多核处理器验证中存储数据错误快速定位机制*

周宏伟,邓让钰,李永进,晏小波,窦 强
(国防科技大学 计算机学院,湖南 长沙 410073)

摘 要:提出并实现的一种数据错误快速定位机制(Fast Fault Location Mechanism, FFLM)面向多核处理器存储系统的功能验证,FFLM基于硬件仿真器构建多端口存储器黄金模型,通过在仿真过程中实时监控存储系统与处理器核之间的访存报文,在线比较被测系统访问真实存储器的数据与黄金模型中的对应数据是否一致,在错误数据从存储系统送入处理器核的时刻就能够发现数据错误。与传统方法相比,FFLM具有仿真速度快、硬件资源代价低以及定位错误时间短的优点。对自主设计的CMP-16多核处理器进行仿真时的统计数据表明:使用FFLM后定位数据错误的速度能够比未使用FFLM时平均提高6.5倍。

关键词:多核处理器;验证;存储数据错误;定位机制
中图分类号:TP302.1 **文献标志码**:A **文章编号**:1001-2486(2012)06-0001-06

A fast location mechanism on memory data error for multi-core processors verification

ZHOU Hongwei, DENG Rangyu, LI Yongjin, YAN Xiaobo, DOU Qiang
(College of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract: A fast fault location mechanism on memory data error, which is called FFLM for a self-made CMP-16 multi-core processor's functional validation, is proposed and realized. FFLM builds a multi-port golden memory model based on the hardware emulation accelerator. It monitors the packages of memory access between memory system and processor cores during the emulation, real-time compares the data from real memory system being tested and the data from golden memory model, judges whether they are consistent, and finds the errors once any wrong data is sent to processor core from memory system. Compared with traditional ways, FFLM has the advantages of fast emulation speed, low hardware cost and low fault Location time cost. Statistical results from the emulation for a self-made CMP-16 multi-processor show that FFLM improves the speed of date fault location in memory system by 6.5 times averagely.

Key words: multi-core processor; verification; memory data error; location mechanism

随着高性能多核微处理器设计复杂度的提升,验证复杂度呈指数级提高。软件模拟器具有调试功能丰富和可观察性强的特点,但是模拟速度非常慢,因此比较适合在模块级进行设计验证,在芯片级和系统级仅支持小规模的验证。硬件仿真器既具有软件模拟的灵活性又具有硬件加速的高性能,主要用于在系统级验证中执行操作系统和应用软件,节省验证时间。处理器的功能验证通常在不同的验证层次使用不同的验证机制[1-3]。近年来,随着自主处理器研制水平的快速发展,国内对处理器验证的相关研究越来越多。郭阳等对微处理器模拟、硬件仿真、形式验证等方法的原理、特点和适用场合进行了探讨[4]。对于大规模处理器芯片,构建基于FPGA的原型验证平台是提高验证速度的有效方法,如"龙腾S2"处理器[5]和飞腾64处理器[6]的验证使用了FPGA原型验证平台,其中文献[5]提出在参考计算机平台上运行测试程序,然后利用IP对照定位错误位置的错误定位方法,文献[6]通过比较参考模拟器导出的运行日志和软模拟环境或者硬件仿真器模拟环境输出的运行日志,检查和定位错误。虽然硬件仿真加速器没有FPGA原型验证平台的仿真速度快,但是具有信号可观察性好,错误定位容易的优点,得到了广泛的研究和使用,例如张珩等针对FPGA原型验证可调试性差的问题,提出使用硬件仿真器的仿真加速作为中间层的解决方案[7],陈华宏等实现了一种由硬件加速器、工作站和终端组成的CPU通用测试平台,通过无损记录运行日志以及自动导出运行日志的方法进行错误检查和定位[8],陈讯等提出了一种通过在线仿真器和逻辑分析仪等设备抓取参考设计中的数据作为参考对象的验证方法[9]。

