ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.4, April 2009, pp.1077−1086
doi: 10.3724/SP.J.1001.2009.03602

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

“申威-1 号”高性能微处理器的功能验证*

黄永勤+, 朱 英, 巨鹏锦, 吴志勇, 陈 诚

(国家高性能集成电路(上海)设计中心,上海 201204)

Functional Verification of “ShenWei-1” High Performance Microprocessor

HUANG Yong-Qin+, ZHU Ying, JU Peng-Jin, WU Zhi-Yong, CHEN Cheng

(National High Performance IC (Shanghai) Design Center, Shanghai 201204, China)

+ Corresponding author: E-mail: yqh1601@pub.wx.jsinfo.net

Huang YQ, Zhu Y, Ju PJ, Wu ZY, Chen C. Functional verification of “ShenWei-1” high performance microprocessor. *Journal of Software*, 2009,20(4):1077−1086. http://www.jos.org.cn/1000-9825/3602.htm

Abstract: Today’s microprocessor designs are becoming more and more complicated, so effective and sufficient verification is one of the key factors to the success of design tape-out. This paper firstly introduces some common theories and methods in microprocessor functional verification, and then introduces the verification strategies and various verification methods used to verify “ShenWei-1” high performance microprocessor. The RTL (register transfer level) verification is highly important for functional verification. Simulation-Based verification is the main method in “ShenWei-1” RTL verification, and this paper introduces how to use all kinds of verification technologies to solve the key problems of the RTL simulation-based verification: Generating high quality stimulus, checking simulation results quickly, reaching the target of verification coverage. At the end, this paper analyses the effects of all kinds of verification methods.

Key words: functional verification; pseudo-random test stimulus; functional coverage; reference model; real-time comparison

摘 要: 微处理器设计日趋复杂,如何对微处理器设计进行有效而充分的验证,成为芯片流片成功的关键因素之一.在介绍微处理器功能验证的一般理论和方法的基础上,介绍了“申威-1 号”高性能微处理器的功能验证所采用的验证策略及各种验证方法.RTL(register transfer level)级验证是功能验证的重点,模拟验证是“申威-1 号”RTL 级验证的主要验证手段.详细介绍了如何综合采用多种验证技术来解决 RTL 级模拟验证的几个关键问题:高质量测试激励生成、模拟结果正确性的快速判断以及验证覆盖率目标的实现.最后对各种验证方法所取得的验证效果进行了分析.

关键词: 功能验证;伪随机测试激励;功能覆盖率;参考模型;实时比较

中图法分类号: TP302 **文献标识码**: A

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2004AA1Z1080 (国家高技术研究发展计划(863))

Received 2008-11-10; Accepted 2009-03-16

1 引 言

1.1 高性能微处理器验证面临的挑战

高性能微处理器为了追求高性能,通常采用超级标量、乱序发射、转移预测、预取、推测执行等先进的微处理器实现技术,实现结构变得越来越复杂,功能验证成为其设计开发过程中的一个重要瓶颈.为了实现一次投片成功的目标,高性能微处理器功能验证面临下面的挑战:

(1) 指令和操作数的验证空间巨大

高性能微处理器指令集通常包括几百条指令,指令的合法组合序列几乎是一个无限大的空间.即使是相同的指令序列,也会因中断、初态等的不同而导致指令执行过程的不同,所以指令的验证空间十分巨大.而同一条指令,由于操作数的不同会引起执行过程的不同.对于 64 位高性能微处理器,任意一个操作数的取值可以有 2^{64} 种情况,因此操作数的验证空间是十分巨大的,尤其是对浮点运算部件而言,这是一个巨大的挑战.

(2) 验证充分性的全面、客观衡量

验证人员需要在验证过程中不断地了解验证对象的正确性程度,对验证充分性进行全面、客观的衡量,从而能够指导验证工作不断深入、全面地开展,达到设计错误的快速收敛,提高验证的效率.所以,如何对验证充分性做出全面、客观的衡量,是验证过程中面临的又一难题.

本文的验证对象是“申威-1 号”微处理器,该处理器为通用 64 位 Load/Store 型 RISC 结构,指令集包含 200 多种指令,采用了超级标量、乱序发射、转移预测、推测执行等实现技术,包含多级片上 Cache,并支持多机的 Cache 一致性,支持浮点乘加和 SIMD 运算,全片包含几千万只晶体管.这款逻辑极其复杂的高性能微处理的功能验证向设计人员提出了严峻的挑战,其验证工作所用时间接近整个设计周期的 60%.

1.2 高性能微处理器验证技术的发展现状

面对高性能微处理器验证这一业界难题,世界著名的微处器研发公司都在这方面投入了大量的人力和财力,通常设计与验证人员之比达到 1:3~1:4.目前,微处理器的功能验证主要采用模拟验证、形式验证、硬件仿真加速等验证方法[1−3],这些验证方法各有千秋.

