PCI总线从接口的设计与验证

闫福鑫,许志宏,刘佑宝
(西安微电子技术研究所,陕西 西安 710065)

摘 要:讨论了一种包括配置空间和I/O空间的从PCI(PCI-slave)接口电路的Verilog HDL设计。重点介绍了顶层的系统架构,对其进行了功能分析和结构划分,并详细阐述了各子模块电路的设计和实现。根据PCI总线交易时序,给出使用有限状态机实现接收总线信号控制本地逻辑的方法。针对PCI时序的复杂性,提出了一种新颖实用的PCI系统验证方法,并重点讲述了组建验证平台的方法及其优点。通过编写测试激励程序完成了功能仿真,仿真和验证的结果表明,该接口电路在功能和时序上符合PCI技术规范,达到了预定的目标。

关键词:PCI接口;Verilog HDL;状态机;系统验证
中图分类号:TP39 **文献标识码**:A **文章编号**:1673-629X(2012)08-0233-04

Design and Verification of PCI-slave Interface

YAN Fu-xin, XU Zhi-hong, LIU You-bao
(Xi'an Microelectronics Technology Institute, Xi'an 710065, China)

Abstract: The Verilog HDL design for PCI-slave interface including configuration space and I/O space is discussed. A general overview of the top-level architecture is provided, the thought and the function of each module are also introduced in detail. The timing simulation is implemented and the method of using the state machine is put forward to ensure the proper completion of the bus operation. A new verification methodology for PCI system is discussed. In addition, the approach to test bench and it's advantage are also emphasized. The result of timing simulation and experiment shows that the design of PCI-slave interface complies with PCI local bus specification revision and achieves the expectation.

Key words: PCI interface; Verilog HDL; state machine; system verification

0 引 言

PCI总线[1,2]是一种高速的32位或64位多地址多数据外围部件互联局部总线,它具有132MB/S传输速度和33MHz的带宽设计。其接口电路模块在各种计算机系统中发挥着越来越重要的作用。目前PCI总线接口设计多采用以下3种方法:

(1)ASIC PCI;
(2)FPGA + PCI soft IP;
(3)FPGA +PCI hard IP[3]。

由于采用专用的PCI接口芯片用户不可能用到完整的PCI接口功能,而且灵活性相对较差,组件相对增多,提高了成本,所以对于用PCI总线的工程师,通常采用的解决方案为后两个,采用PCI CORE。FPGA制造商都提供PCI接口的宏核,如Mentor公司的MPCI32 Core和ALTERA公司的PCI MegaCore,它们以固核的方式向用户提供IP[4]。其优点是可以根据实际需求有选择性地实现PCI规范中的功能子集,以后进行功能升级也比较方便,只需重新进行逻辑设计,而且还可以将其他用户逻辑与PCI接口逻辑集成在一个芯片上,实现紧凑的系统设计[5]。

笔者通过对协议的深入了解,成功实现了PCI从接口功能的Verilog HDL设计,有效的实现了接口电路和系统之间数据传输,设计传输位宽为32bits,频率为33MHz。文中介绍了该PCI接口的设计思路和方法,以及子模块的功能设计,并采用Synopsys公司的SmartModle模型库组[6]及Mentor公司的Test System共同搭建的验证平台来验证该设计。

1 PCI接口的设计原理

1.1 系统结构

如图1所示的系统结构中,整个设计分为五大块:PCI总线接口逻辑、Backend接口逻辑、寄存器接口控制逻辑、用作数据存储的FIFO接口及配置空间。PCI总线接口逻辑包括CMD译码、数据选择输出、奇偶校验产生和输出、地址溢出检查、状态机、BIOS ROM接口、可选的扩展接口等功能模块。Backend接口逻辑

