第 34 卷 第 5 期 计 算 机 工 程 2008 年 3 月
Vol.34 No.5 Computer Engineering March 2008
·博士论文· 文章编号：1000—3428(2008)05—0036—03 文献标识码：A 中图分类号：TP302

多处理器芯片组中交叉开关的设计与性能优化

方志斌[1,2,3]，安学军[1,3]，胡 鹏[1,2,3]
(1. 中国科学院计算技术研究所，北京 100080；2. 中国科学院研究生院，北京 100039；
3. 中国科学院计算机系统结构重点实验室，北京 100080)

摘 要：交叉开关是交换芯片和芯片组的核心逻辑。该文设计并实现了多处理器芯片组中的交叉开关，其工作频率在 FPGA 布局布线后可以达到 100 MHz。通过实践采样，对延迟和带宽进行测试，提出性能优化的策略，目前该交叉开关已稳定运行于龙芯 2E 多处理器系统中。
关键词：交叉开关；多处理器；芯片组

Design and Performance Optimization of Crossbar in Multi-processor Chipset

FANG Zhi-bin[1,2,3], AN Xue-jun[1,3], HU Peng[1,2,3]
(1. Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080; 2. Graduate School, Chinese Academy of Sciences, Beijing 100039; 3. Key Laboratory of Computer System and Architecture, Chinese Academy of Sciences, Beijing 100080)

【Abstract】 Crossbar is the core logic of switch and chipset. This paper designs and implements a crossbar in multi-processor chipset, and the frequency of the crossbar is up to 100 MHz after placing and routing on FPGA platform. The delay and bandwidth of the crossbar is tested based on implementation, and some optimization strategies of the crossbar are proposed. The crossbar runs in multi-processor system composed by Godson 2E CPU.
【Key words】 crossbar; multi-processor; chipset

为了顺应 CPU 多核的发展趋势，笔者在低成本高性能计算机的研究中将多个龙芯 2E 处理器通过芯片组直接互连，实现了全局统一编址。芯片组设计有 2 个系统总线端口，每个系统总线端口以总线形式连接 2 个处理器，每个处理器有独立的内存控制器直连内存，可以直接访问全部物理内存，PCI 和 LPC 端口实现了 I/O 功能，互联端口实现了芯片组间互连，利用交叉开关采用全互连方式实现了各端口间的数据交互。

1 交叉开关的设计和实现

芯片组内的交叉开关需要实现各个端口的通信，因此，需要先对其内部通信机制进行分析。

1.1 交叉开关的通信协议

芯片组的核心逻辑是一个交叉开关模块，交叉开关用于互联其他所有接口，每个内部互连接口包含一个 Master 接口和一个 Slave 接口。Master 接口接收来自处理器、外设或互联接口的外部请求，按照地址总线指定的地址窗口向相应的接收方 Slave 接口转发此操作请求，接收方 Slave 接口设置仲裁器，只有获得仲裁响应的 Master 请求才能在此 Slave 接口上输出。

多处理器芯片组交叉开关通信协议采用切入通信机制，地址线与数据线分离，数据线 64 位。内部总线协议分成 Slave 和 Master 两部分，通过交叉开关实现点对点通信，Master 用于传送数据，是总线交互的发起者，通过 CYC 信号有效标志一次传送请求的开始。Slave 用于接收数据，是总线交互的接收者，Slave 看到 CYC 有效后，如果当前缓冲区容量大于或等于 Master 请求传输的数据量，就向 Master 发出 ACK 信号，反之则不发出 ACK 信号。Master 看到 ACK 信号后把数据传送给 Slave，数据是否有效依靠 STB 指示，数据有效期间内 STB 为高电平。CYC 信号有效时 CMD 指示了本次传送的固定长度，具体时序如图 1 所示。

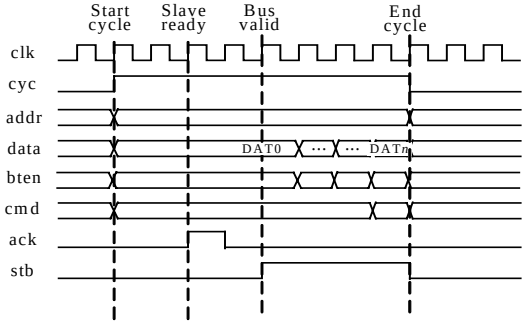

图 1 交叉开关通信协议时序图

1.2 交叉开关的设计

交叉开关的设计包括流控、调度和路由机制[1]，龙芯 2E 多处理器芯片组交叉开关采用分布式流控、轮循调度和全局地址路由。

交叉开关的流控机制指接收缓冲区判断能否接纳发送方请求传输数据量的机制，可以采用请求方比较和接收方比较 2 种方式实现。在请求方比较的机制中，交叉开关设置全局

基金项目：中国科学院创新课题基金资助项目“新一代机群关键技术研究”(20054010)；中国科学院计算技术研究所创新课题基金资助项目“基于龙芯的 SMP 芯片组及其相关系统的研究”(20046080)
作者简介：方志斌(1973－)，男，博士研究生，主研方向：计算机体系结构；安学军，副研究员、博士；胡 鹏，博士研究生
收稿日期：2007-04-28 **E-mail：**fzb@ncic.ac.cn

