基于 Xilinx SoPC 的可重构嵌入式计算系统的研究与设计

张 宇 冯 丹
(华中科技大学计算机科学与技术学院 武汉 430074)

摘 要 由于应用种类、实时性以及处理效率等要求,高性能嵌入式计算硬件平台需要具备相当的计算能力以及一定的适应性。为此提出了一种基于 Xilinx FPGA 的动态可重构的片上系统设计方案。系统采用专用硬件来执行计算密集型任务,运用动态可重构技术来支持硬件处理模块功能的动态配置。研究了 Xilinx 可编程片上系统上的 3 种硬件加速方案:CPU 协处理器、PLB 扩展加速器和 MPMC 扩展加速器。实验数据表明 MPMC 加速器性能最优。在 Virtex5 FPGA 器件上实现了可动态重构的 MPMC 加速器,以 128 位 AES 加密、解密两个功能模块为例,从硬件资源占用率、重构延时等角度考察了可重构系统的特点。

关键词 嵌入式计算,可编程片上系统,可重构计算,协处理,加速器
中图法分类号 TP36 **文献标识码** A

Study and Design of Reconfigurable Embedded Computing System Based on Xilinx SoPC

ZHANG Yu FENG Dan
(School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China)

Abstract The high performance embedded computing systems need considerable computational power and flexibility to meet various application requirements. A reconfigurable SoPC design based on Xilinx FPGA was presented. The system uses a dedicated hardware accelerator to process computational intensive tasks, and the accelerator can be dynamically configured during run-time. The hardware processing engine can be coupled to the host system as a CPU coprocessor, a PLB accelerator or an MPMC accelerator. Based on the experimental result that the MPMC accelerator had the highest performance, a reconfigurable MPMC accelerator was designed and integrated into the SoPC system. The experiments used 128-bit AES encryption and decryption as case studies. Features such as hardware resource utilization and reconfiguration latency for the reconfigurable system were also studied.

Keywords Embedded computing, System on programmable chip, Reconfigurable computing, Co-processing, Accelerator

1 引言

嵌入式计算系统需具备较强的计算能力,还需满足较好的实时性、低功耗和低成本等要求[1],其设计具有一定的挑战性。过去几十年中,在摩尔定律的推动下,处理器的性能得到了飞速的发展。以处理器为核心部件的嵌入式计算系统比较依赖软件的解决方案。出于进一步增强计算性能的需要,人们提出了 MPSoC(Multi-Processor System on Chip)架构,MPSoC 在 SoC(System on Chip)上集成多个处理器核心来提高程序的并行性。与服务器、PC 上多采用同构的多处理核心架构不同,MPSoC 多采用异构的体系架构,这主要是由于 SoC 系统成本、功耗需求严格,硬件架构需根据应用来定制,以达到量体裁衣的效果。相对于基于软件程序处理方案的 MPSoC,某些情况下我们还需要采用专用硬件实现对一些复杂算法的加速,这是由于:

①一些复杂且已被定制为标准的算法本身不再会改变,适合利用专用硬件执行。

②一些运算操作的执行不适合采用软件方式。比如 bit 一级上的数据操作在绝大多数处理器上执行十分低效;一些算法操作需要数量巨大的寄存器,利用 CPU 执行困难。

③需在短时间内做输入、输出操作的功能模块适合采用专用硬件加速;假如大量的数据必须在短时间被读入,随即处理,最后输出,这种情况下专用硬件比 CPU 更为有效。

专用硬件加速器具有性能高、硬件成本低和功耗低等优势,但其缺陷是灵活性不够,针对各种算法的硬件模块需提前设计好并集成到系统中,模块无法进行功能改变、功能升级;多个模块的可扩展性不好。改进的方法是采用硬件的动态可重构技术[2,3]。值得强调的是一部分 FPGA(Field Programmable Gate Array)器件支持动态局部重构(Dynamic Partial Reconfiguration),即芯片在工作运行时可以动态配置部分区域的电路结构,而片内其它区域的功能模块将不受到影响。系统可根据应用需要在运行过程中动态地配置硬件处理模块,而不需要将所有模块都预先集成到芯片内部,因此可以减少逻辑资源消耗,降低系统成本和功耗。

