

引用格式:余振波,聂言硕,宋宇鲲,等.一种自适应可重构通用浮点加速器的设计[J].微电子学与计算机,2021,38(1):89-94.
[YU Z B, NIE Y S, SONG Y K, et al. Design of an adaptive reconfigurable general-purpose floating-point accelerator[J]. Microelectronics & Computer, 2021, 38(1): 89-94.]

一种自适应可重构通用浮点加速器的设计

余振波¹, 聂言硕¹, 宋宇鲲¹, 侯 宁²

(1 合肥工业大学 电子科学与应用物理学院, 安徽 合肥 230009;

2 河南城建学院 电气与控制工程学院, 河南 平顶山 467041)

摘 要: 用加速器因其专业性过强往往缺乏一定的灵活性, 在处理不同类型的应用时不可避免的导致能效比的下降. 本文设计了一款自适应可重构浮点加速器, 它可以根据计算任务需求和可重构计算资源使用情况, 在运行时将计算任务映射到可重构计算资源, 具有自适应可重构的能力. 该浮点加速器整体采用“RISC-V+可重构浮点运算单元”的架构, 可重构浮点运算单元由一系列粗粒度浮点运算器构成, 负责具体的浮点计算. 该设计在 Xilinx Ultra-scale XCVU440FPGA 芯片上进行了原型验证, 结果表明, 该浮点加速器具有较广的应用面, 运算效率高, 算法适应性强.

关键词: 可重构; 浮点加速器; RISC-V; 高密度计算; FPGA

中图分类号: TN47

文献标识码: A

文章编号: 1000-7180(2021)01-0089-04

Design of an adaptive reconfigurable general-purpose floating-point accelerator

YU Zhen-bo¹, NIE Yan-shuo¹, SONG Yu-kun¹, HOU Ning²

(1 School of Electronic Science and Applied Physics, Hefei University of Technology, Hefei 230009, China;

2 School of Electrical and Control Engineering, Henan University of Urban Construction, Pingdingshan 467041, China)

Abstract: Application-specific accelerators often lack flexibility due to their professionalism, which inevitably leads to the decrease of energy efficiency during the execution of a wide range of applications. In this paper, an adaptive reconfigurable floating-point accelerator is designed, which can automatically map computing tasks to reconfigurable computing resources at runtime basing on computing task requirements and the status of reconfigurable computing resources. The floating-point accelerator uses the architecture of "RISC-V + Reconfigurable Floating-point Arithmetic Unit". The reconfigurable floating-point arithmetic unit consists of a series of coarse-grained floating-point arithmetic units, which are responsible for specific floating-point calculations. The design has been prototypical verified on the Xilinx Ultrascale XCVU440 FPGA chip, and the results show that the floating-point accelerator has wider applicability, high operation efficiency, and high algorithm adaptability.

Key words: reconfiguration; floating-point accelerator; RISC-V; high-density computing; FPGA

1 引言

在语音信号通信、数字图像处理、科学计算等研

究领域,对数据处理的精度和实时性的要求越来越高,配套的硬件系统大都需要具备很强的浮点运算能力,为了获得可观的能效比,业界开始倾向于通过

专用加速器来获得性能提升。但是当硬件系统需要运行不同类型的应用时,专用硬件加速器因为其专业性过强而不可避免的导致片上资源的使用率降低以及能效比的降低^[1]。

在这种情况下,可重构系统成为一种非常具有吸引力的解决方案。可重构系统在执行计算任务时通过配置流来动态改变运算单元阵列的功能(功能的改变往往消耗几个或几十个时钟周期),然后通过数据流来驱动运算单元阵列进行计算。这种特殊的计算方式让可重构系统具有专用加速器所没有的灵活性和接近专用加速器的能效比。

针对以上讨论的内容,本文提出一种自适应可重构通用浮点加速器(Reconfigurable General-purpose Float-point Accelerator, RGFA),整体采用“RISC-V+可重构浮点运算单元”的架构,RISC-V使用自定义编程接口下发运算指令,硬件动态解析运算指令以生成配置信息的同时,利用浮点运算流水线计算时间隐藏动态生成配置信息的延时;可重构浮点运算单元由一系列粗粒度基础浮点运算器构成,加速器能够结合计算任务需求和加速器内部可重构计算资源的使用情况,在运行时将计算任务映射到可重构计算资源,具有自适应可重构的能力。