* **收稿日期**:2012-05-30
基金项目:国家"核高基"重大专项资助项目(2009ZX01028-002-002);国家自然科学基金资助项目(61103011,61170045)
作者简介:周宏伟(1980—),男,陕西宝鸡人,助理研究员,博士后,E-mail: hongw.zhou@gmail.com

传统的基于FPGA或者仿真加速器的处理器功能验证方法主要通过比较参考环境的运行日志和被测处理器实际执行的日志进行错误检查和定位,存在以下局限性:首先,必须构建处理器体系结构级模拟器参考模型或者拥有参考设计,抓取模拟器或者参考设计中的日志作为被测处理器验证环境的参考对象;其次,这些方法不能实时监测存储系统的数据访问,发现错误的时机较晚,效率低且定位错误时间长;最后,日志中通常保存的是寄存器的值,指令的地址等,这些信息只能反映出错的部分信息,对于错误根源的定位仍然不足。在多核处理器中,由于存储系统中的一个错误数据被处理器核读取后,处理器核可能会执行相当长的一段时间后才会报错,这造成发现测试程序执行错误的时间点远晚于产生错误的时间点,致使验证人员一方面无法及时定位错误,另一方面很难确定错误类型及出错数据的地址。为了最大化发挥仿真加速器在实际应用中的价值、缩短错误定位时间,针对硬件仿真器调试模式的特点进行二次开发非常必要。

本文提出一种面向多核处理器的能够快速定位存储系统数据错误的机制(Fast Fault Location Mechanism, FFLM),FFLM基于硬件仿真器构建的存储器黄金模型(Golden Memory,GM),通过使用多端口存储器构建GM,并使用特殊的端口映射机制尽量减少存储器的读写端口数,可以在保证GM访问带宽的前提下提高仿真速度。由于GM提供了足够的访问带宽,因此能够使访存事务监控器中读写事务FIFO的深度与CPU处理器核中实际的读写队列深度保持相同数量级,减少存储器资源的占用。在线访存数据检查器能够判断读返回的数据是否正确,更准确地确定错误位置。

1　存储系统数据错误快速定位机制

多核处理器通常拥有复杂的存储层次和Cache一致性协议,给功能验证带来了很大的挑战。Cadence公司最新的硬件仿真加速器Palladium XP(本文中简称PXP)具有丰富的调试模式,允许仿真过程中暂停时钟和重启时钟,支持动态探针和无限踪迹(InfiniTrace)功能,提供大容量的存储器资源,具有友好的软硬件协同仿真环境,为用户进行验证环境的二次开发提供了有力支持。为了提高错误定位速度,首先提出了两种可行的基于PXP的错误定位机制,在分析比较的基础上进一步优化,最终确定并实现了一种能够快速定位存储系统数据错误的FFLM机制。

1.1　两种基于PXP的错误定位机制

方案一利用PXP的调试模式能够支持无限踪迹的特性,通过访存踪迹加快错误定位。该方案的基本步骤如下:第1步,PXP仿真过程中记录访存踪迹(trace),每隔一定的时间间隔,PXP模拟暂停,记录检查点,将trace文件输出,进入后处理过程;第2步,对已产生的trace文件进行后处理,图1是后处理的流程图,如图所示,通过从trace中获取监测信号的值,产生读写事务序列,然后根据读写事务序列对离线的Golden Memory进行访问,读事务将比较来自Golden Memory的黄金数据(Golden Data)和来自trace的读返回数据:如果相等,则继续处理trace,直到处理完毕后退出后处理过程,跳到第1步继续执行仿真;如果不相等,则产生触发条件(trigger),退出后处理过程,进入第3步;第3步,恢复检查点并从该点重新仿真,执行过程中设置触发条件,执行程序直到出错点,通过观察波形等方式定位错误。