到稿日期:2009-06-15 返修日期:2009-08-24 本文受 973 国家基础研究项目(No. 2004CB318201)资助。
张 宇(1979—),男,博士生,主要研究方向为网络接口、可重构计算,E-mail:yuzhang13@gmail.com;冯 丹(1970—),女,博士生导师,主要研究方向为海量存储系统。

动态可重构计算技术在学术界已有一段研究历史。相关研究诸如加州伯克利提出了名为 Pleiades 的异构片上系统架构[4]，Pleiades 采用了粗粒度的可重构模块构建，主要针对高性能、低功耗的数字信号处理应用。Pleiades 由一颗通用片上处理器完成主控调度，在中央处理器周围集成了一些独立的协处理器，这些协处理器通过可重构的片上系统网络进行通讯。Chameleon Systems 提出一种名为 CS2112 的可重构的片上系统架构[5]，该系统主要针对通讯应用。它将一个通用处理器与多个 32 位处理器耦合。这些处理器在被分置在芯片内部不同的层面上并可以动态配置。芯片的最底层存放了整个系统的配置信息。Chameleon 内部有一个状态机，由它来决定一个层面是否有效。片内集成的通用处理器负责可重构的配置，并由它来从最底部的分层中读出比特流配置信息。加州大学欧文分校的 MorphoSys 系统在片上采用了类似 MIPS 的 TinyRISC 处理器。TinyRISC 处理器与一个由 28 位运算器组成的可重构矩阵相连，可提供用户自定制指令集的扩展。MorphoSys 系统由 TinyRISC 控制，它包含了多个独立的配置存储空间并能实现动态重构[6]。

工业界也推出了一些具备可重构能力的异构结构的可编程器件。例如 Xilinx 公司在单颗芯片上将可编程逻辑资源与 PowerPC 硬核处理器结合，提出平台 FPGA 的概念[7]。Altera 公司提供了内嵌 ARM 微处理器的 Excalibur FPGA 器件[8]。Atmel 公司在低端市场上推出了集成 8 比特 AVR 处理器并拥有一定量可编程逻辑资源的芯片[9]。

在业界内，广泛采用 Xilinx 公司的可编程器件，它的市场份额最大，其大容量 FPGA 器件（如 Virtex2、Virtex4 和 Virtex5）可支持动态局部重构。本文主要针对 Xilinx 公司的 FPGA 器件及其对应的 SoPC（System on Programmable Chip）架构，研究并设计了适用于嵌入式计算的可动态重构的片上系统。其主要思路是采用处理器结合专用硬件处理模块的架构来加速性能。硬件处理模块基于动态重构技术实现，可以在系统运行过程中灵活地动态配置。本文的主要工作包含以下两个方面：

• 研究了 Xilinx SoPC 架构内 3 种形式的硬件加速方案：(1)与 CPU 耦合的协处理器；(2)挂接在 PLB 总线上的加速器；(3)挂接在 MPMC(Multi-Port Memory Controller)Switch Fabric 上的加速器。

• 针对性能最优的 MPMC Switch Fabric 上的加速器方案，给出了其可重构硬件的设计方法。

在实验环节，我们构建了一个集成 128 位 AES 加密、解密运算模块的嵌入式计算系统，在 Xilinx Virtex5 FPGA 上完成了原型系统的实现。SoPC 中采用 Microblaze 处理器。采用可重构技术集成加密、解密硬件处理模块，并从性能、逻辑资源消耗、重构延时等方面对系统进行了评估。

本文第 2 节研究 Xilinx SoPC 的硬件架构下 3 种硬件加速方案的特点；第 3 节描述可重构系统的硬件的设计以及实验情况；最后对全文进行总结。

2 三种硬件加速方案及其特点

Xilinx 片上系统可以支持 PowerPC 硬核处理器（需器件内部集成）和 Microblaze 软核处理器。硬核处理器性能高，但器件成本也高；与之相比，软核处理器性能较低，但由于采用 FPGA 可编程资源构建，设计灵活且成本低廉。本文主要研究 Microblaze 软核处理器结合专用协处理器硬件的方案。实验结果证明该方案性能远远高于基于硬核处理器的软件实现方案。

• CPU 协处理器

协处理器本身可被视为 CPU 的算术逻辑运算单元的功能扩展，用户须调用专用的指令集以实现对协处理器的控制。Microblaze 支持 FSL（Fast Simplex Link）接口扩展协处理器。集成有协处理器的 SoPC 架构如图 1 所示。

