DOI:10.19651/j.cnki.emt.2208937

基于 FPGA 的 DDR4 多通道控制器设计

翁天恒 袁永春 周 榕 李迎春 张俊杰
(上海大学特种光纤与光接入网重点实验室 上海 200444)

摘 要: 在网络通信和图像处理等系统中存在多个子系统同时访问外部存储器的情况,多通道存储控制器则可以有效地解决这个问题。然而随着数据量的激增和处理单元性能的提升,传统的多通道控制器带宽利用率低,难以满足系统对存储器高速存取的要求。因此,针对上述问题,本文设计了基于 FPGA 的 DDR4 多通道控制器。该控制器定义了简化的用户接口并支持网络通信中的循环缓冲区设置,降低了使用的复杂度,同时提高了设计的通用性。设计采用基于循环优先级仲裁器的系统转换结构,高效地解决多通道访问冲突问题,提升了系统的带宽利用率。此外,系统采用分片机制实现循环缓冲区内的访问回绕。基于 Xilinx KCU116 FPGA 的板级测试表明,本文所设计的多通道访问结构的测试结果与仿真一致。在访问长度为 4 069 字节时,系统最高有效带宽为 78.3 Gbps,带宽利用率达到 94.0%。

关键词: 多通道存储控制器;DDR4 SDRAM;FPGA
中图分类号: TN919 **文献标识码:** A **国家标准学科分类代码:** 510.50

Design of DDR4 multi-channel controller based on FPGA

Weng Tianheng Yuan Yongchun Zhou Rong Li Yingchun Zhang Junjie
(Key Laboratory of Specialty Fiber and Optics Access Networks, Shanghai University, Shanghai 200444, China)

Abstract: In network communication system, image processing system, and so on, multiple subsystems would access external memory simultaneously. Multi-channel memory controller can solve this problem effectively. With the increasement of data and the improvement of processing unit performance, traditional multi-channel controllers can not meet the requirement of high-speed memory access for system because of low bandwidth utilization. To solve the aforementioned problems, we propose a new kind of DDR4 multi-channel controller on the FPGA in this paper. The controller is defined by simplified user interface and supports ring buffer in network communication, which reduces the use complexity and improves the universality. Multi-channel access conflicts can be solved efficiently by adopting circular priority arbiter. Meanwhile, the bandwidth utilization of the system has been improved. Besides, the rewinding access to the ring buffer is realized by a sharding mechanism. The simulation results are consistent with that on Xilinx KCU116 FPGA. When testing 4 096 MB records, the maximum effective bandwidth of the system is 78.3 Gbps, and the bandwidth utilization rate reaches 94.0%.

Keywords: multi-channel memory controller;DDR4 SDRAM;FPGA

0 引 言

随着现代网络电子技术的发展,在网络通信以及图像处理等领域中,数据缓存量日益庞大,使得大容量外部存储器的应用越来越广泛。而双倍速率同步动态随机存储器系列产品由于高速的数据传输频率,被广泛地使用在各个领域的缓存系统中[1-3]。目前,DDR4 SDRAM(double-data-rate fourth generation synchronous dynamic random access memory)拥有可以满足大多数系统的理论峰值带宽,但是要将这一理论峰值带宽中很大一部分交付给多个工作负载,需要一个能够有效利用片外接口的多通道内存控制器。

DDR4 SDRAM 全称为第四代双倍速率同步动态随机存储器,主要应用于计算、服务器以及嵌入式设备等[4]。此前 DDR3 SDRAM 被应用于智能手机、平板电脑以及一些微型服务器系统中。但是,随着产品中处理器芯片主频和带宽的提升,存储器逐渐成为整体系统性能提升的瓶颈,DDR3 SDRAM 已经不能满足系统对存储带宽的需求。相比于 DDR3 SDRAM,DDR4 SDRAM 拥有更低的工作电压、更高的数据速率、更快的突发访问和更高的系统可靠性[5]。因此,本文主要研究 DDR4 多通道控制器。

收稿日期:2022-01-26

文献[6]根据 JEDEC(Joint Electron Device Engineering Council,联合电子器件工程委员会)规范实现了对 DDR4 SDRAM 接口的访问,并通过在结构中设计并行组,提高了其数据吞吐率。但是,该设计并不能在 DDR4 通知错误状态时对进行中的事务做出正确的处理。由于 DDR4 SDRAM 片外接口控制的高复杂度,很多内存控制器在设计上并不够完备,而 Xilinx 公司提供的存储接口生成器(memory interface generator,MIG)知识产权(intelligent property,IP)核解决了 DDR SDRAM 的接口控制逻辑复杂的问题,给设计者提供了新的开发方案。文献[7]分析了 MIG 良好的稳定性和灵活性。然而 MIG 只提供一组读写接口,为了实现多个用户访问,需要在 MIG 控制器的基础上设计多通道仲裁方案。

多通道控制器的设计首先需要解决通道间的访问冲突问题,文献[7]基于 MIG 实现了多通道读写防冲突设计,有效解决了期货行情数据处理中多通道同时访问 DDR3 的冲突问题。但是,设计中采用了 Xilinx 定义的应用程序接口(application interface,APP)总线,不利于模块的移植。论文[9]将 MIG 配置为高级可扩展接口(advanced eXtensible interface 4,AXI4)接口,并在前级利用 Xilinx 的 AXI-Interconnect IP 解决多个通道间的总线访问冲突,但是将 AXI4 总线作为用户接口,增加了使用的复杂度,同时,AXI-interconnect 作为商用 IP 不利于系统的移植。文献[10]中,作者使用 AXI4-Stream 和 AXI4-Litc 接口分别传输数据和命令,简化了访问流程。结构上先仲裁后接口转换(带宽转换),未针对访问冲突进行带宽优化。为了提高多通道同时访问时的 DDR 带宽利用率,文献[11]整体采用先接口转换(带宽转换)后仲裁的结构。在设计中首先将每组用户接口转换为高带宽的 AXI4 总线,再进行多通道仲裁,带宽利用率有很大的提升,最高可达 93%。但是,在接口转换实现上未利用 AXI4 总线 Outstanding 的特性进行地址与数据通道的流水线设计,系统带宽利用率仍有提升空间。

