

可重构异构系统结构研究

姜晶菲, 唐玉华, 刘福东, 胡毅

(国防科学技术大学 计算机学院, 湖南 长沙 410073)

摘要:可重构异构系统是由通用微处理器、可重构模块、专用 ASIC、IO 接口等资源构成的异构并行处理系统, 文中提出的可重构异构系统结构融合了不同的计算资源, 使系统中的某些资源能够很大限度地满足某种应用的模式和处理要求。系统中可重构模块的硬件功能可以通过在线重构技术加以改变, 各模块之间的互连关系可通过重构互连控制器调节和仲裁。这种可变性使计算系统能适应更大范围的应用需求, 向一体化和高性能的方向发展。

关键词:异构; 重构互连控制器; 可重构模块

中图分类号: TP303

文献标识码: A

文章编号: 1673-629X(2009)01-0025-04

Research on Reconfigurable Heterogeneous System

JIANG Jing-fei, TANG Yu-hua, LIU Fu-dong, HU Yi

(School of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract: Reconfigurable heterogeneous system(RHS) is composed of different modules, such as microprocessor, reconfigurable module, A-SIC, IO interface module and etc. This kind of system combines several different computing resources to satisfy the processing mode of distinct applications thoroughly. The hardware functions of the reconfigurable modules can be changed with run-time reconfiguration technology. The interconnection relationship of all modules in RHS can be arbitrated and modulated by reconfigurable connection controller. The flexibility of RHS made the system incorporate and suitable for applications in larger scope, which can achieve higher performance.

Key words: heterogeneous; reconfigurable connection controller; reconfigurable module

0 引言

微处理器 + 专用 ASIC 是高性能计算机系统保障软件灵活性, 同时提升性能的常用结构, 这样的异构系统能够解决微处理器的适应性问题, 通过合理配置异构的芯片资源, 计算系统能更好地适应特定应用的计算模式, 达到较高处理性能。可重构技术指依靠软件来改变硬件电路结构的技术, 芯片在生产出来之后硬件线路还能根据应用需求加以改变, 这样既保留微处理器的软件灵活性, 又发挥专用 ASIC 的优势。

可重构异构系统(RHS, Reconfigurable Heterogeneous System)是由微处理器、可重构模块(RM, Reconfigurable Module)、专用 ASIC 部件、IO 接口等资源构成的异构并行处理系统。这样的系统能够灵活地融合多种不同的计算资源, 使一个系统中的某种(或某些)资源能够很大限度地满足某种(或某些)应用的模式和处理

要求, 达到较高的性能。系统中集成的一个或多个 RM 能实现硬件的功能改变, 各模块之间的互连关系可以通过配置和仲裁调节。这种可变性能使系统适应更大范围的应用需求, 向一体化、小型化、可搬移、高性能的方向发展。

1 相关工作

并行结构和多处理器技术是目前提高特定领域计算机系统处理能力的有效手段。可重构异构结构结合了并行处理和可重构计算两种技术, 相比于传统的“单处理器 + 单部件”的可重构系统来说, 异构系统中处理器和 RM 的类型和数量可能都比较多, 控制更加复杂。可编程器件是一类最常用的细粒度可重构模块, Splash 和 DECPeRLe-1 是较早的可重构处理系统, 应用于复杂位级计算, 具有细粒度、松耦合、静态配置等特征, 但其应用范围小、资源利用率不高。紧耦合结构把 RM 和微处理器更紧凑地连接在一起, 可有效提高数据传输带宽, 缓解配置压力。如 REMARC^[1]是面向多媒体应用的可重构协处理器, Morphosys^[2]系统是加速计算密集型应用的片上可重构系统。RM 和主处理器间的

收稿日期: 2008-04-28

基金项目: 国家自然科学基金(NSFC60736013)

