

引用格式:牛云鹏,张多利,唐旭,等.一种可重构多核系统的仿真加速方案[J].微电子学与计算机,2021,38(2):19-24. [NIU Y P, ZHANG D L, TANG X, et al. An acceleration scheme for reconfigurable multi-core system[J]. Microelectronics & Computer, 2021, 38(2): 19-24.]

一种可重构多核系统的仿真加速方案

牛云鹏¹, 张多利^{1,2}, 唐旭¹, 宋宇鲲^{1,2}

(1 合肥工业大学 微电子设计研究所, 安徽 合肥 230601;

2 教育部 IC 设计网上合作研究中心, 安徽 合肥 230601)

摘 要: 针对可重构多核系统多任务仿真耗时长、并发性差的问题, 提供了一种分层的、以资源请求异步响应为特点的松耦合多精度多线程系统建模方案. 采用此方案的系统模型在任务仿真时, 不仅能够提供寄存器级的实时数据反馈, 同时也具有优异的任务并发性能. 实验显示, 相对于传统上位机软件数据收集和计算, 该模拟器周期预测相对误差小于 3%, 模拟计算相对误差小于 $2e-5$, 仿真耗时最大缩短 73.52%; 在启用任务异步响应优化后, 任务并行度平均提升 28.21%. 该设计在硬件目标系统的快速性能评估方面具有良好的实用价值.

关键词: 多核系统; 多线程建模; 验证加速; 性能评估

中图分类号: TP337

文献标识码: A

文章编号: 1000—7180(2021)02—0019—06

An acceleration scheme for reconfigurable multi-core system

NIU Yun-peng¹, ZHANG Duo-li^{1,2}, TANG Xu¹, SONG Yu-kun^{1,2}

(1 Hefei University of Technology Institute of VLSI Design, Hefei 230601, Anhui, China;

2 IC Design Web-cooperation Research Center of MOE, Hefei 230601, Anhui, China)

Abstract: This paper provides a layered, loosely coupled, multi-precision, multi-threaded system modeling for solving the problem of time-consuming of multi-task simulation and poor concurrency in reconfigurable multi-core system. The system model using this scheme can not only provide register-level real-time data feedback during task simulation, but also have excellent task concurrency performance. Experiments show that, compared with the data collection and calculation of traditional host computer software, the relative error of the simulator's period prediction is less than 3%, the relative error of simulation calculation is less than $2e-5$, and the simulation time is shortened by a maximum of 73.52%; after enabling the task asynchronous response optimization, Task parallelism increased by an average of 28.21%. The design has good practical value in the rapid performance evaluation of the hardware target system.

Key words: multi-core system; multi-thread modeling; simulation acceleration; performance evaluation

1 引言

基于片上网络通信的可重构多核系统不仅实现了高带宽核间通信, 其自身也可根据运算需求灵活实现网络资源节点重构. 当系统规模增大时, 基于硬件层任务加载的测试复杂性大幅上升, 这将严重制

约系统软硬件开发效率^[1]. 为了确保任务测试集的完备性, 同时提高系统开发效率, 大规模硬件系统设计时, 常采用混合精度建模与硬件原型协同验证的方案^[2].

目前针对可重构多核系统的验证和架构探索, 主要包括基于数学推导的形式化验证和系统

高层建模两类方法. 由于当前形式化验证的理论基础和工具链尚不完善, 以高层次建模为基础的软硬件协同仿真是主要验证方案^{[3][4]}. 高层建模基础组件常利用 SESC、QEMU、GEM5 等开源模拟器进行辅助构造^[5]. 基于 X86 和 ARM 等通用架构的计算系统, 仅需对上述模拟器组件进行少量二次开发就可构建指令精确的硬件全系统模型, 但当前通用处理器模拟器均缺乏对片上网络方案的完备支持.

