

基于 PLB 总线的 NOR FLASH 控制器设计

陈超文, 彭国杰, 王忆文, 李 辉

(电子科技大学 微电子与固体电子学院, 四川 成都 610054)

摘 要: 为了使不同型号的 NOR FLASH 都可以高效率地与 PLB 总线交互, 提出了一种软硬件相结合的方法实现接口控制器. 仿真结果表明, 该控制器可以正确高效地实现 NOR FLASH 的各种功能. 整个控制器最后在 Xilinx ml507 评估平台上通过了验证, 实现了对 NOR FLASH 的正确操作.

关键词: 高效率; PLB; NOR FLASH; 控制器

中图分类号: TN47

文献标识码: A

文章编号: 1000-7180(2014)05-0083-04

Design of NOR FLASH Controller Based on PLB General Bus

CHEN Chao-wen, PENG Guo-jie, WANG Yi-wen, LI Hui

(School of Microelectronics and Solid-State Electronics,
University of Electronic Science and Technology of China, Chengdu 610054, China)

Abstract: In order to make different types of NOR FLASH can be interacted with the PLB bus efficiently, a combination method of hardware and software is put forward in the design to implement the interface controller. Simulation results show that the controller is able to implement various functions of the NOR FLASH correctly and efficiently. Finally, the controller is achieved to control the NOR FLASH correctly through the Xilinx ml507 evaluation platform.

Key words: high efficiency; PLB; NOR FLASH; controller

1 引言

随着便携式电子产品的普及, NAND FLASH 以其价格低廉的优势被越来越多的用于大容量数据存储^[1-2]. 然而, NOR FLASH 拥有的高传输效率、高可靠性和易于使用等优点是 NAND FLASH 无法取代的^[3]. 正是基于这个原因, NOR FLASH 被广泛应用于各种嵌入式系统的代码存储中.

处理器与 NOR FLASH 通过 PLB 总线^[4]进行交互时需要接口支持. 采用纯硬件的方式实现控制器有其局限性, 因为不同型号的 NOR FLASH 命令不尽相同, 控制器所能支持的命令又受其状态机的限制. 采用纯软件的方式虽然可以解决这个问题, 但是速度慢, 不适合于高速通信系统中. 为了兼顾 IP 复用性^[5-7]和高效率传输两个方面, 在本设计中采用软硬件相结合的方法^[8]来实现基于 PLB 总线的

NOR FLASH 控制器. 整个设计在 Xilinx ml507 评估平台上正确实现了对 NOR FLASH 的高效控制.

2 NOR FLASH 基本操作

NOR FLASH 的接口信号主要由双向数据总线 DQ、地址总线 ADDR 以及 3 个主要的控制信号: CEn、OEn 和 WEn 组成. 总的来说, 对任意型号的 NOR FLASH 进行读、写、擦除、解锁、上锁等一系列操作都可以转化为向 FLASH 写入命令字和从 FLASH 读取数据两种操作.

2.1 读操作

NOR FLASH 读操作相对比较简单, 其时序如图 1 所示. 其中, R1 为一个读周期的时间, R2 为地址到输出开始有效的的时间, R3 为芯片使能到输出开始有效的的时间, R4 为输出使能到输出开始有效的的时间.

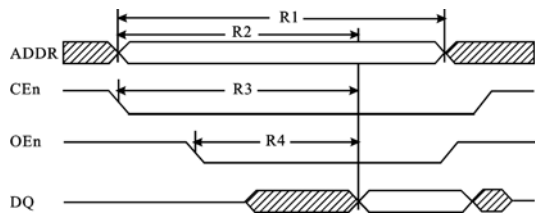


图 1 读操作时序

由于不同型号的 NOR FLASH 读操作时序不尽相同,但是在进行读操作的时候输出使能和芯片使能这两个主要的控制信号之间一般没有时序要求,所以在设计控制器的时候只要把 R1 参数化即可满足大多数不同型号 NOR FLASH 的读操作要求。