收稿日期:2011-12-26;**修回日期**:2012-03-30

作者简介:闫福鑫(1986-),男,硕士研究生,研究方向为IC设计技术;刘佑宝,研究员,博士生导师,研究方向为IC设计技术。

模拟了本地接口的模式转换及其它控制逻辑。寄存器接口逻辑主要包括内部控制寄存器,数据存储单元为用作传输缓存的读写 FIFO。

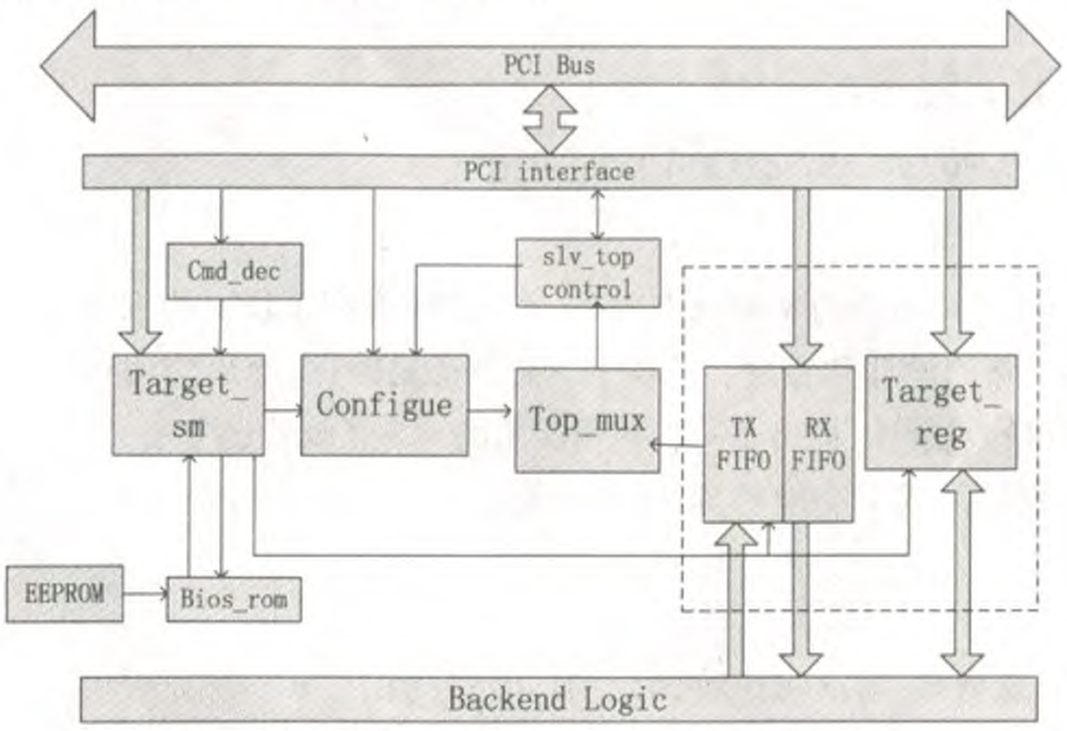

图1 接口系统构造图

1.2 功能模块说明

CMD 译码逻辑主要实现对 PCI 总线信号 C/BE#[3:0]命令的译码,检查 PCI 主设备发起的操作命令是否为本设备所支持的命令形式。

配置寄存器模块实现由 PCI 总线配置部分寄存器,同时将 BIOS ROM 配置的寄存器值通过该模块输出至 Output Data MUX 中。如果设计中引入可选的 SPI EEPROM,则一些寄存器可以通过此模块 boot 加载,同时 Backend 接口也可对寄存器进行配置,特别是对只读存储器也可以实现写入配置。

Top_mux 模块主要依据接口电路的当前操作周期(命令周期、地址周期及数据周期)将配置空间输出,FIFO 数据输出进行选择并有效地输出至总线。

顶层控制模块实现将接口电路产生的校验状态,终止状态等经此模块处理后输入至配置空间的状态寄存器,同时将状态机产生的 PCI 总线控制信号及数据经时钟树同步后输出至总线。

BIOS ROM 模块主要实现扩展与 SPI 串行连接的 EEPROM 接口,实现对配置空间的部分寄存器 bootload 加载。

寄存器接口则主要负责逻辑控制,该模块独立于接口电路的设计,易于区分各自独立的信号,用户可根据需要选择相应的电路进行置位或者复位相应的寄存器位,其支持 32 位,单周期数据传输。在控制上主要实现了地址译码、仲裁及中断功能,现将这三种功能详细介绍如下:

地址译码:主要通过对配置空间的写操作,实现了对 I/O,MEM 的访问使能,以及通过配置寄存器写使能产生 I/O,MEM 访问的有效状态输出逻辑。

仲裁机制:在默认情况下,寄存器操作授权于 PCI 总线,由 PCI 总线上主机对其进行读写操作,当 Backend 逻辑同时需要对其进行相关操作时,首先发出 req 信号,待 PCI 主机操作完成后,寄存器接口会向 Backend 逻辑返回 gnt 信号,授权 Backend 逻辑对进行读写操作。