形式验证是通过数学的方法来证明设计的正确性,这种方法保证设计实现的所有可能行为都被考虑到,能够证明设计没有缺陷,可达到 100%的状态覆盖率.常用的形式验证工具有等价性检查工具(equivalence checkers)和设计属性检查工具(property checkers),其中,设计属性检查工具又称为 model checkers[4].等价性验证工具主要用于 RTL 级设计与物理实现之间的等价性验证;设计属性检查工具主要用于设计的功能验证,这类工具使用形式数学的方法来检查设计行为是否遵循设计者定义的一些规则.由于大规模设计的状态空间巨大,超出了目前工具的验证能力,因此设计属性检查工具主要应用于局部复杂逻辑的验证[5,6].

模拟验证是通过检查设计描述在测试激励作用下的行为正确与否来判断设计的正确性,当设计复杂时,一般很难达到 100%的验证覆盖率,模拟验证的效果主要取决于测试激励的质量、模拟结果正确性判断方法、覆盖率分析等几个方面[7].在微处理器的验证中,通常采用手工编写的焦点测试激励、系统和应用程序[8]、伪随机指令序列作为模拟验证的激励[9],并采用参考模型比较法、断言检查法、事务检查法、程序自检查法这些正确性判断方法[1−3].针对微处理器的巨大验证空间,通常会基于模拟结果快速判断机制来建立大规模的自动模拟验证平台进行海量验证[2].

硬件仿真加速是通过频率指标低于设计指标的硬件来实现设计描述,验证速度远远大于模拟验证.微处理器的硬件仿真加速可以基于商用硬件加速器[2,3]或采用 FPGA(field-programmable gate array)来实现 RTL 级设计描述[10].在此基础上建立的系统级验证平台,使得在流片前就可以进行操作系统和应用程序的正确性验证,并可进行性能快速评估.

形式验证目前主要用于验证局部逻辑区域的功能正确性.硬件仿真加速虽然能够有效提升验证速度,但缺点是错误定位比较困难.模拟验证一般不会受限于设计规模的大小,具有错误定位方便的特点,而且可以通过综合使用各种测试激励生成和结果检查方法使验证达到很好的效果,至今仍是高性能微处理器功能验证采用的

主要手段.因此,如何提高这种验证方法的验证效果是高性能微处理器验证中需要解决的一个关键问题.

本文在“申威-1 号”的验证中也采用了这些验证手段,而模拟验证是本文采用的主要方法,硬件仿真加速则作为模拟验证的有效补充.效率是模拟验证的关键,而模拟验证的效率就是发现错误、定位错误的能力.为了提高“申威-1 号”RTL 级模拟验证的效率,本文对以下几个方面进行了深入研究:

(a) 突破了系统和应用程序规模大、模拟时间长的验证难点;

(b) 开发了高效的伪随机指令序列生成器,使得伪随机海量验证成为模拟验证的主要手段[9];

(c) 采用参考模型比较法、断言检查法、程序自检查法这些正确性判断方法[1−3],极大地提高了错误发现和定位的效率;

(d) 针对微处理器的巨大验证空间,利用集群计算机系统建立了大规模的自动模拟验证平台,进行海量自动验证[2].

2 RTL 级验证策略和验证环境的创建

由于“申威-1 号”结构复杂,为了保证验证的充分性,RTL 级验证采取了层次化的验证策略,按照模块级、部件级、全片级、系统级的先后层次进行验证.基于层次化验证策略创建的“申威-1 号”RTL 级验证环境包括模块级和部件级模拟验证环境、全片级模拟验证环境以及 FPGA 物理原型验证平台这几个部分,其中,全片级模拟验证环境如图 1 所示.

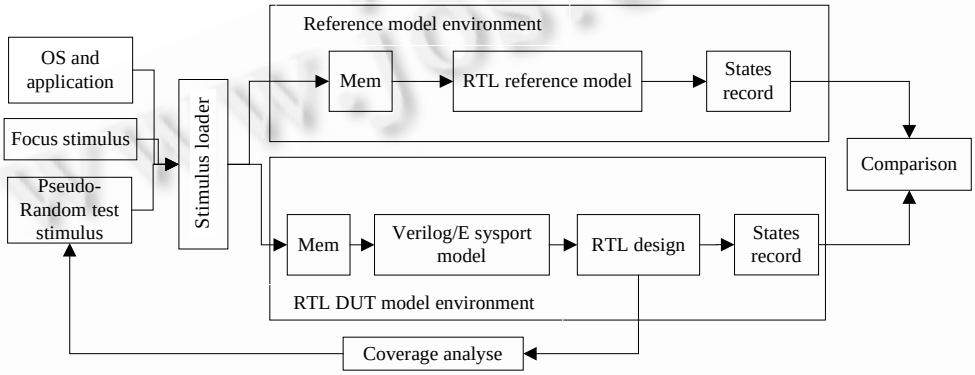

Fig.1 RTL chip-level simulation based verification environment

图 1 RTL 级全片模拟验证环境

RTL 级全片模拟验证环境中包含“申威-1 号”的 RTL 级参考模型,RTL 级参考模型是 RTL 级设计的高层抽象,用以实现基于参考模型的模拟结果自动判断.模拟验证过程中,测试激励被同时加载到参考模型和 RTL 级设计模型上,通过比较两者的关键状态信息来快速判断 RTL 级设计的正确性.模拟过程中统计的覆盖率信息用以指导测试激励的生成[11],以提高激励的生成质量.

