

基于协议控制器的 DDR3 访存控制器的设计及优化

陈胜刚, 付兴飞, 曾 思

(国防科学技术大学 计算机学院, 湖南 长沙 410073)

摘 要: 为了完成 SoC 系统中片外 DDR3 存储器高效访问, 实现了基于协议控制器的 DDR3 访存控制器. 针对 SoC 系统层访问事务属性与 DDR3 存储器突发访问属性要求之间的不匹配问题, 设计了一种访问请求平滑处理机制, 充分利用协议控制器的 DDR3 访问命令缓冲, 用更少的 DDR3 存储器访问请求完成 SoC 上层读写事务, 提高了整个 DDR3 访存控制器的访问效率. 采用 Verilog 语言完成了逻辑设计和物理实现, 并采用人工综合激励的方法对访问请求平滑机制进行了评估, 实验证明该优化机制可对 DDR3 访存控制器实现有效加速.

关键词: DDR3; 访存控制; 请求平滑

中图分类号: TP331. 2

文献标识码: A

文章编号: 1000—7180(2016)06—0006—05

Design and Optimization of a DDR3 Memory Controller with Protocol Controller

CHEN Sheng-gang, FU Xing-fei, ZENG Si

(College of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract: A DDR3 Memory controller with protocol controller is implemented to efficiently access the off-chip DDR3 memory in a SoC system. Considering the unmatched property between the SoC system level read-write transactions and the DDR3 SDRAM burst access requirement, a memory transaction smoothing mechanism is implemented in the DDR3 memory controller, which makes well use of the protocol controller's DDR3 SDRAM memory command pipeline and can reduce the times of the actual off-chip SDRAM access to improve efficiency of the entire DDR3 memory controller. Verilog model and physical implementation of the memory controller are carried out. Evaluation based on synthetic stimulus show that the smoothing mechanism can improve the efficiency of the DDR3 memory controller.

Key words: DDR3; memory controller; transaction smoothing

1 引言

DDR3 SDRAM^[1] (下文简称 DDR3) 是 SoC 系统中常用的一种片外存储器^[2], 其主要优点是成本低、集成度高、容量大、速率快、峰值带宽高. DDR3 在较高的频率下采用源同步的并行总线传输信号, 这对时序和信号完整性的要求极高. 因此, SoC 系统直接集成专用的 DDR3 协议控制器 (包含协议层控制器和物理层 PHY). 协议控制器往往只能将系统读写请求按照顺序和个数原样转换为 DDR3 命令

序列去执行, 并对 DDR3 命令完成简单的调度, 即完成读写请求到 DDR3 标准协议之间的协议转换工作. 更为复杂的工作都在上层 DDR3 存储控制器中完成.

DDR3 存储控制器的主要任务需要完成 SoC 系统层与协议控制器之间的对接、系统层读写事务相关性的控制 (强序)、DDR3 访问效率的提升等工作. DDR3 峰值带宽较高但实际效率与读写事务发生模式关系较大, 因此存储控制器的优化工作显得极为重要.

为了完成与SOC系统层的对接,DDR3存储控制器多采用标准接口协议AMBA AXI^[3];同时考虑到系统实现开销,总线位宽一般在128 bit左右或者更小,且支持突发模式和多种访问粒度。DDR3存储体是采用固定的突发访问模式(一般为8)^[1]。在图像和视频等应用中,应用层的数据粒度一般较小且呈矩阵连续分布。协议控制器将这种小粒度的请求直接翻译为DDR3命令序列,会引起对DDR3存储体多次冗余访问,造成DDR3带宽的进一步下降。

针对DDR3控制器设计中SOC系统层访问属性与DDR3存储体突发属性之间的不同,本文设计了一种具备请求平滑功能的DDR3存储控制器,以提高DDR3处理此类问题的效率。

2 DDR3 访存及优化

DDR3存储体组织方式是行(Row)、列(Column)、Bank(体)的三维方式,其中Bank通常为8,一行也称为一个页面(Page)。为提高带宽和DDR3的可用性,DDR3颗粒往往封装为位宽更宽的双列直插式存储模块(Dual Inline Memory Modules, DIMM)的形式使用,每个DIMM上通常包含多个DDR3芯片,位宽通常为64 bit或者32 bit。

