用FPGA实现PCI-E接口和DMA控制器设计

王嘉良,赵曙光
(东华大学,上海 201620)

摘 要:PCI-Express接口是第三代总线通信接口标准,它采用点对点串行连接方式,具有高速率和高带宽等特点,这是传统PCI并行结构所不能比拟的。为了实现FPGA与CPU之间高速相互通信,介绍了一种用单片FPGA实现PCI-Express接口和高速DMA控制器的设计方法,并在Xilinx Virtex-6 FPGA系列平台上实现。运用Xilinx提供的PCI-Express端点硬核,提出并设计了基于PCI-Express总线的DMA数据传输方案。通过仿真及硬件测试表明,该设计方案成本低、高效,可满足如10Gb以太网数据帧抓取等高速数据采集及分析系统的需要。

关键词:PCI-Express总线;FPGA;直接存储器访问
中图分类号:TP39 **文献标识码**:A **文章编号**:1673-629X(2011)06-0181-04

PCI-E Interface Implementation and DMA Engine Design with FPGA

WANG Jia-liang, ZHAO Shu-guang
(Donghua University, Shanghai 201620, China)

Abstract: PCI-Express interface is the 3rd generation bus interface standard, which is point to point serial interconnection with high speed and high bandwidth, what the traditional PCI parallel structure cannot be compared with. Propose a design of high-speed communication with CPU through the PCI-Express interface and a DMA engine method based on a single FPGA. It is implemented on the Xilinx Virtex-6 FPGA series platform. With the hard PCI-Express endpoint IP core, the transaction of data through DMA based on PCI-Express bus is achieved. The results of simulation and hardware test show that it is applicable and can be used to the 10Gb Ethernet environment and other data acquisition systems.

Key words: PCI-Express bus; FPGA; DMA

0 引 言

随着网络的飞速发展,人们可获取的信息量日益增长,数据的处理及存储速率的要求也越来越高。万兆网(10Gb以太网)的普及,高速存储设备的应用(如DDR2,传输速率可达800M)对系统带宽带来极大的挑战。而传统的总线架构,虽历经多年发展,但其带宽已经成为制约系统性能的主要瓶颈。文中介绍了一种基于高速的I/O串行互联技术,PCI-Express(简称PCIe),提出了在系统设计中基于FPGA的PCIe总线接口设计及应用于该总线的高速DMA Engine设计方案。

1 PCI-Express总线简介

PCIe总线是Intel公司于1997年提出的第3代I/O总线技术,用以取代PCI总线和芯片内部连接,并在2002年提出PCI Express 1.0标准[1]。与PCI并行总线相比,PCIe采用差分LVDS串行传输方式,一条数据链路(称为PCIe x1)由2对LVDS差分信号线来实现发送和接收,信号传输速率为2.5Gbps。与PCI采用共享总线方式不同,PCIe采用点对点连接,设备均可以通过专用数据链路直接与总线相连,可以组合成x1,x2,x4,x8,x16及最新x32链路模式。这种总线拓扑方式使得PCIe具有很好的扩展性和灵活性。串行数据传输采用工业标准的8b/10b编码实现数据恢复。PCIe x1可以提供500MB/s带宽,这相当于普通PCI带宽的3.78倍[2]。PCIe总线不但提供了更高的性能,更大大减少了接口管脚,降低电路板布线难度,并减少了信号线间产生的互相干扰。

2 FPGA系统设计方案

2.1 传统FPGA设计方案

传统FPGA并不具备LVDS信号驱动能力。一般以PEX 8311桥接芯片实现PCIe物理层接口,再配合CPLD实现用户逻辑设计[3]。PEX 8311提供2个端

收稿日期:2010-10-28;**修回日期**:2011-03-03
基金项目:国家自然科学基金(60073053,60133010)
作者简介:王嘉良(1986-),男,硕士研究生,研究方向为通信系统、SoC设计与验证;赵曙光,教授,博士生导师,研究方向为电子设计自动化、进化电子系统等。

口，一个为兼容 PCI Express 1.0 标准的 PCIe 接口，另一个为 Local Bus 总线接口，用于与 CPLD 相连。Local Bus 总线工作于 66MHz 时钟，32-bit 总线宽度，可以提供 266MB/s 带宽。同时，PEX 8311 还提供了 2 组 DMA 通道[4]。使用 PEX 8311 可以方便地从 PCI 平滑过渡到 PCIe，从而降低开发成本，因此被广泛使用[5]。然而，PEX 8311 并不能有效发挥出 PCIe 高带宽优势，也缺乏可配置性和灵活性，难以提高系统整体性能。