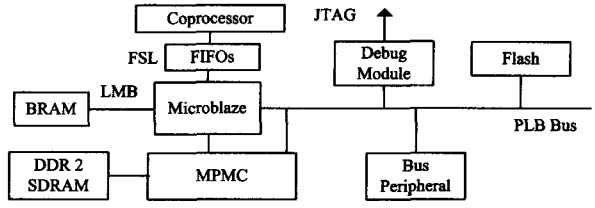

图 1 Microblaze 协处理器系统

工作流程方面，程序用 put 指令将寄存器内的数据写到 FSL FIFO。协处理器将需要处理的数据从 FIFO 中读出，完成处理后将结果写入 FIFO。程序再利用 get 指令将数据从 FSL FIFO 读到 CPU 寄存器中，最后将其写回内存，完成一次处理。

• PLB 加速器

与协处理器不同，硬件加速器主要是挂接在系统总线上。加速器会映射出一组硬件寄存器，CPU 则通过内存寻址空间（Memory Space）读写这些寄存器来完成对加速器的操控。对于挂接在 PLB 总线上的硬件加速器，其系统架构如图 2 所示。

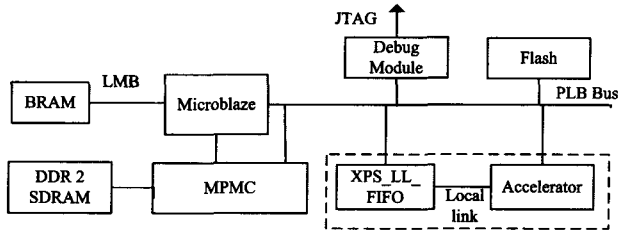

图 2 基于 PLB 总线扩展的硬件加速器

PLB 加速器的工作采用非 DMA 模式，加速器的数据通路与 XPS_LL_FIFO 相连接。工作流程是：CPU 将数据从内存读到寄存器中。通过 PLB 总线将数据从寄存器写入 XPS_LL_FIFO。当全部数据写入 FIFO 后，加速器通过 Local Link 接口将数据取出，进行处理；处理完毕后，将结果写回 XPS_LL_FIFO，产生中断。Microblaze CPU 再将结果从 FIFO 以 32 位方式读到其寄存器中，最后写回内存。

• MPMC 加速器

加速器的数据传输环节也可以依赖 DMA 控制器。MPMC 控制器内部集成了 DMA 控制器，MPMC 的 DMA 是基于描述符机制进行数据传输。集成 MPMC 加速器的 SoPC 架构如图 3 所示。工作流程是：设置好描述符链表，开启 MPMC 的 DMA 控制器，DMA 控制器将待处理数据从内存读入自身的缓冲区。加速器通过 Local Link 接口将 DMA 缓冲区内的数据读出并进行处理，处理后将结果写入 DMA 缓冲区。最后 DMA 控制器将数据写回内存，产生中断，通知主系统处理完毕。

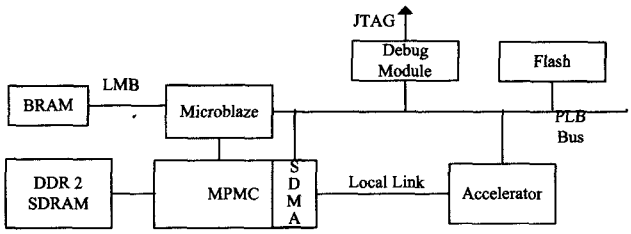

图 3　基于 MPMC 交换式总线扩展的加速器

为了研究 3 种硬件加速方案的特点，本文选取了 128 位 AES 加密运算作为性能评估案例。硬件处理模块的设计参考了 Opencores 上的 AES 开源项目[10]，在其基础上针对 Xilinx 器件做了修改。模块采用 128-bit 的数据输入输出，完成 128 bits 明文加密需要 22 个时钟周期(10 轮处理占用 20 个时钟周期、密钥扩展占用 1 个时钟周期以及结果输出占用 1 个周期)。硬件模块运行在 100MHz 时钟频率下时，理论上可以达到 73MB/s 的数据加密吞吐率。