针对以上不足,本文设计了基于现场可编程门阵列(field programmable gate array,FPGA)的 DDR4 多通道控制器。主要工作如下:

1) 设计高效简洁的用户接口,支持以字节为单位的非对齐突发传输,支持地址与数据通道的流水线传输。

2) 设计先接口转换后仲裁的系统结构,并在接口转换中进行流水化设计,使系统最高带宽利用率达 94.0%。并针对通信、网络领域中循环缓冲区的需求,加入缓存区访问回绕的功能。

3) 设计 DDR4 多通道控制器的仿真和板级测试,并针对结果进行分析。

综上所述,本系统能够高效、可靠地完成 DDR4 多通道访问工作。在网络通信、图像处理等场景具有较高应用价值。

1 系统结构设计

1.1 系统整体结构

DDR4 多通道控制器框图如图 1 所示,系统包含 3 种总线接口:用户接口、DDR4 接口以及 AXI4 总线,AXI4 总线的突发模式为递增模式,突发传输宽度与数据位宽相同。

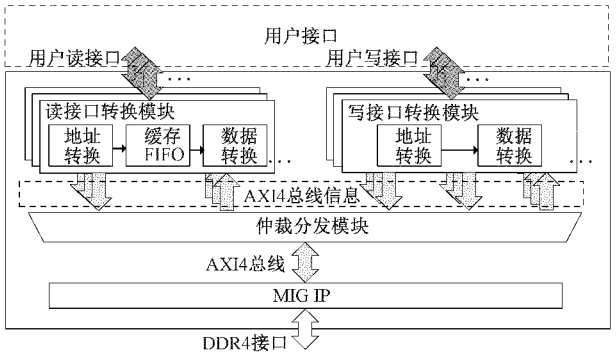


图 1 系统结构框图

如图 1 所示,设计采用先接口转换后仲裁的结构,先由每路读和写接口转换模块将低带宽的用户接口转换为高带宽的 AXI4 总线,AXI4 总线带宽与 DDR4 接口带宽相同,接着仲裁分发模块对高带宽的 AXI4 总线仲裁。在 DDR4 接口带宽相同的情况下,该结构最大化地提升了系统的带宽利用率。

系统主要由读写接口转换模块、仲裁分发模块以及 MIG 组成。3 个部分的功能如下:

1) 读和写接口转换模块。该模块主要由地址转换与数据转换部分组成。每组读写接口转换模块控制一路用户接口,功能是根据用户接口的访问请求,生成符合 AXI4 协议的控制与数据信息。同时,该模块支持访问回绕,能够让用户的突发访问到达循环缓冲区边界后,重新回到缓存区起始地址。在通信、网络领域中更具实用性。

2) 仲裁分发模块。由于多路用户接口访问的都是同一个 DDR4,而操作 DDR4 接口的 MIG 也只有一组 AXI4 总线。为了实现多通道访问的要求,仲裁分发模块对多路访问请求进行仲裁输出并对返回数据按通道分发。

3) MIG IP。负责 DDR4 SDRAM 的初始化、自动刷新、时序同步以及存储器访问等任务,具有高效、稳定的特点[12]。系统设计中使用标准 AXI4 接口的 MIG 进行 DDR4 接口控制,有利于模块的移植,提高了系统的通用性。

系统数据流向如图 1 中箭头所示,每路用户接口分为读接口和写接口,分别由读和写接口转换模块控制。仲裁分发模块将多路读写接口转换模块封装的 AXI4 总线信息选择一路输出。MIG 收到仲裁分发模块的 AXI4 访问信息后操控 DDR4 接口完成存储器访问。

1.2 用户接口定义

基于突发传输的 AXI4 总线作为先进微控制器总线架

构(advanced microcontroller bus architecture，AMBA)第三代协议，支持更高性能和更高频率的系统，具有高带宽低延迟的特点。并且 AXI4 协议支持非对齐的传输，满足存储器随机访问的特点[13]。但是，直接操作 AXI4 总线具有一定的复杂度，主要体现在以下 3 个方面：

1) AXI4 总线拥有 5 个相互独立的通道，每个通道都需要进行握手。

2) AXI4 总线支持地址非对齐的传输，但是总线上的数据都是按固有地址边界对齐后进行传输的，使用时需要对数据进行处理。

3) AXI4 协议规定了每次突发传输不能跨过 4 KB 地址边界，传输时需要判断突发是否越界并对越界传输进行拆分。

针对以上的问题，本文设计了一种操作简单的用户总线如图 2，该总线基于突发传输，包含五个通道：写地址通道、写数据通道、写响应通道、读地址通道以及读数据通道。每个通道的数据流都是单向的，并且支持流水线传输，具有高吞吐率的特点。