3 RTL 级模拟验证

RTL 级验证是微处理器功能验证的重点,而模拟验证是“申威-1 号”RTL 级验证所采用的主要验证手段之一.鉴于该处理器验证空间巨大,模拟验证需要解决 3 个方面的关键问题:

(1) 高质量测试激励生成;

(2) 模拟结果正确性快速检查;

(3) 验证覆盖率分析及覆盖率目标的实现.

本节将具体阐述在“申威-1 号”的验证过程中如何解决这 3 个方面的关键问题.

3.1 高质量测试激励生成

测试激励质量是影响模拟验证效果的重要因素之一.在“申威-1 号”的测试激励开发中,考虑到验证效率、全面性等方面的因素,主要采用了 3 种测试激励:

(1) 手工焦点测试激励;

(2) 伪随机测试激励;

(3) 系统和应用程序.

为了确保各类激励的验证效果,还实现了覆盖率指导下的测试激励生成.

3.1.1 手工焦点测试激励

焦点测试激励主要用于初期验证和一些逻辑边界的验证.焦点测试激励是手工编写的各种汇编程序,如单指令、指令集的两两组合序列等.验证初期设计错误较多,通过焦点测试对设计的基本功能进行验证,错误定位方便.验证的中后期,焦点测试主要针对一些特定逻辑区域,尤其是对逻辑边界情况的测试非常有效.

3.1.2 可重用伪随机测试激励

“申威-1 号”可重用伪随机测试激励自动生成环境如图 2 所示.该环境的开发主要基于伪随机序列技术,由伪随机指令序列库、指令集描述、伪随机指令序列生成器(pseudo-random instruction generator,简称 PRISG)等几部分组成.伪随机指令序列生成器根据伪随机指令序列描述生成符合验证约束的伪随机指令序列.伪随机指令序列描述遵循制定的序列可重用规则,符合可重用规则的伪随机指令序列在开发新的序列时可以被方便地引用,进行组合和嵌套,生成更复杂的测试激励,通过测试激励高度复用,实现激励生成的高效性.

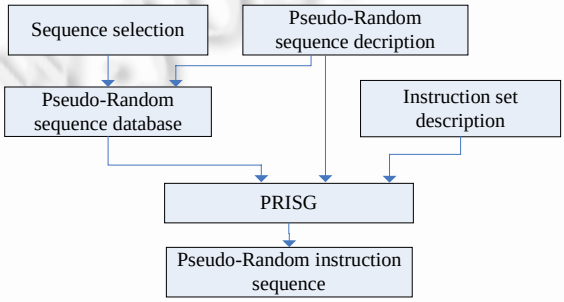

Fig.2 Pseudo-Random tests stimulus auto generation environment

图 2 伪随机激励自动生成环境

3.1.3 系统和应用程序

为了充分验证“申威-1 号”在运行实际系统和应用程序时的正确性,需要将这些程序同时在 FPGA 验证平台和模拟环境下进行验证.由于操作系统(operating system,简称 OS)和应用程序运行规模都很大(一般达千万级指令),在模拟验证中由于受到验证速度的限制,必须对 OS 和应用程序进行必要的精简.

3.1.4 双覆盖率指导下的测试激励生成

为了提高测试激励的生成质量,“申威-1 号”同时采用了代码覆盖率和功能覆盖率两种覆盖率分析手段,实现了双覆盖率指导下的测试激励生成.

代码覆盖率分析的对象是 RTL 设计代码,包括语句、分支、状态机等内容;功能覆盖率分析的对象则是“申威-1 号”的微结构功能和逻辑边界.选择两种覆盖率分析机制,可以兼顾实用性、易用性、分析代价几个方面.在验证初期,采用代码覆盖率分析,二次开发的工作量较小,可以迅速定位未验证到的 HDL(hardware design language)代码,同时分析时耗费的计算资源也比较少,实用性和易用性都很高.但是随着验证的深入,仅分析某一代码行是否被覆盖到是远远不够的,为此,我们开发了“申威-1 号”的全片功能覆盖率模型,通过该模型,可以充分分析用户所定义的功能点是否被覆盖.事实证明,功能覆盖率分析在保证“申威-1 号”验证的充分性中发挥了重要作用.