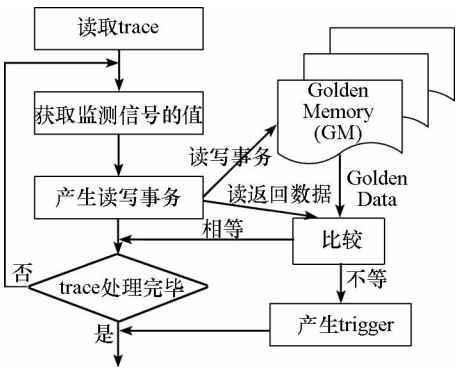

图1　利用trace的错误定位方法中后处理流程图

Fig. 1　The flow chart of post-process in fault location method using trace

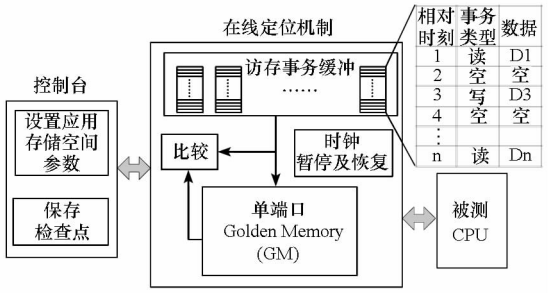

图2　利用大容量数据缓冲和在线GM的加快错误定位机制

Fig. 2　The mechanism of speeding up fault location with high-capacity data buffer and on-line GM

方案二利用大容量数据缓冲和在线黄金模型加快错误定位,其结构示意图如图2所示。控制台负责设置应用的存储空间参数和保存检查点。在线定位机制在PXP上通过专用硬件实现,访存

事务缓冲和 GM 使用 PXP 内的 1 读 1 写随机访问存储器(RAM)实现。在仿真过程中,访存事务缓冲监测被测 CPU,记录探测到的访存数据,由于多核处理器具有多个处理器核心,因此需要多个缓冲器同时记录。每个事务缓冲项包括:表示相对时刻的时钟指针、事务类型和数据。每周期将事务存入到时钟指针对应的项中,若没有事务,则该项为空。为了精确定位出错的时钟周期,每当事务缓冲器记录满时,时钟暂停及恢复模块暂停 CPU 的时钟,同时按时间顺序从事务缓冲中读取事务,启动访问 GM,写事务的数据写入 GM,读事务的数据和从 GM 读出的数据进行比较。出现数据比较错误时,通过时钟指针定位出错的时钟周期。如果未出现比较错误,则当访存事务缓冲中的事务全部读取完毕后,恢复 CPU 的时钟,继续执行。

以上方案与传统的软件模拟和 FPGA 仿真方法相比,其优缺点如表 1 所示。通过分析,基于硬件仿真器的方案一与方案二仍然不能较好平衡仿真速度和定位错误的难易度,因此,我们在这两个方案的基础上进一步提出了 FFLM 测试结构。

表 1　各种错误定位机制的优缺点比较

Tab. 1　The advantages and disadvantages of various fault location mechanism

错误定位机制	优缺点
软件模拟	优点:支持行为级设计,信号可观测性好,错误容易定位,支持检查点机制; 缺点:模拟速度慢,可模拟的设计规模受限
FPGA 仿真	优点:仿真速度快; 缺点:仅支持可综合的设计,信号可观测性差,错误不容易定位,不支持检查点机制,实现在线 GM 困难,可仿真的设计规模受 FPGA 芯片规模的限制
基于 PXP 的方案一	优点:仿真速度较快,支持混合仿真,支持检查点机制,不需要在 PXP 上使用专门的硬件机制,定位错误主要通过软件后处理实现,错误容易定位,GM 通过软件实现,容量不受限制; 缺点:导出 trace 文件的时间远远超过仿真时间,不适合仿真大规模程序
基于 PXP 的方案二	优点:仿真速度较快,支持检查点机制,通过在 PXP 上使用专门的硬件机制支持在线错误定位,错误容易定位; 缺点:需要专门的硬件机制,需要控制 CPU 的时钟暂停和恢复,GM 容量受 PXP 存储器资源限制,读写事务缓冲满后需要停止仿真,读写事务依次访问单端口 GM 影响仿真速度