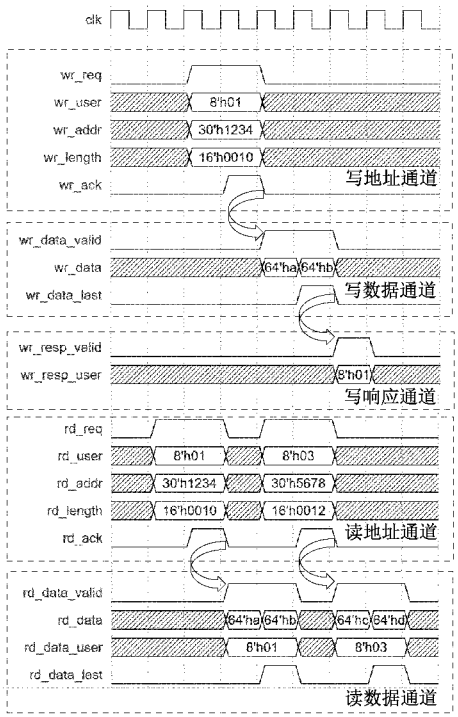


图 2　用户总线信号时序图

相比于 AXI4 总线，用户接口具有以下的优势：

1) 用户接口采用单向握手和通道依赖关系来进行数据流控制。如图 2 中读与写地址通道的时序图所示，访问方首先发起请求并准备好地址与长度信息，被访问方就绪后，产生应答信号表示可以传输数据。图 2 中弧形箭头表示通道间信号的依赖关系，箭头指向的信号只有在箭头起始处的信号置位后才能有效，即只有地址通道握手完成后，才能进行数据通道的数据传输。省去了读和写地址通道以外的握手流程，简化了访问方式。

2) 用户接口的读和写数据传输无需考虑字节对齐的问题。如图 2 中读和写数据通道所示，以 64 bit 宽度的用户数据为例，有效字节按照从低到高的字节顺序填充总线数据并进行传输，直到一次访问结束。与 AXI4 总线相比，用户总线的首次传输无需根据突发地址进行对齐填充，更加符合各类系统中的数据流传输特点。

3) 用户总线的突发传输不需要考虑跨边界问题。用户按图 2 的时序产生访问请求，就可以完成数据读取与写入。进一步简化了用户接口使用逻辑。

综上，本文设计的用户接口使用地址通道的单向握手控制数据流，简化了操作流程。用户总线的数据无需字节对齐，简化了用户接口使用逻辑，更加方便用户数据的写入与读取。本文基于 FPGA 的多通道 DDR4 控制器就是围绕该接口展开的。

1.3　设计实现的关键问题

基于简化的用户接口，系统将复杂的 AXI4 总线控制逻辑封装在每路的读和写接口转换模块中，再由仲裁分发模块实现多通道与单通道间的数据传输。而高效可靠地完成上述功能需要解决以下几个问题：

1) 跨 4 KB 地址边界的分片处理。因为 AXI4 协议规定了突发传输不能跨越 4 KB 地址边界，所以设计中会按 4 KB 边界将跨边界的传输拆分为多个突发传输。而由于分片机制的存在，设计中就需要对每次 AXI4 突发传输进行标记，以确定一帧数据写入或者读取结束。因此，系统巧妙的利用了 AXI4 的 ID 标签来标记一帧突发传输结束。

2) 循环缓冲区设计。访问地址到达系统设置的循环缓冲区边界后，会重新回到缓存区的起始地址。循环缓冲区的访问回绕机制与跨 4 KB 地址边界的处理方法相同，都是将单次访问进行分片，不同点在于访问回绕分片后的地址为缓存区的起始地址。为了便于硬件实现，设计中将两者统一处理，增加了系统的可靠性。

3) 数据对齐处理。AXI4 总线进行突发传输时，总线数据都按照其数据宽度的自然地址边界进行对齐的。而用户总线数据传输都是将用户访问地址作为边界进行对齐。因此，设计中需要根据地址信息进行用户数据与 AXI4 数据间对齐方式的转换，而转换的关键在于解决转换前后的传输变长问题，即转换前后总线上传输的次数不一定相同。对于地址信息的获取，本文在此进行了流水线设计，如图 1 中读接口转换模块所示，数据转换需要的地址等信息在地址转换周期内写入先进先出队列(first input first output，FIFO)，两者独立运行，提升了模块的吞吐率。

4) 多通道识别。多通道访问的核心问题是多路仲裁与多路分发。仲裁模块可以直接获得仲裁请求的通道号，多路分发模块只能通过数据总线上的标识来判别其所属的通道信息。而 AXI4 总线上自带 ID 标签，设计中将 ID 标签合理使用，通过扩展其位宽，并在扩展的位宽中定义含通

道信息的字段,使通道信息存在于 AXI4 总线传输中,方便分发模块正确地将总线数据传输到对应的通道中去。

5) 仲裁算法选择。通常的仲裁算法有两类:固定优先级仲裁与循环优先级仲裁。固定优先级仲裁就是指各个通道的优先级设计开始就被确定,并在系统运行过程中不会改变。这种算法的优点在于,通过赋予重要通道较高的优先权,保证系统的可靠性。但是,缺点在于,有可能发生某个通道"撑死"或者"饿死"的现象。而循环优先级仲裁可以通过对于优先级的改变,将仲裁通道的优先级变为最低,保证各通道获得优先仲裁的机会均等,能够有效的克服上述问题[14]。因此,设计中采用循环优先级算法作为系统的仲裁机制。

2　系统关键模块设计

2.1　支持访问回绕的地址转换设计

地址转换设计是将用户地址通道的控制信息(用户访问地址和长度)转换为符合 AXI4 协议的信号,同时在设计中支持访问回绕,主要完成对 AXI4 协议跨 4 KB 地址边界的分片处理和回绕机制的实现。