DDR3采用动态随机存储器(Dynamic Random Access Memory, DRAM)存储数据,其访问方式与静态随机存储器的直接读写不一样,需要相对复杂的操作序列才能完成读写事务^[4],且需要定期刷新以保持数据。正因为如此,DDR3在访问某一页面的时候,需要先将其数据线预充电(称为打开页),然后才能进行访问;当需要切换到另外一个页面的时候,需要将之前打开的页关闭(数据写回)。在打开和关闭页面的过程中是不能进行读写操作的,且充放电过程较慢,耗时较长。这是DDR3访问效率低的主要原因。现代DDR3提供多个Bank,每个Bank同时支持打开一个页面,这从一定意义上提高了DDR3访问效率;同时DDR3提供突发访问属性,以进一步提高访问效率。

DDR3存储控制器提高DDR3效率的着手点主要是如何优化对打开页面的访问、尽量降低切换页面的可见开销。主要的方法包括设置虚拟打开页、进行任务级调度^[5-6],或者存储命令调度^[7]等。虚拟打开页是在DDR3存储控制器中设置较大缓冲区域,用类似Cache的预取技术提前完成对DDR3的访问并保存访问结果,或延后执行真实的DDR3访问。但虚拟打开页开销巨大,因此研究较少。任务级调度

主要根据DDR3的特征,尽量将同一个页面的访问请求调度到连续时间内完成,从而尽量减少页面切换的开销。任务调度的有效性受限于请求发生的模式以及读写强序要求,且需要较为复杂的控制逻辑,将加大单个请求的完成时间。存储命令调度则是将任务级读写事务转换为存储命令序列后,对存储命令进行一定的调度。比如Activate命令^[1]可以提早发出,有些关闭存储体的Precharge命令可以省略,不同的存储体可以打开多个页面等。存储命令调度要遵循严格的DDR3协议的定时要求以及读写请求的强序要求,可调度范围不大,且一般需要结合协议控制器完成。

3 基于协议控制器的DDR3访存控制器

图1为本文设计的基于协议控制器的DDR3访存控制器的结构示意图。其中,应用层访存请求接口为AXI协议接口,支持突发读写,数据粒度从1 Byte到32 Byte可变。访存请求控制单元负责与应用层AXI端口进行握手,将请求转发给协议控制器,并同时完成一些访存控制。请求平滑处理机制将多个小粒度的AXI访存请求转换为大粒度的协议控制器请求,提高DDR3访存效率。协议控制器将上层处理后的读写请求转换为符合协议控制接口协议的读写请求。协议控制器将上层的读写请求转换成相应的DDR3存储命令序列,放入DDR3存储命令队列中,并根据访问时序要求,进行一定程度的调度,然后将这些命令执行并发送给DDR3 DIMM,并接收来自DDR3 DIMM的读返回数据,返回给上层访存控制器。数据返回控制和重排序缓冲根据应用层请求的顺序和粒度,将数据依次返回给应用层。本文将协议控制器和DIMM一起称为DRAM系统。

3.1 DDR3 存储命令调度

协议控制器将接收到的有效R/W请求扩展为DDR3命令序列(Memory Command Sequence),并存储在DDR3存储命令队列(Memory Command Pipeline, MCP)中。命令之间的时序是按照DDR3的标准存储协议规则来定时的。每个周期控制器将队列头的命令发送给DIMM执行。根据DDR3协议,存储命令包括以下三个:Precharge(P)、Activate(A)与Column(C),其中Column命令可能为Read或者Write命令,同时使能(Cp)或者未使能(C)Auto Precharge过程。如图2所示。