中断功能:对于中断操作,支持中断功能的标识,内部设计中断向量寄存器指向 ram 的零地址(Backend 发中断请求信号,实际指向专用的中断向量寄存器),具体操作时,由 Backend 逻辑向寄存器接口逻辑发出 32 位中断请求,并存至中断向量寄存器,同时发出的 32 位中断使能信号对中断向量寄存器的相应位具有屏蔽作用,只有在没有完全信号屏蔽的情况下,接口逻辑才会向 PCI 发中断请求。PCI 主机可以读取中断向量,并可以通过写命令对中断向量寄存器的相应位写一清中断。

读写 FIFO 主要实现了数据的传输缓存,有效地将 PCI 总线端和本地端隔开,方便实现了 Posted Write 以及 Delay Read 功能。其支持突发传输,在同步时钟控制下,当读写控制使能有效时,产生同步 FIFO 的读写地址自增,并结合 FIFO 存储 RAM(本设计的 RAM 深度 16,宽度 36),实现了同步 FIFO 的结构设计。

FIFO 的读写操作受同一时钟控制,写操作和满标志,读操作和空标志工作在同一时钟(PCI 时钟)下,写读指针的差值表明 FIFO 当前剩余空间及接受数据个数 情况。FIFO 内部保存数据特点:写入和读出数据顺序一致,读写地址是一个循环列表,当指针移动到最后位置时,又重新回到初始位置。另外,FIFO 的读写地址变化以 gray 码形式递增,最大限度减少数据的变化位数,有效降低了亚稳态的发生。

1.3 状态机设计

状态机(FSM)是设计 PCI 总线接口的核心与主干,状态机通过监视 PCI 总线上的控制信号、配置空间的状态信号,来切换到不同的状态,同时在各种状态下发出相应的控制信号并且控制数据流向。本设计中目标设备状态机主要依据 CMD 译码逻辑和 PCI 总线请求生成从 PCI 接口的控制时序,状态机在当前地址访问错误或 Backend 忙于发送或接受数据的情况下终止当前操作或请求重试操作[7]。

图2 所示为从模式状态机的状态转换图,为了简明,没有画出各状态在自身停留的转移过程,根据从 PCI 接口电路功能,将从模式状态机划分为九个状态:

