第 35 卷 第 5 期 计 算 机 工 程 2009 年 3 月
Vol.35 No.5 Computer Engineering March 2009
·工程应用技术与实现· 文章编号：1000—3428(2009)05—0227—03 文献标识码：A 中图分类号：TP336

通用 PCI 目标控制器的设计与实现

王新胜，王晨旭，付明旺
(哈尔滨工业大学(威海)微电子中心，威海 264209)

摘 要：在研究 PCI 总线规范的基础上，完成 PCI 目标控制器的功能描述和验证，从整体设计、数据通路和控制路径的建立上阐述对目标控制器的设计，实现了 FIFO 型数据接口和寄存器型数据接口，以满足通用性。使用总线功能模型完成对 PCI 目标控制器的功能验证，测试结果显示该模型满足功能要求，同时进行了 FPGA 实现，完全符合 PCI v2.2 规范要求。

关键词：PCI 总线；目标控制器；总线功能模型

Design and Implementation of Common PCI Target Controller

WANG Xin-sheng, WANG Chen-xu, FU Ming-wang
(Microelectronics Center, Harbin Institute of Technology in Weihai, Weihai 264209)

【Abstract】This paper develops a PCI target controller and verifies it, based on study of PCI bus specification, which focuses on the whole design, data path and controlling, particularly implements FIFO interface and register interface purposing to common uses. The target controller is verified using bus function models and implemented in FPGA, which proves the design is met with functional demand and conformed to PCI v2.2 specification.

【Key words】Peripheral Component Interconnect(PCI) bus; target controller; bus function models

1 概述

PCI(Peripheral Component Interconnect)总线是微型计算机中处理器/存储器与外围控制部件、扩展卡之间的互联接口。PCI 总线规范是互联机构的协议以及电气和机械配置的规范，是当今高性能微型计算机事实上的总线标准[1]。

由于 PCI 总线是目前技术成熟的总线中速度最快、性能最稳定的总线技术，因此其相关功能设备的市场需求很大。现在市场上 PCI 总线的相关产品很多，但大多来自国外的 PLX, AMCC 等公司，而国内的公司鲜有相关产品问世。

PCI 总线控制器为 PCI 总线和用户设备提供操作接口，协调 PCI 总线信号和用户设备接口信号，使用户设备能够按照 PCI 总线的规范进行数据的传输。PCI 总线控制器的设计比较复杂，根据用户设备的不同性质，PCI 设备分为主设备和目标设备，因此，PCI 控制器也就有主设备控制器和目标设备控制器之分[2]。PCI 总线主设备可以主动发起总线上的数据传输，而目标设备不具有这种能力，它只能被动配合主设备完成主设备要求的数据传输。本文设计了一个 PCI 目标控制器，此目标控制器具有较强的通用性，尤其实现了 FIFO 型数据接口和寄存器型数据接口，更好地满足通用性。

2 PCI 目标控制器的设计

PCI 总线目标设备在总线传输中处于被动地位，它不会申请对总线的使用权。当 PCI 总线上的某一主设备发起对本地资源的访问时，整个控制路径的建立和数据传输的过程都通过目标控制器实现。

2.1 功能及整体设计

在 PCI 系统中，目标设备处于被动的地位，它不会主动申请总线使用权，只是在地址周期内响应总线主设备的传输要求，并配合主设备完成整个数据传输[3]。图 1 为目标控制器的详细结构。

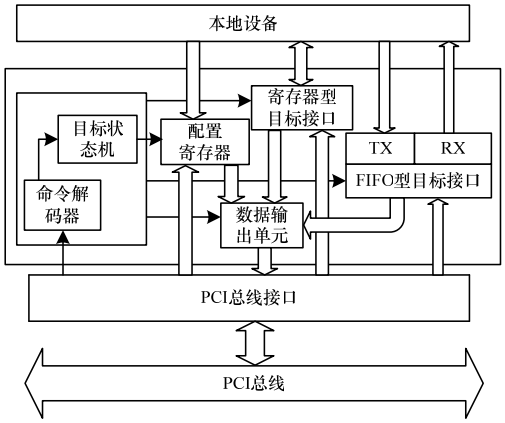

图 1 目标控制器结构

PCI 主设备发起某类型的操作，目标控制器在地址周期内解析相应的地址和命令信息。在解析地址确定为本次操作的目标设备后，目标控制器会启动目标设备状态机根据当前解码的信息进行一系列的控制操作。

