

GPU 虚拟化环境下的数据通信策略研究

张玉洁, 吕相文, 张云洲

(南京航空航天大学 计算机科学与技术学院, 江苏 南京 210016)

摘要:虚拟化技术能够以较低的成本和能源消耗共享有效的资源,一些应用程序往往需要利用图形处理器来加快它们的计算以提高性能。但是由于虚拟化本身的特点,在 GPU 虚拟化环境下进行 CUDA 应用开发会带来很大的性能开销。此外,当采用多 GPU 并行处理大规模的程序时,传统的 GPU 之间的数据交互方式是通过 CPU 来中转,不仅会带来“路程”上的开销,同时 PCI-E 相对于 GPU 显存的低带宽更是限制了数据传输的速率。针对以上问题,文中在 Xen 和 VMware 虚拟化平台下,针对 CUDA 应用的延迟和吞吐率找出最优的虚拟机间通讯方式,针对 GPU 之间不同的数据传输方式,找出最优通信方案,并从理论上和实验中分析出影响多 GPU 协同运算效率的因素。

关键词:GPU 通用计算;虚拟化;CUDA;数据通信

中图分类号:TP31

文献标识码:A

文章编号:1673-629X(2015)08-0024-05

doi:10.3969/j.issn.1673-629X.2015.08.005

Research on Data Communication Strategy under GPU Virtualization

ZHANG Yu-jie, LÜ Xiang-wen, ZHANG Yun-zhou

(College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics,
Nanjing 210016, China)

Abstract: Virtualization, as a technology that enables easy and effective resource sharing with a low cost and energy footprint, applications with stringent performance often need to make use of graphics processors for accelerating their computations. Due to the characteristics of virtualization itself, it will be brought significant performance overhead when doing CUDA application development under GPU virtualization environments. Besides, it will be studied when processing a large-scale data by using multi-GPU in parallel way. The traditional way of data exchange between the GPUs is transferring by the CPU. Thus it not only will bring the "walk" cost, but also limit the bandwidth of the data transfer rate by using PCI-E with respect to the GPU bandwidth. To solve the above problems, try to identify the optimal communication between virtual machines under Xen and VMware virtualization platform for CUDA applications in this paper. For different data transmission between the GPUs, try to find the optimal communication scheme, and to analyze the factors that affect the efficiency of multi-GPU collaborative computing through theory and experiment.

Key words: general-purpose computation on GPU; virtualization; CUDA; data communication

0 引言

虚拟化技术对物理资源进行逻辑抽象和统一表示,降低了资源管理的复杂度,提高了物理资源的利用率和运营效率,从而有效地降低了运营成本^[1]。同时,现代 GPU 的计算能力比同时代 CPU 的计算能力胜出几个数量级,因而商业界、学术界纷纷致力于 GPU 虚拟化技术的开发^[2]。

然而在目前的虚拟化环境下并没有一个通用且高效率的 RPC 系统,传统的如 XMLRPC^[3]、CORBA^[4]、

ICE^[5]等应用则只针对开放的网络或分布式环境,当把它们移植到虚拟化环境中则会在系统吞吐率、延迟以及 CPU 占用方面出现很大的劣势,基本无法应用于实际项目。2006 年, NVIDIA 公司统一架构 CUDA^[6] (Compute Unified Device Architecture) 的诞生,引发了 GPU 通用计算领域 (General Purpose computing on Graphics Processing Units, GPGPU) 的迅猛发展,大大简化了 GPU 的编程环境。当采用多 GPU 并行处理大规模的数据时,数据通信的选择问题,已经成为在虚拟

收稿日期:2014-09-06

修回日期:2014-12-09

网络出版时间:2015-07-21

基金项目:国家“863”高技术发展计划项目(2009AA044601);国家自然科学基金重点项目(61139002);江苏高校优势学科建设工程资助项目;南京航空航天大学基本科研业务费专项科研项目(NP2013308)