实验平台是 Xilinx ML505 开发板，板上配备了 Virtex5 系列 5vlx50tff1136-1 FPGA 器件。开发工具是 Xilinx ISE v10.1.3 以及 EDK v10.1.3。实验中我们对 16kB 的数据进行加密，待处理的数据都已经预先存放到了 DDR2 内存中，整个处理环节包括从内存取数据、数据处理以及结果写回内存 3 个步骤。性能测试结果如表 1(a)所列。采用 XPS_TIMER，以系统时钟周期数(100MHz)为单位进行计时。用软件处理方式实现了 AES 加密算法(参考 Optimized ANSI C code for the Rijndael cipher[11])，该算法在集成有 PowerPC440 (32kB cache，400MHz)硬核处理器的系统上运行，加密吞吐率达 3807kB/s (约为 3.718 MB/s)。

表 1

(a)性能测试结果对比

	Execution Cycles	Execution Time	Throughput
HW-FSL 1	1034200	0.01034sec	1.51MB/s
HW-FSL 2	113198	0.00113sec	13.80MB/s
HW-PLB1	2003663	0.02004sec	0.78MB/s
HW-PLB2	246606	0.00247sec	6.34MB/s
HW-MPMC 1	51867	0.00052sec	30.125MB/s
HW-MPMC 2	37827	0.00038sec	41.036MB/s

(b)SoPC 硬件资源消耗对比

	Slice Registers	Slice LUTs	BlockRAM
HW-FSL 2	7177	5984	35
HW-PLB 2	8027	7166	36
HW-MPMC 2	9043	7857	43

HW-FSL 1：MicroBalze v7.10d (100MHz，NO cache)，DDR2 (200MHz)

HW-FSL 2：MicroBalze v7.10d(100MHz，8kB I/D cache)，DDR2 (200MHz)

HW-PLB 1：MicroBalze v7.10d (100MHz，NO cache)，DDR2 (200MHz)

HW-PLB 2：MicroBalze v7.10d(100MHz，8kB I/D cache)，DDR2 (200MHz)

HW-MPMC 1：MicroBalze v7.10d(100MHz，NO cache)，DDR2 (200MHz)

HW-MPMC 2：MicroBalze v7.10d(100MHz，8kB I/D cache)，DDR2(200MHz)

表 1(b)归纳了 3 种硬件加速方案在 Virtex-5 5vlx50tff1136上综合、映射后的硬件资源消耗情况。实验总结如下：

①对于 FSL 协处理器以及 PLB 加速器这两种方案，CPU 的改善对性能提升效果显著。例如，当采用了 8kB 的 Cache 时，系统运行较未配置 Cache 而言性能增幅十分明显。这主要是由于上述方案中数据搬运、系统同步环节都由处理器直接负责。

②PLB 加速器方案性能远不及另外两种硬件处理方案，只达到了 AES-128 加密引擎理论最大吞吐率(73MB/s)的 8.7%，这主要是由于数据需要在 PLB 总线上往返传输，而 PLB 总线传输的开销较大，导致了性能降低。

③三者中 MPMC 加速器方案性能也最优，为 41MB/s，达到了 AES-128 加密引擎理论最大吞吐率的 56%。由于数据传输采用了 DMA，CPU 操作较少，因此该方案受处理器性能的影响较小。缺点在于硬件资源消耗最多，同时驱动设计比较复杂，软件开发难度较大。

④CPU 协处理器方案性能在三者内居中，该方案可编程逻辑硬件资源消耗较少，同时软件设计简单。但由于处理过程中 CPU 对协处理器操作频繁(主要是数据输送环节)，因此 CPU 占用率较高。

综合上述因素考虑，本文将基于性能最高的 MPMC 加速器方案来设计可重构的面向高性能嵌入式计算 Xilinx SoPC 系统。

3　动态可重构系统的设计与实验

Xilinx 公司 2006 年 3 月推出了 EAPR(Early Access Partial Reconfiguration)动态局部重构电路设计流程。EAPR 可重构设计采用自底层向上的开发方式。在设计初期每个子模块(包含顶层模块、静态模块和动态模块)必须独立地进行综合，形成各自的电路网表。接下来由这些模块组合成为一个初始的设计，最后将动态模块逐一插入设计中。设计者需要预先在芯片内部划分出一定大小的区域，以容纳动态模块。Xilinx 提供了名为 HWICAP 的 FPGA 配置控制器 IP 核，可以通过 PLB 总线集成到片上系统中，用户利用驱动程序接口实现对 FPGA 器件的动态配置。

