Journal of Sensor Technology and Application 传感器技术与应用, 2022, 10(3), 369-377
Published Online July 2022 in Hans. http://www.hanspub.org/journal/jsta
https://doi.org/10.12677/jsta.2022.103044

Hans汉斯

多核数据管理器仲裁机制的研究与设计

马　燕

合肥工业大学，安徽　合肥

收稿日期：2022年5月9日；录用日期：2022年6月28日；发布日期：2022年7月5日

摘　要

多核片上数据传输是指在片上系统中，通过共享存储器、片上网络等方式进行通讯，实现多个处理器协同处理任务的技术。为了达到多核片上传输技术在进行多个处理器核和多个外部设备之间数据传输时，同时兼顾数据包发送和接收、有效监控数据交换和灵活派发数据包的目标，本文提出了一种基于大规模数据的拆解/封包传输的多核数据仲裁机制，新机制采用优先级多级调整策略，实现数据包有序进出队列管理器，从而减少内部通信方面的负担，提升系统整体性能，同时完成了新机制的HDL建模和仿真，实验结果表明，所提出的设计能够保证多组数据的优先级次序，提高传输时的数据利用率，平衡响应速度，在多数据场景下，实现公平有效地传输。

关键词

多核处理器，仲裁机制，数据传输技术，优先权扩展

Research and Design of Arbitration Mechanism for Multi-Core Data Manager

Yan Ma

Hefei University of Technology, Hefei Anhui

Received: May 9th, 2022; accepted: Jun. 28th, 2022; published: Jul. 5th, 2022

Abstract

Multi-core on-chip data transmission refers to a technology that communicates through shared memory, on-chip network, etc., in a system-on-chip to realize collaborative processing tasks by multiple processors. In order to achieve the goal of multi-core on-chip transmission technology in data transmission between multiple processor cores and multiple external devices, while taking into account data packet sending and receiving, effectively monitoring data exchange and flexibly dispatching data packets, this paper proposes a method based on a multi-core data arbitration me-

文章引用: 马燕. 多核数据管理器仲裁机制的研究与设计[J]. 传感器技术与应用, 2022, 10(3): 369-377.
DOI: 10.12677/jsta.2022.103044

chanism for large-scale data disassembly/packet transmission. The new mechanism adopts a priority multi-level adjustment strategy to realize the orderly entry and exit of data packets into the queue manager, thereby reducing the burden of internal communication and improving the overall performance of the system. The HDL modeling and simulation of the new mechanism are carried out. The experimental results show that the proposed design can ensure the priority order of multiple groups of data, improve the data utilization rate during transmission, balance the response speed, and achieve fair and efficient transmission in multi-data scenarios.

Keywords

Multi-Core Processor, Arbitration Mechanism, Data Transmission Technology, Priority Extension

1. 引言

随着多核技术的成熟，对片上数据交互的性能要求也日益提高[1] [2]。由于传统数据传输模型在数据交换过程缺乏实时调整等能力，所以研究一种新型的数据传输管理机制以满足更多应用场景下的用户需求显得尤为紧迫[3] [4] [5]。本文设计的多核数据管理器机制首先将大规模数据分解为若干数据包的形式，然后将数据包特征抽象成队列元素，作为数据传输的管理对象，通过仲裁机制管理队列元素进出队列，最终完成数据包传输[6]，在此过程中加入多级仲裁，确保大批量数据处于合理优先级次序，提高数据传输利用率，保证响应速度平衡，使得数据在多个主机核与设备间灵活搬运，实现多核系统内海量数据的实时高效交互[7] [8] [9]。

2. 系统概述

本文设计的多核数据管理器仲裁机制是多核片上数据传输系统的一部分。队列元素是被传输数据特征的抽象，也是多核数据传输系统调度和管理的对象。队列元素中包含数据包在数据存储空间的起止地址等信息，通过操作队列元素即可完成数据的读写。

数据传输的第一步操作，源端数据经主机解析形成目的端编号信息、数据包存储地址信息、数据包尺寸信息以及数据包原始优先级信息，同时完成队列元素的入队操作。数据包实体存放于系统存储空间，通过仲裁队列元素，间接控制数据包的传输。经过三级仲裁，筛选出出队的队列元素传递至目的端，目的端核或设备接收来自源端的队列元素，解析队列元素中包含的数据包地址信息，获取数据包数据，从而完成数据包由源端到目的端的传输。