适用于片上网络研究的工具主要是以 Pacman 为代表的周期精确模拟器^[6], 它支持虫孔路由、全链接的交叉开关、以及 mesh 和星形等结构, 其通信层建模采用工作状态机, 实现了处理器和路由单元模型一致性, 简化了模型验证步骤. 片上网络模拟器对网络拓扑结构、互联方式、通信协议等方面的研究工作具有积极的促进作用, 但因处理器节点不支持运算单元建模, 所以此类模拟器仅局限于多核系统通信层行为研究, 无法通过自定义组件和添加扩展实现系统应用层计算类任务的加载和模拟.

本文根据可重构多核目标系统硬件架构和任务执行特点, 以提高系统验证效率为设计目的, 使用 C++ 建模构建了一套组件松耦合且支持多线程任务执行的系统仿真加速平台.

2 目标硬件结构

如图 1 所示的 HMCS 是基于 2D-Mesh 的双端口 NoC 可重构多核系统. 主要含异构运算簇 (PUs)、主控制器 (MC)、NoC 通信组件、多通道存储器 (Memory) 和通用交互接口 (IO) 等部件.

HMCS 规模可根据任务资源需求弹性配置. 网络底层组件以数据、配置和节点信息构成的三层协议栈作为通信基础以保证通信效率. 主控制器、存储组件和异构运算簇等资源节点挂载于 NoC 双端口路由节点, 并通过三层网络协议栈进行任务执行中的信息转发和接收.

以太网接口等通用接口主要负责 HMCS 内部存储器件与外部设施的数据交互. 主控制器作为系统的资源调度和配置中心, 其收到下游局部状态管理器 C_n 汇总的状态请求信息之后, 根据任务优先级从内部配置缓存获取下游节点配置, 并按照协议栈封后通过网络下发到目的节点, 实现任务执行中的配置调度和任务分发. 异构运算簇是包括大维度矩阵运算和 FFT 等在内的专用 IP 集合, 其网络挂载

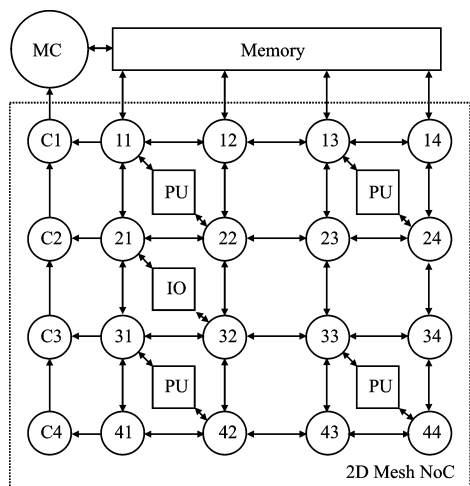


图 1 HMCS 硬件架构示意图

坐标以及组合方案可由系统初始化配置文件进行重构. 存储单元采用多通道存储设计以支持运算簇间数据交换.

HMCS 基于 NoC 的多核通信机制, 能够灵活实现规模重配置和运算簇重构, 为适应不同类型的密集型计算提供了一种通用解决方案. 系统底层采用的三层网络协议栈设计, 相比于传统多核总线通信, 能够有效提高多核通信效率和运算簇资源利用率.

3 模拟器设计

3.1 整体结构

系统模拟器按照模拟粒度划分主要分为周期精确和指令精确级^[7]. 本文模拟器系统结构如图 2 所示, 模拟器由任务预处理层、全局管理层、HMCS 混合精度建模和系统分析层四个功能层面组成. 模拟器系统中 HMCS 硬件建模部分采用混合精度设计, 运算簇和存储类部件采用指令级精确建模, 网络组件在指令精确级建模的基础上内置时钟探针时序统计模块. 采用混合精度建模在兼顾仿真速度的同时, 也能保证局部时序信息的精确反馈.

任务预处理层负责从用户输入中获取系统初始化规则以及用户任务, 并将其进行本地化指令映射, 最终将输入分解为基于管理层寄存器操作的子任务指令集序列, 其输出片段示例如图 3 所示.