在 MPMC 上扩展动态可重构的外设可按图 4 所示实现，但需要注意以下几点：

①在 Xilinx EDK 工具下搭建静态的 SoPC 系统时，需要设计一个虚拟外设模块(XPS_LL_SOCKET)，该模块用于定义外设寄存器的空间地址映射。

②动态电路与静态电路之间需用总线宏(Bus Macro)模块来进行连接。

③用通用 IO 模块(GPIO)输出两个控制信号：BM_Enable 和 User_Reset。由于 PLB 总线是一种共享式总线，重构过程中模块的某些信号会产生非稳定数值而影响到总线上其它模块的正常工作。重构之前将 BM_Enable 信号置为无效，Bus Macro 的输出恒定为低电平。

④SoPC 系统在上电后由一个 reset 控制器来对 CPU 以及各个外设模块逐一进行 RESET 初始化。否则硬件模块内部的状态机可能被置于一个死循环点而导致不能工作。动态模块在重构完毕后，系统不能自发地产生 reset 信号对硬件进行初始化置位。需要设计一个程序可以控制的软置位。在此利用 GPIO 来产生一个软件可控的 User_Reset 信号，用于动

态模块的初始化置位操作。

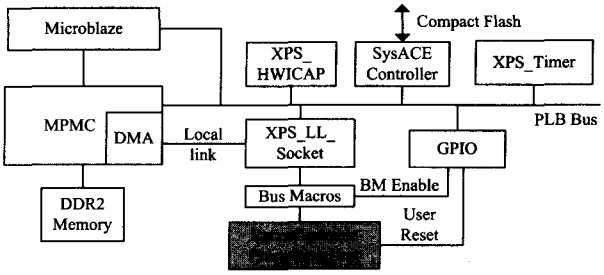

图 4　可重构 MPMC 加速器

图 4 所示的是集成了一个可重构处理模块的最小化系统；由于 MPMC switch fabric 最多可以支持 8 个端口的扩展，因此实际使用中可以集成多个可重构处理模块。

实验中采用了 AES-128 加密与 AES-128 解密两个功能模块作为案例。解密算法的硬件设计仍以 Opencores 上的 AES-128 项目为参考。我们将设计针对 Xilinx 器件做了移植，对于 128 位数据的加密、解密操作各需 22 个时钟周期完成。若不采用动态重构技术，将两个加速器都集成到片上系统中，设计在 MAP 过程中会产生 overmapped 错误。原因在于设计总共需要用到 61 个 BlockRAM 模块，而器件本身只能提供 60 个。采用 Xilinx EAPR 流程，我们在图 4 所示的 SoPC 的 RPU 上实现加密和解密功能。表 2 总结了设计的资源占用情况，其实验平台环境如下：

①FPGA 器件是 Virtex-5 5vlx50tff1136。

②设计开发工具：EDK v10. 1. 3，ISE v10. 1. 3，EAPR v9. 2_PR12。

③SoPC 的主要部件是 Microblaze CPU v7. 10d(8kB I/D cache)，PLBv46 v1. 03a，MPMC v4. 03a，XPS_HWICAP v1. 00a，XPS_GPIO v1. 00a，XPS_ SYSACE v1. 00a，XPS_ UART16550 v2. 00b，XPS_TIMER v1. 00a，XPS_INTC v1. 00a。

④Microblaze CPU 以及 PLB 总线频率是 100MHz，DDR2 SDRAM 控制器频率为 200MHz。

表 2　硬件资源消耗

	Slice LUT (Num/Total)	Slice Register (Num/Total)	Block RAM (Num/Total)
AES-128 Encryption Accelerator	818/28800	1204/28800	660
AES-128 Decryption Accelerator	1127/28800	1241/28800	10/60
Static SoPC Design (with two accelerators)	10426/28800	11751/28800	61/60
Reconf. SoPC Design (configured as En. Acc)	10102/28800	9567/28800	43/60
Reconf. SoPC Design (configured as De. Acc)	10408/28800	9604/28800	47/60

可以看出，在具备相同功能模块的情况下，可重构的设计方案将节省硬件资源。器件上的布局布线(floorplan)如图 5 所示。预先在芯片上划分出一块重构区域，该区域内的资源需容得下两个加速器之中的资源消耗最多者。