2 相关工作

可重构系统能够根据特定任务需求对系统内部的运算电路结构进行调整以适应不同应用场景,从而获得更高的能效。它通常包含可重构逻辑,配置信息存储器和一个通用处理器。应用程序中占用大部分执行时间的主要代码块会被优化,放在可重构运算电路中执行,其余的指令则在通用处理器上执行。

目前,可重构系统研究主要应用于以整型计算为主的嵌入式应用领域^[2],使用整数运算单元组成同构可重构阵列。经典的可重构系统如 Morpho-sys^[3],Adres^[4],等采用静态调度的方法生成通用处理器的控制码和可重构硬件的配置信息。这一类方法使用编译器和专门的优化工具,对程序进行分析,提取出可以优化的代码块,依据可重构系统底层电路结构生成静态的配置信息。在高精度数据密集型计算领域,浮点运算单元种类较多且计算周期各自不一,由其组成的异构可重构阵列在静态调度上则更加困难。还有一类可重构系统^[5-6]不用软件编译器而用片上的硬件资源去产生配置信息(片上硬件实现编译的功能),再去驱动可重构阵列进行计算。这一类方法实时性较好,能够在通用处理器执行程序

期间并行完成指令相关性分析并生成配置信息。但是,由于硬件资源有限的天然限制,这类可重构系统无法连续对多条指令进行解析,更无法进行循环级(loop-level)的并行处理,从而带来可重构阵列资源使用率低下问题。

3 可重构通用浮点加速器体系结构

本文提出自适应可重构浮点加速器结构目的在于为计算任务动态分配计算资源,最大程度的改善加速器对各类加速场景的适应能力。

3.1 RGFA 整体结构

如图 1 所示, RGFA 由微控制器(RISC-V)、重构控制单元(Reconfiguration Controller)、数据存储单元(Data memory)、访存单元(Load/Store)、可重构浮点运算单元(RFU)五个主要模块构成,可以分为两个层次:(1)以 RISC-V 为核心的指令层;(2)以 RFU 为核心的数据层。重构控制单元是指令层和数据层之间的接口,负责解析指令下发的运算指令,生成数据层可以使用的配置信息。

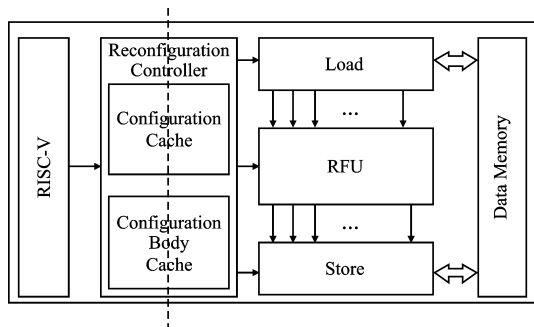


图 1 浮点加速器架构

在指令层, RGFA 的功耗主要来源于指令在 RISC-V 核中的执行过程。RISC-V 核实现了 RISC-V 指令集架构中的 RV32I 指令集, RV32I 作为一组足够编译器、汇编器和链接器使用的最小完备指令集,使得 RGFA 在使用最少的硬件资源实现处理器核的同时,具备完整的可编程能力。在使用 RGFA 对程序中的核心计算部分,比如循环,进行加速时,编程人员首先手动或者通过自动化 CAD 工具将循环展开,然后将展开的代码按照 RGFA 的运算指令格式编入 RGFA 程序, RGFA 在运行时根据这些运算指令和运行时状态自行组织数据路径,将计算任务映射到可重构计算资源,实现重构。

在数据层, RGFA 的功耗主要来源于数据在 RFU 中数据的计算过程。RFU 的内部结构如图 2 所示。RFU 是由若干个处理单元(Processing Ele-

ment, PE) 构成的一维可重构阵列。一个 PE 是一个计算资源,不同的 PE 具有不同的计算延迟周期和不同的浮点运算功能,比如有浮点加法、浮点乘法、浮点除法和浮点开方等功能,因此一维可重构阵列是异构的。异构运算单元在静态调度上是困难的, RGFA 采用一套 valid/ready 握手机制来同步不同 PE 的计算过程,实现动态调度。