为了方便硬件实现同时兼顾使用的合理性,设计中通过参数化的方式设置循环缓冲区的地址宽度,并将最小缓存区限制为 4 KB 地址空间,使循环缓冲区边界与 AXI4 协议规定的 4 KB 地址边界一致。因此,本文所提出的设计需要处理两个核心问题:1)确定跨 4 KB 地址边界的分片次数;2)确定每次分片的突发地址与突发长度。

针对上述问题,本文采用的方法如下:假设用户访问地址为 A_0,用户访问长度为 L_0。设用户访问的分片次数为 N,由式(1)计算得到。在循环缓冲区地址宽度为 x bits 的情况下,第 n 次分片的访问地址 $A(n)$ 和访问长度 $L(n)$ 如式(2)、(3)所示,其中 n 为正整数,$n \in [1,N]$。x 为正整数,$x \in [12,X]$,X 表示存储器的地址位宽。

$$N = \lceil (A_0 \% 4096 + L_0 - 1)/4\,096 \rceil \tag{1}$$

式中:"%"表示取余运算,"/"表示除法运算;"⌈ ⌉"表示向上取整。

$$A(n) = \begin{cases} A_0, & n = 1 \\ (A_0 - A_0 \% 2^x) + \lceil (A_0 - A_0 \% 4\,096) + 4\,096 \times (n-1) \rceil \% 2^x, & 1 < n \leqslant N \end{cases} \tag{2}$$

$$L(n) = \begin{cases} L_0, & N = 1 \\ 4\,096 \quad A_0 \% 4\,096, & N \neq 1, n = 1 \\ 4\,096, & N \neq 1, 1 < n < N \\ L_0 - [(n-1) \times 4\,096 - A_0 \% 4\,096], & N \neq 1, n = N \end{cases} \tag{3}$$

支持地址回绕的分片策略如上所述,首先根据用户访问地址与长度计算访问分片的次数,然后通过式(2)和(3)算出每次分片所对应的访问地址与长度,完成访问分片。

由于式(3)算出的访问长度以字节为单位,为了转换为符合 AXI4 协议的突发长度,本文进一步计算的得到 AXI4 突发长度 $L_A(n)$,如式(4)所示。

$$L_A(n) = \lfloor (A(n) \% M + L(n) - 1)/M \rfloor \tag{4}$$

式中:M 表示 AXI4 总线数据传输的字节宽度;⌊ ⌋表示向下取整。

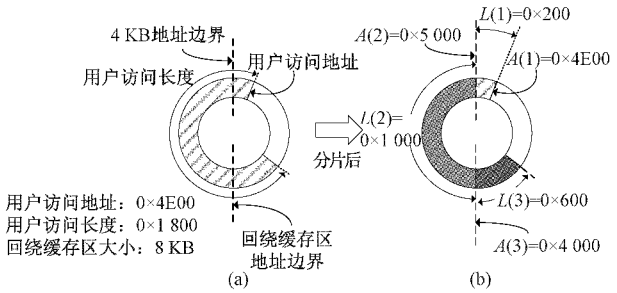


图 3　用户访问分片示意图

以上为本文的地址转换设计,为了使上述的分片机制更加清晰,下面举例说明本文分片过程。假设用户访问地址为 0x4E00,访问长度为 0x1800,循环缓冲区大小为 8 KB,图 3(a)表示该用户访问的地址空间。由图可知,该访问同时跨过 4KB 地址边界与循环缓冲区边界,因此,系统须根据这两个地址边界将用户访问拆分为 3 次传输。拆分后的访问地址依次为:用户访问地址 0x4E00、4 KB 地址边界 0x5000 与循环缓冲区起始地址 0x4000。分片结果如图 3(b)所示。整个分片过程均可由式(1)～(3)共同表示。

为了进一步验证设计的正确性,本文进行了功能性仿真,仿真软件采用 Mentor Graphics 的 Modelsim 10.7。

地址转换模块 4 KB 地址边界分片与回绕仿真结果如图 4 所示,图中用户写地址通道产生访问地址为 0x4E00,访问长度为 0x1800 的写请求,系统中每块循环缓冲区大小为 8 KB,AXI4 总线数据宽度为 32 字节。编号①～④表示用户写地址通道转换过程,地址转换设计首先在①处与用户写请求进行单向握手,接着开始计算分片次数以及每次分片的访问地址与访问长度,分片过程已在前文中举例说明,分片结果由图 3(b)所示。接着通过式(4)将以字节为单位的访问地址转换为 AXI4 总线的突发长度,3 次转换结果分别为:0xF、0x7F、0x2F。②③④表示访问分片后 3 次 AXI4 总线写地址通道的传输,由图可知,地址转换的结果是正确的。因此本设计能够实现支持分片与访问回绕的地址转换。

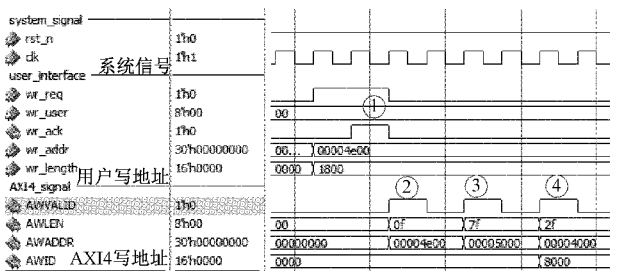


图 4　地址转换设计仿真图

2.2　用户接口数据与 AXI4 总线数据的对齐转换设计

数据转换设计是完成用户接口与 AXI4 总线接口间数据传输格式的转换，关键在于实现数据传输的对齐转换。对齐转换设计存在以下几个核心问题：1）由于对齐方式的不同，对齐转换后传输的次数可能会存在差异，这一差异会导致转换后的传输次数发生改变；2）由于分片机制的存在，对齐转换过程中需要对跨 4 KB 地址边界的突发传输进行拆分。