作者简介:张玉洁(1991-),女,硕士,研究方向为虚拟化、高性能计算。

网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20150721.1439.026.html

化环境下进行 CUDA 应用开发不可越过的难题。

为了提高在虚拟化环境下多 GPU 协同计算的性能,文中首先在系统层分析由于虚拟化带来性能影响的主要因素,通过对比同一虚拟化平台下的不同数据通信策略,得出特定虚拟化平台的最佳通信方式。在应用层方面,通过研究多 GPU 卡的数据传输机制,得到影响多 GPU 协同运算效率的主要因素。

1 虚拟机域间通信策略

1.1 基于 Xen 的虚拟机通信方式

Xen 平台下虚拟机域间通信优化的解决方案主要有三种,分别是 XenSocket^[7]、XWAY^[8] 和 XenLoop^[9]。

XenSocket 是在 Xen 中基于 socket 的解决方案,旨在提高虚拟机域间通信的吞吐率。XenSocket 的 API 采用标准的 socket API 接口,在 socket 的接口下,它使用 Xen 提供的共享内存来实现虚拟机之间高速的数据传输^[10]。

XWAY 使得在同一物理机不同虚拟机之间运行的程序可以通过标准的 socket API 来进行通信,不仅实现了域间通信的高带宽、低延迟,同时也充分地保证了对二进制的兼容性,使任何基于 socket API 的网络程序都可以直接享用 XWAY 带来的高速域间通信的体验^[11]。

XenLoop 是一个 Xen 平台下另一虚拟机域间通信方案,该方案是一个完全透明的、高性能的虚拟机域间的网络环回通道。XenLoop 可使同一物理机不同虚拟机之间直接进行数据通信而不需要第三方软件的参与,同时可以不牺牲应用层的透明性,应用程序不用作任何修改便可以无缝地运行。XenLoop 还实现了在 Xen 虚拟化平台下虚拟机域间的高速通信^[12]。

1.2 VMware

VMware 开发了一套相应的虚拟机通信接口 VMCI (Virtual Machine Communication Interface)。VMCI^[13] 是一个作用在应用层的方案,提供在 VMware 系列虚

拟机平台下的快速、高效的域间通信通道,使得客户端虚拟机与客户端虚拟机、客户端虚拟机与物理机之间能够实现高速通信。VMCI 实现两类接口,一类是数据报 API,一类是共享内存 API。VMCI 属于 VMware 的商业产品,内部实现细节目前并未披露^[14]。由于目前并不存在适用于虚拟机中的显卡驱动,文中通过在 ESXi 中某一虚拟机上使用 PCI pass-through (也即 VMware 的设备直通) 技术,使得该虚拟机获得物理机中 GPU 的访问权。

2 GPU 内部数据传输

随着 2011 年 CUDA 4.0 由 NVIDIA 作为一个全新版本发布后,其功能特性大幅度增加,主要涉及应用程序移植的简化、多 GPU 编程的加速、开发工具的增加和改进三个方面。在多 GPU 通信方面,在 CUDA 4.0 之前,多个 GPU 在同一节点内相互访问,需要首先将 GPU 中的数据传输给 CPU,再通过 CPU 的中转完成 GPU 间的数据交互。而在 CUDA 4.0 之后,NVIDIA 实现了不同 GPU 可以直接进行传输,且传输可以一次完成。另外通过 NVIDIA 发布的 GPU Direct 2.0^[15] 技术使一台服务器内多个 GPU 之间直接点对点通信,不仅绕开了主机端的参与,同时让多 GPU 编程更加轻松且传输效率大大增强。

2.1 统一虚拟地址空间

统一虚拟地址空间 (Unified Virtual Addressing, UVA) 模型为主机与系统内所有的设备提供了单一地址空间^[16]。这一模型将 CUDA 中包括 CPU 以及 GPU 在内的所有指令执行过程放入同一个地址空间。这之前每个 GPU 和 CPU 都使用它们自己的虚拟地址空间,从而导致许多额外的开销,UVA 的引入为 GPU 之间的点对点数据传输提供了逻辑支持,使得开发者体验到更方便的内存管理。

