而单核验证环境跨芯片的重用性很强, 投入验证较快, 因此很快发现了大量设计问题, 而访存部件验证环境则在中后期体现出其验证强度大的优势, 发现的问题多为复杂情况下的设计错误。而无论是 PCI-E 还是访存部件, 在芯片级环境验证中仍然发现了约 10% 的设计错误, 这主要是因为部件级环境和核心级环境在创建时就将验证的目标定位在设计功能实现的重点和难点上, 而与芯片全局相关的功能都必须依赖于芯片级模拟验证环境。

多层次的验证环境可以提高覆盖率指导下的验证效率, 加快覆盖率收敛速度。本文同样以 PCI-E 和访存部件的验证覆盖率为例, PCI-E 的覆盖率指导下验证采用了部件级和全片级两种环境, 访存部件则同时采用了部件级、核心级和全片级 3 种环境, 最终覆盖率目标是各层次环境下的覆盖率合并后达到 100%。本文通过综合考虑覆盖点贡献率和 CPU 消耗时间两个因素, 采用覆盖率排名分析 (ranking

图 10 层次化环境对设计的交叉验证效果

Figure 10 Hierarchical verification effect on design. (a) Ratio of errors found by PCIE environments; (b) ratio of errors found by memory unit environments

表 2 PCI-E 的覆盖率结果分析

Table 2 Coverage analysis of PCI-E design

Verification environment level (%)	Ratio of valid stimulus (%)	Cumulative coverage (%)
Component level	87	85
Chip level	13	57

表 3 访存部件的覆盖率结果分析

Table 3 Coverage analysis of memory unit

Verification environment level (%)	Ratio of valid stimulus (%)	Cumulative coverage (%)
Component level	31	65
Core level	62	89
Chip level	7	48

and ordering tests) 工具⁴⁾ 统计得到最小有效激励集合, 然后按照环境层次对集合内的有效激励进行分类, 并对有效激励占比和累积覆盖率进行统计, 如表 2、表 3 所示. 有效激励占比是指在最小有效集合中某层次环境下的激励所占的百分比, 它反映出在覆盖率指导下的验证中不同层次环境所发挥的作用大小; 累积覆盖率是指某层次环境下的有效激励覆盖率合并后的累积值, 不同层次环境间累积覆盖率存在交集.

从表 2 和表 3 可以看出, PCI-E 的部件级验证在覆盖率指导下的验证中发挥了主要作用, 有效激励占比达到了 87%, 全片级 PCI-E 环境则作为有效补充, 在部件级验证不足的方面发挥了重要作用; 访存部件的覆盖率指导下验证则以核心级验证为主, 部件级验证为辅, 全片级作为全局验证的有效补充. 需要说明的是, 层次化的验证环境是交叉互补的关系, 以上数据仅从覆盖率角度阐述了层次化环境的有效性, 在实际验证中应根据验证对象的特点, 制定合理的验证环境创建方案和测试激励开发计划, 以实现验证覆盖率尽快收敛的目标.

4) Cadence. ICC User Guide Product Version 12.1. <http://www.cadence.com>, 2014.

5 总结

本文在国产多核处理器的验证中建立的层次化可重用模拟环境, 分为部件级、核心级和芯片级, 这些不同层次的验证环境在处理器的验证中既存在交叉又相互补充; 通过系统地规划处理器各部分设计的验证层次, 既保证了功能验证的充分性, 同时又满足了模拟验证的速度要求, 降低了对模拟验证资源的消耗. 可重构的芯片级环境解决了大规模多核处理器芯片级模拟验证的难题, 在满足验证需求的同时提高了验证效率. 环境创建中开发的一系列面向处理器结构的可重用验证组件, 构成了最基础的处理器验证环境组件库, 将随处理器的发展而不断扩充完善, 进而为将来处理器模拟验证环境的快速开发提供有力保障.