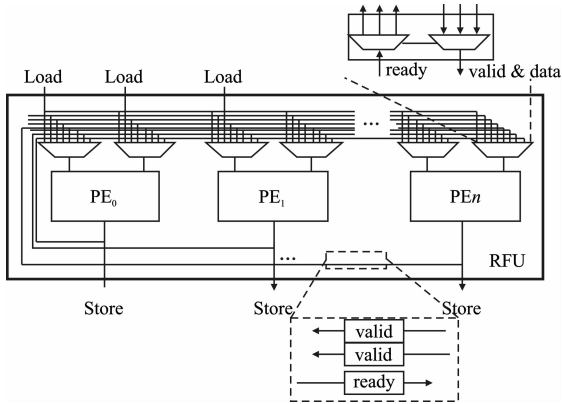


图 2 RFU 结构

图 2 同时给出了多路选择器的内部结构。多路选择器在选择操作数的同时向上级反馈握手信号。当上游数据(总线上数据来自于存储器或者其它的 PE 计算结果)已经放在总线上而下游 PE 没有准备好接收时,通过握手信号的反馈,上游 PE 暂停访问存储器或暂停计算。图 3 显示了 PE 的具体结构。PE 内部具备一个配置信息寄存器(Context Register),用于配置数据路径,关于配置信息的介绍将在 3.2 节进行介绍。

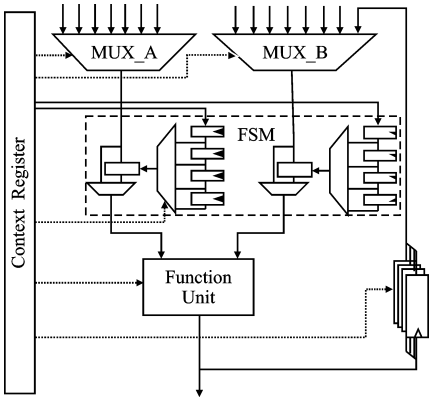


图 3 PE 结构

由于 PE 具有浮点计算延迟周期的特性,在组织数据路径时,有时需要对数据节拍进行控制,PE 中的延迟单元提供了这一功能,延迟单元由有限状态机进行控制。每个 PE 配备了一定数据量的本地

寄存器,用于缓存计算结果。本地寄存器的数量等于 PE 的计算延迟周期数。例如,如果执行浮点加法的 PE 的计算延迟周期数为 4,则该 PE 具有 4 个本地寄存器,用于缓存累加运算最后的 4 个结果数据。最后,PE 的计算结果传回到其它 PE 的输入端口或者通过 STORE 单元写回存储器。RFU 的上游是 LOAD 单元,负责将 Data Memory 中的数据“泵”入 RFU。RFU 的下游是 STORE 单元,负责将 RFU 的最终计算结果写回 Data Memory。在数据层,一次完整的计算周期总是从 LOAD 单元读出数据开始,到 STORE 单元写回数据结束。

重构控制单元是指令层和数据层之间的接口。用户在指令层下发运算指令时只需给出当前派遣的运算指令与前序运算指令的数据依赖关系和当前派遣的运算指令的计算类型,无需关心数据层具体计算资源的使用和数据路径的组织。重构控制单元提供的自适应重构机制负责完成运算指令到具体的计算资源的映射。重构控制单元包含配置信息体缓存(Configuration Body Cache, CBC)和配置信息缓存(Configuration Cache, CC),它们是 RGFA 体系结构中额外的存储部件,用于实现 PE 计算资源自动分配和生成配置信息的功能。CBC 保存每一条运算指令正在使用的 PE 的编号。CBC 是一个双路缓存,保证每次读操作读出来的值是最近写入的值。CBC 可以被配置信息体(Configuration Body, CB)中的 MUX_A/MUX_B 字段索引,索引命中则表示找到此次配置源数据的上一级。CC 则保存了重构控制单元生成的要下发到 RFU 的配置信息。配置信息在 CC 中可以被 PE 的序号索引。关于 CBC 和 CC 的使用方法将在 3.2 节进行介绍。

3.2 自适应可重构机制

本文提出的自适应可重构机制是一套以运算指令的逻辑 ID 到可重构计算资源 PE 的物理 ID 的转化为核心的计算任务映射方案。图 4 显示了运算指令从指令层到数据层的映射过程。首先,如图 4(a)所示,从程序代码块提出数据流程图,在计算资源和数据依赖关系的约束条件下,将其划分为不同的组(Set),并按顺序为组内各个节点分配逻辑编号(CBID)。图 4(b)显示了 RGFA 自适应可重构机制为运算指令分配 PE 计算资源的过程, RGFA 以组为单位对 RFU 进行重构,目前支持对两组运算指令的并行执行。