全局管理层的硬件原型是主控制器单元, 内部具有任务调度、线程映射和网络资源控制寄存器组等组件, 对任务队列指令执行取指和译码后通过 NoC 状态管理器对网络资源节点和内部线程池进

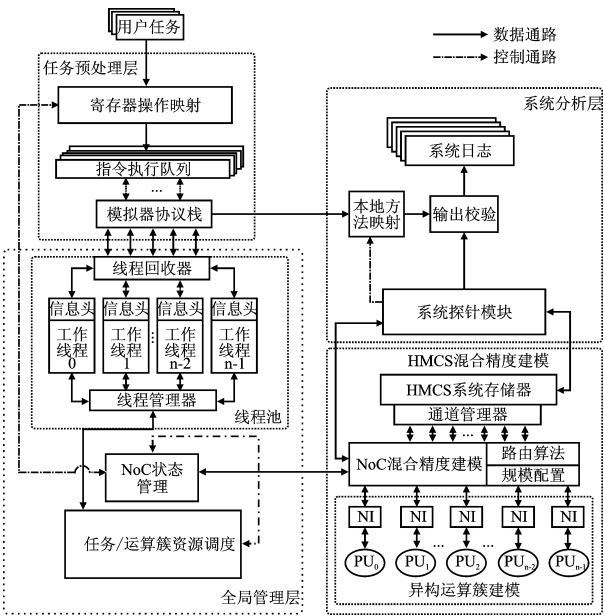


图 2 模拟器结构设计

行映射和调度。

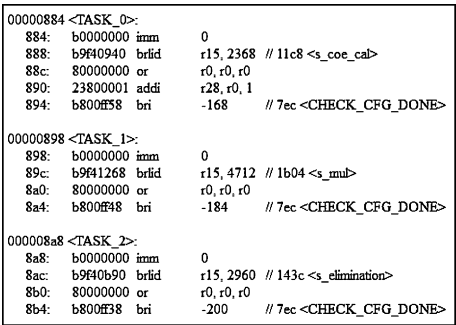


图 3 任务分解片段

系统分析层在任务执行中负责监控和收集 NoC 系统的网络状态和下游运算簇指令执行状态，同时在 HMCS 的子任务执行完毕后进行运算簇周期信息反馈和数据输出，并生成系统日志。HMCS 混合精度建模是对 NoC 可重构硬件系统的松耦合建模，网络布局采用以遗传算法为基础的配置策略，算法配置模块通过模拟器协议栈封装，可以按需替换。资源类组件包括存储行为精确的多通道存储器，其不仅为全局管理层中寄存器组提供存储映射，也为任务执行中多线程数据同步提供媒介。指令精确建模的异构运算簇模型 PUs 通过网络接口 NI 挂载于网络指定坐标，并能够根据用户任务输入的初始化规则进行坐标替换和运算簇资源重构。本文设计对硬件系统任务执行模拟中，能同时实现网络节点时序检测和输出校验，与使用通用模拟器对比更符合目标系统应用需求。

表 1 模拟器功能对比

名称	编程语言	多核模拟	NoC 模拟	异构模拟
SESC	C++	支持	不支持	支持
QEMU	C	支持	不支持	支持
GEM5	C++, Python	支持	不支持	支持
Pacman	SystemC	支持	支持	不支持
本设计	C++	支持	支持	支持

3.2 功能设计

3.2.1 任务输入优化

模拟器系统的任务预处理层根据多线程任务执行特点，对输入任务进行子任务拆分和通用算法的本地化映射。如图 4 流程图所示，用户根据硬件编程规范使用编写任务输入文件，通过初始化加载至系统模拟器。模拟器首先输入进行空值处理、边界检查及源数据正则校验，之后对程序中形成执行链的耦合变量进行静态替换。同时热点统计基于代码中多次执行的函数和代码段映射为模拟器平台相关的本地机器码，以提高任务中重复段的执行效率，最后利用数据不相关性条件后执行寄存器重命名手段优化。本设计在执行热点代码本地化映射后，在当前子任务未执行完毕前保留寄存器重命名之前的中间文件副本，在执行中侦测到热点优化异常时可以回退状态并重新生成任务指令序列。