图 1 为多存储器模型和统一地址空间模型的结构图。

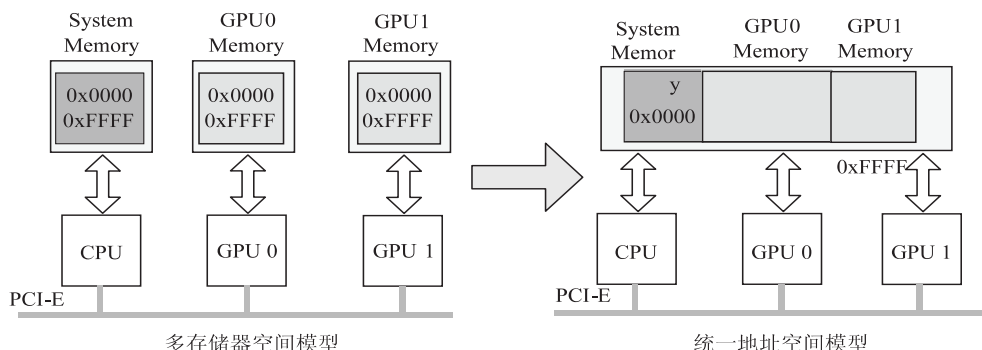


图 1 多存储器空间模型和统一地址空间模型

2.2 CPU 中转传输

在 CUDA 3.2 及其之前的版本中,多 GPU 的设备

存储器和主机端的内存被视为独立的存储器块,各自拥有独立的地址空间。GPU 之间如果需要进行通信

的话则必须首先通过将数据从 GPU 端拷贝到主机端内存,再由主机端内存传输至目标 GPU 中的显存。

2.3 GPU 点对点传输

在 CUDA 4.0 之前的多 GPU 程序开发中,每个 GPU 被看成是独立的核心。而在 CUDA 4.0 中,每颗核心及其具备的各种存储资源被当作一个整体的工作网络中的节点,各个节点之间的通信和同步是对等的(形成了一个共享式内存的 SMP 网络)。host 端的内存资源和 GPU 上的设备存储资源被当作一块统一的存储器池,存储器地址统一编码,多 GPU 之间可以通过 PCI-E 总线直接进行通信,而不再需要通过内存进行中转。

2.4 结果预测

虽然 GPU 之间的直接数据传输绕开了主机端内存的中转,但由于基于 Fermi 架构的显卡没有对 GPU 之间的数据传输提供直接的硬件支持,在数据传输时仍然需要将数据经由总线,其实质是少了一次经由 PCI-E 总线的数据传输的时间。因此预测随着计算规模的增大,G_G(GPU 点对点传输)相对于 G_H(CPU 中转传输)的传输方式的计算时间差应该有所提升,提升的速度应该基本呈线性增长的趋势,其时间差应与数据的规模成正比,与总线带宽成反比。

以矩阵复合运算 $A * B + C * D$ 为例,在一定的数据规模下,设 GPU 的矩阵计算时间为 T_0 ,将四个矩阵拷贝至 4 个 GPU 的总时间为 T_1 ,将计算结果拷贝回 CPU 端的时间为 T_2 ,单 GPU 对中间结果的单向数据传输时间为 ΔT ,则 G_G 的理论计算时间为 $T_0 + T_1 + T_2 + 3\Delta T$,G_H 的理论计算时间为 $T_0 + T_1 + T_2 + 8\Delta T$,时间差为 $5\Delta T$ 。