图2给出了一种可能的DDR3存储命令序列。其中第 n 个命令序列对应一个命中当前打开页面的

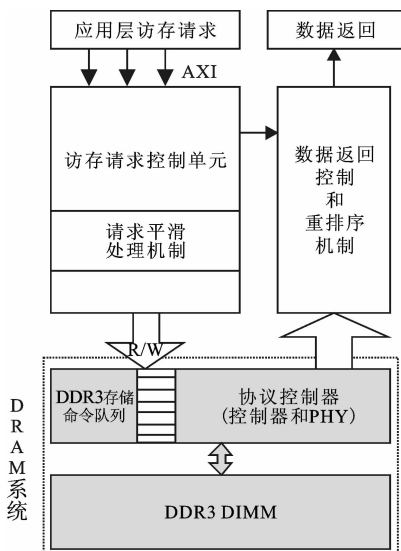


图1 基于协议控制器的DDR3访存控制器结构示意图

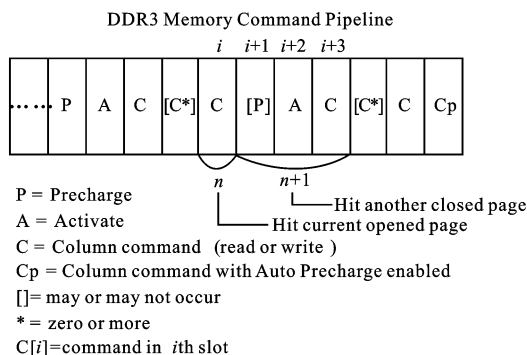


图2 一种可能的DDR3存储命令序列

R/W 请求,仅为一个 $C[i]$ ($C[i]$ 表示处于第 i 个 MCP 槽的 C 命令) 命令,而第 $n+1$ 个命令序列则为命中另外一个未打开的页面的 R/W 请求,因此必须首先用 $P[i+1]$ 命令关闭当前打开的页面(若请求 n 和请求 $n+1$ 命中同一个 Bank),并随后用 A 命令激活命令的页面,再发出 C 命令进行正常的访问。通常 C 命令的顺序与 R/W 接口的读写顺序严格保持一致,但是 P/A 命令则在满足 DDR3 标准协议的前提下,会尽可能早地被发送给 PHY 执行,这有利于降低打开和关闭不同 Bank 的页面的开销。

通常情况下,R/W 的 Burst 请求是对连续地址的读写,会被翻译为多个 C/Cp 命令,占据多个的 DDR3 命令队列槽。降低 DDR3 命令序列中命令的个数,尽早让 P、A 命令处于被调度的窗口内,是提高 DDR3 系统访存效率的一种有效方法。

3.2 请求平滑机制

应用层请求包含两种属性:粒度(Data Size, DS)和长度(Burst Length, BL)。粒度决定了访问请

求的宽度,长度决定需要访问多少个 DS 大小的数据。SoC 的应用可能会向 DDR3 存储控制器发出多种 DS 和 BL 组合的读写请求。协议控制器在将 R/W 请求转换成 DDR3 命令序列的时候,会直接将 Burst 请求展开,忠实体现读写请求的意图。

假定应用层 AXI 请求的有效数据粒度 DS 记为 $w(\text{bit})$,一次 DDR3 burst 访问的数据量(即协议控制器 R/W 接口的有效数据粒度)记为 $W(\text{bit})$ 。因此, $DS=w/\text{BL}=n$ 的请求将会直接被翻译为包含 n 个 C 命令的 DDR3 访问序列,从而对 DDR3 DIMM 访问 n 次。而 DDR3 一次最小访问数据量为 W 等于 512 bit,即 8 个 64 bit DIMM 位置;若 w 小于 512 bit,那么协议控制器将通过掩码(写)或者数据丢弃(读)的方式处理非需求的位置或者数据。另外,DDR3 访问命令将造成对 DDR3 命令队列槽的耗费,从而降低 P 和 A 命令的调度效率。

根据 DDR3 存储命令队列的特征,本文在访存控制中设计了一种请求平滑机制,可以有效降低一个 R/W 请求对应的存储命令序列中的命令个数,提高 DRAM 系统访存效率。

应用层 AXI 请求访问的属性如下:

$$DS=w \quad BL=n \quad \text{StartAddr}=a,$$

其中,StartAddr 为字节地址

那么处理后的请求属性如下:

$$\begin{aligned} DS &= W \\ BL &= \text{ceil}((n * w + (a * 8) \% W) / W) \\ \text{StartAddr} &= a - (a \% (W / 8)) \end{aligned}$$

其中函数 $\text{ceil}(\ast)$ 的含义是取不小于 $\ast$ 的最小整数,即向上取整。

对 BL 的长度处理要考虑起始地址是否对齐的 W 的边界,若对齐,则 $BL=n * w/W$;否则需要考虑非对齐部分应该占一个突发长度。同时,起始地址必须设置为不大于原始地址的最大一个对齐 W 边界的地址。

图 3 给出了一个 $DS=3, BL=4, w=128, W=64 * 8=512$, $\text{StartAddr}=\text{baseA}+6 * 8$ (baseA 地址为 512 bit 对齐,一个 DDR 地址数据量为 8 Byte) 的应用层 AXI 请求在经过优化处理前后的访存过程。图中由于起始地址为 6 地址,这是一个非对齐地址,按照上面的变化法则得到 $DS=512 \text{ bit}, BL=3$, $\text{StartAddr}=\text{baseA}$ 。由于起始地址未对齐 W 边界,那么协议控制器将会对 0~5 地址的数据采取丢弃(读)或掩码写(写)的方式处理,不保证应用层能获得这几个地址的有效数据;若是写,则会通过 DDR3