3.2 模拟结果正确性快速判断

由于高性能微处理器的验证空间巨大,所以需要有高效的模拟结果正确性判断机制来提高模拟验证的效率.“申威-1 号”采用了多种当今流行的手段来进行模拟结果正确性判断,主要有如下几种:

(1) 测试激励自校验法

当采用指令序列作为测试激励时,有些激励的最终运行结果有一个确定的期望值,这种激励本身可以将模拟运行结果与期望值进行比较,通过这种比较来检查激励在设计模型上模拟运行是否正确.但这种激励自校验方法必须等到程序模拟完成之后才能知道结果是否正确,错误定位调试较为困难.

(2) 人工检查法

人工检查主要是通过在模拟过程中全程跟踪记录模拟波形,对模拟波形进行观察分析来判断模拟结果的正确性.这种方法不能实现自动化回归测试.

(3) 断言检查法

断言检查是通过在设计模型的描述中增加有关设计对象的属性或规则描述,在模拟过程中实时地判断模拟结果是否违反相应的属性或规则.断言检查器可以快速定位出错误点,但过多的断言描述会影响模拟速度.

(4) 参考模型比较法

参考模型比较法通过在参考模型和 RTL 级设计模型上运行相同的测试激励,并比较两者的关键状态输出是否相同,来判断设计模型模拟结果的正确性,从而发现相应的错误点.我们将这种方法称为基于参考模型的验证方法(reference verification methodology,简称 RVM).

为了实现“申威-1 号”基于参考模型的 RTL 级验证,本文用 C 语言开发了“申威-1 号”的 RTL 级参考模型,并通过在模拟环境的 Testbench 中增加相应的监测功能模块来实时监测记录关键状态信息,记录的关键状态信息包括提交指令数、程序计数器(PC)值、整数和浮点寄存器的值、内部控制寄存器(IPR)的值、内部和外部存储系统的状态和数据等微处理器对软件可见的全部状态.

由于参考模型与 RTL 级设计模型在描述抽象层次、描述语言上存在差异,导致两种模型的模拟速度差异较大.为了实现模拟过程中两种模型的同步,“申威-1 号”的 RTL 级验证平台中采用 E 语言,解决了 RTL 级设计模型与参考模型之间的同步问题,实现了基于参考模型的模拟结果指令级实时自动判断.E 语言内嵌的 PLI 接口可以控制 RTL 级设计模型的模拟运行,同时,通过 E 语言与 C 语言间的函数调用,可以合理调度参考模型的运行.每条指令完成时刻对设计模型和参考模型的关键信息进行比较,只有在发现比较有不同时才记录设计模型和参考模型运行的出错信息,如出错指令和出错节拍等.这种模拟结果实时自动判断方法,有效地提高了模拟验证的速度和错误定位的效率.而且通过 RTL 级设计与参考模型之间的同步,在中断、多机一致性请求等异步事件干扰的情况下仍能实现模拟结果的实时自动判断,错误自动定位精度达到节拍级.

3.3 验证覆盖率目标的实现

针对“申威-1 号”巨大的验证空间,为了尽快实现验证覆盖率目标,主要采取如下几方面的验证手段.

3.3.1 浮点部件的专项验证

浮点部件的验证是高性能微处理器验证中的一大难点.“申威-1 号”是 64 位处理器,支持 IEEE 标准的浮点数据格式,支持多种浮点算术运算,浮点部件验证空间巨大.为此,我们开发了两个浮点专项验证环境:(1) 浮点部件级模拟验证环境;(2) FPGA 物理原型验证平台上的浮点运算部件自验证环境(在第 4.2 节详细给出介绍).“申威-1 号”浮点部件的验证工作主要基于这两个专项验证环境和一个 RTL 级全片模拟验证环境来展开.

浮点部件级模拟验证环境(floating-point unit verification environment,简称 FUVE)的结构如图 3 所示,该环境的主要构成为测试激励生成引擎、浮点参考模型、测试激励加载和结果比较 Testbench、功能覆盖率描述和浮点部件 RTL 级设计.基于该环境进行验证时,首先描述测试激励的约束,测试激励生成引擎解析这些约束生成测试激励,然后根据浮点部件要求的时序将测试激励传送给浮点部件.测试激励在传送给浮点部件的同时也传送给浮点参考模型,浮点部件和参考模型在测试激励的驱动下开始运算,两者运算的结果由结果比较程序加以

比较,根据比较结果就可以判断浮点部件运算是否正确.在浮点部件进行运算的同时,可以根据定义的事件和对象跟踪覆盖率,覆盖率分析结果可以指导下一步测试激励的生成.

3 种验证环境在验证重点、验证效率上各有不同,见表 1.通过表 1 中的对比可以知道,这 3 个环境在验证功能上具有互补性,协同采用高效地实现了浮点部件的验证覆盖率目标.在制定验证计划时,我们将浮点算法逻辑的验证主要放在部件级模拟验证环境来实施,浮点部件其他逻辑的验证在全片级模拟环境中实现,而约束较少的大量随机测试在 FPGA 自验证环境上实施.