2.2 基于 PCIe 硬核接口设计

Xilinx 新推出 Virtex-6 系列的 FPGA，集成了 PCIe IP 硬核模块。该模块兼容 PCI Express 2.0 标准分层协议（为物理层、数据链路层和事物层，又称传输层），提供了系统接口（SYS）、外部传输接口（PCI EXP）、配置接口（CFG）、事物接口（TRN）和物理链路接口（PL）[6]。其 Virtex-6 GTX 可配置高速串行传输器为 PCIe 信号可靠传输提供了保障。文中便是针对该硬核的事物层提出片上系统的 PCIe 设计方案。

3 PCI-Express 接口及应用层设计

一个基本的高速数据采集系统，由微处理器（CPU）、FPGA 和外部存储器等组成[7]。CPU 与 FPGA 通过 PCIe 总线进行数据传输，FPGA 通过 DMA 方式对外部存储器进行数据读写操作。文中给出基于 PCIe x1 传输方式的接口设计方案。

3.1 PCIe 接口及事物层设计

较之使用 PEX 8311 桥接芯片的系统设计[8]，采用基于 Xilinx 的 IP 核的设计方案，用户可以根据自己的需求灵活设计面向事物层接口电路，并方便加入特殊功能，如与内部总线连接、实现 DMA 传输等。该设计充分利用 FPGA 集成度高、可配置性强等特点[9]，来发挥 PCIe 接口性能。FPGA 的 PCIe 接口设计包括 Xilinx PCIe 端点硬核和面向事物层的应用逻辑设计 2 个部分，这里将详细给出应用层设计，如图 1 所示。

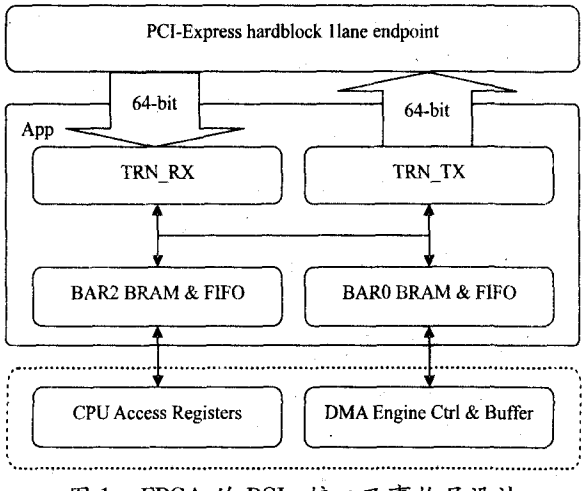

图 1 FPGA 的 PCIe 接口及事物层设计

Xilinx PCIe 硬核支持 64-bit 数据通路，分别用于发送和接收数据。该硬核提供 6 个 32-bit 基地址寄存器 BAR0 ~ BAR5（Bass Address Register，简称 BAR），可以根据用户设计需求进行配置。在本设计中使用 BAR0 和 BAR1 组成一组 64-bit 地址空间，用于存储外部 CPU 访问 FPGA 内部寄存器地址。使用 BAR2 和 BAR3 组成另一组 64-bit 地址空间，用于存储 DMA Engine 控制器和缓存器地址。通过判断 trn_rbar_hit_n [6:0] 来区分 BAR0 和 BAR2，其主要代码如下：

```
… …
TRN_RX_MEM_RD64_DW1DW2: begin
  … …
    req_bar0_o<= ~trn_rbar_hit_n_q [0];
    req_bar2_o<= ~trn_rbar_hit_n_q [2];
    … …
end
```

以外部 CPU 读 FPGA 内部寄存器为例，FPGA 将收到的 PCIe 总线上数据帧经由硬核的事物层，以 64-bit 带宽送出，应用层通过 TRN_RX 接收状态机来判断请求访问的地址空间，然后将请求数据缓写入一个双端口的 req_fifo[10]；用户逻辑通过读取 req_fifo，将有效的寄存器数据写入另一个双端口 data_fifo 中[11]，最后通过 TRN_TX 发送状态机发送给硬核事物层。