这是因为,G_H 采用的是 4 块 GPU 全部将数据传输至 CPU,再从 CPU 中转至其中的一块 GPU(如 GPU 0),总计算时间为 $T_0 + T_1 + T_2 + 8\Delta T$ 。而 G_G 则不然,这是因为 GPU 0 在计算加法运算之前需要进行 $(A/2) * B$,而计算结果并不需要进行传输,直接驻留在 GPU 0 中,因此中间结果不仅绕开了主机端的参与,同时也省略了 $(A/2) * B$ 的两次数据传输。文中设计的矩阵的数据类型是单精度的浮点数 float,如果在 GPU 中分配和 $C/2$ 与 D 同样大小的数据,采用 CUDA 4.0 SDK 中的 simpleP2P 样例来测试 GPU 之间的数据带宽(实际上就是 PCI-E 的实际带宽),则数据总量与 PCI-E 总线实际带宽的商 ΔT_1 应该与 ΔT 在理论上相等。

3 实验结果与性能分析

3.1 实验环境

实验所用平台的参数如表 1 所示。

表 1 实验参数

实验环境	参数
操作系统	Ubuntu 12.04
虚拟化平台	Xen 4.0.1, ESXi 5.0
GPU 平台	NVIDIA Tesla C2050 *4
CPU 平台	Intel Xeon E7-4830
GPU 编译器	nvcc
CPU 编译器	gcc 4.4.7, g++ 4.4.7
CUDA 版本	CUDA 3.2.16, CUDA 4.

3.2 域间通信方式选择

本节从通信策略支持的通信方向、对标准协议(TCP、UDP)的支持、延迟以及对用户的透明性等角度来分析在基于 Xen 的虚拟化环境下不同的通信策略对 CUDA 应用的性能影响。

尽管 XenSocket 和 XWAY 能够在基于 Xen 的虚拟化平台下提供高效率的域间通信,但是这些通信机制都存在某方面的不足。从客户端和服务的通信方向、对标准协议的支持、TCP 提交的延迟(单位为 TCP 报文的往返时间)以及对用户的透明性三种通信方式进行比较,如表 2 所示。

表 2 Xen 通信方式总结

通信方式	通信方向	标准协议支持	延迟	透明性
XenSocket	半双工	不支持	较长	否
XWAY	全双工	只支持 TCP	较长	是
XenLoop	全双工	TCP 和 UDP	很短	是

从表 2 可见,XenSocket 的功能比较简单,只支持单向通信,而传统的 socket 通信和 RPC 都是双向的,这就使得其无法充分利用 RPC 机制的透明性。由于它的立足点在于增加虚拟机域间的传输带宽,对标准的 TCP 和 UDP 协议的支持也不完善。同时,XenSocket 虽然在数据量少(一般在 16 kB 以下)时的单向吞吐量是 XenLoop 的两倍多,但由于其采用固定的缓冲区大小使得其无法满足更大数据量的传输需求。

XWAY 通过拦截 socket 底层的 TCP 调用来为面向 TCP 连接的应用提供透明的虚拟机域间数据传输。但它同样存在诸多不足,如它需要对网络协议栈进行修改、不支持 UDP 协议、不支持套接字的在线迁移,同时,XWAY 也未对共享内存的安全性进行考虑。

XenLoop 方案对用户和系统都是完全透明的,提供了高性能的虚拟机域间网络回环通道。它支持 TCP 和 UDP 两种协议,同时利用 Xen 提供的授权表机制在数据通道的两端建立共享缓冲,实现全双工通信,并实现了对共享区的同步机制和安全机制。文中将在 Xen 平台下使用 XenLoop 加速虚拟间的通信。

由于 VMware 的闭源性,在学术界很少有基于 VMware 虚拟化平台的通信解决方案。但 VMware 公

司开发了一套相应的虚拟机通信接口 VMCI,它是一个工作在应用层的解决方案,提供在 VMware 系列虚拟机平台(Workstation、ESXi 等)下的快速、高效的域间通信通道,使得客户虚拟机与客户虚拟机、客户虚拟机与宿主虚拟机之间能够高效通信。文中在 VMware 虚拟化系列平台下均采用 VMCI 进行域间加速。