可重构系统可以动态加载或卸载硬件模块，其使用比传统的静态电路更加灵活。但重构过程需要一定的时间，这就带来了重构延时的问题，重构延时有可能使得系统整体性能降低。表 3 归纳了实验中获得的动态模块的重构参数，其中重构延时是通过 XPS_TIMER 进行计时得出的(计时单位是时钟周期数，时钟频率为 100MHz)；规定 bit-stream 配置文件已经从 Compact flash 读入 DDR2 内存中。本文中的重构延时定义为处理器将 bit-stream 从 DDR2 内存中写入 XPS_HWICAP 模块，再由 XPS_HWICAP 执行配置两个环节的时间总和。

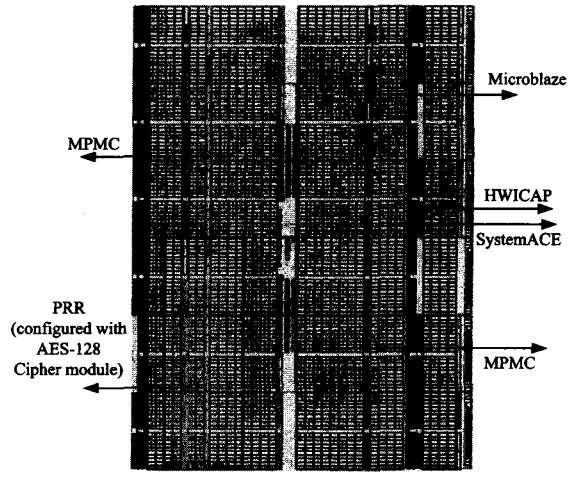

图 5　芯片布局图

表 3　重构延时

	Reconfiguration Latency (Clock Cycles@100MHz)	Bit-stream Size (Bytes)	Reconfiguration Throughput(kB/s)
Encryption module	1282041	182468	13899
Decryption module	1307166	185968	13893

可以看出重构的延时在 10 ms 这个数量级上。对于任务切换比较频繁的重构系统而言，重构延时是一不可忽视的开销。利用 XPS_HWICAP 模块进行重构配置，重构延时的计算定义为：

$$T_{reconf}=T_{data\_transfer}+T_{ICAP\_conf}$$

式中，$T_{data\_transfer}$ 指 bit-stream 从 DDR2 内存中的读取所需时间加上在 PLB 总线上的传输时间，T_{ICAP_conf} 指 ICAP 模块的配置时间。对于 ICAP，其理论上最大的配置吞吐率可以由以下公式表示：

$$MTT=\frac{IDIT}{ClockPeriod}$$

式中，IDIT(ICAP Data Input Width)指 ICAP 模块的数据 I/O 位宽；Virtex4，Virtex5 器件内部的 ICAP I/O 均为 32 位。那么当频率为 100MHz 时，MTT(Maximum Theoretical Throughput)为 400MB/s。表 3 中 XPS_HWICAP 的吞吐率约为 13. 6MB/s，$T_{data\_transfer}$ 占到了整个重构延时的 96%，此环节中，处理器、DDR2 SDRAM 的性能以及 PLB 总线的数据传输效率都是决定因素。

结束语　嵌入式计算平台需具备强大的处理能力和较好的灵活性，以适应不同的应用。针对于此，本文提出了一种基于动态重构的硬件协处理的片上系统解决方案。在 Virtex5 FPGA 器件上实现了可动态重构的 MPMC 加速器。实验以 128 位 AES 加密、解密运算为例，证明了系统具备高处理性能、低硬件资源消耗率的特点。可重构系统具备较好的灵活性，不足之处是重构过程需要一定的延时。在实际使用中可

(下转第 281 页)

构，利用时间冗余技术实现微处理器容错机制的方法。介绍了其中指令复制方法的实现，并着重分析了利用指令复制实现控制流检测。从仿真结果看，这种设计方法可以有效检测出处理器正常指令流的程序计数器及相关电路中出现的瞬态故障。提出的方法将程序控制流检测和指令复制机制融合起来，与类似的利用时间冗余技术实现的容错处理器相比，进一步节省了硬件资源，减少了额外的执行时间。