多核数据管理器的仲裁机制总体设计分为 4 个模块，分别是入口阶段仲裁模块、中间阶段优先权扩展模块、出口阶段仲裁模块和多对多仲裁模块。

入口阶段仲裁模块从 8 个入口子队列的顶端元素中选取出当前优先级最高的元素进入下一阶段。

中间阶段优先权扩展模块负责对入口阶段选取的队列元素，进行优先权值的扩展，为每个队列元素生成一个新的优先权值。

出口阶段仲裁模块负责对出口队列中的队列元素进行两轮弹出申请，对得到响应的队列元素做实际弹出操作，对未得到响应的队列元素做二次弹出申请，对两轮弹出申请均未得到响应的队列元素做弹出回收。

多对多仲裁模块完成12个输入到5个输出间的仲裁，负责将源端输入的元素弹出至对应目的端接收。多核数据管理器仲裁机制系统总体原理如图1所示。该系统实现共计12个源端的输入和5个目的端的输出，源端包括8个处理器核源端和4个外部设备源端，目的端包括4个外部设备目的端和1个核间传输公共目的端，其中公共端由8个源端核输入和8个目的端核输出构成。

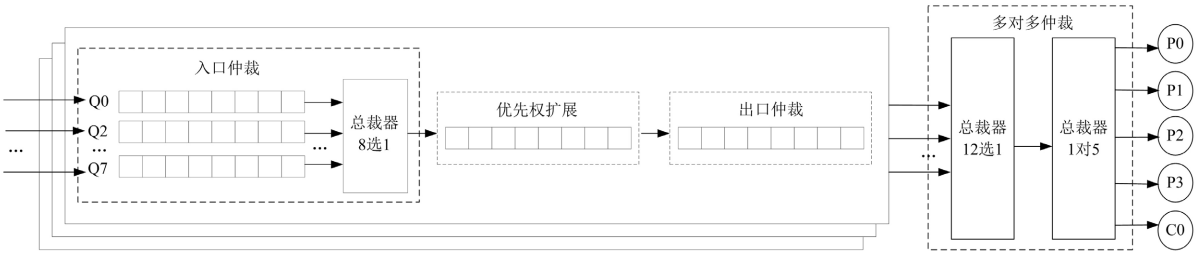

Figure 1. Overall structure diagram
图 1. 总体结构框图

3. 系统硬件实现

3.1. 入口阶段仲裁模块

入口队列中分为8个深度为8的子队列，最多容纳64个元素。入口阶段仲裁模块对缓存在入口队列中的新入队元素进行裁决输出。入口阶段仲裁模块采用的是轮询式仲裁策略(Round-Robin) [10]，依次响应子队列Q1~Q7的弹出申请，当最后一个子队列Q7执行完毕时再返回至第一个子队列Q1，重新开始依次响应每个子队列。当每个子队列的顶部队列元素成功弹出后，整个队列执行一次前移，由原来位于第二位的队列元素成为新的队列顶部元素，并等待响应。入口阶段仲裁模块工作示意如图2所示。

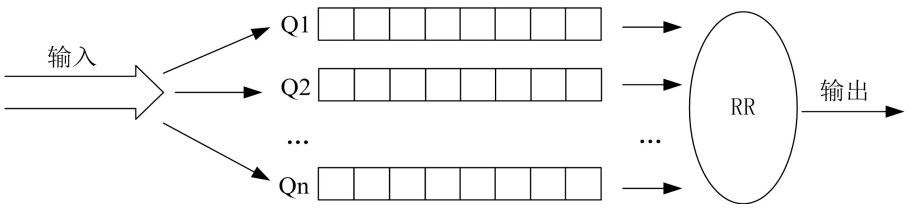

Figure 2. Schematic diagram of entry-level arbitration module
图 2. 入口阶段仲裁工作示意图

3.2. 中间阶段优先权扩展模块

中间阶段队列是一个深度为8的队列，包含8个队列元素。中间阶段优先权扩展模块依据当前队列中8个元素的原始优先权值，以及表示数据包尺寸等参数信息，完成对8个队列元素的优先权值扩展。原始优先权值是系统根据队列元素所表示的数据包属性赋予的优先级次序 original_priority，原始优先级共有0~7八个优先权值，其中优先权值越大表示该数据传输请求要求越先得到执行。