热点任务段包括被频繁调用的表达式以及循环代码段。本文利用的检测方法是基于计数实现，利用预处理时为每个操作代码段内部标记并维护一个回调计数器 C_1 和循环计数器 C_2 ，其分别具有设定阈值 T 。 C_1 的阈值与其自身的执行时间耦合，其表达式如下所示：

$$C_1' = \begin{cases} C_1' \times 0.75 & t - t' \geq \Delta t, \quad C_1' < T \\ C_1' & 0 \leq t - t' < \Delta t, \quad C_1' < T \\ C_1' + 1 & 0 \leq t - t', \quad C_1' \geq T \end{cases} \quad (1)$$

公式(1)中 t 时刻为当前 C_1 统计时刻， t' 为先前 C_1 递增时刻。非循环段超过阈值后 C_1 执行热度统计，若在时限 Δt 内未触发热点阈值，其 C_1 值会缩减 25%，这是为了避免时间跨度较大的同一子任务函数段被多次调用时发生热点误判，其余情况 C_1 保持不变。 C_2 与时间无关的循环体计数，当达到阈值时 C_2 计数。 C_2 和 C_1 之和共同决定热点任务段标记，当代码段两计数器和超过阈值 T ，则子任务程序段被标记为热点代码，并对这段代码实行本地化函数映射后替换原任务指令段。本地映射执行完毕后 C_2 和 C_1 值分别减半，防止重复统计导致的过度优化降

低预处理速度.

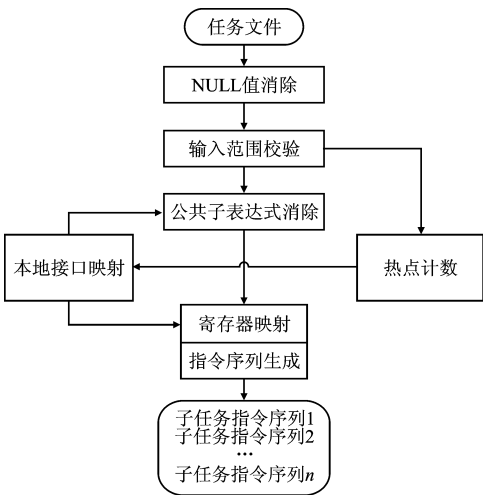


图 4 指令优化处理流程

3.2.2 时钟探针设计

仿真粒度较小的周期精确模拟器设计,能详细反馈目标系统在任务执行中寄存器组的时序信息,但也正由于需要对系统时序信息进行精确统计的原因,采用全组件周期精确建模的系统,其仿真速度相对于面向行为和功能模拟的指令精确建模劣势明显.

为了平衡仿真粒度和速度的问题,本设计仅在模拟器系统的 HMCS 硬件模型中的 NoC 网络组件中保留了局部周期精确建模,以满足网络的时序信息统计需求,其核心是负责时序信息生成的时钟探针模块.

时钟探针模块主要变量跳变如表 2 所示,其中 $n (n \geq 1)$ 是电路仿真周期数:

表 2 时钟探针变量跳变

状态量	前态	现态	次态
ptr	$(n-1)\%3$	$n\%3$	$(n+1)\%3$
t_ptr	$(n-1)\%3+1$	$n\%3+1$	$(n+1)\%3+1$
stb[0]	stb[2]	stb[0]	stb[1]
stb[1]	stb[0]	stb[1]	stb[2]
stb[2]	stb[1]	stb[2]	stb[0]

时钟探针通过在指令精确模型中引入时钟变量,模拟硬件原型每周期的寄存器组刷新行为. 硬件电路的周期精确建模需要对其寄存器值设置更新前和更新后状态. 为了体现这个特性需要内部维护三个私有变量记录电路前态、现态和次态. 模拟器中时钟变量代表的有效沿到来时, stb 状态寄存器的值会在三个状态间依次翻转,并且时钟变量的也会同

步刷新,时钟变量在 $[0, 2]$ 间翻转,状态寄存器组 stb 跟随时钟变量同步循环更新,从而从抽象的数据流角度对物理寄存器随有效时钟沿到来而发生跳变的时序行为进行模型映射. 通过时钟探针模块可以在模拟器系统运行时获取硬件目标系统时序信息预估.