(1)Idle 状态:总线空闲。

(2)Cmd_start 状态:开始地址命令译码。

(3)Check_addr 状态:检测地址是否匹配(对于延时读写操作,当 master 再次请求数据传输时,检测地址是否匹配)。

(4)Check_abort 状态:检测目标设备是否请求终

止操作。

(5)Start_single 状态:开始单次读/写。

(6)Start_multiple 状态:开始突发读/写。

(7)Start_retry 状态:Master 重试操作。

(8)Start_abort 状态:目标设备终止总线操作。

(9)Start_wait 状态:Master 读写等待。

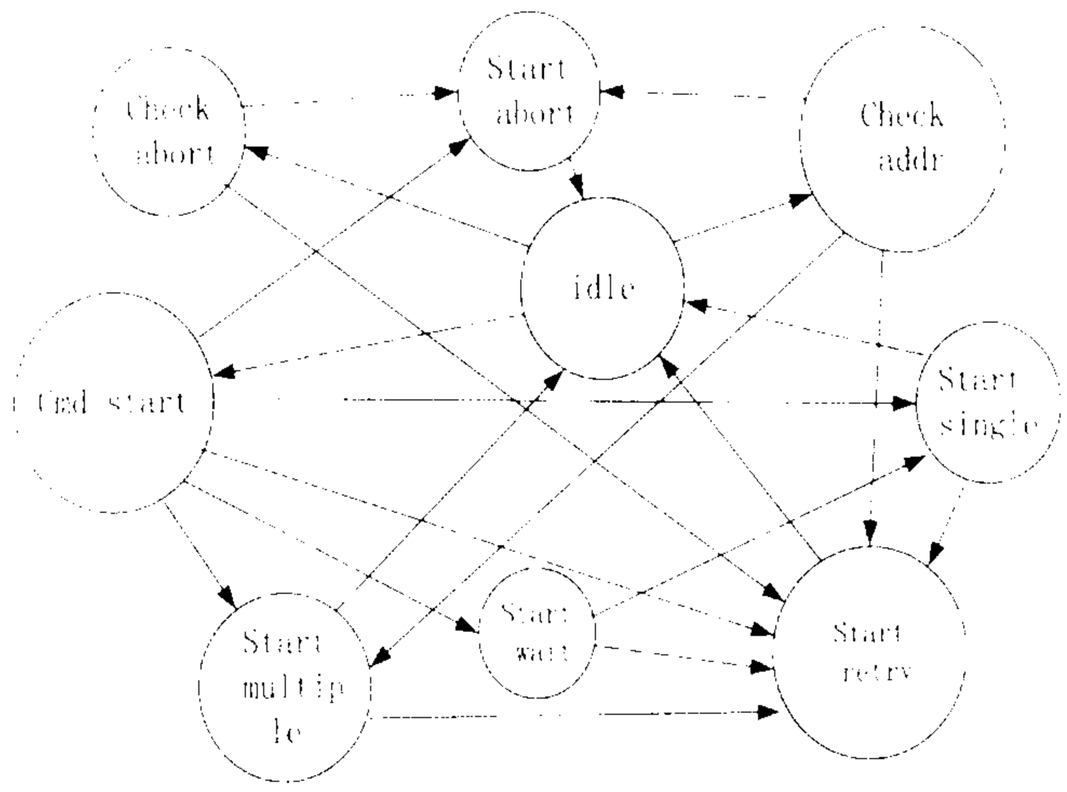

图 2　PCI 总线从模式状态转换图

从图 2 中可以看出,状态机主要实现对内部控制寄存器和目标设备的读写。对内部控制寄存器的读写只支持单次传输,而对目标设备及内部 FIFO 的读写支持单次传输和突发传输。在传输过程中,可能会产生重试操作,主要有三种情况:首先是出现无效地址模式时(对存储空间执行操作时,地址的后两位不为“00”)。其次是当执行突发传输时,地址超过了地址空间范围。再次是总线主设备进行写操作而 RX_FIFO 处于满状态或者总线主设备进行读操作而 TX_FIFO 处于空状态。一次基本的单次传输状态转移过程为 Idle→Cmd_start→Start_single→Idle,一次基本的突发传输状态转移过程为 Idle→Cmd_start→Wait→Start_multiple→Idle。

本设计支持交易终止(Transaction Termination)的具体操作为向 PCI 总线发出断开信号(disconnect),方式有 start_abort、start_retry,对于 start_abort 状态,进入此状态的方式主要有两种,首先当前状态处于 idle 状态,如 PCI 主机空闲,且 BIOS、MEM、IO 或配置空间访问操作正在进行中,此时 frame#拉低,或者目标设备已被锁定,其它主机对其访问将被终止,在这两种情况下,状态机将进入 check_abort 状态。而在此状态下,当发生地址、命令奇偶校验错误,Backend 强制终止或 IO 操作发生错误时将直接进入 start_abort 状态。其次当前状态为 cmd_start 状态,如果同样产生上述引起终止的条件,将直接进入 start_abort 状态。而对于 start_retry 状态,进入此状态的方式共有五种:对于当前为 cmd_start、check_addr 或 check_abort 状态,如果当前操作为延时读操作,此时 Backend 逻辑正进行写操作,则进入 start_retry 状态。对于当前为 start_single 状态,如 PCI 主机需要继续数据传输(frame#,irdy#同时拉低),则进入 start_retry 状态。而对于当前为 start_multiple 状态,当出现无效地址时即进行重试操作,如果 PCI 总线仍需继续数据传输,且进行突发传输时,地址超出地址边界或者 PCI 主机进行写操作时,RX_FIFO 几乎满,则进行重试操作,进入 start_retry 状态。

对于进行中的突发传输,如果 PCI 总线仍需数据传输(frame#,irdy#同时拉低),对于读操作如果 TX_FIFO 非空或者对于写操作如果 RX_FIFO 非满,则不需要插入等待状态(start_wait)。而对于立即读模式,如果 TX_FIFO 处于空状态或者读空时需要插入等待状态。另外,对于突发传输,如果对目标设备的操作不在地址空间范围内(目标设备地址空间设置为 64K),则将断开此次操作,中止本次数据传输。同时,如果当前操作需要进行重试,则当前地址和命令需要不断保存,并与之前所保存的不断进行比较,判断是否匹配。