2.2 写操作

写操作时序如图 2 所示.从时序图可以看出,各个信号的持续时间都是以写使能信号的变化为中心.一般的, W2 和 W7 的最小持续时间可以为 0. 所以,在设计控制器的时候,需要着重关注的是 W3, W4 和 W5. 其中 W3 是芯片使能信号有效到写使能信号有效的建立时间, W4 是写使能信号有效的持续时间, W5 是写使能信号无效时芯片使能信号还要保持有效的保持时间. 在控制器的设计中把这三个时间参数化即可满足大多数不同型号 NOR FLASH 的写操作要求。

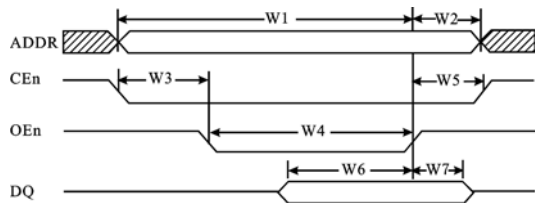


图 2 写操作时序

3 NOR FLASH 控制器设计

为了兼顾 IP 的复用性和传输的高效性两个方面,控制器的设计采用软硬件相结合的方法实现,即硬件部分负责时序的转换,软件部分负责命令的发送. 这样,由于命令是由软件发出的,将不会像用纯硬件方式实现的那样受到状态机的限制;同时由硬件完成数据传输保证了其读写效率。

虽然,命令的发送是由软件来完成的,会占用一定的 CPU 时间;但是从总体上看,在与 NOR FLASH 通信过程中的读写操作占据了绝大部分比例,而这部分是由硬件来完成的,相对而言,软件部分只占用了很少的时间. 所以,与采用纯硬件方式实现的控制器相比,软硬件相结合的方式虽然牺牲了

一小部分时间性能,却换来了对不同 NOR FLASH 型号的支持. 所以,这种划分方案是可行的. 此外,由于硬件只需要负责执行时序转换,整个控制器通过软件的命令字控制不断重复调用硬件去执行读写操作,大大简化了硬件实现。

3.1 硬件设计

硬件部分只需要负责时序的转换,整个 NOR FLASH 控制器的硬件部分框图如图 3 所示. 读写时序状态机和寄存器等都是在同步时钟的控制下工作的,为了使整个硬件部分框图看起来更加简洁明了,在图中没有画出时钟信号和复位信号的连接。

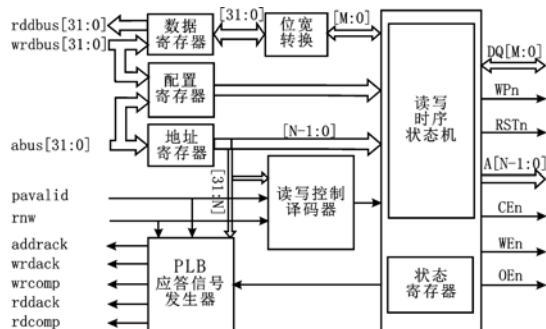


图 3 NOR FLASH 控制器硬件部分框图

一般的, PLB 总线信号和 NOR FLASH 信号位宽不同,所以需要先进行位宽转换. PLB 地址总线的高位用于译码判断当前的操作是不是对 FLASH 进行操作. 读写控制译码器通过采样 PLB 地址总线的高位 $abus[31:N]$ 、读写信号 rnw 和指示总线周期有效性的信号 $pavalid$ 来译码产生控制信号送给时序状态机以确定是进入读时序还是写时序. PLB 应答信号发生器通过采样 $abus[31:N]$ 、 rnw 、 $pavalid$ 和状态寄存器的信号来适时的给处理器以地址应答、读应答、写应答等反馈信号. 硬件部分包含了 PLB 应答信号发生器使得 FLASH 可以通过该控制器直接挂载在 PLB 总线上而不再需要其他接口转换模块,方便于与系统中的其他模块通信. 配置寄存器是针对不同型号的 NOR FLASH 而设置的,主要用于配置读写时序中的时间参数、数据位宽和地址位宽。