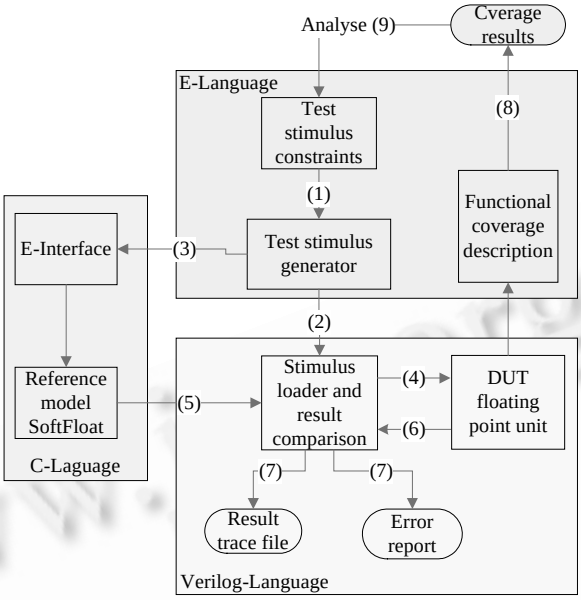

Fig.3 Framework of floating-point unit verification environment

图 3 浮点部件验证环境结构图

Table 1 Comparison between three floating-point-unit verification environments

表 1 3 种浮点验证环境对比表

	Float unit level environment	Full chip level environment	FPGA self-test environment
Speed	2000 cycles/s	500 cycles/s	12×10^6 cycles/s
Verification focus	The correctness of floating-point operation algorithms: “+”,“–”,“×”,“÷”etc	All types of float instructions: Branch and jump, memory access, and float exceptions etc	Verify float unit by using integer arithmetic unit
Merits	Easy implementation, fast debug capability, and can shorten the time of getting high coverage	The interface timing is the same as the real chip	Fast running speed

3.3.2 大规模测试激励的加速验证

高性能微处理器投片之前一般需要经过操作系统和应用程序的验证.这些测试激励包含的指令数量较大,一般达上千万条指令,当“申威-1 号”采用这些测试激励进行模拟验证时,会面临如下难点:

(1) 每更换一个应用程序,需要对操作系统初始化过程进行重复验证,这种重复的验证过程严重影响了模拟验证的效率;

(2) 操作系统和应用程序的规模较大,一个完整程序的验证需要长达几十或上百小时的验证时间,一旦在验证中间发生错误,如何快速地重现、定位模拟验证过程中的错误是提高其验证效率的关键之一.

本文在“申威-1 号”的验证过程中,通过下面两个手段实现了大规模测试激励的加速验证:

(1) 实现基于微处理器结构特征的断点恢复机制

本文根据“申威-1 号”的结构特点,开发并实现了一套基于微处理器结构特征的断点恢复机制.这套基于结

构特征的断点恢复机制在模拟运行断点处只保留“申威-1 号”对软件可见的全部结构状态,在恢复模拟运行时恢复这些结构状态.当采用应用程序进行验证时,通过这种断点恢复机制,可以在模拟速度较快的参考模型上完成OS初始化过程运行,然后将参考模型上初始化过程结束时刻的状态加以保留,并恢复到“申威-1号”设计模型上,直接开始应用程序的验证,极大地加速了应用程序的模拟验证过程.基于这一机制,还可以在大型测试激励的模拟过程中先不记录波形轨迹,而是按一定的时间间隔设置多个断点,这样,一旦发现错误,可以从离错误点最近的断点恢复模拟过程,跟踪记录波形,从而有效地提高了错误定位的效率.

(2) 简化测试激励

在对操作系统核心进行验证时,将对正确性验证贡献不大的部分进行压缩简化,如磁盘影像解压缩部分可进行大幅度压缩.应用程序验证通常以实际应用为蓝本,将其缩小规模或截取其核心算法及对验证有用的部分来进行验证.

3.3.3 随机化的外部系统环境

微处理器的行为不仅取决于指令序列,还依赖于外部系统环境,一般高性能微处理器验证环境都包含一个外部系统环境模型.为了保证验证覆盖率,外部系统环境模型应描述尽可能多的系统行为.

为了充分验证“申威-1 号”在各种系统使用环境下的正确性,在全片级模拟验证环境中,用 E 语言开发了一个伪随机外部系统事务生成器来模拟实际系统的存储器访问、I/O 访问、一致性请求及中断请求等操作,实现了一个随机化的外部系统环境,它与“申威-1 号”设计结合成一个完整的可运行模拟环境.该模拟环境在满足“申威-1 号”系统接口协议的前提下,充分发挥伪随机技术的优点,可以实现两个方面的随机化:(1) 系统给 CPU 的响应命令类型和响应延迟的随机化;(2) 系统发给 CPU 的一致性请求、中断等异步事件的随机化.实现这两方面的随机化,可以最大程度地验证“申威-1 号”与系统接口相关部分逻辑的正确性.