1.2　基于 FFLM 的测试结构

基于 FFLM 的测试结构如图 3 所示,FFLM 包括多端口存储器黄金模型(Multi-Port Memory Golden Model,MP-MGM)、低开销访存操作监控器(Low-Cost Memory Operation Monitor,LC-MOM)和在线访存数据检查器(On-Line Memory Data Checker, OL-MDC)。被测系统是一个自主研制的具有 16 个处理器核心的微处理器 CMP-16,每个处理器核拥有私有的 Cache,一级 Cache(L1Cache)位于处理器核内部,二级 Cache(L2Cache)位于核外,L2Cache 通过片上网络互连。下面我们将分别阐述 FFLM 中各模块的具体功能和实现方法。

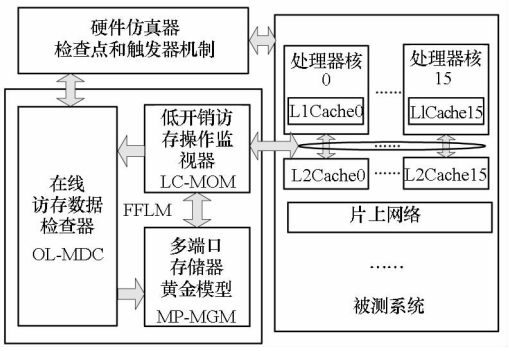

图 3　基于 FFLM 的测试结构示意图

Fig. 3　The schematic of test structure based on FFLM

1.2.1　低开销访存操作监控器

为了能够尽早发现处理器的访存错误,验证系统需要对处理器核的访存操作进行实时监控,捕获每一个处理器核的所有访存操作。CMP-16 中 L1Cache 的数据是 L2Cache 内数据的子集,L2Cache 与 L1Cache 之间的接口是理想的访存操作监控接口。在该接口中,L2Cache 收到来自处理器核的请求报文有:读请求报文、写请求报文和同时具有读、写语义的原子请求报文。L2Cache 返回给处理器核的响应报文有:读响应报文、写响应报文和原子响应报文。原子请求由于同时具有读语义和写语义,因此会收到两个响应报文,一个是数据返回(Return)类响应报文,另一个是应答(Ack)类响应报文。

CMP-16 中每个处理器核发送的访存请求具有以下特点:(1)接口数据位宽 64 位,最多连续发送 4 个地址连续的读请求报文,读响应返回的顺序与读请求发出的顺序一致;(2)支持以字节

为粒度的部分写,每个线程发出一个写请求后,若要发送对另一个不同地址的写请求,必须等待之前的写请求的响应返回,但是在部分写模式下,允许发送最多8个地址相同但是掩码不同的写请求,写响应返回的顺序与写请求发出的顺序一致;(3)发出原子请求后,必须等到收到原子应答后,才能够发送其他请求。