自适应可重构机制依赖于有关的表项,这些表项用于记录运算指令解析过程中产生的相关信息,

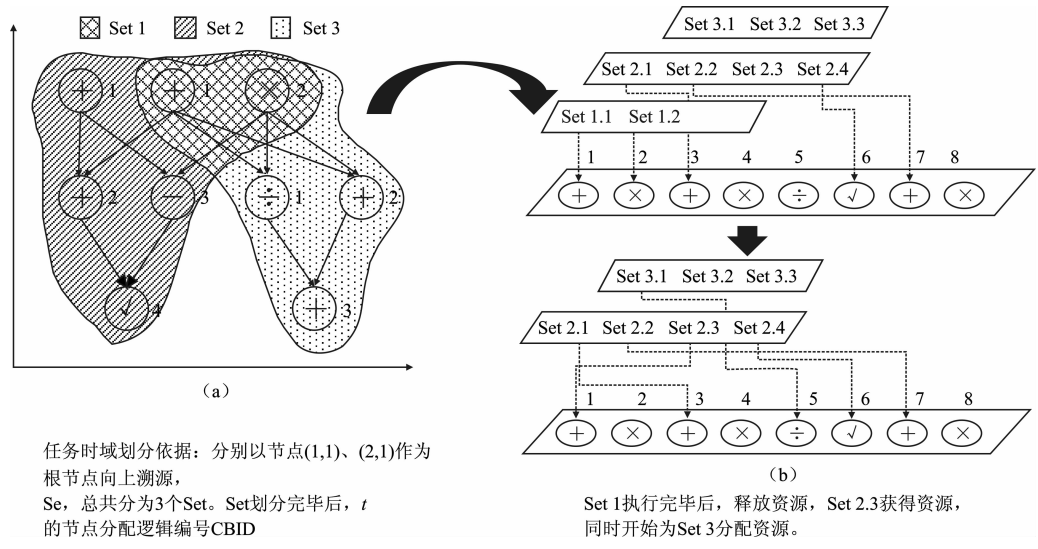


图 4 任务映射示意图

比如，操作数的路由信息和不同 PE 单元的计算延迟信息等。配置信息体(Configuration Body,CB)是表项之一，结构如图 5 所示，CB 由若干比特域构成，用于保存指令层下发的运算指令。运算指令分为两类：PE 运算指令和 LOAD/STORE 访存指令，两类指令有各自的比特域格式。无论哪种类型的指令，都具备 CB_ID 和 OP 这两个比特域，CB_ID 指明运算指令在组内的逻辑序号，OP 指明运算指令的操作类型。在 PE 运算指令中，MUX_A/MUX_B 比特域指向前序运算指令的 CB_ID，指明该条运算指令的源数据来源。在存储器访存指令中，ADDR_INC 比特域指明访存单元的访存地址的递增增量，访存所需要的基地址则由 RISC-V 的自定义 CSR 寄存器硬连线至访存单元。BATCH_LEN 比特域指明访存操作需要进行的次数，这种连续读/写的计算能力可以极大的提高 RGFA 的吞吐量。

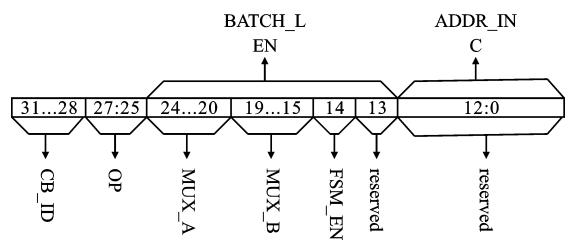


图 5 配置信息体结构

CC 中的表项与 PE 呈一一对应的映射关系。表项中 Busy 比特域指明 PE 当前是否正在被使用；CBC 中的表项与指令层下发的运算指令呈一一对应的映射关系，每个表项可以包含两个 PE 编号，两组运算指令可以并行执行，组间具有相同 CB_ID 的

运算指令由双路 CBC 中的一个表项来维护。在指令层将运算指令下发至 CB 后，重构控制单元使用 CB 中的 OP 域查找 CC 中的表项，将空闲的 PE 的编号写入 CBC 中。同时，重构控制单元使用 CB 中的 MUX_A/MUX_B 域索引 CBC，将读出的 PE 编号写入 CC 中，完成运算指令的逻辑 ID 到 PE 计算资源的物理 ID 的转化。图 6 中的电路结构展示了这一转换过程。重构控制单元也会将 CB 中的其它信息写入到 CC 中，如 CB_ID。算法 1 中的伪代码展示了这一重构过程的部分行为。