作者简介: 姜晶菲(1974-), 女, 博士, 副教授, CCF 会员, 研究方向为计算机系统结构、高性能嵌入式计算机系统、高性能微处理器设计、密码处理。

连接带宽是影响系统性能的主要因素之一。

目前可重构系统的应用还限制在比较特定的核心算法上^[3],计算模式比较单一,常采用静态配置 RM 后,进行特定算法加速的方式。RM 与处理器的配合效果并不十分理想,系统效率常受到某些瓶颈的影响。RHS 可能包括具有不同粒度的 RM 和多个处理器资源,不同部件间数据和配置传输的效率将会更严重地影响系统性能的发挥,怎样通过有效的连接方式和重构方式平衡各部件的性能仍是需要重点研究的问题。

文中提出一种融合多个处理器模块、粗粒度 RM、细粒度 RM、全局存储模块的 RHS 结构,通过重构互连模块的管理,各个模块能够进行有效的数据通信和功能重构,从而实现针对不同算法的加速。

2 RHS 框架

RHS 框架如图 1 所示。该系统是由多个处理器模块(CPU0、CPU1...)、RM、存储模块、互连模块和 IO 接口构成的并行系统。处理器模块和 RM 是负责系统运行计算任务以及控制任务的处理结点,处理结点都带有各自的存储子系统(EEPROM, SRAM, SDRAM, flash 等),用于存储程序、计算数据、配置信息等;RM 分为专用 RM 和细粒度 RM 两种类型,专用 RM 以粒度为 8 的处理阵列构成,专用于加速具有规则计算模式的特定应用算法(如乘累加算法、对称密码算法等),细粒度 RM 指 FPGA 结构,可用于实现数字信号处理算法核、序列密码算法及其它随机逻辑电路。RM 的配置过程由处理器发起,配置数据可存于全局存储器、处理器的局部存储系统,或 RM 的局部存储器中;处理器通过配置指定 RM 功能后,可以向该模块传送命令,启动并完成一定数据集在 RM 上的快速处理,源数据和计算结果可传递于全局/处理器局部存储器与 RM 局部存储器之间。

全局存储模块提供全局可见的存储空间,在硬件上,所有的处理节点均可访问全部存储区域,而在软件上,所有处理器都把此存储区域映射为各自 IO 空间的一部分,某个处理器作为“主处理器”负责全局存储区域的划分和管理,解决数据冲突,并配合互连模块中的硬件逻辑来完成存储区保护。

各处理器模块均可对外提供基本的系统接口(如 I2C、串口等),用于调试监控等。RHS 对外提供统一的 IO 接口,由主处理器通过可重构桥接模块与外界进行通信。桥接模块实现了主处理器存储总线或 IO 总线(如 PCI)到特定 IO 接口的转接,因此该模块由进行总线协议转换的桥接部件及特定 IO 接口控制器两部分组成,这使得 RHS 通过转接提供不同 IO 接口,提高

了对外连接的适应性。

重构互连模块是整个 RHS 的核心。该模块负责把来自各个异构模块(处理器、RM、存储模块)的各类总线转换为一致的内部互连,以提供模块间高速的数据传输通道;它负责控制系统中所有 RM 配置数据的传输和具体的配置过程;它负责解析系统定义的各种配置命令和处理命令,进行冲突仲裁、资源分配和保护,并记录传输状态。

RHS 的重构可以在不同的级别上进行:在体系结构级,通过调整各个芯片之间的互连关系,可以针对特定应用要求更合理地分配资源,实现灵活的软硬件划分,形成可重构并行处理系统;在部件级,专用 RM 内部可以实现功能单元间静态或动态的可重构通路,以及单元内部的功能重构,形成粗粒度的可重构计算结构,可有效加速特定算法的执行;在逻辑级,对细粒度 RM 进行静态或动态配置,定制程序需要的特定功能,实现高度并行的功能单元,可灵活提高特定算法的执行速度。