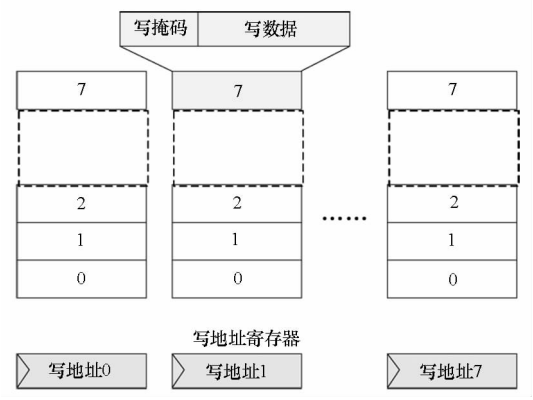

图4　LC-MOM中保存写请求信息的机制

Fig. 4　The mechanism of saving the write requests information in LC-MOM

当L2Cache返回给处理器核读或写响应报文时,需要对GM进行访问,由于无法从响应报文获得读写请求的访问地址、写数据以及其他一些所需信息,因此需要在LC-MOM中设计一套保存请求报文相关信息的机制。由于FFLM中采用多端口GM,GM具有足够的带宽,因此监测每个核的访存事务队列的深度只需与CPU处理器核中实际的读写队列深度保持相同即可,节省硬件资源。对于读请求,每个核仅需要设计一个4深度的FIFO保存最多4个读请求的信息,16个核共需64深度。对于写请求,采用如图4所示的机制保存请求信息,每个处理器核使用8个写地址寄存器保存8个不同的写请求地址,每个地址对应一个8深度的写请求FIFO队列,每个核共需64深度。写请求队列的每一项保存写请求的写掩码(指示被写字节的位置)和写数据。对于监测到的一个新的写请求,首先从8个写地址寄存器中查找是否有匹配的有效写地址,如果有,则将写请求信息写入该地址对应的FIFO队列,如果没有,则分配一个空闲的写地址寄存器保存当前的写地址,并将写请求信息写入该寄存器对应的FIFO队列。写响应报文中本身不带地址,LC-MOM从L2Cache的流水线中捕获写响应地址,通过将该地址与自己保存的8个写地址寄存器中的有效地址进行匹配,从匹配地址对应的队列中读出写掩码和写数据,对GM进行写操作。

1.2.2　多端口存储器黄金模型

MP-MGM负责存储程序执行过程中所有访存操作访问过的数据。MP-MGM中的GM容量取决于被测程序的访存空间,如果设置太大,将需要大量的存储器资源,设置太小又会限制测试程序的规模。如何在保证访问GM带宽的同时降低GM的访问端口数是提高仿真速度的关键。图5是能够支持16个处理器核的MP-MGM的体系结构示意图。

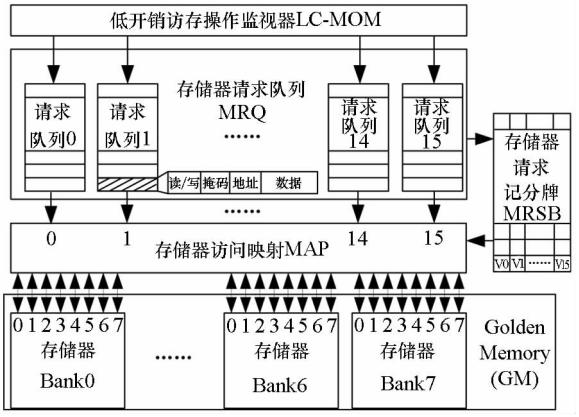

图5　MP-MGM体系结构示意图

Fig. 5　The schematic of MP-MGM architecture

如图5所示,MP-MGM中的GM容量为2GB,通过特殊设计的存储器请求队列MRQ(Memory Request Queue)、存储器访问映射MAP(Memory Access Mapping)和存储器请求记分牌MRSB(Memory Request Score Board)逻辑对访问端口数目进行优化,在保证GM访问带宽的前提下尽量提高模拟的速度。MRQ用于缓存来自LC-MOM的存储器访问请求,每个核对应一个独立的FIFO请求队列。请求队列中每一项包含了访问的类型、访问的字节掩码、访问地址和访问的数据(写请求时携带写数据,读请求时携带读响应数据)。MRSB记分牌是一个FIFO结构,每一项为一个16位的请求向量,向量中每一位对应一个请求队列,用于记录同一时刻哪些核向其对应的请求队列写入了新的操作。为了支持部分字节访问时以字节为粒度进行存储器访问,GM分为8个体,每个体的位宽为一个字节。为了在保证访问带宽的同时提高仿真速度,GM每个存储体设计为8个读写端口,存储器访问映射逻辑MAP完成16端口到8端口的访问映射。MAP从MRSB中读取请求向量,若请求向量中为1的位数小于等于8,则使用一个时钟节拍即可完成所有为1的向量位对应的请求队列中的访存请求对GM的访问。若请求向量中为1的位数大于8,则需要分两个时钟节拍进行GM访问。MAP逻辑能够根据访存请求中的字节掩码控制对