3.3.4 海量自动验证

基于本文实现的伪随机测试激励自动生成和模拟结果自动判断机制,在集群计算机系统上建立的“申威-1号”模拟验证环境可以根据指令序列的约束连续自动生成伪随机测试激励,自动判断模拟结果的正确性,最终在该验证环境上实现了伪随机测试激励海量自动验证,每天可以自动完成近两亿条指令的自动验证,为“申威-1号”模拟验证覆盖率目标的实现提供了有力保障.

3.3.5 覆盖率分析

“申威-1 号”的功能验证同时采用了代码和功能两种覆盖率分析机制[12],验证过程中完成了多次覆盖率分析,并用覆盖率分析结果及时指导下一步测试激励的开发,特别是伪随机测试激励的开发.这种覆盖率指导下的测试激励开发,有效地加快了错误的收敛速度.代码覆盖率在验证中期就接近了 100%,在验证的中后期达到了100%;流片之前全片的功能覆盖率达到了 99.7%,全片只有 26 个功能点没有在全片模拟环境下覆盖到.最终,这些没有覆盖到的功能点通过设计确认、在模拟环境下强制信号状态等验证手段进行了验证.

4 物理原型验证

物理原型验证是高性能微处理器验证通常采用的一种验证手段.为了加快“申威-1 号”的功能验证进度,我们还开发了物理原型验证平台.

4.1 物理原型验证平台

“申威-1号”物理原型验证平台使用多片大容量FPGA实现了“申威-1号”的全部逻辑,并用DDR(double data rate)传输技术解决了多个芯片之间互连线超过芯片信号引脚数的问题,并用 FPGA 实现了外部系统控制逻辑.物理原型验证平台能够同时运行 LINUX 和 UNIX 操作系统,具有完善的用户界面,“申威-1 号”工作频率可达12MHZ,能够方便、快捷地对各种应用程序和性能测试 Benchmark 进行验证,并具有各种调试支持功能.

“申威-1 号”物理原型验证平台实现了系统环境下的软、硬件协同验证,在样片返回之前,较好地支持了操作系统等系统软件的调试.在物理原型验证平台上,通过操作系统和应用程序的测试,往往可以发现模拟验证不易发现的错误.在该平台上,通过性能测试 Benchmark 的测试,还能快速评估芯片的各项性能指标.

4.2 浮点运算部件自验证环境

针对浮点验证空间巨大的特点,本文在“申威-1 号”物理原型验证平台上建立了浮点部件自验证环境.该环境能够按照约束自动生成浮点汇编程序,将此程序作为浮点参考模型的输入.由于浮点参考模型采用整数运算来实现浮点运算,所以浮点参考模型的运行将由整数部件来执行;再由浮点部件执行此浮点汇编程序得到一个结果,通过比较整数部件与浮点部件的两个浮点运算结果是否一致,就能判断出浮点部件运算的正确性.这种自验证环境由于其快速性,可以有效地提高浮点运算部件相关的验证覆盖率.

5 验证效果分析

“申威-1 号”的研发从方案设计到流片,时间长达 3 年多,验证工作贯穿整个研制周期,而 RTL 级的验证从验证环境开发到整个验证工作结束历时也将近两年.我们对 RTL 级验证过程中发现的错误用数据库进行了管理,基于这一错误管理数据库,本文进行了一些统计分析.

5.1 各类错误的汇总分析

图 4 是 RTL 级验证过程中累计错误数统计,验证初期发现的错误总数增长得很快,验证的中期错误总数基本是匀速增长,这个阶段持续时间很长,到了流片前的验证后期,错误总数基本保持不变.验证主管可以根据错误收敛趋势对目前所处的验证阶段进行判断,并进一步制定下一步的验证策略.

图 5 是 RTL 级验证过程中发现的各种类型错误的统计结果.我们可以看到,在验证发现的各种错误中,设计缺陷错误达到了一半以上,而由于修改而引入的错误超过了 20%.这说明,由于设计的复杂性,往往针对一个设计缺陷的修改又会引入新的设计错误.笔误在所有错误中排名第三,这是因为,我们的设计队伍普遍是初次接触 Verilog 语言,因此在代码编写上还不够熟练,所以引入的笔误较多.协议错误比较少,这是因为该微处理器采用的是自上而下的设计流程,因此大多数的接口协议错误在算法级验证中就已被发现.之所以优化错误最少,是因为对代码的优化是在 RTL 设计和验证已初步完成时,才根据电路设计要求以及 FPGA 平台的性能测试结果进行了少量的代码调整.