3.3 GPU 内部数据传输

由于受计算能力和内存容量方面的限制,很多并行程序对数据进行拆分并通过数据交换来实现原来程序的功能。文中通过矩阵复合运算 $A * B + C * D$ 来完成整个实验数据的验证。

其中, $A * B$ 分别由 device 0 和 device 1 完成, $C * D$ 分别由 device 2 和 device 3 来完成,乘法完成后通过两种方式分别将计算结果传递给 device 0,由于矩阵加法的复杂度较低,所以在最后所有的加法运算全部交由一块 GPU 来完成。本节给出了 4 个 GPU 在两种不同数据传输方式下的性能比较。表 3 所示为 G_H 和 G_G 随着矩阵规模逐渐增大后的统计特征对比,统计特征包括 kernel 函数的迭代次数、实验方差。其中, kernel 执行次数即为在主函数中执行 kernel 的次数,也即迭代次数;方差则表示若干次迭代次数的时间的统计方差,单位为 s^2 。

表 3 两种不同数据传输方式的统计特征

阶数	kernel 执行 次数(G_H)	kernel 执行 次数(G_G)	方差 (G_H)	方差 (G_G)
1 000	100	100	0.000 020	0.000 012
2 000	100	100	0.000 037	0.000 049
3 000	50	50	0.000 053	0.000 066
4 000	50	50	0.000 194	0.000 099
5 000	25	25	0.000 849	0.000 622
6 000	25	25	0.001 547	0.001 124
7 000	10	10	0.008 421	0.006 429
8 000	5	5	0.011 462	0.009 456

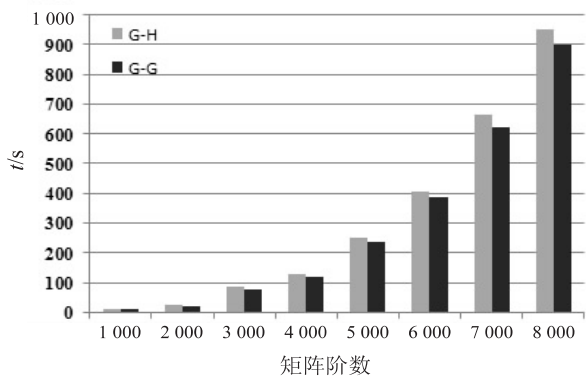


图 2 不同数据传输方式的时间对比

图 2 所示为通过两种数据传输方式的时间对比,横轴是矩阵的阶数,纵轴是执行的时间。为了使最后

统计的数据统一,图中的执行时间均为将迭代次数换算至 100 次后的时间,同时,矩阵复合运算的执行时间是输入数据初始化完毕后开始计时,数据从 GPU 端拷回主机端结束计时,因而未将由于虚拟化带来的性能开销(即访问延迟)考虑在内。

从图中可以看出,随着矩阵规模的增加,G_H 相对于 G_G 的执行时间差也越来越大,变化速度基本呈线性增长的态势。由 2.4 节的理论分析知,这是由于 G_G 相对于 G_H 少了 $5\Delta T$,其中 ΔT 为单 GPU 对中间结果的单向数据传输时间。G_G 相对于 G_H 不仅少了中间结果在总线上一次数据传输,同时 GPU 0 在计算加法运算之前进行 $(A/2) * B$ 的计算结果并不需要进行传输,直接驻留在 GPU 0 中,因此中间结果不仅绕开了主机端的参与,同时也省略了 $(A/2) * B$ 的两次数据传输。

表 4 为实际测量的时间差与理论计算值对比情况。其中,理论值为单个 GPU 传输 $(C/2) * D$ 的结果乘以系数 5 后的结果;测量值为在主机端生成同理论值相同数据量的数据将其传输至 GPU 中所花费的时间乘以系数 5 后的结果;实际值即如图 4 中的 G_H 和 G_G 的时间差;实际-理论差即为理论计算值与 G_H 和 G_G 时间差的差值;同理,测量实际差为“等价”的测量值与实际的时间差的差值。