本设计将目标控制器按照功能分为几个功能模块实现，即地址命令解析模块、状态机、配置寄存器、FIFO 类型目标接口和寄存器型目标接口。FIFO 类型通道用于大批量的数据传输，可将本地存储器块挂接在该类型的数据通道上；寄存器类型接口用于数据量小的数据传输，该接口的后面可以挂接一些本地功能设备的控制/状态寄存器块。这样，PCI 总线上的主设备既可以访问到本地的存储器设备，也可以访问本地的寄存器。

作者简介：王新胜(1978－)，男，助教，主研方向：VLSI 设计，计算机体系结构；王晨旭，讲师；付明旺，硕士

收稿日期：2008-06-18 **E-mail：**wsun515@126.com

2.2 目标控制器的数据路径实现

目标控制器的数据路径具体就是指 2 种类型的目标接口：寄存器型目标接口和 FIFO 型目标接口。

本地可以有一些功能设备，这些设备都有相应的控制/状态寄存器。PCI 总线上的主设备要对这些控制寄存器进行编程，进而控制相应的功能设备完成特定功能。寄存器型目标接口就为本地的功能设备提供了这样一种接口，使得 PCI 总线上的主设备可以通过它来访问功能设备的寄存器。由于 PCI 总线主设备的访问目标是一些寄存器，因此该接口无需支持突发传输，主设备每一次访问最多只涉及到 32 位的数据传输。

FIFO 型的目标通道主要用于大量的数据传输，在设计中可以将本地的存储器挂接在该接口上，这样 PCI 总线主设备可以通过 FIFO 型目标通道访问到本地的存储器。由于是对存储器的访问，数据访问量大，因此需要考虑到访问的效率。针对 PCI 主设备对本地存储器的读和写操作，设置了 2 个异步 FIFO：读 FIFO 和写 FIFO。

对 PCI 主设备发起的写操作，数据会先以 PCI 时钟写入接口内的写 FIFO，同时 PCI 控制器通知本地存储器控制器有数据写入。本地存储器控制器待 FIFO 中达到一定数据量时以自己的时钟读取 FIFO 中的数据存入本地存储器，如果 FIFO 读空，存储器会暂停等待直到 FIFO 中有新的数据写入，然后进行读取。

对于 PCI 主设备发起的读本地存储器的操作，为了提高读数据的效率，设计中根据不同数据传输的特点，把读操作划分为 2 个类型，即立即读和延迟读。

(1)立即读：在 PCI 总线主设备发起读操作后，将一直占用总线等待。PCI 目标控制器向本地存储器请求对应的数据，直到本地存储器将数据写入读 FIFO 中，PCI 控制器读取 FIFO 中数据，开始 PCI 总线上的数据传输。这样，PCI 总线上的主设备在数据未到达总线之前一直是空闲的，但它也一直占用总线，降低了总线的利用率。所以，这种类型的读操作一般适用于进行数据量小的传输，且本地存储器可以在较短的时间内将要求的数据送到 PCI 总线，因为 PCI 规范规定目标设备不能无限制地插入等待周期，必须在 16 个 PCI 时钟周期内提供第 1 个数据。

(2)延迟读：针对立即读总线利用率低的缺点，设计中引入了延迟读。PCI 总线主设备的读请求到达 PCI 目标总线控制器时，控制器锁存地址和命令信息，并回应主设备让它重试。同时 PCI 目标总线控制器向本地存储器请求数据，本地存储器将数据写入总线控制器中的 FIFO 中。此时如果 PCI 总线上的主设备重试先前的数据传输请求，PCI 目标控制器就可以响应请求将数据发送到 PCI 总线上。这个过程中，PCI 总线主设备在收到重试回应后就会释放 PCI 总线，允许其他的主设备占用总线进行数据传输，从而提高了总线的利用率。为保证 FIFO 中的数据是主设备要求的数据，重试读的过程中，PCI 目标总线控制器设计为能对重试的地址和命令信息进行核对。

2.3 目标控制器的控制路径实现

目标控制器的控制由 2 个部分完成：命令解码器和目标状态机。命令解码器解码来自 PCI 总线上的地址命令信息作为目标状态机的输入，状态机鉴于此输入执行相应的操作。

(1)命令解码器。PCI 总线主设备在地址周期内通过命令数据线(C/BE#)设置不同的编码可以发起以下的操作：I/O 读写，存储器读写，配置读写，存储器行读/多行读和存储器写及使无效。命令解码器的解码情况如表 1 所示。需要注意的是，对于存储器行读/多行读和存储器写及使无效这 2 个命令，目标控制器将其分别转化为存储器读和存储器写命令来实现其对应的操作。