可访问寄存器指示各接收缓冲区的空闲容量，每个端口在发送请求前比较缓冲区空闲容量以决定是否发送请求，本文将其称为集中式流控机制。在接收方比较的机制中，交叉开关没有全局可访问寄存器指示各接收缓冲区空闲容量，发送请求在仲裁得到与接收方建立连接的许可后，由接收方比较发送请求数据量和本地缓冲区空闲容量以决定是否接收发送请求，本文将其称为分布式流控机制[2]。

随着半导体工艺进入深亚微米，线延迟超过门延迟，对全局寄存器的访问已经成为微体系结构设计的首要约束，因此，交叉开关集中式流控机制面临线延迟的挑战。在分布式流控机制中请求方在发送请求前无法判断其能否被接收，接收缓冲区一旦饱和，将引起无效重试，降低交叉开关物理通道的利用率，这种状况在长消息传输和集合通信中更明显。

交叉开关的调度机制包括随机和轮循等方式[3]。当多个指向同一接收方的请求被仲裁，其中只有一个请求被容许与接收方连接后，其余请求可以采取 2 种方式处理：(1)该请求一直有效，但此方式将使请求方一直占用交叉开关请求线，无法利用请求方虚通道的优点。在传输长消息的通信中无效等待时间增加，交叉开关利用率将进一步降低；(2)撤销该请求，将其延迟等待若干周期后重试。由于第(1)种方式效率低下，芯片组交叉开关将基于第(2)种方式进行，即请求仲裁后无法与接收方连接，将撤销该请求进行重试。

交叉开关的路由机制包括表路由和全局地址路由。表路由指所有路由信息存放在表中，交叉开关需要对请求方的目的地通过查表以确定接收端口；全局地址路由指按全局地址进行路由，交叉开关内部设置各端口的地址空间，对请求方的地址进行解析并确定接收端口。

1.3 交叉开关的实现

龙芯 2E 多处理器芯片组交叉开关使用 Verilog HDL 硬件描述语言实现，在 Xilinx 环境中开发，采用 Xilinx 公司的 Virtex 4 系列 XC4VLX60 芯片在项目组设计的 PCB 上进行验证，交叉开关在布局布线后的工作频率可达到 100 MHz。

2 交叉开关的测试

利用 Xilinx 公司的 Chipscope 调试工具，在项目组实现的 PCB 上进行采样得到交叉开关的延迟参数，并利用系统软件测试交叉开关的通信带宽。笔者开发了虚拟外部设备以及虚拟内部总线接口，虚拟内部总线接口按照交叉开关通信协议和内部实现方法设计，该仿真环境用于测试交叉开关重试机制和各种优化策略。

2.1 交叉开关的延迟测试

利用 Chipscope 调试工具测得芯片组内部各通道的时序周期如表 1 所示。

表 1 交叉开关各通道时序周期表

通道类型	时序周期
交叉开关 Master 从接收请求方 cyc 到转发给交叉开关 Slave	5
交叉开关 Slave 从接收请求 cyc 到接收到应答方 ack	1
交叉开关 Slave 将应答 ack 转发回交叉开关 Master	1
交叉开关 Master 从接收 ack 到接收到请求方 stb	1
交叉开关 Master 将应答 stb 转发给交叉开关 Slave	2
单个 Cache 块传输 Stb 周期	4
8 个 Cache 块传输 Stb 周期	32

由表 1 可知，处理器总线单个 Cache 块的传输跨越交叉开关需要经过 14 个周期，延迟为 140 ns；8 个 Cache 块的传输跨越交叉开关需要经过 42 个周期，延迟为 420 ns。

2.2 交叉开关的带宽利用率分析

经过测试，交叉开关传输单个 Cache 块的带宽为 267 MB/s；传输 8 个 Cache 块的带宽为 640 MB/s。在原型系统实现中，Master 从转发请求方 cyc 到接收到请求方 stb 需要经历 8 个周期。在用路由表实现芯片组内部路由的情况下，其周期比用全局地址空间实现增加 2 个，这使交叉开关 Master 从转发请求方 cyc 到接收到请求方 stb 要经历 10 个周期。在交叉开关用全局地址路由实现的原型系统和表路由实现的 2 种情况下，对一次交易传输不同长度数据的带宽利用率进行比较，结果如图 2 所示。