本设计根据 PCI 总线规范设计有限状态机,能有效地实现 PCI 总线的各种基本操作,从而在使用 Verilog HDL 实现 PCI 总线从接口电路的设计时更加便利,并且代码更加清晰和易于维护。

2　验证平台

对于设计结果是否符合设计所定义的功能期望,电路的逻辑功能是否有效,需要通过仿真来验证[8]。仿真贯穿设计的整个过程,为了进行仿真,编写验证平台(TestBench)是必不可少的步骤[9]。

针对 PCI 时序的复杂性,编写相应的验证平台很难全面地将 PCI 协议的各个细节兼顾,这给模拟仿真带来了较大的复杂性和不确定性[10]。为了简单有效地对设计进行全面的验证,文中采用了 Synopsys 公司提供的 PCI/PCI-X FlexModels 行为级仿真模型,并根据验证平台的需要配置成 PCI/PCI-X 模型,包括 pci_master_fx, pci_slave_fx, pci_monitor_fx 等一系列组件,且提供了模型库[11]。而 Mentor 公司提供的 Test System 所包括的 core_target_reg, core_target_fifo 等组件也有效模拟了 Backend 逻辑端的行为,响应 master 发起的总线操作。

文中利用 PCI/PCI-X FlexModels 模型及 Test System 共同组建验证平台,从而实现了一个系统的验证环境,由 Master_model 发起总线操作启动 PCI 接口进行数据传输,而 core_target_reg, core_target_fifo 等负责响应总线操作。如图 3 所示,PCI-slave interface 为文中的设计。设计者只需通过编写命令语句产生激励来控制 Master_model,发起所需的 PCI 总线操作。最后通