针对以上问题，本文采用移位拼接的方式进行数据格式的转换，并根据用户访问地址与长度计算得到转换后的传输次数，以确定最后一次传输结束。同时，本文通过插入传输结束标识的方法对突发传输进行拆分，以达到传输分片的目的。

由于设计中的用户接口数据与 AXI4 总线数据需要进行相互的转换，所以本文的内容包括：1）用户写数据的按固有地址边界对齐转换；2）AXI4 总线读数据的按用户访问地址对齐转换。

针对用户接口数据的对齐转换，本文采用的方法如下：假设传输的数据宽度为 Wbits。设突发传输中用户写数据通道第 n 次传输的数据为 $D_{wu}(n)$，则按固有地址边界对齐转换后的数据 $D_{wn}(n)$ 如式(5)所示，并且每次传输对应的结束信号 $last_{wn}(n)$ 如式(6)所示。

$$D_{wn}(n)=\begin{cases}D_{wu}(n)\ll W_s, & n=1\\ \{D_{wu}(n)\ll W_s\}\mid\{D_{wu}(n-1)\gg(W-W_s)\}, & 1<n\leqslant N_u\\ \{D_{wu}(n\quad 1)\gg(W\quad W_s), & n=N_u,N_u<N_n\end{cases}\tag{5}$$

式中："≪"为左移运算符；"≫"为右移运算符；"|"为按位或运算符；n 为正整数，$n\in[1,N_n]$；$W_s=(A_0\%W_b)\times 8$，$W_b=W/8$；N_u 表示按用户访问地址对齐的数据传输次数，计算方法为 $N_u=(L_0-1)/W_b$。N_u 表示按固有 W −bit 地址边界对齐的数据传输次数，计算方法为 $N_n=(A_0\%W_b+L_0-1)/W_b$。

$$last_{wn}(n)=\begin{cases}0, & n\neq N_n,p(n)\neq 0\\ 1, & n\neq N_n,p(n)=0\\ 1, & n=N_n\end{cases}\tag{6}$$

其中，$p(n)=(A_0-A_0\%W_b+n\times W_b)\%4\,096$。

用户写数据的对齐转换设计将每次传输的用户写数据通对式(5)进行移位拼接，生成按固有 W-bit 地址边界对齐的数据 $D_{wn}(n)$，同时计算与之对应的结束标识 $last_{wn}(n)$，来确定转换后的传输次数以及进行传输分片。

AXI4 总线读数据的对齐转换方式与用户接口数据的对齐转换相似：设 AXI4 总线的读数据通道第 m 次传输的数据为 $D_{rn}(m)$，移位拼接的方式如式(7)所示。由于分片机制的存在，系统需要对对应同一个用户读请求的多次 AXI4 总线读数据通道的传输进行合并，本文通过控制传输结束信号 $last_{wn}(m)$ 完成上述功能，具体方法如式(8)所示。

$$D_{ru}(m)=\begin{cases}D_{rn}(m)\gg W_s, & N_n-1\\ \{D_{rn}(m)\gg W_s\}\mid\{D_{rn}(m\quad 1)\ll(W-W_s)\}, & N_n\neq 1,1\leqslant m\leqslant N_n-1\\ \{D_{rn}(m)\gg W_s, & N_n\neq 1,m=N_n\end{cases}\tag{7}$$

式中：m 为正整数，$m\in[1,N_u]$。

$$last_{wn}(m)=\begin{cases}0, & m\neq N_u\\ 1, & m=N_u\end{cases}\tag{8}$$

综上，本文对齐转换设计通过移位拼接的方法解决传输中数据对齐方式不同的问题，并控制传输结束信号处理对齐转换前后传输次数不一致与传输分片的问题。

为了验证对齐转换设计的正确性，本文设计仿真实验来模拟用户写数据对齐转换及用户写数据传输跨 4 KB 地址边界的过程。如图 5 所示，wr_addr 和 wr_length 为用户访问地址与长度，wr_data_valid、wr_data、wr_data_last 表示用户写数据通道中的数据有效信号、数据、传输结束信号。convert_valid、convert_data、convert_last 为用户写数据对齐转换后的数据传输信号，本次仿真中传输的数据宽度为 64 bit。

对齐转换设计根据用户访问地址 0xFFF 和访问长度 0x18 计算得出转换后的传输次数为"4"，并通过访问地址对用户写数据进行移位拼接。因为传输跨越了 4 KB 地址边界，所以系统需要通过插入有效的传输结束信号使传输分片。由图 5 可知，对齐转换后的结果是正确的。因此，本文能够完成 AXI4 总线与用户接口间的对齐转换。

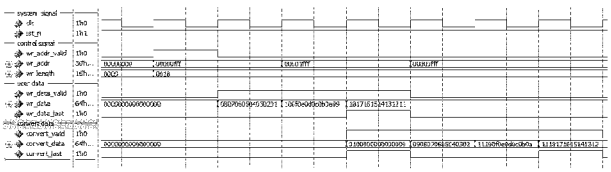


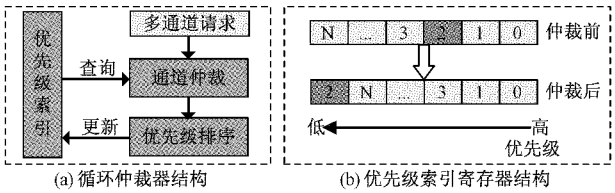
图 5　对齐转换仿真图

2.3　循环优先级仲裁器设计

图 6 为循环优先级仲裁器的设计框图，通道仲裁部分根据优先级索引的优先顺序响应来自多个通道的请求，并在仲裁结束后对优先级索引进行重排序，将已仲裁的通道优先级调整为最低。