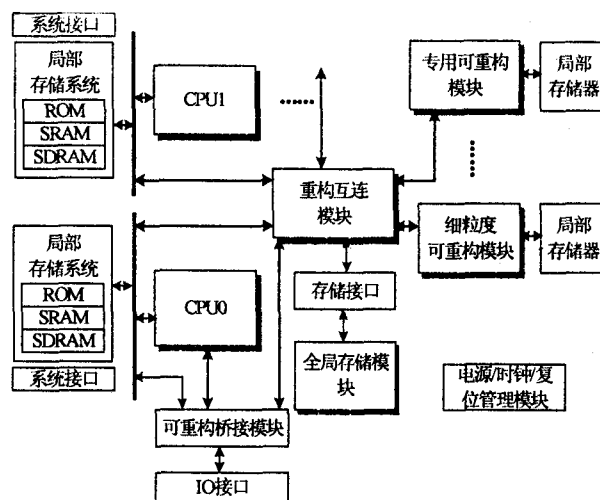


图 1 RHS 框架

3 重构互连控制

3.1 重构互连模块结构

图 2 显示了重构互连模块(Reconfiguration Connection Controller, RCC)结构。RCC 把发自各个模块的数据访问都看成是对存储器设备的访问,CPU、RM 可作为主设备发出访问请求,RM、全局存储器、被重构的 RM 可作为从设备接收访问请求。RCC 可连接的 CPU 均采用存储器映射 IO 方式,CPU 把其可访问的其它设备映射成 IO 空间的地址区域,访问请求由其存储总线传送给 RCC,经过对应接口逻辑的转换,变为 RCC 内部统一的存储总线形式(数据的数据 data、地址 addr、读写 rw、使能 cs 和就绪 rdy);CPU 的 IO 空间中

RCC可访问模块及相应控制寄存器的地址映射关系记录在RCC配置寄存器中,RCC可根据这种映射关系来解析存储总线命令,形成相应访问时序。RM是用于批量数据处理的计算模块,其内部的局部存储区用于存储源数据和计算结果;RM作为从设备被某CPU访问时,其内部存储区映射为此CPU的某块IO空间,RM也可作为主设备访问全局存储器。

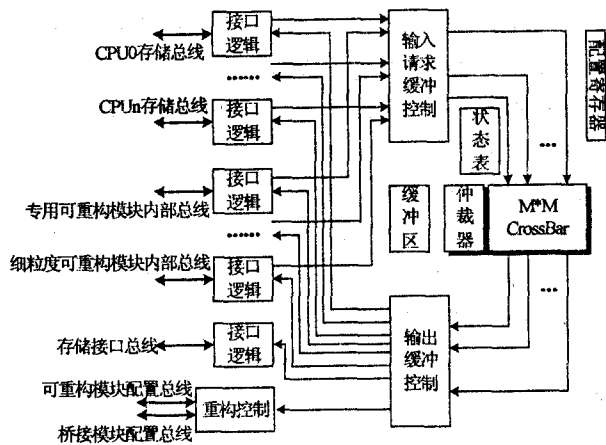


图2 重构互连模块(RCC)结构

全局存储器是可被CPU或RM访问的从设备,各个主设备模块都是通过把一定范围的全局存储区映射为自己的某部分IO空间来实现存储访问的,对全局存储区的映射关系(地址范围)保存在RCC映射寄存器中,这种映射关系可以通过具有管理权限的一个CPU(如CPU0)确定。当两个主设备把同一片全局区域映射为自己IO空间的一部分时,该片存储区就成了两个主设备的一片共享存储区,RCC负责检测并处理对共享存储区的访问冲突,而访问共享区的双方设备负责维护数据的一致性和通信关系。

CPU之间的通信在RHS中通常是少量控制信息的通信,由于大容量全局存储区通常用flash等构造,比较适合大量的页数据的传输,且CPU不能作为从设备来实时接收数据,所以利用其中的共享区来完成CPU间通信速度较慢、实时性不强。在RCC中设置小规模共享缓冲来保存CPU之间的通信数据,并设置一定的通信中断可解决这一问题。RCC负责接收源CPU的发送请求,保存在共享缓冲中,并根据中断允许状态向目标CPU发送通信中断。目标CPU可通过响应中断或查询方式获知通信请求,并主动读取共享缓冲数据。各CPU中也同样为共享缓冲分配各自的IO地址空间。