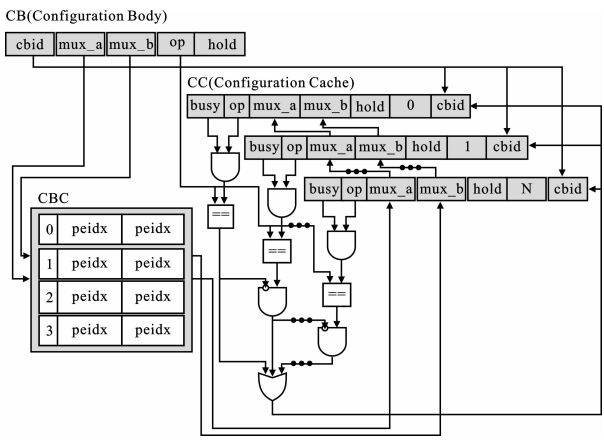


图 6 分配 PE 计算资源

算法 1：重构控制单元部分算法

```
for (i=0; i<pe. number; i++){
    if(cb. operation == cc[i]. operation && ! cc[i].
    busy){
        cb_cache[cbid] = i;
        cc[i]. cb = cb. cbid;
        for PE operation & STORE: cc[i]. mux_a =
```

```
cb_cache[cb_mux_a];
for PE operation: cc[i].mux_b = cb_cache[cb_mux_b];
for PE operation & STORE: cc[i].l=fsm_en = cb.fsm_en;
for LOAD & STORE: cc[i].addr_inc = cb.addr_inc;
for LOAD & STORE: cc[i].base_addr = CSRs[i];
for LOAD: cc[i].batch_len = cb.batch_len;
cc.busy = 1;
}
}
```

最后,重构控制单元将得到的配置信息从 CC 发送到 RFU 中对应的 PE 的配置信息寄存器.在该 PE 完成计算,被释放之前,CC 不会再向该 PE 发送配置信息.

4 实验与性能测试

本文为 RGFA 配备了 8 个基础浮点运算单元和 16KB 的数据存储器,数据存储器分为 8 块,具有 8 个读端口和 4 个写端口.本文使用 Verilog HDL 语言完成通用浮点处理器和多核系统的硬件描述,并在 Xilinx Ultrascale 系列 xcvu440 的 FPGA 芯片上进行了原型验证,系统可以稳定工作在 200MHz 频率下, RGFA 的资源消耗情况如表 1 所示.

表 1 加速器资源消耗				
name	CLB LUTs	Block RAM	DSPs	CLB
RISC-V	1 158	4	0	254
RFU	9 318	0	14	1 665
Data memory	1 638	16	0	331
Controller	1 118	0	0	421
LOAD/STORE	121	2	0	122
RGFA	13 341	22	14	2 386

为了评估可重构浮点加速器的性能,本文为 RGFA 分别挂载 8/16 个基础浮点运算单元,在 FP-GA 平台上进行了原型验证.作为对比,本文采用 Xilinx ZYNQ-7000 全可编程系统芯片中的单核 ARM Cortex-A9 处理器的测试性能作为对比对象,对多个应用进行了分析,分别统计出它们在软件和浮点加速平台下实现的周期数.本文将各类应用中的核心算法提取出来,在保证算法原理一致的情况下,依据硬件特性做适应性修改,将算法映射到 Cortex-A9 和 RGFA 中.采取的应用则是一些典型的数据密集型和计算密集型应用,包括科学计算、LTE、机器学习等,分别选取了科学计算中常用的

复数乘法、点积和矩阵乘法;通信中的 8 点 FFT;机器学习中的数据降维常用的求矩阵特征值/特征向量(本文采用 jacobi 算法求矩阵特征值/特征向量).表 2 统计了多种测试应用的 3 种实现方案的执行周期数.

应用类型	Cortex-A9	浮点加速器	浮点加速器
		(8 个基础运算单元)	(16 个基础运算单元)
Complex_mul_1K	43 205	1 437	744
Dot_product_1K	14 049	7 802	4 493
Matrix_mul_64x64	5 959 203	1 163 264	678 384
FFT_8_1K	88 236	21 482	16 712
Jacobi_16x16	3 680 273	1 523 465	1 274 406

为了直观的看出多种应用在不同结构下的加速效果,图 7 列出了不同应用运行的归一化时间,以 Cortex-A9 实现方式作为基准,其它的实现方式基于 Cortex-A9 实现做出归一化调整.