3.2.1. 优先权扩展算法

优先权扩展模块从入口队列中获取8个队列元素，在原始优先权值的基础上，将队列元素所包含的数据包尺寸考虑进来，依据优先权扩展公式，对队列元素进行一次优先级扩展重定序工作，使得不同尺寸不同优先级的数据包能够获得更加公平有序的仲裁次序[11]，这对于队列的整形、控制数据包拥塞至关重要[12]。优先权扩展公式如所示：

$$\mathrm{Pri\_now}[i]=\mathrm{Pri\_ori}[i]+\frac{\mathrm{Pkt\_size\_max}}{\mathrm{Pkt\_size}[i]}$$

式中，$\mathrm{Pri_now}[i]$当前优先权参数，$\mathrm{Pri_ori}[i]$原始优先权参数，为预先设置的初始配置优先权值，Pkt_size_max 为最大数据包尺寸，$\mathrm{Pkt_size}[i]$当前数据包尺寸。在进行优先权值生成时，一方面需要考虑原始优先权值得因素，另一方面需要根据当前时刻队列中数据包的具体执行因素，采用 Pkt_size_max 与 $\mathrm{Pkt_size}[i]$的比值形式，首先寻找出当前排队数据包中最大尺寸得数据包，然后排出各数据包相对最大尺寸数据包的比重值，让原始优先权值高且尺寸较小的数据包获得优先执行，使加权后的综合优先权值提前。该算法一方面考虑了不同队列优先级的差异，另一方面考虑了不同队列的公平性，是一种较平衡的解决方案[13]。

3.2.2. 主要工作流程

优先权扩展流程如图 3 所示，详细流程步骤如下：

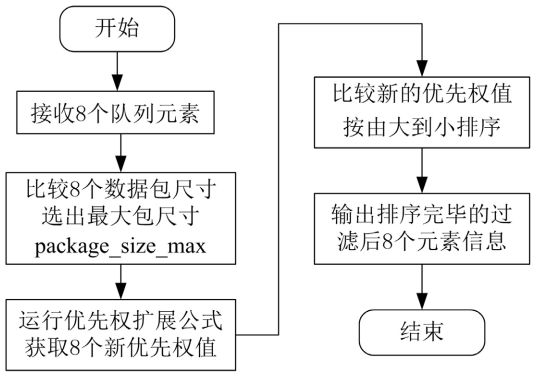

Figure 3. Flow chart of priority expansion
图 3. 优先权扩展流程图

1) 接收入口队列弹出的 8 个 64 bits 队列元素。

2) 比较 8 个元素信息中包含的数据包尺寸信息 package_size，计算出当前 8 个数据包中的最大数据包尺寸 package_size_max。

3) 执行公式计算，计算出每个数据包尺寸同当前最大数据包尺寸的比值，并同原始优先级数 package_priority 做加法处理，为每个数据包获得新的优先权值 priority_now。

4) 比较 8 个新优先权值 priority_now，为 8 个优先权值做从大到小的排序处理。

5) 将排序完毕的优先权值对应的队列元素过滤后的有效信息，即 16 bits 队列元素，按照优先权值由大到小的次序排列后，做 8*16 bits 输出。

3.3. 出口阶段仲裁模块

在出口阶段仲裁模块之前设置缓存 FIFO (First Input First Output)，用来缓存中间阶段优先权扩展模块处理完毕的队列元素的同时，为出口阶段仲裁模块提供等待弹出的队列元素。缓存 FIFO 的宽度 width 为 128 bits，深度 depth 为 2^12，128 bits 位宽包含 8 个 16 bits 的队列元素数据。

3.3.1. 设计原理

出口队列包含 8 个队列元素。出口阶段仲裁模块主要完成已扩展优先权队列元素的弹出工作。出口阶段仲裁模块从上级缓存模块中一次获取 8*16 bits 的队列元素，当出口队列所在的通道获得目的端响应