3.2 软件设计

软件部分只需要负责命令字的发送. 软件编程时只需要按照读、写、擦除等操作的流程图编程即可. 本文以 Intel JS28F256P30T95 NOR FLASH 为例叙述控制器软件部分的设计. 图 4 所示为字写操作的流程图. 下面就以字写操作为例给出示意代码,其他操作与此类似。

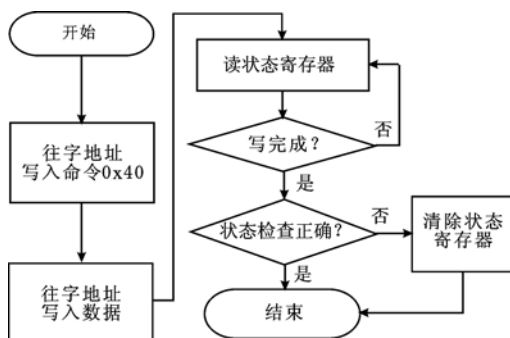


图4 字写操作流程

```

typedef unsigned int u32;
typedef unsigned short u16;
#define word_write_cmd 0x40
#define clr_SR_cmd 0x50
#define complete_mask 0x80
#define write_error_check_mask 0x1a
u16 flash_word_write(u32 address, u16 data)
{
    u16 SR;
    u32 result;
    /* 先写入写操作命令字 0x40 */
    * (u16 *) (address) = word_write_cmd;
    * (u16 *) (address) = data; // 写数据
    do
    {
        SR = * (u16 *) (address); // 读状态
    }
    /* 如果 SR[7] 不为 1, 说明写操作未完成 */
    while (! (SR & complete_mask));
    /* 如果写出错, 执行清除状态寄存器命令 */
    if (SR & write_error_check_mask)
    {
        * (u16 *) (address) = clr_SR_cmd;
        return SR;
    }
    else return 0;
}

```

状态寄存器有 8 位, 其默认值为 0x80。写操作完成与否主要检查状态寄存器的最高位 SR[7] 是否为 1, 所以程序中只需要把读回来的状态寄存器的值与 0x80 相与就可以判断 SR[7] 是否为 1。至于写的的数据对不对, 还需要检查 SR[4:3] 和 SR[1] 是否都为 0。同理, 在程序中把状态寄存器的值与 0x1a 相与即可。如果不为 0 说明写数据出错, 需要执行清除状态寄存器的命令。

4 实验验证

ml507 是一款通用的 FPGA、RocketIOTM GTX 和嵌入式系统开发板, 如图 5 所示。本设计主

要用到开发板上的 FPGA、SRAM 和 NOR FLASH 等资源。

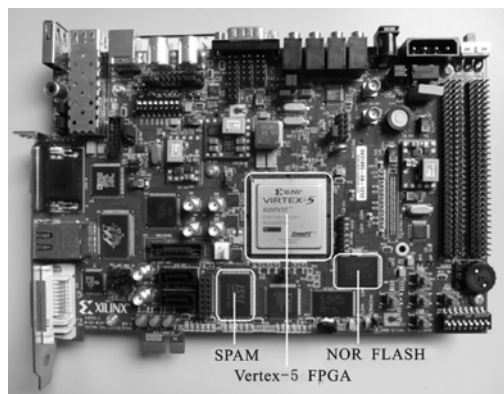


图5 ml507 评估平台

整个控制器最后在 Xilinx ml507 评估平台上搭建的 PowerPC 系统里进行了详细的验证, 验证系统框图如图 6 所示。虚线框部分是由 FPGA 实现的, NOR FLASH 控制器的硬件部分在图中已经标明, 软件部分则由 PowerPC 实现。