过对仿真波形的观察,分析待验证的模块是否符合PCI协议和设计需求。

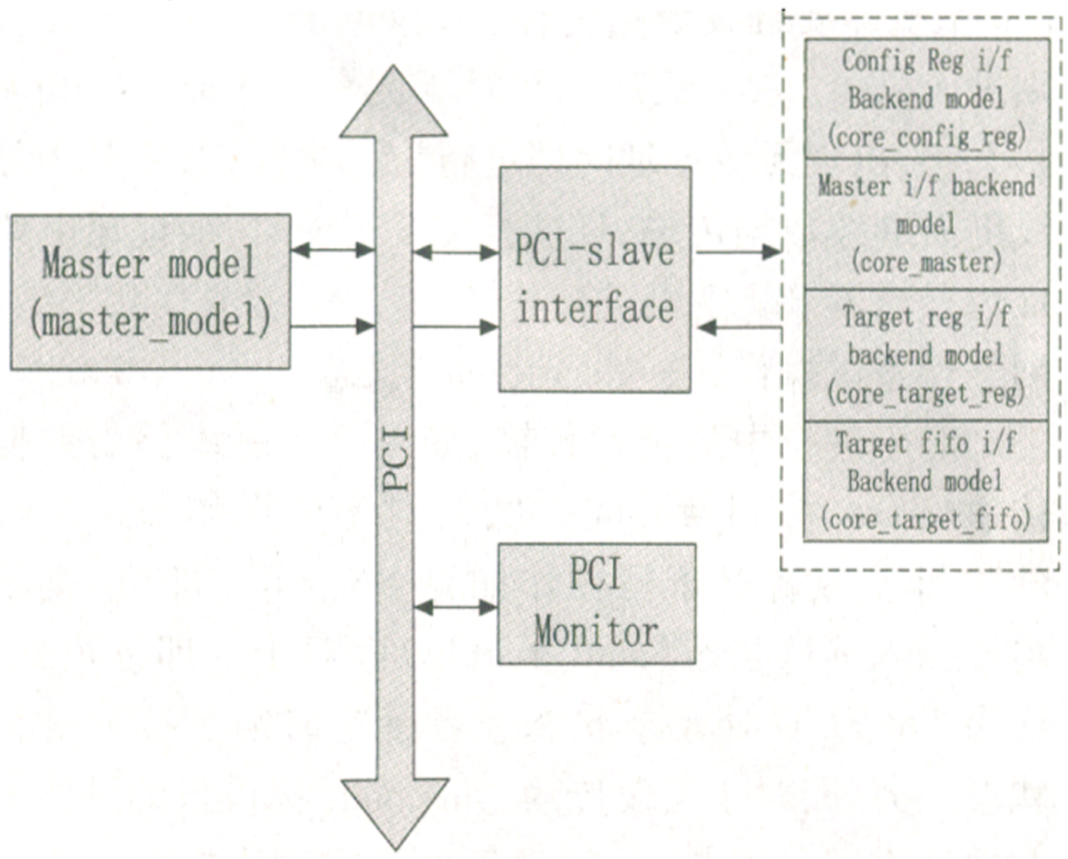

图3 验证系统示意图

在图3所示的验证平台中,只需使用HDL command stream方式将命令语句直接写在验证平台的Verilog HDL程序中,控制Master_model的操作,产生配置空间读写、存储器空间读写、I/O空间读写、DMA读写等总线操作。Command stream实际是调用存在于库文件中一些过程(procedure),用户根据验证的需要加入或者修改相应的参数,即可产生所需的激励条件[12]。例如,为了验证对I/O空间的访问(不支持突发传输),通过I/O空间写命pci_cbe==0011,向pci_ad==32'h1000_0000的从设备的I/O空间写入数值32'h0000_1111。命令语句如下,时序仿真结果如图4所示。

pcimaster_write_cycle(id_10, PCIMASTER_IO_WRITE,'h10000000,1,8'h00,32'h00001111,0,FLEX_FALSE,FLEX_WAIT_F,status)

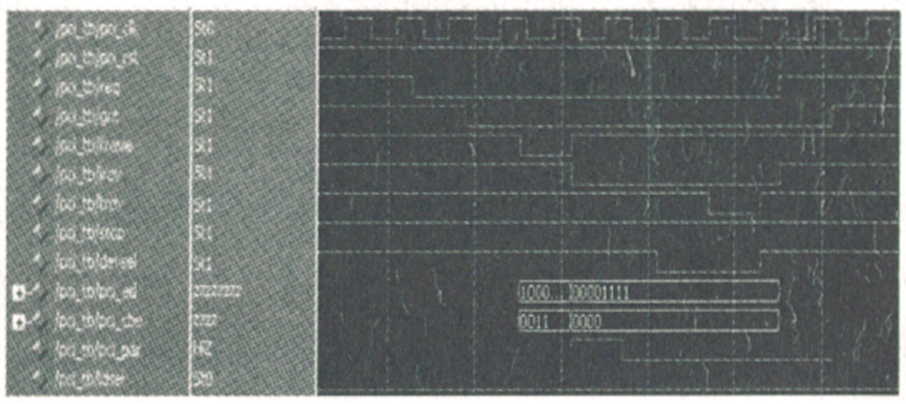

图4 对I/O空间的单次写操作

为了验证对本地存储设备的读写操作(支持突发操作),通过存储器空间写命令pci_cbe==0111向pci_ad==32'h2000_0004起始的从设备存储器空间,连续写入32'h2222_0000、32'h2222_0001、32'h2222_0002、32'h2222_0003、32'h2222_0004。命令语句如下,时序仿真结果如图5所示。

pcimaster_write_cycle(id_10, PCIMASTER_MEM_WRITE,'h20000004,5,8'h00,32'h22220000,0,FLEX_FALSE,FLEX_WAIT_F,status);

pcimaster_write_continue(id_10,8'h00,32'h22220001,0,FLEX_WAIT_F,status);

pcimaster_write_continue(id_10,8'h00,32'h22220002,0,FLEX_WAIT_F,status);

pcimaster_write_continue(id_10,8'h00,32'h22220003,0,FLEX_WAIT_F,status);

pcimaster_write_continue(id_10,8'h00,32'h22220004,0,FLEX_WAIT_F,status);