的 MASK 方法^[1],不修改这几个地址的数据。

从图 3 可以看出,优化后访存次数将从 4 次降低为 3 次。这是一个最不高效率的请求转换的例子,若原始请求的 BL=5,处理后的请求仍然仅需 3 次 C

命令,从而对 DDR3 命令队列的占有率也会降低,增加协议处理器调度 P 和 A 请求的窗口,也加快了 DIMM 对读写请求的响应速率。

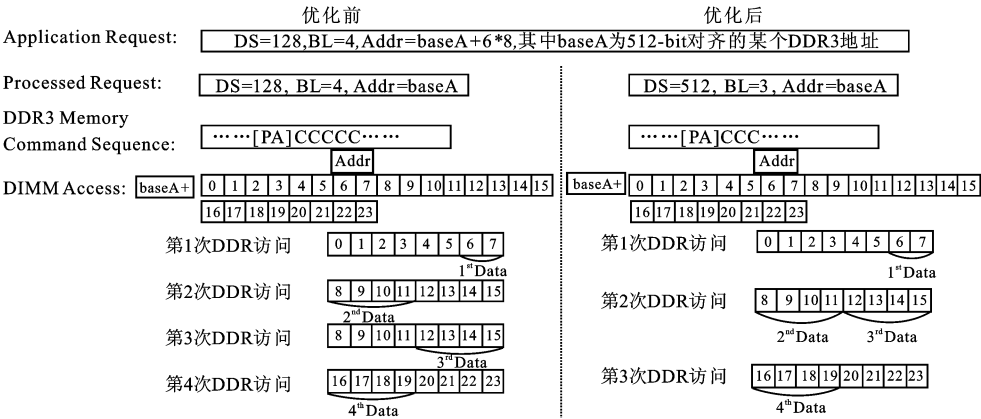


图 3 一个 DS=128 的请求的优化访问过程

3.3 数据返回控制

应用层一般期望存储控制器能保持请求的执行顺序,以及数据返回的顺序;同时保证返回数据与请求的粒度和长度保持一致。因此,需要对平滑处理后的返回数据进行解包处理。

数据返回控制单元用一个未完成读请求队列来保存原始读请求信息,并根据请求信息对返回数据队列中的数据进行分包处理,按照原始请求的需求返回给应用层。

同时,读数据返回控制单元还会根据当前返回数据队列和未完成读请求之间的关系对应用发起的读请求进行阻塞,防止请求接收过快,导致返回数据队列超负荷。如图 4 所示。

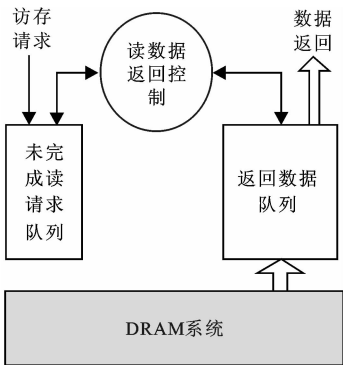


图 4 数据返回控制

4 性能评估

4.1 实验设置

本文用 Verilog 描述了本文所设计的 DDR3 访

存控制器,用 Micron 提供的 DDR3 仿真模型对其进行了模拟验证和性能评估。实际实现过程中,考虑到开销,将 W 设置为 256 bit, w 可取值为 8 bit, 16 bit, 32 bit, 64 bit, 128 bit 和 256 bit, BL 可从 1~16 变化。DIMM 的工作速率设置为 1 600 MT/s,应用层请求的工作频率设置为 500 MHz。

采用综合的激励产生方法对 DDR3 访存控制器的效率进行评估,激励涵盖了所有的粒度和长度的组合,并包含多种粒度和长度的随即组合。普通测试激励包含 1 000 组以上的请求,随机测试包含 2 000组以上的读写请求。

4.2 性能评估

表 1 给出当数据粒度为 64 bit,在变化不同的请求长度(BL)情况下请求平滑机制优化带来的加速比效果。从表中可以看出,当 BL 等于 1 的时候,使用请求平滑机制优化并不能带来好处,这是因为在这种情况下,请求的粒度和长度属性并不会发生变化;而当 BL 变大,请求的属性就会在请求平滑机制的优化下发生变化,访存次数降低,从而提高 DIMM 向应用层供数的能力。同时对于写请求,访存控制器从应用层接收数据的能力也会得到提升,从而提高性能。