3.2 仲裁机制

由于CPU、RM模块数量可能较多,为折中仲裁效率和复杂性,RCC采用二级仲裁策略。RCC首先记录各个模块的忙闲状态,一级仲裁在发出访问请求的

CPU全集和RM全集中各选择 P 个CPU请求子集和 Q 个RM请求子集,并保存需要缓冲的、未被选择的请求;二级仲裁根据 $P+Q$ 子集中的目标地址查询映射寄存器,确定目标设备和目标区域,根据优先级寄存器中的优先级选择不产生冲突的 M 路请求,建立数据传输通道,完成数据传输。例如当 $P=2, Q=3$ 时,在图2中,经过仲裁RCC通过 3×3 的Crossbar可支持诸如CPUa \rightarrow RMa,CPUb \rightarrow RMb,RMc \rightarrow 全局flash共三路同时访问。

各模块最基本的请求方式是单个访存周期的存储访问,RCC一次仲裁只完成若干路请求各一个字数据的传输。对于批量传输的数据集,这种方式的仲裁开销过大,影响了数据交换带宽,因此,RCC中设置了仲裁粒度寄存器,用于记录某个模块一次申请仲裁的请求个数,各模块可通过一定的命令设置各自的仲裁粒度寄存器,当某模块获得RCC仲裁通路后,可连续服务次数等于该寄存器值的访存请求。

3.3 重构控制机制

具有管理权限的CPU可通过特殊访存序列对RM、桥接模块等进行功能重构,RCC负责把这样的访存序列转换成具体配置时序。RHS把专用RM的配置接口规范为普通的访存接口,对该类模块的配置与普通数据读写无异。FPGA具有专用的配置接口,RCC通过内部的配置管理逻辑及FPGA的SelectMAP^[4]接口来完成对其的在线重构。主要流程为:

- (1) RCC解析CPU访存请求,若为配置命令,则设置对应RM在RCC中的配置使能寄存器;
- (2) 根据CPU访存命令设置和清除配置状态寄存器中编程位,保持初始化时间间隔;
- (3) 读取配置初始化寄存器(由RM配置输出线置位),查询是否初始化完毕;
- (4) 当初始化完毕后,根据CPU访存命令按字节向RM配置空间写入配置数据;
- (5) 当配置结束后,根据CPU访存命令查询配置状态寄存器,清除配置使能寄存器。

图3显示了CPU、RM和RCC三种不同类型模块连接关系的两种实现结构。图3(a)示意了各种模块作为独立部件时连接的实现方式。图3(b)示意了把CPU、RCC逻辑和各类RM逻辑集成在统一的FPGA中实现的连接结构,利用FPGA的部分重构功能^[5],RCC可部分地改变FPGA中细粒度RM的逻辑功能,而使其它逻辑和接口处于正常工作状态,这样RCC可从同一FPGA的CPU逻辑中接收配置请求,再转而配置该FPGA中RM的功能,形成集成度更高的连接结构。

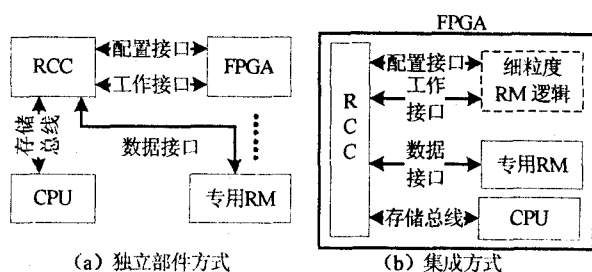


图 3 CPU, RM 和 RCC 连接关系的实现结构

4 关键技术验证

RHS 关键技术的模拟验证包括以下内容:

(1) RCC 逻辑用 Verilog 语言进行设计,最小能仲裁 3×3 路 CrossBar 的同时访问,通过设定数目参数其能支持可变模块数的互连访问;RCC 的 testbench 中构造了两个 CPU、两个 RM 及一个全局 flash 的访问模型,通过模拟验证了 RCC 仲裁功能的正确性。

(2) 利用 Xilinx 提供的 FFT、CODIC、LSFR 等 IP 核完成这些算法在 FPGA 上的加速。

(3) 采用流水化结构设计了一种专用 RM 结构, n 段功能单元能组成深度流水的处理结构,在其上映射数据密集型的 MAC(乘累加)算法,形成高速的流水处理结构。