时，出口阶段仲裁模块仲裁一次，将位于出口队列顶部最高优先级的队列元素弹出，并等待接收端检测传输是否建立，且发回反馈信号。对于得到成功传输的队列元素，出口队列整体前移一次，由原来位于第二位的队列元素成为新的顶部元素，并等待下一次响应。对于未得到成功传输的队列元素，重新进入出口队列尾部，同时其他队列元素出口队列整体前移一次，由原来位于第二位的队列元素成为新的顶部元素，并等待下一次响应。

在此过程中，采用的仲裁方式为固定优先级仲裁(Fixed-Priroity) [14]，即优先级最高的顶部元素优先获得仲裁机会。第一轮响应中，队列元素按照扩展后的优先权排序，优先级最高的队列元素排在队列顶部，优先级最低的队列元素排在队列尾部。第二轮响应中，第一次响应中未得到传输的队列元素从队列尾部重新加入队列时，默认其在当前队列中的优先级降到最低。

出口队列中的每个队列元素均有两次等待响应的机会，第一次响应但未获得传输的队列元素从队列尾部再次进入队列并等待第二次响应。出口队列中的 8 个队列元素均完成至少一次响应后，队列才会清空，并进行下一批次 8*16 bits 队列元素的推入和仲裁操作。对于两次响应均未获得传输的队列元素进行收集和上报主机，由主机决定是否再进入队列系统进行队列管理操作。

出口阶段仲裁模块的两轮等待机制，一方面可以避免顶部最高优先级队列元素由于迟迟得不到响应带来的队列阻塞和“饥饿”问题[15]，另一方面可以使每个数据包任务拥有合理均衡的响应弹出机会，从而提升整个队列管理器的处理性能[16]。

3.3.2. 出口阶段仲裁状态机描述

出口阶段仲裁模块具体状态转移如图 4 所示。

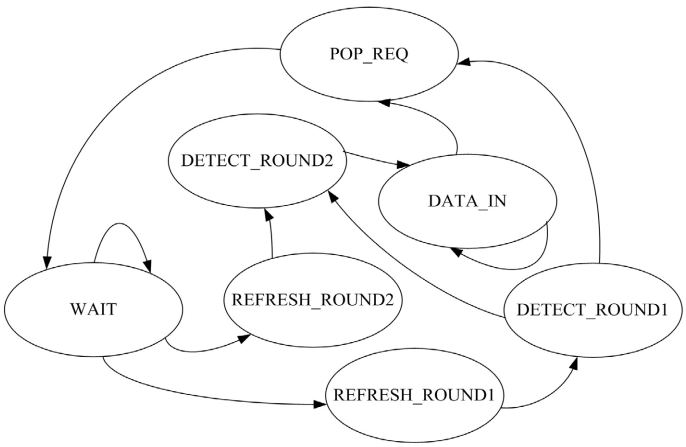

Figure 4. State transition diagram of export-level arbitration module
图 4. 出口阶段仲裁模块状态转移图

数据接收模块接收效数据后，向下级模块产生弹出请求 pop_req，并等待下级模块的处理响应信号，当接收到响应完成信号 finish 且 detect_cnt 小于 8，对当前队列进行第一轮弹出刷新。队列中 8 个元素均完成第一轮弹出响应结束后，开始第二轮弹出响应。检测 detect_cnt_round 2 是否小于 8，对当前队列进行第二轮弹出刷新。第二轮弹出响应结束后，开始下一轮有效数据的接收。

3.4. 多对多仲裁模块

多对多仲裁模块完成一个多输入多输出间的仲裁，输入主要是来自源端的出口队列中弹出的队列元素。该模块共有 12 个源端，包括 8 个 DSP 核心和 4 个外部设备。输出的目的端包括 1 个用于核间通信

的公共接收端和 4 个外部设备。在进行多对多仲裁时，可将该模块分成多选一和一对多两个步骤来进行。当源端均有队列元素请求传输时，先对 12 个源端进行 12 选 1 的操作，采用轮询仲裁机制依次响应各个有数据传输请求的源端，每次只产生一个仲裁结果，并将该结果送至 1 对 5 选择模块中，由该模块根据队列元素中包含的具体目的端编号信息，将队列元素输出至对应目的端。