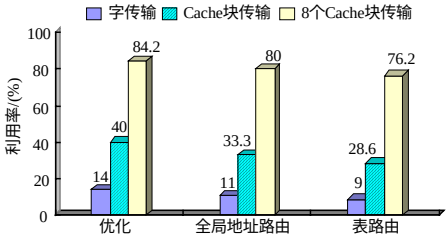

图 2 交叉开关带宽利用率分析

由图 2 可以看出，芯片组在全局地址路由实现的原型系统中传输 8 个 Cache 块数据时带宽利用率达到 80%，而传输字数据时带宽利用率仅达到 11%，这表明芯片组比较适合于大块数据通信的 MPI 应用。在 8 个 Cache 块数据传输时，利用全局地址实现的路由比表路由的带宽利用率提高 3.8%，体现了全局地址空间的优势。

2.3 交叉开关的重试机制测试

交叉开关在请求无法满足时延迟等待若干周期后重试，其采用重试机制可以减小重试请求数目，降低交叉开关无效的仲裁、连接和断开数目，提高其发送端口的性能。图 3 显示了交叉开关在重试延迟周期为 1 时传输不同长度数据时的重试数目测试结果。由图 3 看出，在传输小块数据时不存在重试数的负面效应，但传输大块数据时重试数明显提高，其负面效应加大。同时当基本交易传输数据从 32 B 增加到 256 B 时，即内部通信带宽提高的条件下，平均重试数目增大。

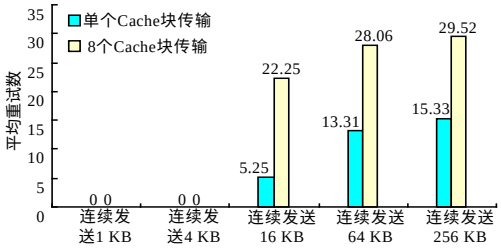

图 3 不同交易长度下传输 1 KB 的平均重试数

图 4 显示了交叉开关传输 64 KB 长度数据时在不同重试延迟周期下重试数目测试结果。

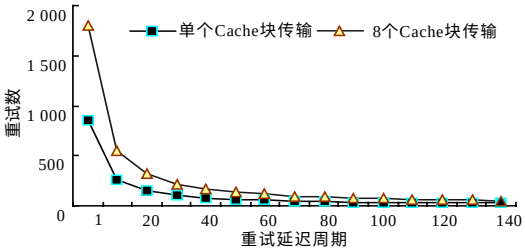

图 4 连续发送 64 KB 数据时的延迟和重试数目

可以看出，在 2 种不同内部通信带宽配置下，在等待延迟周期从 1 周期调整到 40 周期的范围内重试数急剧下降，而

从 40 周期调整到 140 周期的范围内重试数目变化不大。

3 交叉开关性能优化

在设计和测试的基础上，本文提出了交叉开关性能优化的策略。

3.1 交叉开关时序优化

通过对原型系统的 8 个周期进行分析，在交叉开关 Master 从接收请求到转发给交叉开关 Slave 的 5 个周期中，数据通路有 2 周期的延迟，有 1 个周期用于地址比较实现芯片组内部路由，有 2 周期用于交叉开关仲裁。据此可以对交叉开关优化，将数据通路延迟减小 1 个周期，同时消除 Slave 将应答 ack 转发给交叉开关 Master 的 1 个周期。因此，在优化情况下，交叉开关 Master 从转发请求方 cyc 到收到请求方 stb 只需要 6 个周期。

对交叉开关在原型系统实现的全局地址路由和理论优化 2 种情况下，一次交易传输不同长度数据的带宽利用率进行比较，结果如图 2。优化情况下比目前的实现带宽利用率提高 4.2%，这表明当前实现的芯片组仍有优化的潜力。

3.2 交叉开关重试机制的优化

在交叉开关的通信中，存在 2 类重试：(1)由调度机制引起；(2)由分布式流控机制引起。为减小这 2 类无效重试，芯片组为每一信道设置一重试延迟等待寄存器，该寄存器由接收方通过交叉开关内部独立的数据线完成设置，当请求方的请求被撤销时可按该寄存器等待若干周期后进行重试交易。

在调度产生重试的情况下，接收方仲裁器将根据当前所有请求的优先级对各请求反馈回不同的重试延迟周期，并通过交叉开关内部独立的数据线完成设置对应的寄存器，请求方以此寄存器进行重试将使每次重试请求均有效。

在分布式流控机制中，由图 4 可看出重试延迟等待周期与内部通信基本交易传输长度有关，当基本交易传输长度低时所需延迟周期小，反之延迟周期大。在接收方缓冲区饱和时，接收方将根据当前请求的基本交易传输长度对各请求反馈回不同的重试延迟周期，并通过交叉开关内部独立的数据线完成设置对应的寄存器。经过测试，在基本交易为单个 Cache 块长度下，连续发送 64 KB 数据时，请求方以 50 周期延迟进行优化重试将使第 1 次请求的成功率从 5.6%提高到 89.3%。