不同存储体的访问。对于需要初始值的测试程序,MP-MGM 支持直接从存储映像导入数据到 GM。

由于 16 端口到 8 端口的映射会花费额外的时钟周期,因此 CPU 存储系统和 FFLM 的仿真频率必须高于 CPU 中处理器核的仿真频率,当二者的频率比为 2 时 GM 访问带宽与 CPU 核实际访存带宽匹配,可以保证 MRQ 中的队列不会溢出。如果访存操作产生的速度是平均每 2 拍产生一个访存操作,则可以设置频率比为 1,即在相同时钟频率下工作。

1.2.3　*在线访存数据检查器*

根据 LC-MOM 监测获得的访存报文,访存数据在线检查器 OL-MDC 负责产生读写存储器请求并写入到 MP-MGM 中对应的 MRQ 中。OL-MDC 一共设计有 16 个,每一个处理器核对应一个。产生写存储器请求的条件是 LC-MOM 监测到写请求应答报文,写操作的地址、数据和写掩码能够根据图 4 所示的机制获得。产生读存储器请求的条件是 LC-MOM 监测到读请求返回报文,请求地址和读掩码通过 LC-MOM 中保存读请求信息的队列获得,读返回数据直接从读请求返回报文的数据域获得。OL-MDC 的一个重要功能是当读存储器请求从 GM 中读出黄金数据后,OL-MDC 控制发出一个读比较操作,将黄金数据和读存储器请求中携带的数据进行比较,如果两个数据相同,则说明从被测存储系统中返回的数据正确,如果不同,则说明被测存储系统中存在数据错,出错数据的地址为读存储器请求中携带的地址,出错数据的正确值为被比较的黄金数据。根据数据访问的局部性原理,在出错点附近分析仿真波形文件可以确定出错的具体位置,如果记录的波形文件不够,可以从最近的一个检查点重新启动仿真,设置 PXP 仿真暂停的触发条件为访存地址等于出错地址,通过分析从检查点之后的每一次对出错地址的访存操作,能够快速定位出错的位置和原因。

2　实验及结果分析

在验证 CMP-16 多核处理器的过程中,对各种定位机制的模拟/仿真速度进行了实验和分析。实验中软模拟环境使用 Cadence 公司的 NC-sim 模拟器,硬件仿真环境 使用 PalladiumXP 硬件仿真加速器。根据 PXP 能够记录的 trace 的容量,在方案一中每执行 10 万个时钟周期导出一次 trace,相应地,方案二中访存事务缓冲的各队列深度为 10 万,每项位宽为 109 位(事务类型 5 位、字节掩码 8 位、地址 32 位和数据 64 位),因此每个队列容量为 10.9Mb,16 个队列共计 174.4Mb。在 FFLM 中,LC-MOM 中每个处理器核对应的读请求 FIFO 深度为 4,位宽为 109 位,对应的 8 个写请求 FIFO 中每个深度为 8,位宽为 109 位,共计 7.4Kb,因此 16 个处理器核对应的请求队列容量为 118.6Kb。MP-MGM 的 MRQ 中每个请求队列深度为 64,位宽为 109 位,16 个队列总容量为 111.6Kb。综上,FFLM 中所有队列所需的存储位为 230.2Kb,队列所需存储器资源远小于方案二。表 2 列出了各模拟/仿真方法定位机制的硬件资源代价和仿真速度。由于 CMP-16 规模较大,仅能够通过 FPGA 平台实现 4 核系统仿真,且 FPGA 仿真实现在线 Golden Memory 困难,因此所列数据中不包含 FPGA 仿真的数据。