为了方便硬件实现，本文对优先级索引寄存器进行如图 6(b)所示的设计，图中每个单元格表示一个寄存器，每个寄存器本身对应一个优先级，寄存器中的数值代表对应优先级的仲裁通道。系统按优先级顺序查询各个寄存器对应通道的仲裁请求，当某一通道被仲裁后，重新排序索引寄存器中的通道号，排序过程如图 6(b)所示，图中深色单元格中的数值表示被仲裁的通道号。

图 7 为系统多路仲裁器的仿真结果，仿真中设置仲裁通道为 8 路，初始优先级从通道 0 至通道 7 依次递减。系统中仲裁接口采用 FIFO 接口，所以图中 fifo_rena0、fifo_rena1…、fifo_rena7 表示系统对各个通道仲裁信号，fifo_empty0、fifo_empty1…、fifo_empty7 为仲裁请求 FIFO 的



(a) 循环仲裁器结构　　(b) 优先级索引寄存器结构

图 6　循环优先级仲裁器结构框图

空标志,fifo_rena0 与 fifo_empty0 的后缀表示通道号。图中 index_prio0、index_prio1…、index_prio7 表示 8 个优先级索引寄存器,优先级顺序从 index_prio0 到 index_prio7 递减。

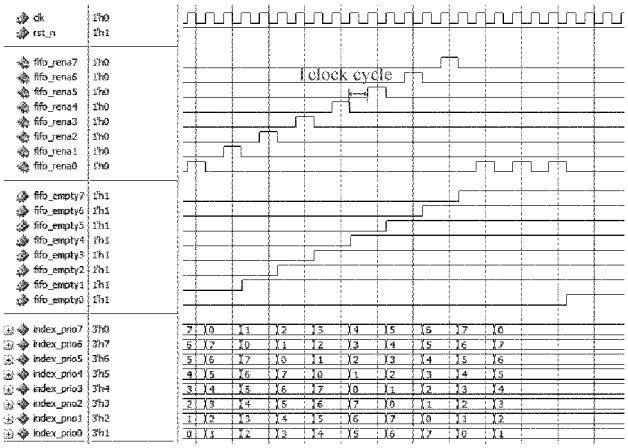


图 7　循环优先级仲裁器仿真

由图中优先级索引寄存器可知,初始化时系统的优先级顺序从通道 0 至通道 7 递减,首先,fifo_rena0 置“0”表示有来自通道 0 的仲裁请求,系统随即对该通道进行仲裁,并更改通道 0 的优先级为最低,同时更新其他通道的优先级。虽然在第一次仲裁后通道 0 仍有仲裁请求,但是由于其他通道的优先级高于通道 0,所以系统在多个通道同时请求的情况下优先响应其他通道。系统仲裁完通道 1 到通道 7 后,继续对通道 0 仲裁,最后在没有其他通道进行仲裁请求的情况下,系统对通道 0 进行了连续仲裁。图 7 表示了系统对 8 个通道的仲裁过程,由图可知,本文的循环优先级仲裁器的仲裁结果是正确的,同时通道间的仲裁间隔只有一个时钟周期,具有高效仲裁的特点。

3　系统测试

为了测试系统的正确性与性能,本文对所设计的 DDR4 多通道控制器设计进行上板验证。

测试选用 Xilinx 的 Ultra-Scale+系列 KCU116 评估板,FPGA 型号为 XCKU5P-FFVB676-2-E。测试评估板上的 DDR4 SDRAM 使用两颗 Micron 公司的 MT40A256M16GE-075E 芯片,每片芯片物理位宽为 16 bit,容量为 4 GB,等效带宽为 2 666 MHz^{-16}。

测试系统结构如图 8 所示,系统时钟为 333 MHz,DDR4 多通道控制器中用户接口的数目可由参数设置,为了方便测试,循环缓冲区设置为 8 KB 大小。控制器中的 AXI4 总线数据宽度为 256 bit。测试的用户访问请求由 FPGA 内部产生,整个测试系统均使用 Verilog 语言在 KCU116 评估板上实现,FPGA 开发平台为 Vivado Design Suite2018.3。

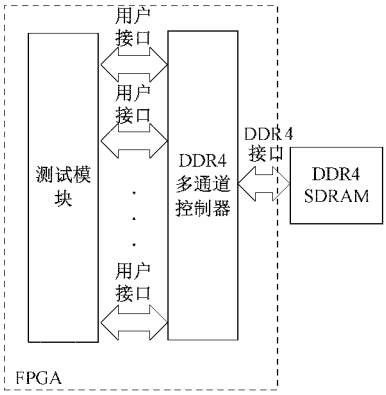


图 8　测试系统框图

3.1　正确性测试

为了验证设计的正确性,本文对系统进行读写一致性测试。测试系统结构如图 9 所示,DDR4 控制器的用户接口设置为 4 路进行多通道测试,用户接口传输的数据宽度为 64 bit。每个用户接口单独处理相应的读写测试模块的访问请求。四个读写测试模块相互独立,访问空间互不干扰,每个模块能够访问的地址空间如图所示。

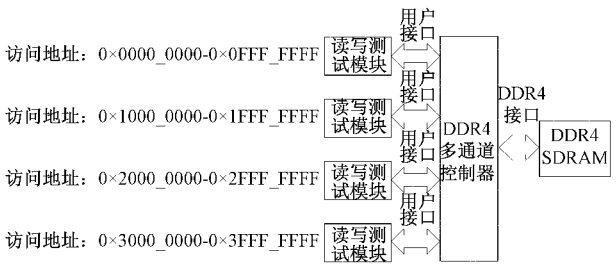


图 9　系统正确性测试系统结构图

系统测试流程如下:首先通过用户接口向 DDR4 SDRAM 中的某段地址空间写入数据,接着将该地址空间的数据读出,在模块内部对比写入与读出数据,最后通过对比实验的结果判断设计功能的正确性。