3.3 交叉开关流控机制的优化

交叉开关分布式流控机制避免了线延迟所带来的负面效应，但其时序不优化，而且当传输大容量数据时存在重试，重试即使采用优化也不能提高到 100%。交叉开关集中式流控机制实现时需要所有端口能对该缓冲区操作，使芯片组提高频率受到很大限制；同时全局寄存器输出需要跨越所有通道，在整个芯片中成为长线，其线延迟限制了芯片组的频率。

本文提出了交叉开关的流控优化策略，即每一接收缓冲区在所有信道请求方设置一对应寄存器。接收方利用交叉开关内部独立的控制线设置该寄存器，当接收方缓冲区容量发生变化时，如果空闲容量高于某一阈值时将所有请求方对应的寄存器设置为 1，此时采用集中式流控机制，所有请求默认请求传输数据量能被接收缓冲区容纳，只须获得仲裁即可建立与接收方的连接，忽略 RTY 流控信号；如果该空闲容量小于该阈值时，将所有请求方对应的寄存器设置为 0，此时采用分布式流控机制，如果接收 RTY 流控信号则按 3.2 节中的重试机制优化策略进行重试。阈值的确定按接收缓冲区能容纳所有端口的请求为基准。当寄存器值变化时，接收方仲裁器使所有请求无效，使其按新寄存器所指定的新方式通信。

此优化策略避免了交叉开关集中式流控机制中全局寄存器线延迟所带来的频率提升困难，当信道缓冲区空闲时使交叉开关延迟减小，当信道缓冲区满时通过设置不同的重试等待延迟周期减缓交叉开关的通信压力。

4 结束语

本文设计了多处理器芯片组的交叉开关，随着芯片组内部接口的增加，交叉开关将面临更大的通信性能压力，未来可以采用分级交叉开关实现更大规模的互联接口扩展。

参考文献

[1] Culler D, Singh J P, Gupta A. Parallel Computer Architecture：A Hardware/Software Approach[M]. [S. l.]: Morgan Kaufmann Press, 1998.

[2] Xilinx Corporation. High-speed Buffered Crossbar Switch Design Using Virtex-EM Devices, XAPP240[Z]. (2000-03-14). http://direct.xilinx.com/bvdocs/appnotes/xapp240.pdf.

[3] Gupta P, McKeown N. Design and Implementation of a Fast Crossbar Scheduler[J]. IEEE Micro Magazine, 1999, 19(1): 20-28.

（上接第 21 页）

算法所检测到。经实验验证，本文算法可检测到宽度小于 2 mm 的裂缝；经过伪裂缝消除的处理后，检测到的裂缝信息准确可靠，能够真实地反映裂缝的形态特征。

6 结束语

本文设计了一种基于三维地形模型的路面裂缝自动检测算法，该算法准确地把握了裂缝的谷状纹理特性及大邻域粗尺度的方向一致性 2 大基本特征，对裂缝进行精细的局部结构分析，克服了常规算法易受噪声、杂物以及光照条件等干扰的问题，对不同状况路面中的裂缝均能取得良好的检测效果，是一种高效实用的图像弱信息处理算法。该算法可满足当前大部分标准化路面的裂缝自动检测的需求，为后期精确地评价路面质量并提供相应的保养措施提供了可靠的依据。

从本文的实验结果可以看到，检测到的裂缝中出现了多处断裂的情况，这一方面是由于裂缝本身存在断裂，另一方面是由于算法仍然存在局限性，对噪声环境中过于微弱的裂缝信号无法检测到。如何根据裂缝的走向进行裂缝断裂的修补并采用合适的数据结构对裂缝进行准确的描述将是下阶段的研究方向。

参考文献

[1] 张洪光, 王 祁, 魏 玮. 基于人工种群的路面裂纹检测[J]. 南京理工大学学报, 2005, 29(4): 389-393.

[2] Sinha S K, Fieguth P W. Automated Detection of Cracks in Buried Concrete Pipe Images[J]. Automation in Construction, 2006, 15(11): 58-72.

[3] Lang V, Belyaev A G, Bogaevsici I A, et al. Fast Algorithms for Ridge Detection[C]//Proceedings of the International Conference on Shape Modeling and Applications. Aizu-Wakamatsu, Japan: [s. n.], 1997: 189-197.

[4] Perona P, Malik J. Scale-space and Edge Detection Using Anisotropic Diffusion[J]. IEEE Transaction on Pattern Analysis and Machine Intelligence, 1990, 12(7): 629-639.