3.2 基于 PCIe 的 DMA Engine 设计

通过 DMA 访问外部存储器的最大优势在于 CPU 在配置完 DMA Engine 后可以继续其他指令操作，DMA Engine 会通过请求 PCIe 总线中断的方式，来完成数据传输。本设计将 PCIe BAR0 地址空间存储 DMA Engine 控制和缓存器地址，其定义如图 2 所示。

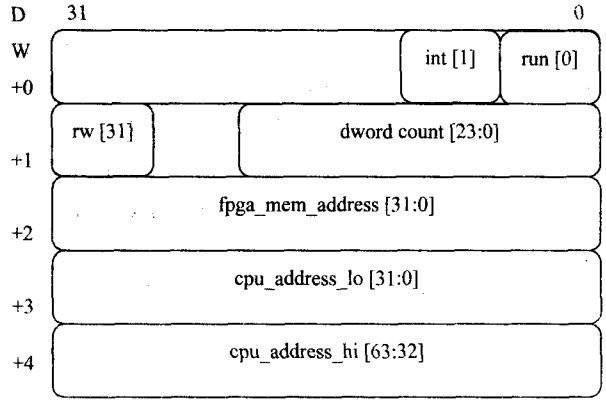

图 2 DMA 控制器和缓存器设计

DWORD 0 [0]，run，写 1 开启 DMA；

DWORD 0 [1]，int，写 1 开启 DMA 中断方式；

DWORD 1 [31] rw，写 1 读 fpga 外部存储器；

DWORD 1 [23:0] dword count，DMA 操作数据个数，以 dword 为单位；

DWORD 2 [31:0] fpga_mem_address，fpga 外部存

储器地址;

DWORD 3 [31:0] cpu_address_lo, cpu 低 32 位地址;

DWORD 4 [31:0] cpu_address_hi, cpu 高 32 位地址。

修改 PCIe 数据帧类型(tlp_type <= trn_rd[62:56]),加入 DMA 数据帧,使其有别于对 BAR 的读写操作,部分代码如下:

```
`define TRN_RX_DMA_TLP_TYPE   7'b10_01010; /* DMA 数据类型 */
`define TRN_RX_DMA_DATA1   10'b01_0000_0000; /* DMA 操作状态 */
... ...
case (state)
`RST: begin
......
case (tlp_type)
   ......
   `TRN_RX_DMA_TLP_TYPE: begin
   ......
     state<=`TRN_RX_DMA_DATA1;
end
   ......
`TRN_RX_DMA_DATA1: begin
   ......
end
......
endcase
```

以同样的方式在 TRN_TX 状态机中加入 DMA 请求,其代码如下:

```
`defineTRN_TX_DMA_REQ   10'b01_0000_0000;
......
case (state)
`RST: begin
   if (dma_tlp_ready) begin   /* DMA Engine 准备好读操作数据帧 */
        state<=`TRN_TX_DMA_REQ;
        dma_tlp_rd_en<=1'b1;   /* DMA 缓存读使能 */
end
else if (……)
        ……
end
……
`TRN_TX_DMA_REQ: begin
   ……
   if (~tlp_dma_data_eof)   /* 数据帧结束标志位 */
        state<=`TRN_TX_DONE;
   else
        state<=`TRN_TX_DMA_REQ;
end
```

FPGA 的 DMA Engine 状态转移图如图 3 所示,分 DMA 读操作和写操作 2 部分。DMA Engine 处于 RST 状态时,通过读取 DMA 信息标识符来判断当前是否处于空闲状态,并从 FPGA 内部 RAM 中读取 DMA 控制信息(读或写),并进入相应状态 READ_ST 或 WRITE_ST。以 CPU 写 FPGA 外部存储器为例,CPU 通过 PCIe 总线写 BAR0 地址数据来配置并开启 DMA Engine。FPGA 将发出对 CPU 的 DMA 读请求,然后等待 CPU 发送 DMA 数据。此时 DMA Engine 处于 WAIT_FOR_DATA 状态,等待来自 PCIe 接口的 DMA 数据包。同样,CPU 读 FPGA 外部存储器时,FPGA 将发出对 CPU 的 DMA 写请求,并当 DMA 完成读操作后,等待 PCIe 接口发送 DMA 数据包,并由中断标志位判断是否开启 PCIe 中断。