随着高性能处理器技术的发展, 处理器的核心数量将不断增加, 并进入“众核 (many-core)”时代, 片内核心数量将达到 32~64 核, 甚至数百核心. 然而在可以预见的未来几年内, 模拟验证仍将是处理器功能验证的主要手段. 本文提出的层次化可重用模拟验证环境不但能够应用于多核处理器功能验证, 也同样适用于众核处理器验证; 不但可以解决某一款处理器模拟验证的问题, 还可以形成可持续发展的、完善的、高效的处理器模拟验证环境开发平台, 为后续处理器系列化快速发展提供强有力的支撑.

参考文献

- 1 He J, Wang B. Research on state-of-art and future of general purpose multi-core processor. In: Proceedings of the 15th National Conference on Computer Engineering and Technology and the 1st Microprocessor Forum. Changsha: National University of Defense Technology Press, 2011. 41-47
- 2 Fine S, Ziv A. Coverage directed test generation for functional verification using Bayesian networks. In: Proceedings of Design Automation Conference (DAC), Anaheim, 2003. 286-291
- 3 Bryant R E. A methodology for hardware verification based on logic simulation. J ACM, 1991, 38: 299-328
- 4 Taylor S, Quinn M, Brown D, et al. Functional verification of a multiple-issue, out-of-order, superscalar Alpha processor—the DEC Alpha 21264 microprocessor. In: Proceedings of 35th Design Automation Conference (DAC 98), NewYork, 1998. 638-643
- 5 Chen C. Research on the microprocessor function verification method based on hardware simulator. In: Proceedings of CNDLIVE Cadence User Conference, Beijing, 2012
- 6 Zhu Y, Chen C, Li Y Z, et al. Creation of FPGA verification platform for a high performance multiple-core microprocessor. In: Proceedings of 2012 Annual Conference of Advanced Computer Architecture, 2012
- 7 Ludden J M, Roesner W, Heiling G M, et al. Functional verification of the POWER4 microprocessor and POWER4 multiprocessor systems. IBM J Res Dev, 2002. 46: 53-76
- 8 Schubert K D, Roesner W, Ludden J M, et al. Functional verification of the IBM POWER7 microprocessor and POWER7 multiprocessor systems. IBM J Res Dev, 2011, 55: 1-10
- 9 Jacome M F, Peixoto H P. A survey of digital design reuse. IEEE Des Test Comput, 2001, 18: 98-107
- 10 Sagahyroon A, Lakkraj G, Karunaratne M. Verification components reuse. J Comput, 2012, 7: 2641-2649
- 11 Savage W, Chilton J, Camposano R. IP reuse in the system on a chip era. In: Proceedings of the 13th International Symposium on System Synthesis, Washington, 2000. 2-7
- 12 Hu X D, Guo Y, Zhu Y, et al. Design and application of instruction set simulator on multi-core verification. J Comput Sci Technol, 2010, 25: 267-273

Hierarchical and reusable simulation environment for high-performance processor verification

HU XiangDong*, JU PengJin, ZHU Ying & SUN Lu

National High Performance Integrated Circuit Design Center, Shanghai 201204, China

*E-mail: huxdisme@vip.sina.com

Abstract Multi-core and SoC integration have become the trend for high-performance processor development; this has made processor design more complex and chip scale even larger. Flat simulation verification environments consume a large amount of computer resources, which can cause the speed of the simulation to decline sharply, and consequently, it can result in poor simulation efficiency. However, simulation is still the most widely used method for processor verification, and processor simulation environment is crucial to simulation efficiency. Based on the processor architecture and the rules for reuse of verification components, a hierarchical reusable simulation verification environment can significantly solve the problems of large memory consumption, low simulation speed, and low verification efficiency. This method can also shorten the establishing and debugging periods, reduce mistakes in the environment, and make the simulation environment more easily reusable for different processor verifications.

Keywords processor, multi-core, simulation verification, hierarchical, reusable, verification component

HU XiangDong was born in 1964. He received his B.S. degree in electrical engineering from Zhejiang University, Hangzhou. Currently, he is a senior engineer at the National High-Performance Integrated Circuit Design Center (Shanghai) and also a senior member at China Computer Federation (CCF). His main research interests include high-performance computer architecture and VLSI design and verification.