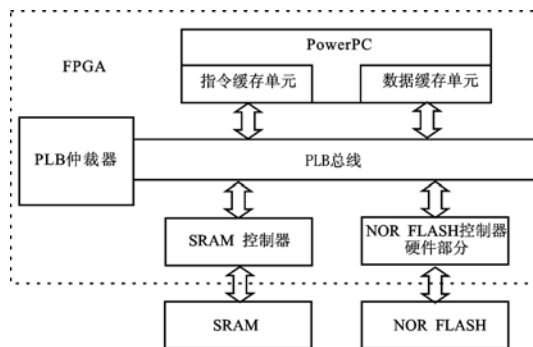


图6 PowerPC 验证系统

通过软件编程控制, 实现了对 NOR FLASH 异步字读、字编程、32 字缓冲编程、块擦除、解锁块、锁块等操作。表 1 给出了一些主要操作的实测执行时间结果。这些实测的时间和数据手册给出的参考数据相当。表 2 给出了 SRAM 中地址范围从 0xFFFFE_0000 到 0xFFFFE_0007 的数据写入到 NOR FLASH 和从 NOR FLASH 读回数据到 SRAM 中地址范围 0xFFFFE_1000 到 0xFFFFE_1007 的对比, 实验结果表明读写操作功能正确。

表1 测试结果

操作	时间
字读	120 ns
字编程	163 μ s
32 字缓冲编程	394 μ s
128 kB 块擦除	1.7 s
锁块	840 ns
解锁块	750 ns

表 2 读写数据对比

SRAM_to_FLASH		FLASH_to_SRAM	
地址	写	地址	读
0xFFFE_0000	0x12	0xFFFE_1000	0x12
0xFFFE_0001	0x34	0xFFFE_1001	0x34
0xFFFE_0002	0x7C	0xFFFE_1002	0x7C
0xFFFE_0003	0x85	0xFFFE_1003	0x85
0xFFFE_0004	0x0F	0xFFFE_1004	0x0F
0xFFFE_0005	0Xac	0xFFFE_1005	0xAC
0xFFFE_0006	0x6B	0xFFFE_1006	0x6B
0xFFFE_0007	0x2E	0xFFFE_1007	0x2E

5 结束语

为了使所设计的 NOR FLASH 控制器具有良好的兼容性和高效率,本文提出了一种采用软硬件相结合的方案实现控制器.实验验证表明,该控制器不仅适用于高速通信系统中,而且对于不同型号的 NOR FLASH,只需要修改硬件中的配置寄存器和软件中的命令参数即可实现移植,控制器具有良好的复用性.

参考文献:

[1] 梁宏明,胡东伟,陈杰. SOC 中 NAND Flash 控制器的

一种新型结构和实现方法研究[J]. 微电子学与计算机,2009,26(6):89-92.

[2] Eyee Hyun Nam, Bryan Suk Joon Kim, Hyeonsang Eom, et al. An Out-of-Order Flash Memory Controller Architecture[J]. IEEE Transactions on Computers, 2011,60(5):653-666.

[3] 王顺. 基于 OPB 总线的 NAND FLASH 控制器设计与验证[D]. 长沙:湖南大学,2009.

[4] 彭维. 基于 PLB 总线的用户 IP 核接口技术研究[D]. 安徽:合肥工业大学,2011.

[5] 夏宇闻. Verilog 数字系统设计[M]. 第 2 版. 北京:北京航空航天大学出版社,2008.

[6] HARRIS David Money, HARRIS Sarah L. Digital design and computer architecture[M]. USA: Morgan Kaufmann, 2007.

[7] 王彬,任艳颖. 数字 IC 系统设计[M]. 西安:西安电子科技大学出版社,2005.

[8] 肖鹏,刘新宁,杜洪华. 基于 AMBA 总线的 NAND FLASH 控制器软硬件划分设计[J]. 现代电子技术, 2008, 31(2):1-3.

作者简介:

陈超文 男,(1989—),硕士研究生. 研究方向为 SOC/SIP 系统芯片技术.

(上接第 82 页)

[7] 郑娟,冯书兴,曹裕华. 军事概念模型分阶段分层次开发过程研究[J]. 计算机仿真,2007,24(7):9-12.

[8] Dms0. Conceptual models of the mission space (CMMS) technical framework[R]. USA: Defense Modeling and Simulation Office,1997.

[9] IEEE Computer Society. IEEE Recommended Practice

for Software Requirements Specifications[S]. IEEE Std 830-1998: Los Alamitos, CA, 1998.

作者简介:

丁晓剑 男,(1982—),工程师. 研究方向为概念模型、系统仿真技术.