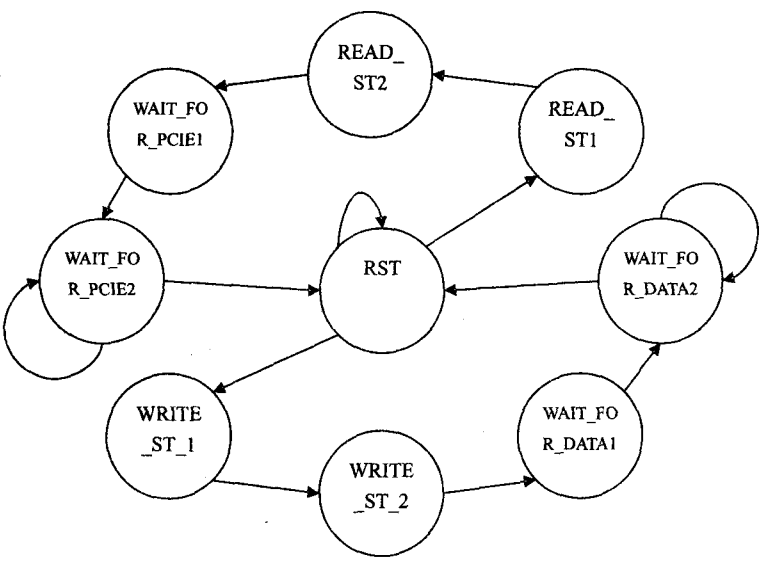

图 3　DMA Engine 状态机描述

4　仿真及测试结果

为了测试该 DMA Engine 设计,对 FPGA 外部 64-bit DDR2 DRAM 进行 32 个 dwords (128 bytes)读与写操作,得到的仿真波形如图 4 和图 5 所示。图 4 是对 DMA 进行写操作的 PCIe 事物层接口波形。图 5 是对 DMA 进行读操作的 PCIe 事物层接口波形。该设计方案在 questasim 6.3f 下验证通过。可以估算出一次 PCIe x1 写操作为 1720Mb/s,读为 1684Mb/s[12]。

同样,在实际的应用环境中测试本文 DMA Engine

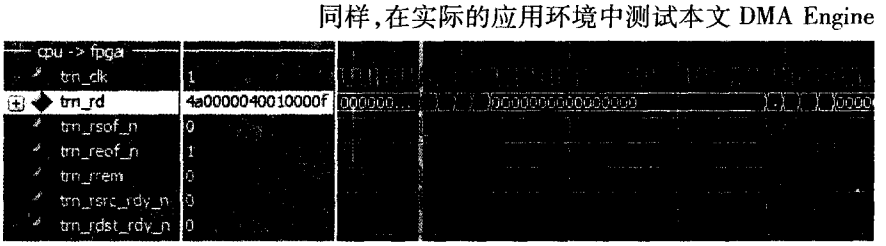

图 4　CPU 通过 PCIe 对 DMA 进行写操作

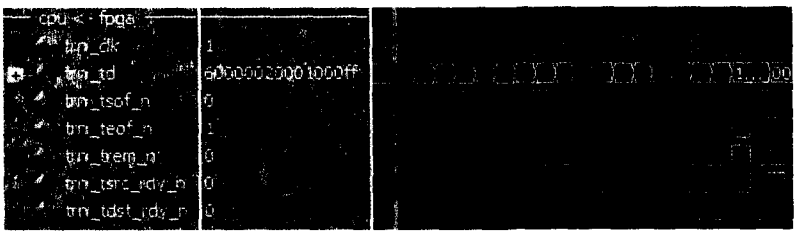

图 5 CPU 通过 PCIe 读 FPGA 外部 DRAM

设计,也获得比较好的结果。图 6 就是系统以本文 DMA 方式进行 4096 bytes 读外部存储器的结果。

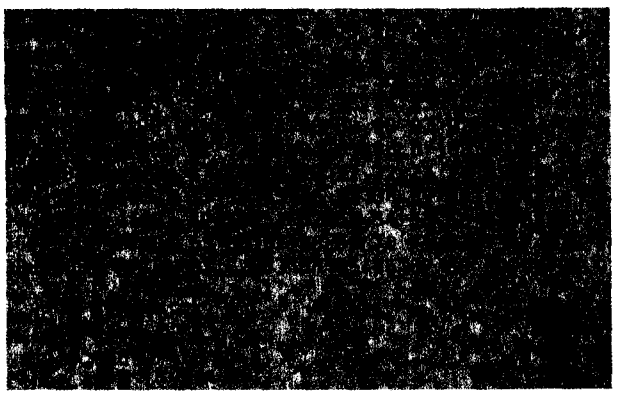

图 6 DMA 读操作及结果

5 结束语

FPGA 较 ASIC 的强大之处,在于灵活性和可配置性。文中凭借 Xilinx PCIe 硬核,很好地实现了 PCIe x1 总线接口,并在此基础上加入高速 DMA Engine 设计,可以有效地提高系统数据存储效率。