表 4 理论值、测量值、实际值对比 s

阶数	理论值	测量值	实际值	实际-理论差	测量实际差
1 000	0.72	0.74	0.74	0.02	0
2 000	2.93	2.95	2.96	0.03	0.01
3 000	6.67	6.71	6.71	0.04	0
4 000	11.80	11.83	11.84	0.04	0.01
5 000	18.57	18.60	18.62	0.05	0.02
6 000	26.77	26.81	26.82	0.05	0.01
7 000	36.45	36.51	36.51	0.06	0
8 000	47.62	47.66	47.68	0.06	0.02

由表 4 不难看出,理论值、测量值与实际值基本相符,同时随着矩阵的阶数不断增加,实际理论差有着小幅度的增加,但都在允许的测量误差范围之内。测量实际差基本上为 0,这是由于测量的数据和实际传输的数据在格式和规模上完全一致,只是简化了数据传输的过程。图 2 和表 4 验证了 2.4 节结果预测的正确性。

4 结束语

文中对现有的虚拟机域间通信的优化方案进行了详细介绍,通过对比几种优化方案,在 Xen 平台下采用 XenLoop 域间通信方式为 GPU 虚拟化环境下的最优通

信方式,在 VMware 虚拟化平台下使用 VMCI 的通讯方式。在单节点多 GPU 内部通信方面,设计了两种方式(G_G 和 G_H)完成 GPU 间的数据传输,以矩阵复合运算为应用场景设计了实验,结果验证了预测结果的正确性。将来的工作是在集群环境下对 GPU 虚拟化的数据传输问题的分析。

参考文献:

- [1] 崔泽永,赵会群. 基于 KVM 的虚拟化研究及应用[J]. 计算机技术与发展,2011,21(6):108-111.
- [2] Overby E. Process virtualization theory and the impact of information technology[J]. Organization Science,2008,19(2):277-291.
- [3] Lambert D, Domingue J. Photorealistic semantic web service groundings: unifying RESTful and XML-RPC groundings using rules, with an application to Flickr[C]//Proc of the 4th international web rule symposium. [s. l.]:[s. n.],2010.
- [4] Vinoski S. CORBA: integrating diverse applications within distributed heterogeneous environments[J]. IEEE Communications Magazine,1997,35(2):46-55.
- [5] Henning M. A new approach to object-oriented middleware[J]. IEEE Internet Computing,2004,8(1):66-75.
- [6] 张舒,褚艳利. GPU 高性能运算之 CUDA[M]. 北京:中国水利水电出版社,2009.
- [7] Zhang X, McIntosh S, Rohatgi P, et al. Xensocket: a high-throughput interdomain transport for virtual machines[C]//

Proc of international middleware conference. Newport Beach, CA:ACM,2007:184-203.

- [8] Kim K, Kim C, Jung S, et al. Inter-domain socket communications supporting high performance and full binary compatibility on Xen[C]//Proc of international conference on virtual execution environments. Seattle:ACM,2008:11-20.
- [9] Wang J, Wright K, Gopalan K. XenLoop: a transparent high performance inter-VM network loopback[C]//Proc of international symposium of high performance distributed computing. Boston:ACM,2008:109-118.
- [10] 陈浩,彭萃芬,孙建华,等. XenRPC: 安全的虚拟机远程过程调用设计与实现[J]. 计算机研究与发展,2012,49(5):996-1004.
- [11] 顾晓峰,王健. 基于 Intel VT-x 的 XEN 全虚拟化实现[J]. 计算机技术与发展,2009,19(9):242-245.
- [12] 孟江涛,卢显良,董贵山. Xen 的虚拟机网络优化研究[J]. 电子科技大学学报,2010,39(1):106-109.
- [13] VMCI overview[EB/OL]. 2014-09-08. [http://pubs. vmware. com/vmci-sdk/](http://pubs.vmware.com/vmci-sdk/).
- [14] 包敬海,周小珠,樊东红. 基于 VMWare 构建虚拟网络实验室的研究[J]. 计算机技术与发展,2010,20(6):242-245.
- [15] Direct 2.0[EB/OL]. 2014-09-16. [http://www. valtra. com/news/6568. asp](http://www.valtra.com/news/6568.asp).
- [16] 董小社,刘超,王恩东,等. 面向 GPU 异构并行系统的多任务流编程模型[J]. 计算机学报,2014,37(7):1638-1646.