基于测试的完备性与可行性,读写测试模块根据以下规则产生用户访问地址与长度:1)设置用户访问长度为 64 字节,在实验过程中保持不变;2)用户访问地址在每次对比实验结束后自增“1”,当到达每个读写测试模块设定的访问地址最大值后自动回绕到最小值。通过以上方法可以完成 DDR4 所有的地址空间的访问,同时检验访问回绕、分片机制以及对齐转换等的功能。

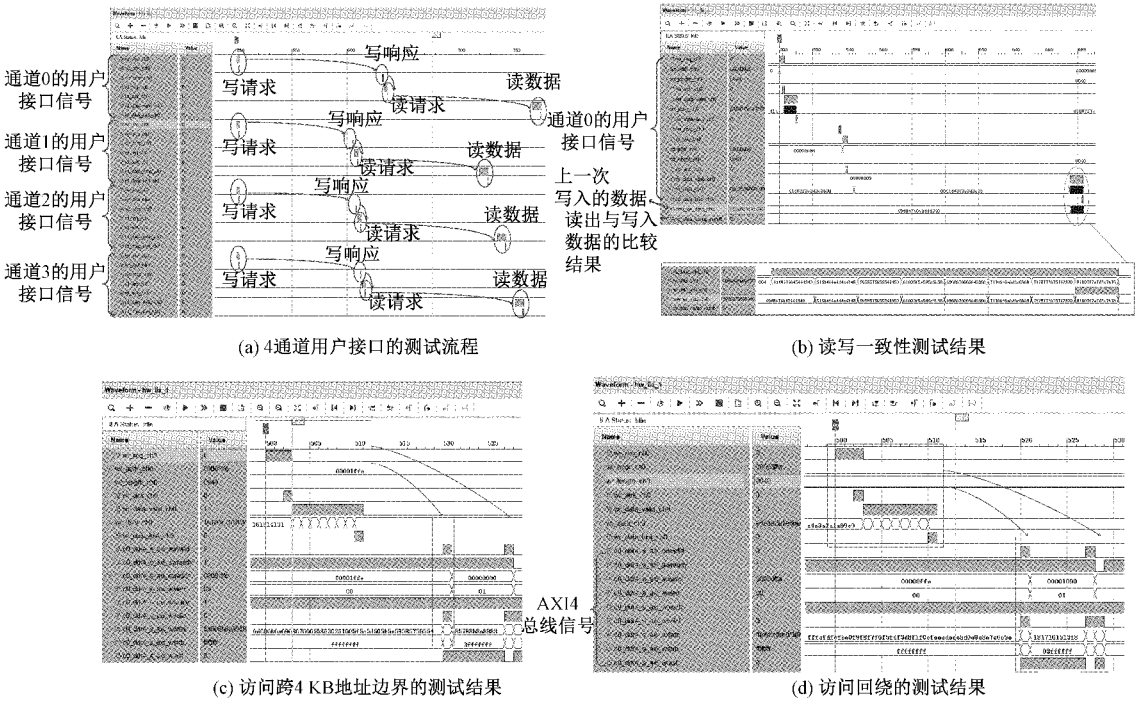
本文使用 Xilinx 提供的在线逻辑分析仪(integrated logic analyzer,ILA)对实验结果做进一步的分析,分别采集

用户接口与 AXI4 总线相关的数据与控制信息，如图 10 所示。

图 10 为系统正确性测试结果，各个信号的含义如图所示。图 10(a)所示为 4 通道用户接口的测试流程，如图所示，每组用户接口均进行先写后读的测试方法，通过将写入与读出的数据进行对比验证设计的正确性。读写一致性测试结果如图 10(b)所示，由图中可知，系统首先向地址 0x09 写入长度为 0x40 字节的数据，最后将该地址空间的数据读出，通过与缓存中上一次写入的数据对比可知读写的数据是一致的，因此可以证明本文设计的 DDR4 多通道控制器能够正确地写入与读出数据。

图 10 (c)为访问跨 4 KB 地址边界的测试结果，如图所示，在用户访问地址为 0xFFA、访问长度为 0x40 时，用户写请求被拆分为如箭头所示的两个部分，这两个部分的突发地址分别为 0xFFA 与 0x1000。同时，写入的数据也被拆分为次突发传输，如图所示。因此，本次实验能够证明用户访问请求的分片机制是正确的。

图 10 (d)为访问回绕的测试结果，如图所示，在用户访问地址为 0x1FFA、访问长度为 0x40 时，分片情况与图 10(c)一致。不同的是，第二次分片后的突发长度为 0x00，表明访问进行了地址回绕。因此，该实验证明了本文回绕机制的实现是正确的。



(a) 4通道用户接口的测试流程

(b) 读写一致性测试结果

(c) 访问跨4 KB地址边界的测试结果

(d) 访问回绕的测试结果

图 10　系统正确性测试结果

综上，DDR4 多通道控制器能够正确的写入与读出数据，并且可以有效的解决多通道访问冲突，同时支持 4 KB 地址边界的分片机制与访问回绕机制。

3.2　性能测试

针对 DDR4 多通道控制器的读写速度，本文采用图 8 所示的结构对系统进行了性能测试，主要计算系统的带宽利用率。

为了突出系统内部流水线设计对性能的影响，同时考虑到多通道访问的仲裁效率，测试系统同时向 DDR4 控制器的多个通道发起持续的用户访问请求，使其系统速率接近极限。在访问完 DDR4 所有地址空间后，根据所消耗的总时间计算带宽利用率，如式(9)所示。

$$\eta=\frac{\text{实际访问带宽}}{DDR4\text{ 理论数据带宽}}=\frac{\text{数据量}/\text{传输时间}}{\text{等效带宽}\times\text{数据总线位宽}}=\frac{8\text{ Gbit}/(timer\_cnt\times 3\text{ ns})}{2\,666\text{ MHz}\times 32\text{ bit}}=\frac{8\text{ G}}{256\times timer\_cnt} \tag{9}$$