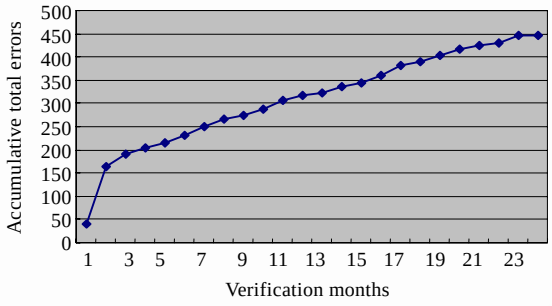

Fig.4 Accumulated errors by month

图 4 按月累计错误数统计图

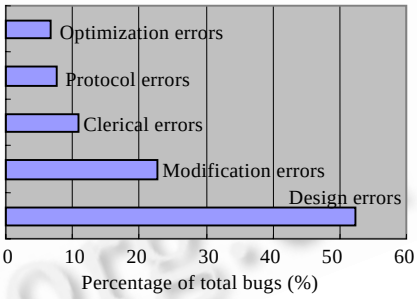

Fig.5 Introduction of bugs

图 5 错误引入原因统计图

图 6 是验证过程中不同的测试激励所发现的错误率的统计,我们可以清晰地看到,伪随机激励测试发现的错误数高达 65%,表明其在“申威-1 号”验证中所起的重要作用;手工编写的焦点测试激励发现了 25%的设计错误;其他的 10%错误通过操作系统、应用程序等验证手段发现.

“申威-1 号”的验证采用了多种验证方法:基于参考模型的验证方法(RVM 方法)、FPGA 原型验证法、静态检查法以及断言检查法等,图 7 是不同的验证方法发现的错误率统计.这里,静态检查法是指利用编码风格检查和可综合性检查等工具对 RTL 级代码进行静态分析以发现代码设计中的错误.RVM 方法发现的错误达到了 83%;FPGA 原型验证方法发现了 12%,这种方法通常能够发现深层的设计错误,是模拟验证的有效补充;静态检查发现了 2%;其他通过断言检查、人工校对以及外围电路验证发现的错误,占了 3%.由此可见,RVM 验证方法是“申威-1 号”的一种主要验证手段,对“申威-1 号”的验证成功发挥了非常重要的作用.

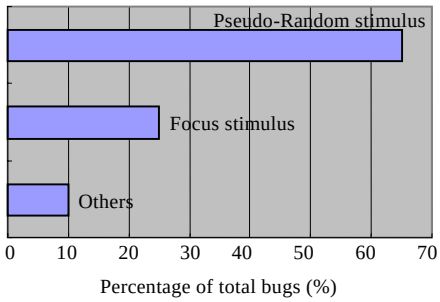

Fig.6 Effectiveness of test category

图 6 各类测试激励发现错误率统计图

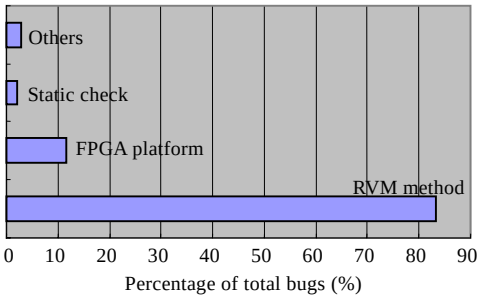

Fig.7 Effectiveness of different verification methods

图 7 各种验证方法发现错误率统计图

5.2 浮点验证效果分析

浮点部件的验证采用部件级专项验证与全片级验证相结合的方法.如图 8 所示,浮点部件级验证环境(FUVE)建立初期的两个月中共发现了 33 个错误,包括设计缺陷、协议理解等类型的错误;该环境在第 2 阶段 3 个月的验证中采取了覆盖率指导下的测试激励生成方法,又发现了 14 个错误;在第 3 阶段的验证工作中共发现 5 个错误,此阶段虽然由于物理设计的需要对设计作了大量的修改,但由修改所引入的错误只发现了 2 个,这应归功于基于部件级验证环境建立的自动回归测试流程.

全片和部件级验证环境发现的浮点错误数对比如图 9 所示,部件级验证环境发现的错误约占总数的 70%,而全片级发现的错误约占 30%.由此可见,部件级验证环境生成的激励更有针对性,与全片验证环境相比,其验证速度更快,更能有效地发现浮点设计的错误,但全片验证环境能够有效发现浮点部件与其他部件之间接口上的错误.验证效果上两种环境互相补充,最终较好地解决了浮点部件的验证问题.