(上接第 23 页)

于多个产品,充分验证了该设计的可行性。

参考文献:

- [1] 郑小军,胡道徐. 一种通用的嵌入式系统 ISP 方法[J]. 电子技术应用,2005,31(7):22-23.
- [2] NXP. In-circuit and in-application programming of the 89C51Rx+/Rx2/66x microcontrollers[S/OL]. 2002. [http://www. nxp. com/documents/application_note/AN461. pdf](http://www.nxp.com/documents/application_note/AN461.pdf).
- [3] 阮军杰,方岑,李时昌. C8051F12x 单片机 128KB Flash 的 BootLoader ISP/IAP 的实现[J]. 工业控制计算机,2013(10):128-129.
- [4] Atmel. In-system programming[S/OL]. 2008. [http://www. atmel. com/Images/doc0943. pdf](http://www.atmel.com/Images/doc0943.pdf).
- [5] 吴军,华更新,刘鸿瑾. SoC 验证方法学研究与应用[J]. 空间控制技术与应用,2012,38(5):27-33.
- [6] 姚露,朱念好. 基于 DW8051 平台的 MPU 设计与验证[J]. 信息技术,2012(1):118-119.
- [7] 夏宇闻. Verilog 数字系统设计教程[M]. 北京:北京航空航天大学出版社,2003.

- [8] Baer Jean-Loup. Microprocessor architecture: from simple pipelines to chip multiprocessors[M]. Cambridge: Cambridge University Press,2009.
- [9] 尹恒,严华. 一种针对嵌入式远程升级安全的存储解决方案[J]. 计算机应用,2011,31(4):942-944.
- [10] Harris D, Harris S. Digital design and computer architecture[M]. 2nd ed. Burlington: Morgan Kaufmann,2010.
- [11] 蒋美娟,郑羽,陈瑞林,等. 基于 LPC1768 汽车故障远程诊断控制器的设计[J]. 计算机技术与发展,2013,23(8):238-241.
- [12] 李国俊,董晶晶,周瑾. 智能卡 COS 安全性测试研究[J]. 计算机技术与发展,2014,24(2):164-167.
- [13] 褚东升,刘滨,蔡声波,等. ISP 技术在智能仪器远程升级中的应用[J]. 单片机与嵌入式系统应用,2002(4):46-48.
- [14] 付超,余本功. 嵌入式无线移动设备的开放式远程现场升级[J]. 计算机工程与应用,2007,43(1):3-5.
- [15] 刘根贤,龚雪容,生拥宏,等. 基于高频 RFID 的微处理器 IAP 技术[J]. 电子技术应用,2013,39(4):29-31.

GPU虚拟化环境下的数据通信策略研究

作者：[张玉洁](#)，[吕相文](#)，[张云洲](#)，[ZHANG Yu-jie](#)，[Lü Xiang-wen](#)，[ZHANG Yun-zhou](#)

作者单位：[南京航空航天大学 计算机科学与技术学院](#)，江苏 南京，210016

刊名：[计算机技术与发展](#)

英文刊名：[Computer Technology and Development](#)

年，卷(期)：2015(8)

引用本文格式：[张玉洁](#)，[吕相文](#)，[张云洲](#)，[ZHANG Yu-jie](#)，[Lü Xiang-wen](#)，[ZHANG Yun-zhou](#) GPU虚拟化环境下的数据通信策略研究[期刊论文]·[计算机技术与发展](#) 2015(8)