表 1　PCI 总线命令解码

C/BE#[3:0]	命令类型	命令解码器
0000	中断应答	忽略
0001	专用周期	忽略
0010	I/O 读	响应
0011	I/O 写	响应
0100	保留	忽略
0101	保留	忽略
0110	存储器读	响应
0111	存储器写	响应
1000	保留	忽略
1001	保留	忽略
1010	配置读	响应
1011	配置写	响应
1100	存储器多行读	转为存储器读
1101	双地址周期	忽略
1110	存储器行读	转为存储器读
1111	存储器写及使无效	转为存储器写

(2)目标状态机：对于 PCI 总线上传来的信号，状态机根据命令解码器的解码信息来控制总线控制器进行相应的操作。如果地址信息不正确或者本地的设备正忙于存取数据，则状态机会向 PCI 总线上发起本次操作的主设备回应目标丢弃或目标重试。目标状态机的具体设计如图 2 所示。

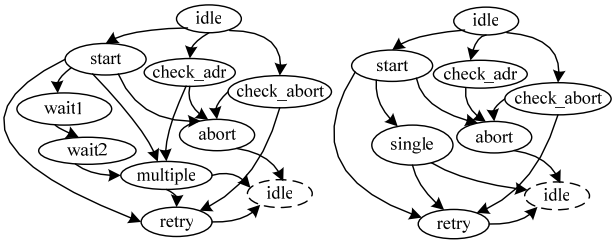

图 2　目标设备接口状态机

状态机共分为 10 个状态，下面对每个状态的意义及状态跳转条件进行详细说明。

(1)idle：目标通道空闲。当 PCI 总线有新的命令但控制器正忙于未完成的数据传输或寄存器配置时，状态机跳转到 check_abort；控制器空闲但寄存器内有记录的未完成的读操作，状态机跳转到 check_adr；PCI 总线上有数据传输要求而又未出现上面的 2 种情况，状态机跳转至 start。

(2)start：启动命令解码。满足放弃条件如奇偶校验错误、强制放弃等条件时，状态机跳转至 abort；不满足放弃条件，若目标设备没有突发传输能力，状态机跳转至 single，若目标设备有突发传输能力且主设备发起的是写命令，状态机跳转至 multiple，如果目标设备有突发传输能力且主设备发起的是延迟读命令，状态机跳至 retry；上述情况均未发生，状态机跳转至 wait1。

(3)check_adr：检查地址匹配。满足放弃条件如奇偶校验错误、强制放弃等条件时，状态机跳转至 abort；如果地址与之前的请求地址一致，对目标设备进行延迟读，状态机跳转至 multiple；如果地址不一致状态机跳转至 retry。

(4)check_abort：检查放弃条件是否满足。满足放弃条件如奇偶校验错误、强制放弃等条件时，状态机跳转至 abort；不满足条件，状态机跳转至 retry。

(5)wait1：FIFO 类型目标设备的等待读。如果 FIFO 中有了待传输的数据，状态机跳转至 wait2；否则一直等待在状态 wait1。

(6)wait2：FIFO 类型目标设备等待读。FIFO 中已经有了所要读取的数据，状态机跳转至 multiple。

(7)abort：控制器回应放弃。一直维持该状态直到主设备声明传输结束，状态机跳转至 idle。

(8)single：单数据传输。如果主设备声明最后一个数据传输，状态机跳转至 idle；如果主设备继续对该目标设备进行读写操作，状态机跳转至 retry。

(9)multiple：数据突发传输。如果剩下最后一个字节传输，状态机跳转到 idle；如果地址信息无效或地址越界，状态机跳转到 retry；如果接收 FIFO 几乎满了而 PCI 主设备继续往其中写入数据，状态机跳转到 retry；其他情况状态机保持 multiple 状态不变。

(10)retry：控制器回应重试。一直维持该状态直到主设备声明传输结束，状态机转至 idle。

目标设备通道通过状态机的状态和当前数据传输的情况产生控制信号，协调完成 PCI 总线上的数据传输，PCI 总线上的传输控制信号也是这样产生的。

DEVSEL#：只要状态机没有处在 idle 或者 abort 状态下，该信号就有效。

TRDY#：状态机处于 sigle 或者 multiple 状态下，该信号有效。

STOP#：状态机处于 abort 或 retry 状态下，该信号有效。

3 目标控制器的验证

在完成目标控制器的 RTL 级功能设计后，必须对设计进行功能验证。据统计功能验证的工作量和投入的力度已经占到项目总投入的 60%~70%，它已经成为整个设计过程中最重要的一个环节[4]。