表 2　不同错误定位机制硬件资源和仿真速度的比较

Tab. 2　Comparison of different fault location mechanism in hardware resources and emulation speed

错误定位机制	定位机制所需硬件资源	仿真速度
软件模拟	无,但是需要软件协同定位机制	单核:上百 Hz,16 核:几十 Hz
基于 PXP 的方案一	无,但是需要软件协同定位机制	每次导出 trace 约 30min,平均速度为 55Hz
基于 PXP 的方案二	174.4Mb FIFO,2GByte GM	16 核全系统:47kHz
基于 PXP 的 FFLM	230.2Kb FIFO,2GByte GM	16 核全系统:480kHz

分析表 2 中的仿真速度可见:软件模拟的速度最慢;基于 PXP 的方案一由于每次导出 trace 的时间远远大于 PXP 实际仿真时间,导致 PXP 的仿真速度优势无法发挥,与软件模拟速度相当;基于 PXP 的方案二的仿真速度得到很大程度的提高,但是仿真速度受到时钟停顿和恢复的影响,仍然没有发挥出全部优势;基于 PXP 的 FFLM 在方案二的基础上进一步提高了约 10 倍的仿真速度。

在对 CMP-16 多核处理器的验证过程中,我们使用软模拟方法在模块级对 CMP-16 中各设计模块进行模拟验证,同时在芯片级对 CMP-16 进行小规模的定制激励的测试,使用硬件仿真器进行系统级的模拟验证,执行操作系统并在其上执行测试应用程序。我们在系统级验证环境中实现了本文提出的 FFLM,用于检查及加速定位存储系统的数据错误。图 6 为 5 月至 12 月共 8 个月

时间中芯片级和系统级测试定位错误的情况,芯片级的测试使用软模拟的方法,系统级测试使用PXP硬件仿真加速器。前6个月软模拟定位的错误较多,主要原因有两个:第一、软模拟执行的是测试人员根据设计文档有针对性编写的测试激励,比较容易暴露设计中的错误;第二、软模拟不加载操作系统,验证的规模小,因此出现错误后定位错误比较容易。后两个月的错误全部基于硬件仿真器的系统级测试定位,原因是硬件加速器仿真速度快,能够加载操作系统,执行真实的软件代码,比软模拟具有更高的测试覆盖率。本文提出的FFLM从10月份起投入到基于硬件仿真器的系统级测试中。3个月时间在系统级测试中共定位了7个设计错误,平均2.33个错误/月,而在未使用FFLM的前5个月,系统级测试总共定位了7个错误,平均1.4个错误/月。使用FFLM之后,系统级测试并定位错误的速度平均提高了66%。

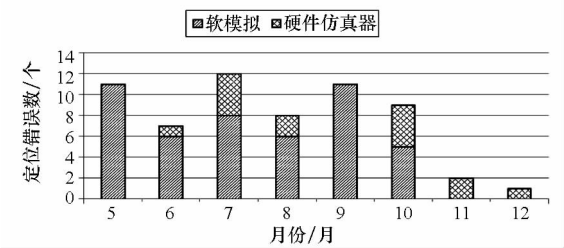

图6　CMP16芯片级和系统级测试中定位错误的情况

Fig. 6　The results of fault location in chip level and system level test for CMP-16 chip