4. 仿真结果与分析

通过 Verilog 编写各个功能模块，利用 Xilinx Vivado 2018.3 综合和功能仿真，验证各模块实现的的功能是否与设计要求的功能相符，对不满足功能设计的模块，通过仿真分析定位设计上的不足和硬件实现上的问题，在 Xilinx 公司的 XCVU440-FLGA2892-1-C 型 FPGA 芯片上完成硬件实现[17] [18]。关键电路模块仿真图如下所述。

4.1. 优先权扩展模块仿真分析

将原始优先级、数据包尺寸均不同的 8 个队列元素发送至优先级扩展模块。在原始优先级较高且数据包尺寸较小的情况下，队列元素经过优先权扩展后，将获得较高扩展后优先权值。

优先权扩展模块仿真如图 5 所示，输入的队列元素数据 pointer_addr_i 按照扩展后的优先权值 priority_now 排序输出结果 pointer_addr。如第一个子队列输入的队列元素数据 pointer_addr_i_0 的扩展优先权值 priority_now 为 9，为当前最大权值，故将 pointer_addr_i_0 排在首位输出，即 pointer_addr_7 的值等于 pointer_addr_i_0 的值。仿真波形证明了不同优先级和尺寸的数据包队列元素可以按照优先权扩展公式获取相应扩展后优先权。

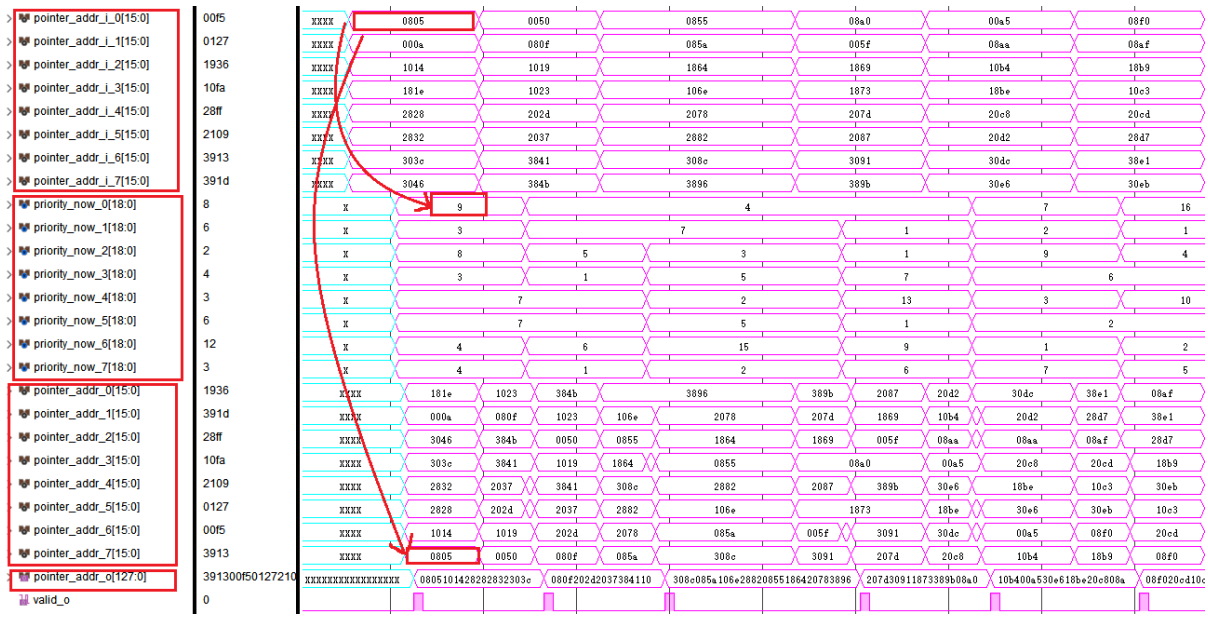

Figure 5. Simulation diagram of priority result output
图 5. 优先权结果输出仿真图

4.2. 出口阶段仲裁模块仿真分析

出口阶段仲裁模块接收来自上级缓存中的 8*16 bits 队列元素数据 data_i，对每个数据做弹出 pop_req 操作。data_detect 为每次申请弹出操作的 16 bits 数据，按照 data_i 中由高位到低位的输出顺序，依次为

data_detect 赋值。得到响应 pdma_ack 的数据 data_o 为出口队列中实际弹出的数据，未得到响应处理的数据在出口队列中会再经历第二轮弹出申请，最终仍未得到响应处理的数据 addr_no_response_o 作为队列中的丢弃数据返回至控制器中做接收记录处理。出口阶段仲裁模块输入输出仿真如图 6 所示，仿真波形证明出口阶段仲裁模块可以实现有效两轮弹出，包括成功响应数据的有效弹出和未成功响应数据的回收弹出。