表 2 给出了不同的粒度的访存请求的平均加速比。平均加速比是对同一个 DataSize 的不同 BL 下的加速比取平均值得到。从表中可以看到,除了 DataSize=256 bit 之外,其他的粒度的实验均取得了约 1.83 的加速比。这是因为在不同的粒度下,访存控制器从应用层接收数据所消耗的节拍,以及返

回给应用层所需要耗费的节拍与数据粒度的关系已经不大,因此才取得了几乎相同的加速比. 而对于粒度为 256 bit 的请求, ω 与 W 相等,无法达到进一步优化目的.

表 1 DataSize=64 bit,BL 不同值的实验结果

Burst Length	Speedup		
	连续读	连续写	读写交替
1	1.00	1.00	1.00
2	1.88	1.56	1.56
3	1.88	1.87	1.87
4	1.88	1.87	1.87
5	1.88	1.87	1.87
6	1.88	1.87	1.87
7	1.88	1.87	1.87
8	1.88	1.87	1.90
9	1.88	1.87	1.90
10	1.88	1.87	1.90
11	1.88	1.87	1.90
12	1.88	1.87	1.89
13	1.88	1.87	1.89
14	1.88	1.87	1.87
15	1.91	1.91	1.87
16	1.91	1.90	1.89
平均值	1.83	1.83	1.81

随机激励则是随机设置访存请求的粒度和长度属性,其实验结果表明也可达到约 1.59 的加速比.

表 2 不同粒度的访存请求的平均加速比

DataSize	Average Speedup
8-bit	1.83
16-bit	1.83
32-bit	1.83
64-bit	1.83
128-bit	1.83
256-bit	1.00
随机激励	1.59

4.3 实现结果

本文设计的 RTL 级代码基于 40 nm 工艺,使用 DesignCompiler 进行了综合. 综合的设置和结果如表 3 所示. 从表中可以看出,访存控制器整体实现开销较小.

表 3 综合设置和结果

项目	参数
工艺	40 nm
时钟	1.6 ns,50% 占空比
访存控制器面积开销(不包括协议控制器)	196 782.3 μm^2
功耗开销	56.7 mW

5 结束语

本文设计了一种基于协议控制器的 DDR3 访存控制器,并针对应用层访问属性与 DDR3 的突发访问之间的不匹配问题,设计了一种访问请求平滑机制,降低对 DRAM 系统的访问次数,从而增强协议控制器的访存能力,提高整个存储器系统的访存效率. 本文用 Verilog 对所设计的访存控制器进行了实现,并用 DC 进行了综合;采用人工综合的测试激励的实验结果证明该方法可达到 1.8 左右的加速比.

参考文献:

[1] JEDEC. JEDEC publishes DDR3 standard [EB/OL]. [2010-07-15]. <http://www.jedec.org/>.

[2] 郭炜,魏继增,郭笋,等. SoC 设计方法与实现[M]. 北京:电子工业出版社,2011.

[3] ARM corporation. AMBA AXI protocol Specification [EB/OL]. (2003-06-16). [2015-05-15]. www.arm.com.

[4] 宋焕章,王保恒,张春元. 计算机原理与设计(下册)[M]. 长沙:国防科技大学出版社,1999.

[5] Pan Guoteng, Luo Li, Ou Guodong, et al. Design and implementation of a DDR3-based memory controller [C]// Proceedings of Third International Conference on Intelligent System Design and Engineering Applications (ISDEA). China,Hongkong,2013: 540-543.

[6] Liu Defeng, Pan Guoteng, Xie Lunguo. Multi-bank memory access scheduler and scalability[C]//Proceedings of 2nd International Conference on Computer Engineering and Technology (ICET). China,Chengdu, 2010(V2):723-727.

[7] Zou Huihui, Ma Peijun, Shi Jiangyi, et al. The optimization and application of DDR controller based on multi-core system[C]// Proceedings of IEEE 11th International Conference on Solid-State and Integrated Circuit Technology (ICSICT). China,Xian,2012:1-3.

作者简介:

陈胜刚 男,(1981-),助理研究员. 研究方向为数字信号处理器设计. E-mail:shgchen@nudt.edu.cn.

付兴飞 男,(1988-),硕士研究生. 研究方向为微处理器设计.

曾 思 男,(1992-),工程师. 研究方向为微处理器设计.