3.2.3 任务资源请求异步响应

系统在执行大规模矩阵计算和大点数 FFT 等任务模拟时,经过分解的子任务内部和不同子任务间其具有指令和数据操作的强耦合. 未进行配置请求和数据请求异步响应的任务调度系统,当下游运算核上执行的任务片未完成时,线程映射器为防止工作线程间的数据同步问题,会阻塞指令发射器的指令分发,从而使下游运算核心无法及时获取配置信息而处于挂起状态,引起子任务执行中的不完全并发. 当任务单次并发线程数超出 NoC 中挂载物理运算核数,一方面线程映射器阻塞下游空载运算核资源请求的几率会增加;另一方面线程映射器的先来先服务策略会立即向获取到任务配置的下游节点分配工作线程,线程映射器无法借由运算核上子任务分时执行的特点创建出多于物理核心的工作线程.

为解决以上问题,全局管理层采用了任务配置和数据资源请求的异步响应设计. 如图 5 所示,完成资源请求异步响应的主要模块是任务请求注册器、线程映射器和任务应答器. 下游资源节点与上游调度组件的交互经由 NoC 状态管理器代理. 任务请求注册器其本质是一个独立的管理层线程,负责循环检测由 NoC 状态管理器转发而来的下游节点请求;任务应答器处理下游的配置类请求,线程映射器负责系统运算资源的实际调度.

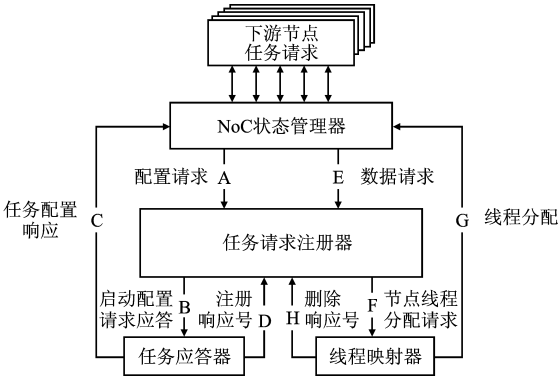


图 5 异步响应示意图

下游节点通过 NoC 状态管理器主动向任务请

求注册器发起任务配置请求,任务注册器检测到请求,会发送通知至任务应答器.任务应答器负责同步当前线程池工作线程,并优先向 NoC 状态管理器发送节点配置请求许可,从而避免因下游节点等待线程分配而造成 IO 阻塞,影响其他运算核任务请求.任务应答器结束配置响应后,请求节点的应答序列号与线程池事件,将传递到任务请求注册器的循环队列,以等待线程分配,同时任务应答器准备响应其他运算核配置请求.

取得配置响应后,节点通过 NoC 状态管理器向任务请求注册器获取工作线程分配,之后任务请求注册器调用线程映射器,根据节点的应答序列号通过线程池中的线程管理器创建子任务工作线程,以配置指令的形式发送至与请求节点同构的空载节点,实现线程映射.当分配线程的运算核的时分任务片执行完毕,其在任务请求注册器中的应答序列号和请求信息将会被删除,线程池同步失效的应答号,通过线程回收器把映射到运算核的工作线程销毁实现线程回收,响应运算节点恢复空载状态准备再次发出任务执行请求.

异步响应策略将下游任务配置请求和线程映射时的指令分发相分离.在具有指令和数据的强耦合操作的场景下,降低了线程映射器工作负载和节点配置请求阻塞几率,下游空载节点能更快获得线程分配,使得多个子任务可在簇内同构节点间分时执行,实现了子任务间和子任务内部并行,提高了系统多核利用率,为用户层多任务并行模拟提供了底层支持.