(4) 利用部分重构设计流程生成包括 LEON3 微处理器核逻辑、IP 核算法处理单元在内的配置流,能完成可变 IP 核算法处理单元的控制和处理,进行了模拟验证。

(5) LEON3 CPU、其它 CPU 访问模型、RCC、专用 RM、IP 核算法处理单元的设计逻辑在由 Xilinx XC2VP50 大容量 FPGA、存储芯片等构造的实验平台上进行了实现验证;在采用 SPARC 微处理器^[6](与 LEON3 具有相同的接口)、CPLD 和 XC2V500 FPGA 构造的实验平台上实现了 IP 核算法处理单元和在线重构控制逻辑,验证了实现的可行性。

上述模拟或实现验证了 RHS 中的几项关键技术: RCC 结构及其实现、基于 FPGA 的 IP 核应用、在线重构控制实现及部分重构流程、RISC 结构的嵌入式处理器设计及应用、专用 RM 结构设计等。在相同的频率

下,实验同一算法(FFT)在 CPU 上软件实现的执行时间是使用 FPGA 硬件加速执行时间的 20 多倍,这说明在上述关键技术的支撑下,带有不同处理模块的 RHS 能为具有不同计算特点的应用提供高效的处理平台。

5 结束语

RHS 能融合多种不同的计算资源,使系统中的某些资源能够很大限度地满足某种应用的模式和处理要求。系统中 RM 硬件功能的在线重构特性能实现硬件的功能改变,RCC 能有效支持自身的配置和对其它模块的重构控制,可在一定的配置设置下完成各模块间的灵活互连,这种可变性使系统能适应更大范围的应用需求,向一体化和高性能的方向发展。下一步的工作将整合各种技术,实现完整的可重构异构原型系统,并进行实际应用的功能划分和算法映射。

参考文献:

- [1] Miyamori T, Olukotun K. A Quantitative Analysis of Reconfigurable Coprocessors for Multimedia Applications[C]//Proc. of IEEE Sym. on FCCM98. Napa, CA: [s. n.], 1998: 2 - 11.
- [2] Singh H, Lee M, Lu G, et al. MorphoSys: case study of a reconfigurable computing system targeting multimedia Applications[C]//Proc. Design Automation Conference. Los Angeles, California: [s. n.], 2000: 573 - 578.
- [3] Margerm S. Reconfigurable Computing in Real - World Applications[J]. FPGA and Structured ASIC Journal, 2006(5): 1 - 8.
- [4] Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode[EB/OL]. [2005 - 09 - 10]. <http://www.xilinx.com>.
- [5] Castillo J, Huerta P, López V, et al. A Secure Self - Reconfiguring Architecture based on Open - Source Hardware[C]//Proceedings of the 2005 International Conference on Reconfigurable Computing and FPGAs. Puebla City, Mexico: [s. n.], 2005: 10 - 17.
- [6] 32 - bit SPARC V8 Embedded Processor. SAILING S698, User's Manual[EB/OL]. [2005 - 10 - 23]. <http://www.orbita.com>.

(上接第 24 页)

- ing via boundary extraction for mining massive point.. data sets [C]//Proceeding of International conference on Geo-computation, 2000. Greenwich: University of Greenwich, 2000: 23 - 25.
- [4] 武晓波,王世新,肖春生. Delaunay 三角网的生成算法研究[J]. 测绘学报, 1999, 28(1): 28 - 35.
 - [5] 吴燕来,朱 莉. Delaunay 三角网生成算法的研究与实现

[J]. 计算机与信息技术, 2007(3): 21 - 22.

- [6] Tsai V J D. Delaunay Triangulations in TIN Creation: an Overview and a Linear - time Algorithm[J]. Int. J. of GIS. 1993, 7(6): 501 - 524.
- [7] Larkin B J. An ANSI C program to determine in expected linear time the vertices of the convex hull of a set of planar points [J]. Computer & Geosciences, 1997, 17(3): 431 - 443.