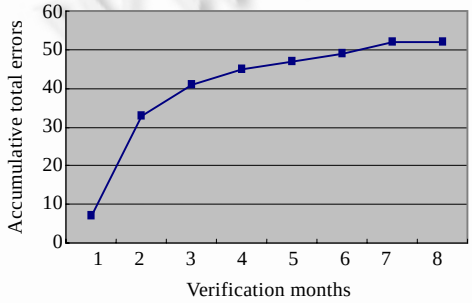

Fig.8 Accumulated bugs found in FUVE

图 8 FUVE 发现错误数累计统计图

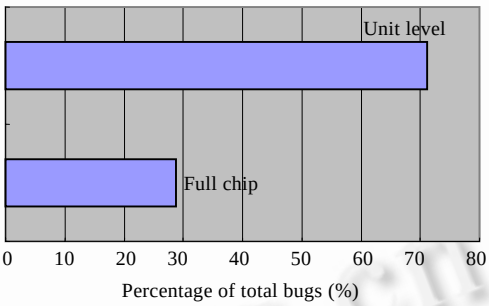

Fig.9 Effectiveness different verification level

图 9 不同验证层次发现错误率统计图

6 总结和展望

本文在“申威-1 号”的验证过程中,建立了一套完整的验证环境.验证环境实现的基于参考模型的模拟结果指令级实时自动判断方法、可重用伪随机测试激励生成方法以及所采取的浮点部件专项验证、大规模测试激励加速验证、FPGA 物理原型验证等验证手段,从多个方面提高了验证的效率,解决了“申威-1 号”指令和操作数的验证空间巨大的问题;代码和功能覆盖率分析在验证过程中的同时采用,使“申威-1 号”的验证充分性有了全面、客观的衡量标准,保证了验证的充分性,使首次投片返回的“申威-1 号”样片能够成功引导操作系统,实现了一次投片成功的目标.我们下一步的研究工作将更多地集中在如何应对多核微处理器的功能验证上,将从并行加速验证、形式验证等多个方面来展开研究.

致谢 在此,我们向对本文的工作给予支持和建议的同事表示感谢.

References:

[1] Taylor S, Quinn M, Brown D, Dohm N, Hildebrandt S, Huggins J, Ramey C. Functional verification of a multiple-issue, out-of-order, superscalar alpha processor—The DEC alpha 21264 microprocessor. In: Proc. of the Digital Equipment Corporation, the 35th Design Automation Conf. San Francisco: ACM, 1998. 638−643.

[2] Ludde JM, Roesner W, Heiling GM, *et al*. Functional verification of the POWER4 microprocessor and POWER4 multiprocessor systems. IBM Journal of Research and Development, 2002,46(1):53−73.

[3] Wazlowski ME, Adiga NR, *et al*. Verification strategy for the Blue Gene/L chip. IBM Journal of. Research and Development, 2005,49(2/3):303−318.

[4] Clarke EM, Wing JM. Formal methods: State of the art and future directions. ACM Computing Surveys (CSUR), 1996,28(4): 626−643.

[5] Bentley B. Validating the Intel® Pentium® 4 microprocessor. In: Proc. of the 2001 Int'l Conf. on Dependable Systems and Networks (formerly: FTCS). 2001. 493−500.

[6] Kaivola R, Narasimhan N. Formal verification of the Pentium 4 floating-point multiplier. In: Proc. of the Design Automation and Test in Europe Conf. and Exhibition. Paris, 2002. 20.

[7] Fournier L, Arbetman Y, Levinger M. Functional verification methodology for microprocessors using the Genesys test-program generator—Application to the x86 microprocessors family. In: Proc. of the Design Automation and Test in Europe Conf. and Exhibition (DATE'99). Munich, 1999. 92−es.

[8] Chang YS, Lee SJ, Park IC, Yung KCM. Verification of a microprocessor using real world applications. In: Proc. of the 36th ACM/IEEE Conf. on Design Automation. New Orleans, 1999. 181−184.

[9] Bartley MG, Galpin D, Blackmore T. A comparison of three verification techniques: Directed testing, pseudo-random testing and property checking. In: Proc. of the 39th Conf. on Design Automation. New Orleans, 2002.

[10] Zhang H, Shen HH. Function verification of Godson2 processor. Journal of Computer Research and Development, 2006,43(6): 974−979 (in Chinese with English abstract).

[11] Fine S, Ziv A. Coverage directed test generation for functional verification using Bayesian networks. In: Proc. of the 40th Conf. on Design Automation. 2003.

[12] Kantrowitz M, Noack LM. I'm done simulating; now what? verification coverage analysis and correctness checking of the DECchip 21164 alpha microprocessor. In: Proc. of the 33rd Design Automation Conf. Las Vegas: ACM, 1996. 325−330.

附中文参考文献:

[10] 张珩,沈海华.龙芯 2 号微处理器的功能验证.计算机研究与发展,2006,43(6):974−979.

黄永勤(1955－),女,广东人,高级工程师,CCF 高级会员,主要研究领域为高性能计算机体系结构,微处理器验证.

朱英(1964－),女,高级工程师,主要研究领域为微处理器验证,微处理器体系结构.

巨鹏锦(1979－),男,工程师,主要研究领域为微处理器验证及性能评测.

吴志勇(1976－),男,工程师,主要研究领域为微处理器验证及性能评测.

陈诚(1980－),男,工程师,主要研究领域为微处理器验证及性能评测.