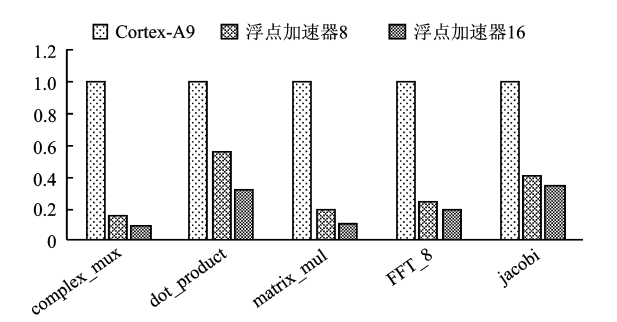


图 7 不同实现方案的性能比较

从表 2 和图 7 可以看出,自适应可重构浮点加速架构对各类数据密集型和计算密集型应用的加速效果明显.对于并行度较高的应用,如矩阵乘法,为 RGFA 配备更多的基础浮点运算单元可以带来更好的性能表现;对于并行度较低的应用,如 FFT_8 中多级蝶形运算无法并行,数据相关性强,增加计算资源带来的收益则不明显;jacobi 算法在实现时整体是一种不确定循环结构且存在循环嵌套,算法映射较为复杂,能够验证通用浮点处理器的算法适应能力.通过以上分析可以看出 RGFA 的可重构特性使其更好地适用于各类加速场景而不仅仅局限于某一类应用.

5 结束语

本文针对高密度计算领域的算法特点,给出了一种自适应可重构通用浮点加速器的设计与实现.本文设计的浮点加速器在指令层完成计算任务划

分,在数据层实现可重构硬件资源的实时分配的配置信息的实时生成,实现可重构资源的动态调度,并利用浮点计算的流水线计算时间来隐藏生成配置信息的延时.实验结果表明,可重构通用浮点加速器具有较高的计算效率和灵活性,对各类计算密集型浮点应用均有良好的加速效果.

参考文献:

[1] CHEN Z Y, ZHOU H, GU J. R-accelerator: an RRAM-based CGRA accelerator with logic contraction[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019, 27 (11): 2655-2667. DOI: 10.1109/TVLSI.2019.2925937.

[2] WIJTVLIET M, WAEIJEN L, CORPORAAL H. Coarse grained reconfigurable architectures in the past 25 years; overview and classification[C]//Proceedings of 2016 International Conference on Embedded Computer Systems: Architectures, Modeling and Simulation. Agios Konstantinos, Greece: IEEE, 2016: 235-244. DOI: 10.1109/SAMOS.2016.7818353.

[3] SINGH H, LEE M H, LU G M, et al. MorphoSys: an integrated reconfigurable system for data-parallel and computation-intensive applications [J]. IEEE Transactions on Computers, 2000, 49(5): 465-481. DOI: 10.1109/12.859540.

[4] MEI B F, VERNALDE S, VERKEST D, et al. ADRES:

an architecture with tightly coupled VLIW processor and coarse-grained reconfigurable matrix[M]//CHEUNG P Y K, CONSTANTINIDES G A. Field Programmable Logic and Application. Berlin, Heidelberg: Springer, 2003: 61. DOI: 10.1007/b12007.

[5] HARBAUM T, SCHADE C, DAMSCHEN M, et al. Auto-SI: an adaptive reconfigurable processor with run-time loop detection and acceleration[C]//Proceedings of the 2017 30th IEEE International System-on-Chip Conference. Munich, Germany: IEEE, 2017: 153-158. DOI: 10.1109/SOCC.2017.8226027.

[6] BECK A C S, RUTZIG M B, CARRO L. A transparent and adaptive reconfigurable system[J]. Microprocessors and Microsystems, 2014, 38(5): 509-524. DOI: 10.1016/j.micpro.2014.03.004.

作者简介:

余振波 男,(1995-),硕士研究生.研究方向为嵌入式系统综合测试.

聂言硕 男,(1995-),硕士研究生.研究方向为集成电路设计与测试.

宋宇鲲 男,(1975-),博士,副研究员.研究方向为面向数据密集与计算机密集应用的 SoC/MPSoC 体系结构 with 实现.

侯 宁(通讯作者) 男,(1982-),博士,副教授.研究方向为 SoC 设计、片上网络、多核系统设计.

E-mail: 30090811@hncj.edu.cn