4 实验测试

本文对模拟器系统的运行测试分精度、速度和并行度三部分.运行模拟器系统的 PC 硬件配置为 CPU Intel i7-4790 3.3 GHz、内存 16GB DDR3 1 600 MHz;对模拟器输出数据进行精度计算时,以 MATLAB 2013a 的输出作为标准数据;现有硬件目标原型设计于 Xilinx Virtex-7 XC7V2000T FPGA 芯片实现;原硬件建模的时序精确模型部分由 SystemC 的 API 实现,环境为 SystemC IDE G-2012.模拟器底层 HMCS 混合精度模型,其网络初始化为 8×8 双端口 NoC,其上随机布局 16 个异构 PU,模拟器加载的用户任务为不同数据规模的复数协方差、矩阵乘以及 FFT 三种类型.

模拟器系统指令精确仿真测试目的是验证其与硬件系统执行相同机器指令时的输出一致度.由于

不进行电路时序信息记录,这种方式能够获得最高的运行速度,表 3 是指令精确仿真模式下,本设计与 SystemC 系统建模的任务模拟速度和误差统计,二者误差统计时均以 MATLAB 执行相同算法的输出作为标准数据.

表 3 指令执行测试

任务	SystemC 模型耗时 (s)	本设计 耗时 (s)	SystemC 相对误差 (%)	本设计 相对误差 (%)
协方差 32×8K	78.31	28.80	1.41e-3	1.47e-3
协方差 64×8K	332.56	104.91	1.54e-3	1.60e-3
2K 组 256 点 FFT	3.68	0.99	1.18e-2	1.33e-2
2K 组 2K 点 FFT	11.63	3.08	1.23e-2	1.38e-2
8K 组 2K 点 FFT	27.92	8.35	1.25e-2	1.41e-2

由表 3 可知本设计相对于 SystemC 上位机模型,用户任务经输入预处理层的本地接口映射和子任务拆分后执行,整体仿真耗时平均缩短 69.68%,最高缩短 73.52%.执行协方差任务时相对误差不高于 1.60e-5,FFT 任务时相对误差不高于 1.41e-4.

HMCS 的混合精度硬件建模层的时钟探针设计替换了原采用的 SystemC 的 API,精简设计提高仿真效率,用于对硬件系统任务执行时的性能预估统计.

实验可知本文模拟器系统在保证功能上与硬件系统保持一致的前提下,可对硬件系统任务执行状态和时序信息进行精确预估,且周期的预估相对误差不高于 3%,执行矩阵乘任务模拟时速度最高每秒可达 30 951.96 cycle,FFT 任务模拟时速度最高每秒可达 3 424.44 cycle.相对于集成 SystemC 的时序模拟接口时每秒约 1k cycle 的周期精确级任务仿真速度,性能提升明显.

根据简化后的 Amdahl 定律,系统的加速比计算如下:

$$A = \frac{T_1}{T_p} \tag{2}$$

本实验中其中 T_1 对应单核串行执行任务周期, T_p 为多核并行执行任务周期, A 的理论值上限等于执行并行处理的核心数.

并行度测试实验利用模拟器执行不同规模的复数矩阵乘任务,图 6 为模拟器系统采用资源请求异步响应策略前后的任务并行度变化.

观察图 6 可知,在执行 16×16 及以下较小规模的复数矩阵乘任务,全局管理层采用资源异步响应设计后并行度要略低于原系统,这是因为在小规模任务执行时,子任务完全并行时最大工作线程不高

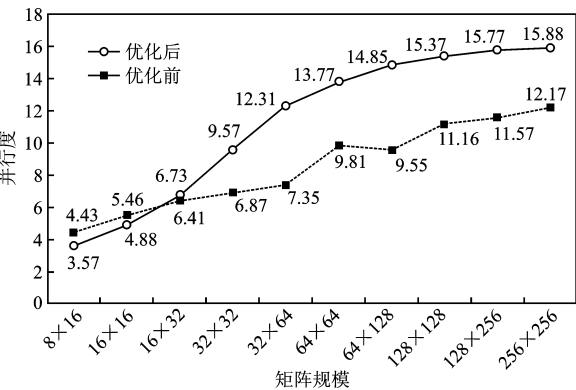


图 6 优化前后并行度测试

于下游运算核数,因此线程池有足够资源进行工作线程创建和运算核映射,下游资源请求能够被立即应答而不会发生应答阻塞和线程池资源耗尽.而异步响应优化后的系统,由于任务请求注册器和任务应答器的存在,需额外执行应答和线程信号量同步过程,因此执行小规模任务时其并行度要略低于优化前系统.