该设计方案通过简单修改也同样适用于 32-bit 带宽传输系统设计。

参考文献:

[1] Bhatt A, Technology and Research Labs, Intel Corporatiob. Creating a PCI Express™ Interconnect [EB/OL]. 2002. http://www.pcisig.com/specifications/pciexpress/resources/PCI_Express_White_Paper.pdf.

[2] PCI Express Base Specification, 1.0a ed [S]. PCI SIG, 2003.

[3] Design Note: Optimizing PEX8311 PCI Express-to-Local Bus DMA Performance[M]. [s.l.]:PLX Technology,2007.

[4] Data Book: PEX 8311Detailed Technical Specifications[M]. [s.l.]:PLX Technology,2009.

[5] 马 萍,唐卫华,李绪志. 基于 PCI Express 总线高速数采卡的设计与实现[J]. 微计算机信息,2008(25):116-118.

[6] LogiCORE™ IP Virtex-6 FPGA Integrated Block Data Sheet [M]. [s.l.]:Xilinx,2009.

[7] Li B, Peng Y, Liu D T. A High Speed DMA Transaction Method for PCI Express Devices[J]. Journal of Electronic Science and Technology of China, 2009,7(4):94-98.

[8] 强 俊,楚 宁,周鸣争. 基于 PCI EXPRESS 总线的视音频采集卡的设计[J]. 计算机技术与发展,2007,17(5):85-88.

[9] 李 攀,田 泽,蔡叶芳,等. 基于 SOPC 的 PCI 通信接口设计与实现[J]. 计算机技术与发展,2009,19(9):217-220.

[10] 张维旭,贺占庄. 基于 FPGA 的异步 FIFO 设计[J]. 计算机技术与发展,2006,16(7):174-176.

[11] 余志强,戎蒙恬,袁丹寿. 一种单端口 SRAM 构成 FIFO 的 ASIC 设计[J]. 计算机技术与发展,2008,18(9):59-61.

[12] 徐君明,裴先登,王海卫,等. 高性能计算机 I/O 技术 PCI Express 分析[J]. 计算机工程,2004,30(12):7-8.

(上接第 180 页)

备的交互。通过绘制机场图片、飞机位置坐标点和飞机图标,在飞行模拟器中实现了飞机位置的实时显示。采用"双缓冲"技术极大地提高了飞行轨迹的绘制效率,绘制出的飞行轨迹无闪烁,画面刷新速度快,完全满足显示要求。

参考文献:

[1] 周长发. C# 面向对象编程[M]. 北京:电子工业出版社,2007.

[2] 陈本峰,苏 琦. Windows GDI +的研究与应用[J]. 计算机应用研究,2003 (3):56-59.

[3] Graus C. Starting with GDI+[EB/OL]. 2003-05. http://www.codeproject.com/KB/GDI-plus/startinggdiplus.aspx, codeproject.

[4] Chand M. GDI + 图形程序设计[M]. 北京: 电子工业出版社,2005.

[5] Nagel C,Evjen B,Glynn J,et al. C# 高级编程[M]. 第 4 版. 北京:清华大学出版社,2006.

[6] Microsoft MSDN[EB/OL]. 2008. http://msdn.microsoft.com/library,Microsoft.

[7] 闫宇晗,常 鑫. 在 C# 中用 GDI + 实现图形动态显示[J]. 计算机技术与发展,2006,16(12):117-118.

[8] 张卫华. 在 GDI+中利用双缓冲技术实现橡皮筋效果[J]. 电脑编程技巧与维护,2009(5):70-80.

[9] Mazdak. Double Buffering With GDI+[EB/OL]. 2002-12. http://www.codeproject.com/GDI-plus/gdiplus.aspx, codeproject.

[10] 齐立波. C#入门经典[M]. 第 4 版. 北京:清华大学出版社,2008.

[11] 梁斌玉. 叩响 c#之门[M]. 北京:电子工业出版社,2009.

[12] 周 靖. Visual C# 2010 从入门到精通[M]. 北京:清华大学出版社,2010.