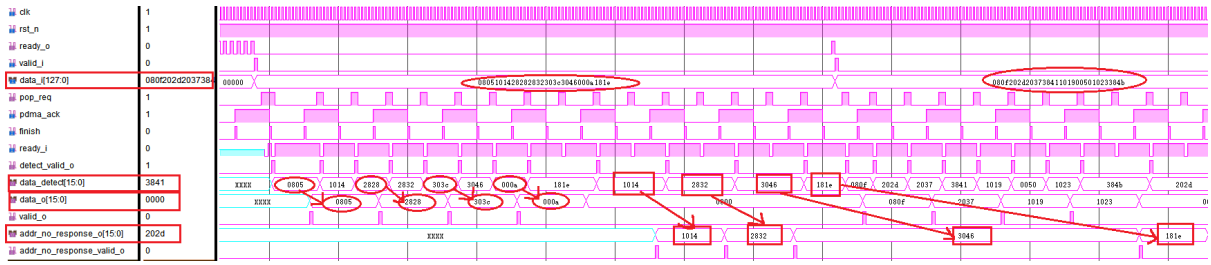

Figure 6. Simulation diagram of export-level arbitration module
图 6. 出口阶段仲裁模块仿真图

4.3. 总体模块仿真分析

在入口队列 ready_o 拉高的条件下，源端连续向队列管理器各通道子队列中发送队列元素，通过多级仲裁管理后，由出口队列发起弹出请求 pop_req，获得目的端响应 pdma_ack 后，完成弹出。其中，输入 64 bits 的 pointer_pacakage_i，输出 16 bits 的待弹出队列元素 data_detect、16 bits 的真实弹出队列元素 exit_data_o。

从处理器核和外设的角度，源端和目的端分别有以下 4 种情形：源端为处理器核，目的端为另一处理器核；源端为外设，目的端为处理器核；源端为处理器核，目的端为外设；源端为外设，目的端为另一外设。

针对以上四种传输，建立具体实验传输情形如下。第一个时间段，处理器核 0 作为源端发送数据，处理器核 1 作为目的端接收；第二个时间段，外设 0 作为源端发送数据，处理器核 2 作为目的端接收；第三个时间段，处理器核 1 作为源端发送数据，外设 1 作为目的端接收；第四个时间段，外设 2 作为源端发送数据，外设 3 作为目的端接收数据。

仿真波形如图 7、图 8 所示，其中图 7 为每种传输在一个时间段内连续发送 8 个 pointer_pacakage_i，图 8 为每种传输在一个时间内连续发送 64 个 pointer_pacakage_i。根据不同源端和目的端编号，依次获取通道号 channel_id 分别为 1、2、13、15。data_detect 为核间传输公共目的端的接收数据，data_detect_p1 为外设 1 的接收数据，data_detect_p3 为外设 3 的接收数据。仿真波形证明数据从目的端输入，经过中间仲裁管理后，可以为目的端接收，本文研究的多核数据管理器仲裁机制可以实现数据的有效仲裁管理。

4.4. 总体模块资源消耗

多核队列管理器仲裁机制资源消耗见表 1 所示。

5. 结束语

本文研究并设计一种基于数据包传输的管理器仲裁机制，通过多级仲裁从而控制数据包对应的队列元素在队列中的进程，完成多核与外部设备之间高效数据交换。为了传输过程中各数据包能得到公平高效的管理和传输，加入基于数据包的优先权扩展机制，通过在原始优先级的基础上根据系统接收情形为每一个排队中的数据包重新赋予权重，达到灵活批量传输多权重数据的效果。同时，采用优先级多级