参 考 文 献

[1] 李红兵，尚利宏，周密，等. 一种执行解耦的容错处理器[C]//第十三届全国容错计算学术会议. 2009

[2] Franklin M. A Study of Time Redundant Fault Tolerance Techniques for Superscalar Processors[C]// Digest of Papers in the 25th International Symposium on Fault Tolerant Computing. 1995:207-215

[3] Sosnowski J. Transient fault tolerance in digital systems [J]. IEEE Micro, Feb. 1994:24-35

[4] Rotenberg E. AR-SMT: A Microarchitectural Approach to Fault Tolerance in Microprocessors [C] // Proceeding of the 29th Fault-Tolerant Computing Symposium. June 1999

[5] Ray J, Hoe J C, Falsafi B. Dual Use of Superscalar Datapath for Transient-Fault Detection and Recovery[C]// Proceeding of 34th Microarchitecture. Dec. 2001

[6] Frohwerk R. Signature analysis: a new digital field service method[J]. Hewlett-Packard J, May 1977:2-8

[7] Burger D, Austin T M. Simplescalar Tool Set Version 2.0[J]. Computer Architecture News, 1997, 25(3):13-25

(上接第 277 页)

集成多个动态模块，多个处理模块的配置可以采用 configuration prefetching, configuration caching 等技术来屏蔽重构延时。

参 考 文 献

[1] Dally W J, Balfour J, et al. efficient Embedded Computing[J]. Computer, 2008, 41(7):27-32

[2] Jou Jer Min, Lee Yun-Lung, et al. A Novel Reconfigurable Computation Unit for DSP Applications[C]// IEEE Computer Society Annual Symposium on VLSI. Porto Alegre, Brazil, 2007

[3] Dutt N, Choi Kiyoung. Configurablef processors for embedded computing[J]. Computer, 2003, 36(1):120-123

[4] Wan M, Zhang H, George V, et al. Design methodology of a low-energy reconfigurable single-chip DSP system[J]. Journal of VLSI Signal Processing, 2001, 28(1):47-61

[5] Salefski B, Caglar L. Reconfigurable computing in wireless [C] // 38th Design Automation Conference. Las Vegas, Nevade, USA, June 2001

[6] Singh H, Lee M H, Lu G, et al. MorphoSys: an integrated reconfigurable system for data-parallel and computation-intensive applications[J]. IEEE Transactions on Computers, 2000, 49(5): 465-481

[7] Xilinx Inc. Virtex-II Pro Platform FPGAs[EB/OL]. www.xilinx.com/virtex2pro

[8] Altera Inc. Excalibur Embedded Processor[EB/OL]. www.altera.com/products/devices

[9] Atmel Inc. Field Programmable System Level Integrated Circuits (FPSLIC devices)[EB/OL]. www.atmel.com/products/FPSLIC

[10] OpenCores. AES project[EB/OL]. http://www.opencores.org/

[11] AES Lounge[EB/OL]. http://www.iaik.tugraz.at/content/research/krypto/AES/

重视中、英文摘要的编写

检索数据库要依据文摘的质量来决定该文章是否被收录，读者要根据文摘提供的信息考虑是否阅读、引用原文，如能被利用，才能体现文章的学术价值，提高原文的引用频次，摘要在一篇文章中的作用非常重要！

中文摘要一般不宜超过 200～300 字，英文文摘的长度一般不超过 250words，不少于 150words，语言简洁。除了实在无变通办法可用以外，摘要中不用图、表、化学结构式、非公知公用的符号和术语。关键词一般为 3～8 个，每个关键词首字母大写。

1. 对文献进行主题分析，通过文摘体现主题概念、主题内容等该篇文献最重要的信息，使读者在没有看到全文的情况下，能够很清楚地了解到该篇文献的中心思想。

2. 英文文摘应与中文文摘相对应。

3. 信息量要完整，能够全面包含论文的关键信息，应包括目的、过程及方法、结果三方面内容。

4. 尽量简化一些措辞和重复的单元，删繁从简。

5. 注意文法(详细请见本刊网站的修改要求)。

6. 取消或减少背景信息。背景信息如果过长或占文摘篇幅的比例过大，则反映出的就是对作者所做的工作描述过于笼统和简单。

7. 作者在文献中谈及的未来计划不纳入文摘。

8. 首句不得简单重复题名中已有的信息。

9. 不用第一人称，用过去时态叙述作者工作，用现在时态叙述作者结论。

10. 文摘中的缩写名称在第一次出现时要有全称。