式中：$timer\_cnt$ 为访问 DDR4 所有地址空间的周期计数；

基于 DDR4 的理论带宽，本文设置 4 组数据位宽为 64 bit 的用户接口进行连续写、连续读与混合读写的性能测试，其中混合读写测试方法为：对两组用户接口进行连续写，对另外两组用户接口进行连续读实验。设置一组数据位宽为 256 bit 的用户接口进行连续写、连续读以及混合读写的性能测试，其中混合读写测试方法为：对该用户接口间隔地发起写请求与读请求。在上述读写测试中，每次访问的长度分别为 64 字节、1 024 字节和 4 096 字节。实验结果如表 1 所示，在性能测试中连续写、连续读、混合读写的最高带宽利用率分别为 81.5%、94.0%和 89.4%，对应的有效带宽分别为 67.9 Gbps、78.3 Gbps 以及 74.5 Gbps。

综上，本文设计的 DDR4 多通道仲裁控制器具有较高的带宽利用率，最高为 94.0%，符合在通信、网络等领域中高速访存的需求，具有实际的应用价值。

表 1 系统带宽利用率测试结果

单次访问长度		64 字节			1 024 字节			4 096 字节		
数据位宽	接口数量	连续写	连续读	读写混合	连续写	连续读	读写混合	连续写	连续读	读写混合
64 bit	4 接口	40.0%	49.9%	49.4%	81.4%	87.0%	79.6%	81.5%	94.0%	89.4%
256 bit	1 接口	22.2%	20.0%	39.9%	81.1%	80.0%	87.3%	81.5%	94.0%	89.3%

4 结 论

本文设计了基于 FPGA 的 DDR4 多通道控制器,定义了一组简洁的用户接口,简化了使用者访问外部存储器的方式。

针对通信、网络领域中循环缓冲区的需求,系统加入缓存区访问回绕的功能,并且通过使用循环优先级仲裁器,高效地解决访问冲突问题,同时采用先转换后仲裁的系统结构,极大地提高了系统的带宽利用率。由此实现了不仅操作简便而且具有高带宽利用率的 DDR4 多通道控制器。

由于本文 DDR4 接口的处理依赖于 MIG 控制器,使得带宽利用率受到限制。同时,在进行用户接口转换时仍有带宽的损耗,没有做到线速处理。未来工作将致力于 DDR4 接口的高效处理与用户接口的线速转化,进一步提升系统的访问效率。

参考文献

[1] 王子懿,沈三民,杨峰,等. 基于 FPGA 的高速大容量存储与传输系统[J]. 电子测量技术,2021,44(13):150-155.

[2] 徐拥军,何文春,刘媛媛,等. 气象大数据存储体系设计与实现[J]. 电子测量技术,2020,43(22):19-25.

[3] 周榕,翁天恒,陈天杨,等. 基于 FPGA 的 160 Gbit/s 网络数据包过滤系统设计[J]. 电子测量技术,2021,44(15):155-161.

[4] 万轶. 高性能 DDR3 存储控制器的研究与实现[D]. 长沙:国防科学技术大学,2008.

[5] ZHENG J, YAN K, ZHANG Y, et al. Design and implementation of DDR4 SDRAM controller based on FPGA[C]. 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference, IEEE, 2018: 421-424.

[6] ISLAM M A, ARAFATH M Y, HASAN M J. Design of DDR4 SDRAM controller [C]. 8th International Conference on Electrical and Computer Engineering, IEEE, 2014: 148-151.

[7] JIAO S, CHENG R. Design of a DDR3 Controller Based on FPGA[J]. Electron. Sci. Technol., 2015, 28(7): 41-43.

[8] 张凤麒,张延彬,王忠勇. 基于 FPGA 的 DDR3 六通道读写防冲突设计[J]. 电子技术应用,2018,44(7):68-71,80.

[9] 郭要强,毛建华. 基于 ARTIX-7 FPGA 多端口 DDR3 读写设计[J]. 电子世界,2019(24):132-133,136.

[10] WANG H, BAI X, WU Q. Multichannel high-speed data caching system on FPGA for RAID storage[C]. International Conference in Communications, Signal Processing, and Systems, 2018: 482-491.

[11] 张宇嘉,杨晓非,姚行中. 基于 AXI4 的卫星接收机 DDR3 多端口存储的设计[J]. 电子器件,2016,39(3):617-622.

[12] 王红兵,强景,周珍龙. Xilinx MIG IP 核的研究及大容量数据缓冲区的实现[J]. 电子产品世界,2016,23(8):43-46.

[13] 孙巍,韩梅,王超,等. Zynq-7000 水声信号采集存储系统设计与实现[J]. 电子测量技术,2018,41(22):112-115.

[14] 胡孔阳,胡海生,王梓. 一种可配优先级 Round-Robin 仲裁器实现[J]. 中国集成电路,2015,24(7):27-29,63.

作者简介

翁天恒,硕士研究生,主要研究方向为 FPGA 网络通信。E-mail: wendien@126.com