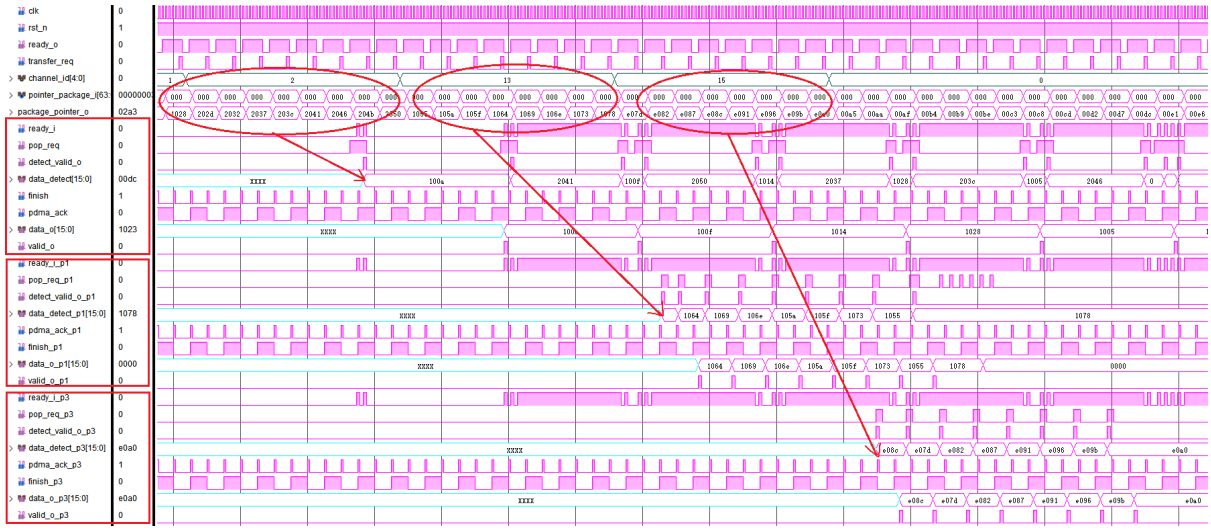

Figure 7. Continuous input of 8 data simulation diagrams
图 7. 连续输入 8 个数据

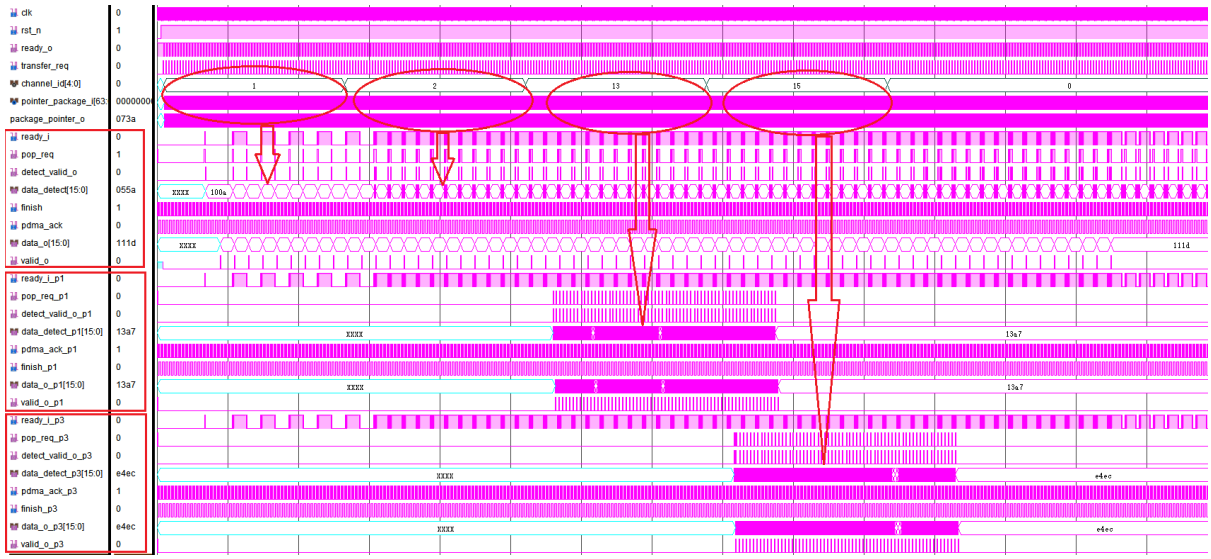

Figure 8. Continuous input of 64 data simulation diagrams
图 8. 连续输入 64 个数据

Table 1. Overall resource consumption table
表 1. 总体资源消耗表

Resources	Estimation	Available	Utilization %
LUT	169,738	2,532,960	6.70
FF	98,219	5,065,920	1.94
BRAM	348	2520	13.81