当任务规模扩大时,未采用异步响应策略设计系统由于资源请求阻塞,导致无法将线程映射至下游空闲运算核,同时导致线程池资源耗尽使得任务并行度发生波动且整体增长缓慢.采用异步响应策略设计的系统由于不受线程池工作线程与运算核一对一映射的影响,其并行度随运算规模增大逐渐趋向于并行度极限.系统优化后相对于优化前,任务并行度平均提升 28.21%.

5 结束语

本文利用构建了一套专用于可重构多核系统的多线程仿真加速平台.文章首先对硬件目标系统结构进行了说明,下一步详细介绍了系统仿真平台的核心设计,最后在仿真平台上运行不同类别和计算规模的用户应用,并与硬件系统任务执行结果进行比对分析.实验结果表明,本文多线程仿真加速平台,在精确度、资源利用率、任务并行度、时间效率等方面均满足硬件系统仿真需求,能够有效提升目标系统的验证效率.

参考文献:

[1] EGGER B, SONG E, LEE H, et al. Random test program generation for verification and validation of the Samsung Reconfigurable Processor[J]. Journal of

Systems Architecture, 2019, 97: 219-238. DOI: 10.1016/j.sysarc.2019.05.007.

[2] NOLTING S, PAYÁ-VAYÁ G, GIESEMANN F, et al. Dynamic self-reconfiguration of a MIPS-based soft-core processor architecture[J]. Journal of Parallel and Distributed Computing, 2019, 133: 391-406. DOI: 10.1016/j.jpdc.2017.09.013.

[3] KIRCHSTEIGER C M, SCHWEITZER H, TRUMMER C, et al. A software performance simulation methodology for rapid system architecture exploration [C]//Proceedings of the 15th IEEE International Conference on Electronics, Circuits and Systems. St. Julien's, Malta: IEEE, 2008: 494-497. DOI: 10.1109/ICECS.2008.4674898.

[4] WEI Y. Construct multi-core DSP software simulation platform based on SystemC[J]. Microelectronics & Computer, 2016, 33(4): 86-90. DOI: 10.19304/j.cnki.issn1000-7180.2016.04.019.

[5] CHEN H, LUO W L, GAN Y Y. Multi_MINT: A multiprocessor simulator based on MINT [J]. Computer Engineering & Science, 2011, 33(12): 37-43. DOI: 10.3969/j.issn.1007-130X.2011.12.007.

[6] HE W Q, YANG L, LU Q. Cycle-accurate DSP processor modeling with SystemC[J]. Microelectronics & Computer, 2013, 30(4): 107-110. DOI: 10.19304/j.cnki.issn1000-7180.2013.04.026.

[7] YU J, LI X, ZHOU X H, et al. Study on retargetable cycle-accurate simulator generation environment [J]. Journal of Chinese Computer Systems, 2007, 28(1): 166-171. DOI: 10.3969/j.issn.1000-1220.2007.01.036.

作者简介:

牛云鹏 男,(1995-),硕士研究生.研究方向为集成电路设计与测试.

张多利(通讯作者) 男,(1976-),博士,研究员,博士生导师.研究方向为多核处理器体系结构与设计方法.

E-mail: zhangduoli@hfut.edu.cn

唐旭 男,(1996-),硕士研究生.研究方向为多核处理器架构.

宋宇鲲 男,(1975-),博士,副研究员,硕士生导师.研究方向为面向数据高密和计算密集应用的 SoC/MPSoC 体系结构与实现.