验证使用了总线功能模型模拟 PCI 目标控制器在实际工作过程中所处的环境。在验证过程中，生成测试数据并将其输出到待验证的总线控制器中，监测模型对激励的响应过程，并报告结果。验证环境的组织结构如图 3 所示。

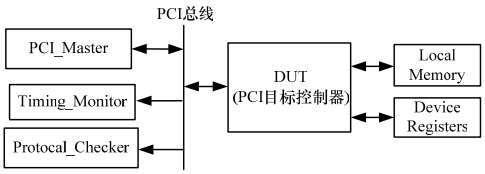

图 3　PCI 目标控制器验证环境

在图 3 中，PCI_Master 模拟总线主设备的功能，对 PCI 目标控制器(Design Under Test, DUT)施加各种传输类型的激励。Local Memory 和 Device Registers 2 个模型分别模拟本地的存储器和寄存器型目标，它们挂接在对应的 FIFO 型目标接口和寄存器型目标接口上。PCI 总线上还有 2 个行为模型，Timing_Monitor 和 Protocal_Checker 是总线上的时序和协议规范监测模块。

验证涉及的验证点有以下几个方面：配置寄存器读写，IO 空间立即读写，存储器空间单次、突发延迟读写，目标设备请求重试，目标设备放弃，主设备放弃。使用 Modelsim SE6.2 完成整个仿真过程，最终 PCI 目标控制器通过验证实现了目标设备功能，完全符合 PCI v2.2 规定。图 4 为 PCI 主设备突发读的仿真结果。

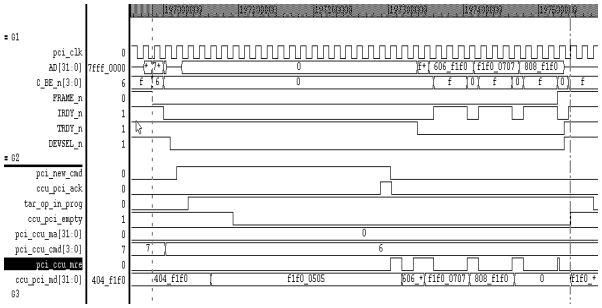

图 4　PCI 总线主设备对本地存储器的突发读

4 结束语

本文在深入研究 PCI 总线规范后，使用 Verilog 硬件描述语言完成了通用 PCI 目标控制器的 RTL 级设计和功能验证。经验证，该设计完全实现了目标控制器的功能，根据目标设备的不同类型，在目标控制器内实现不同的目标接口完成数据传输，使整个设计取得了较高的效率。目前本设计已成功地应用于微电子中心多个 SoC 系统设计中。

参考文献

[1] Shanley T, Anderson D. PCI 系统结构[M]. 林　辉, 译. 北京: 电子工业出版社, 2000.

[2] 沈涵飞, 甘　萌. PCI 总线目标控制器的设计与实现[J]. 计算机工程与设计, 2004, 25(11): 2101-2104.

[3] 刘　红, 李　勃, 常　青, 等. 基于 IP 核的 PCI 总线接口设计与实现[J]. 电子技术应用, 2006, 32(6): 6-9.

[4] Bergeron J. Writing Testbenches——Functional Verification of HDL Models[M]. Boston, USA: Kluwer Academic Publisher, 2000.

编辑　顾逸斐

（上接第 226 页）

5 结束语

综上所述，利用通用微处理器实现 DSP 算法，实际上就是充分利用通用微处理器的片内资源，发挥通用微处理器潜在的计算能力，降低代码冗余度的过程。可运用上述优化策略，在提高速度的同时，极大地提高产品的性价比。虽然这些经验都是针对 ATMEGA16L 芯片的，但也同样适用 AVR 系列单片机的其他芯片，对于其他系列的通用微处理器也有一定的借鉴意义。

参考文献

[1] 纪宗南. DSP 实用技术和应用实例[M]. 北京: 航空工业出版社, 2006.

[2] 张雄伟, 陈　亮, 徐光辉. DSP 芯片的原理与开发应用[M]. 北京: 电子工业出版社, 2003.

[3] Barnett R, Cull L, Cox S. Embedded C Programming and the Atmel AVR[M]. 北京: 清华大学出版社, 2003.

[4] 丁化成, 耿德根, 李君凯. AVR 单片机应用设计[M]. 北京: 北京航空航天大学出版社, 2002.

[5] Glover L, Grant P M. Digital Communications[M]. 北京: 机械工业出版社, 2006.

编辑　顾逸斐