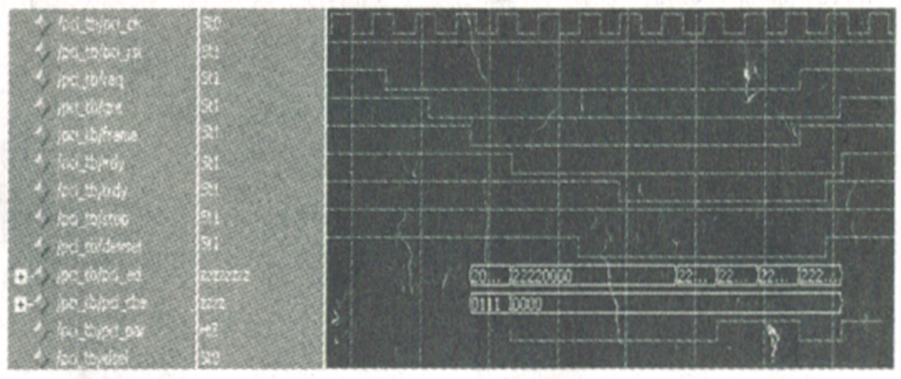

图5 对存储设备的突发写操作

3 结束语

通过对PCI协议及计算机接口技术的深入研究,讨论了PCI从模式的Verilog HDL设计方法,包括顶层结构的系统架构及各子模块的原理和功能,并实现了有限状态机的实时控制、寄存器控制、数据缓存、地址溢出检查等设计方法。并且组建了有效的验证平台,阐述了组建方法,对本设计进行了全面的仿真验证。

参考文献:

[1] 李贵山,陈金鹏. PCI局部总线及其应用[M]. 西安:西安电子科技大学出版社,2003.

[2] PCI Local Bus Specification Revision3.0[S]. [s.l.]:PCI Special Interest Group,2004.

[3] 胡和平,田宜波. 基于FPGA的PCI接口设计[J]. 计算机工程,2003,29(8):31-33.

[4] PCI Megacore Function User Guide, Version 2.3.0[M]. [s.l.]:ALTERA,2003.

[5] 宋克柱,杨小军,王砚方. 基于FPGA的PCI接口设计[J]. 电子技术应用,2001(9):85-89.

[6] Synopsys Inc. Simulator Configuration Guide for Synopsys Models[M]. [s.l.]:Synopsys Inc.,2001.

[7] 李均盛,王省书,胡春生,等. 基于FPGA的PCI从接口设计[J]. 工业控制计算机,2010,23(1): 137-141.

[8] 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,2008.

[9] 周 多,陈章进,郑昌陆. PCI协议接口的设计与验证[J]. 微计算机信息,2005,21(5):77-80.

[10] 何 妍,李树国,羊性滋. 一种具有双PCI接口的ASIC验证系统[J]. 微电子学,2006,36(2):171-175.

[11] Synopsys Inc. PCI/ PCI-X FlexModel User's Manual[M]. [s.l.]:Synopsys Inc.,2001.

[12] 刘道煦,田书林,王 毅. PCI总线接口的VHDL设计与验证[J]. 计算机测量与控制,2006, 10(8):276-281.

淋浴箱组故障自诊断语音报警系统的设计

作者：陈晓东， 王洪喜
作者单位：西安工业大学,陕西西安710032
刊名：计算机技术与发展
英文刊名：Computer Technology and Development
年，卷(期)：2012(8)

参考文献(12条)

1. 李贵山;陈金鹏 PCI局部总线及其应用 2003
2. PCI Local Bus Specification Revision3.0 2004
3. 胡和平;田宜波 基于FPGA的PCI接口设计[期刊论文]-计算机工程 2003(08)
4. PCI Megacore Function User Guide,Version 2.3.0 2003
5. 宋克柱;杨小军;王砚方 基于FPGA的PCI接口设计 2001(09)
6. Synopsys Inc Simulator Configuration Guide for Synopsys Models 2001
7. 李均盛;王省书;胡春生 基于FPGA的PCI从接口设计[期刊论文]-工业控制计算机 2010(01)
8. 夏宇闻 Verilog数字系统设计教程 2008
9. 周多;陈章进;郑昌陆 PCI协议接口的设计与验证 2005(05)
10. 何妍;李树国;羊性滋 一种具有双PCI接口的ASIC验证系统[期刊论文]-微电子学 2006(02)
11. Synopsys Inc PCI/ PCI-X FlexModel User 's Manual 2001
12. 刘道煦;田书林;王毅 PCI总线接口的VHDL设计与验证[期刊论文]-计算机测量与控制 2006(08)