调整策略，在队列入口和队列出口采用轮询调度算法，保证每个队列均有得到响应的机会，避免产生数据包拥塞，实现数据高效搬移。随着多核技术的进一步发展，多核片上数据传输技术也将不断升级，采用更加实时高效的仲裁机制也正成为值得研究的方向之一。

参考文献

[1] 樊超, 詹思维, 李博. 基于多核 DSP 的高清图像跟踪系统[J]. 电子技术, 2021, 50(3): 4-5.

[2] 秦昳, 史晓楠, 巨新刚. 多核处理器核间的通信研究与实现[J]. 现代电子技术, 2016, 39(16): 83-87.

[3] 陈术涛, 沈志, 王春联, 等. 多核 DSP 与 FPGA 高速数据传输系统设计与实现[J]. 电子技术应用, 2018, 44(12): 40-43.

[4] 魏智伟. 多核 DSP 间基于 SRIO 数据传输的设计与实现[J]. 微型机与应用, 2017, 36(4): 36-39.

[5] 王磊, 刘道福, 陈云霁, 等. 片上多核处理器共享资源分配与调度策略研究综述[J]. 计算机研究与发展, 2013, 50(10): 2212-2227.

[6] 蔡想伟. CPU-GPU 异构系统下的片上网络仲裁机制研究[J]. 电子设计工程, 2018, 26(8): 93-101.

[7] Li, Z., Yang, G., Liu, S., *et al*. (2018) A Task-Based Multi-Core Allocation Mechanism for Packet Acceleration. *IEICE Electronics Express*, **15**, 9 p. https://doi.org/10.1587/elex.15.20180414

[8] 牛金海. TMS320C66xKeystone 架构多核 DSP 入门与实例精解[M]. 上海: 上海交通大学出版社, 2015: 1-106.

[9] 夏际金, 赵洪立, 李川. TI C66x 多核 DSP 高级软件开发技术[M]. 北京: 清华大学出版社, 2017: 1-131.

[10] 周超群, 周亦敏. 一种改进的基于复制的异构多核任务调度算法[J]. 电子科技, 2017, 30(6): 57-62.

[11] 钱鹏飞, 乔庐峰, 陈庆华. 一种在大容量交换中可以保证 QoS 的队列调度器设计与实现[J]. 通信技术, 2018, 51(11): 2655-2661.

[12] Guo, Y. and Li, Z. (2018) A Novel Priority-Allocated Scheme for Flow-Based Queue Managers. 2018 *IEEE* 20*th International Conference on High Performance Computing and Communications*; *IEEE* 16*th International Conference on Smart City*; *IEEE* 4*th International Conference on Data Science and Systems* (*HPCC*/*SmartCity*/*DSS*), Exeter, 28-30 June 2018, 1002-1006. https://doi.org/10.1109/HPCC/SmartCity/DSS.2018.00166

[13] 徐金波, 常俊胜, 李琰. 支持多优先级多输出通道的数据队列调度方法和硬件实现[J]. 计算机工程与科学, 2020, 42(10): 1749-1756.

[14] 章婷婷, 彭敏, 周清峰, 等. 无线体域网中动态分配队列长度的调度算法[J]. 合肥工业大学学报(自然科学版), 2019, 42(7): 906-911.

[15] 罗乐, 王春华, 张多利, 等. 一种多核系统改进型列表调度算法[J]. 电子科技, 2020, 33(6): 52-57.

[16] Benacer, I., Boyer, F.R. and Savaria, Y. (2017) A High-Speed Traffic Manager Architecture for Flow-Based Networking. 2017 15*th IEEE International New Circuits and Systems Conference* (*NEWCAS*), Strasbourg, 25-28 June 2017, 161-164. https://doi.org/10.1109/NEWCAS.2017.8010130

[17] 赵海峰. 高性能队列管理与调度技术研究[D]: [硕士学位论文]. 西安: 西安电子科技大学, 2018: 1-65.

[18] 李润泽. 多核数据导航器的设计与验证[D]: [硕士学位论文]. 长沙: 国防科技大学, 2018: 1-54.