FFLM能够提高定位错误速度的原因如下:在芯片级软模拟环境中仅仅运行规模较小的激励,仿真的时间短,因此执行过程中如果发现错误,即使没有记录所有信号的波形,从检查点开始以记录所有信号波形的方式重新执行一遍激励程序的时间也比较短,重新执行加上逻辑验证师分析波形定位错误的时间平均为1天。在未使用FFLM的硬件仿真测试环境中,由于启动操作系统并执行较大规模的应用,程序执行过程中报错的时间点往往和实际出错的时间点间隔很长的时钟周期,而且由于没有出错数据的地址等用于定位错误的关键信息,定位错误需要逻辑验证师和软件人员协同工作,首先软件人员通过查看程序汇编代码等方式分析程序的特性,判断出错的大概原因和位置,然后逻辑验证师根据软件人员的分析结果重新执行程序,通过分析波形定位错误,整个定位过程平均需要15天。在使用FFLM的硬件仿真环境中,由于FFLM能够提供出错数据的地址,通过检查点和条件触发机制,定位错误所需时间平均为2天,定位速度比未使用FFLM时平均提高了6.5倍。

3　结论

与传统方法相比,FFLM具有仿真速度快、硬件资源代价低以及定位错误时间短的优点。在使用FFLM的硬件仿真验证环境中,FFLM能够提供出错数据的地址,通过检查点和条件触发机制,定位错误的速度比未使用FFLM时平均提高了6.5倍。

参考文献(References)

[1] Schubert K D, Roesner W, et al. Functional verification of the IBM POWER7 microprocessor and POWER7 multiprocessor systems[J]. IBM Journal of Research and Development, 2011, 55(3):1-17.

[2] Krygowski C A, Almog E, et al. Key advances in the presilicon functional verification of the IBM zEnterprise microprocessor and storage hierarchy[J]. IBM Journal of Research and Development, 2011, 56(1):1-16.

[3] 屈婉霞,郭阳,庞征斌,等. 基于伪临界值的Cache一致性协议验证方法[J]. 国防科技大学学报,2008,30(6):47-52.
QU Wanxia, GUO Yang, PANG Zhengbin, et al. An efficient verification method of cache coherence protocol based on pseudo-cutoff[J]. Journal of National University of Defense Technology, 2008, 30(6):47-52. (in Chinese)

[4] 郭阳, 李暾, 李思昆. 微处理器功能验证方法研究[J]. 计算机工程与应用, 2003, 39(5):35-37.
GUO Yang, LI Tun, LI Sikun. Functional verification methodology for microprocessor[J]. Computer Engineering and Applications, 2003, 39(5):35-37. (in Chinese)

[5] 叶磊. "龙腾S2"验证平台的设计[D]. 西安: 西北工业大学, 2006.
YE Lei. Design of "Longtuim S2" verification platform[D]. Xi'an: Northwestern Polytechnical University, 2006. (in Chinese)

[6] 胡建国. 高性能微处理器的验证技术研究[D]. 长沙:国防科学技术大学, 2004.
HU Jianguo. Study of high-performance microprocessor verification method[D]. Changsha: National University of Defense Technology, 2004. (in Chinese)

[7] 张珩, 沈海华. 龙芯2号处理器的功能验证[J]. 计算机研究与发展, 2006, 43(1):974-979.
ZHANG Heng, SHEN Haihua. Function verification of godson-2 processor[J]. Journal of Computer Research and Development, 2006, 43(1):974-979. (in Chinese)

[8] 陈华宏. 64位高性能微处理器系统功能验证方法的研究与实现[D]. 长沙:国防科学技术大学, 2005.
CHEN Huahong. Research and implementation of the system function verification methods on 64 bit high-performance microprocessor[D]. Changsha: National University of Defense Technology, 2005. (in Chinese)

[9] 陈迅, 梁斌, 陈跃跃, 等. 全定制微处理器的FPGA原型验证方法[C]//2005年全国计算机工程工艺技术年会论文集. 济南:山东大学计算机学院,2005:379-381.
CHEN Xun, LIANG Bin, CHEN Yueyue, et al. Study of full custom high performance microprocessor function verification using FPGA[C]//CCF TCCET. NCCET'05. Jinan: Computer school of Shandong University, 2005: 379-381. (in Chinese)