

基于FPGA的SRIO多通道控制系统设计与实现*

薛培¹, 官剑¹, 邵春伟², 张鑫刚¹, 郑思旭¹

(1. 无锡华普微电子有限公司, 江苏 无锡 214000; 2. 中国电子科技集团公司第五十八研究所, 江苏 无锡 214000)

摘要: 在板间互联及芯片互联方式上, SRIO具有更高的带宽及实时性。使用MSG(消息)接口的HELLO格式, 发送端采用多接口设计方法, 内部采用Round-Robin处理机制, 实现了多通道接口在同时发送数据时共用一个SRIO接口的竞争处理; 同时封装为多通道的输入输出的方式, 支持接口数量、时钟域的任意扩展。经过测试验证, 该系统最大可实现64个不同时钟下通道的数据收发, 单通道下包和包之间延时最低可到4 μs, 在SRIO传输应用方向上, 具有较好的应用价值。

关键词: XILINX; SRIO; 多通道; Round-Robin

中图分类号: TP334.4

文献标志码: A

DOI: 10.16157/j.issn.0258-7998.222857

中文引用格式: 薛培, 官剑, 邵春伟, 等. 基于FPGA的SRIO多通道控制系统设计与实现[J]. 电子技术应用, 2023, 49(1): 107-113.

英文引用格式: Xue Pei, Guan Jian, Shao Chunwei, et al. Design and implementation of SRIO multiple channel control system based on FPGA[J]. Application of Electronic Technique, 2023, 49(1): 107-113.

Design and implementation of SRIO multiple channel control system based on FPGA

Xue Pei¹, Guan Jian¹, Shao Chunwei², Zhang Xingang¹, Zheng Sixu¹

(1. Wuxi Hope Microelectronics Co., Ltd., Wuxi 214000, China;

2. No.58 Research Institute of China Electronics Technology Group Corporation, Wuxi 214000, China)

Abstract: In board and chip interconnection, SRIO(Serial RapidIO) has higher bandwidth and reliability than other protocol. In this paper, the HELLO format of the MSG (message) interface is used to multi-interface design method at the sending end, and the Round-Robin processing mechanism is used internally to realize the competition processing of the multi-channel interface sharing one SRIO interface when sending data at the same time. It supports the arbitrary expansion of the number of interfaces and the clock domain. After testing and verification, the system can realize the data sending and receiving of 64 channels under different clocks at most, and the delay between the single channel and the packet can be as low as 4 μs. It has good application value in the direction of SRIO transmission application.

Key words: XILINX; SRIO; multiple-channel; Round-Robin

0 引言

随着嵌入式系统性能的快速提升, 板间互连、板内芯片互连对数据带宽、可靠性、灵活性要求逐步提升, 传统的并行传输方式已经很难满足实际的需求。Serial RapidIO(SRIO)高速串行总线协议, 成为一种使用范围极广的高速协议^[1]。SRIO是一种基于数据包交换的互联体系结构, 在芯片互联、实时视频传输、小型化系统等场合得到了广泛的使用^[2]。

在国内外有关SRIO文献的研究中, 研究较多的有SRIO和FPGA之间通信^[3]、FPGA实现多协议转换SRIO设计^[4], 本文即为研究多协议接口和SRIO之间的转换。

本文以XILINX公司FPGA提供的SRIO IP为基础, 设计了多通道发送和接收控制系统的设计方法, 其中发送通道采用了一种Round-Robin数据处理方法, 有效控制多通道数据同时发送时可能造成的数据拥塞。基于该设计方法而开发的SRIO多通道控制系统, 经过仿真与调试, 已用于某型号项目通信系统, 达到了高带宽、低延时的效果, 具有很好的通道扩展性。

* 基金项目: 辽宁省科技厅联合开放基金机器人学国家重点实验室开放基金资助项目(2021KF2205)

本系统采用MSG消息方式,利用MSG接口中的 mailbox 空间作为通道ID区分,采用该方法,相较于其他多接口 SRIO 设计中采用Nwrite/Nread的方法。具有以下优点:

(1)无需占用 SRIO payload 区域,提高了单个包的带宽。

(2)最大支持 64 个通道的输入输出,具备良好的扩展性。

(3)发送端采用 Round-Robin 机制,多协议接口具备优先级控制,相较于其他方法,在不同优先级通道同时发送数据时,可实现通道差异化。

1 SRIO 简介

SRIO 采用三层分级体系结构,该结构如图 1 所示。

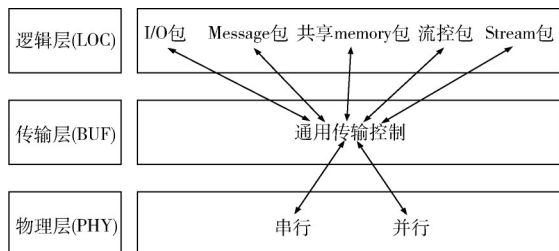


图1 SRIO 三层分级体系

其中逻辑层位于最高层,定义了包的格式,包为端点发起和完成事务提供必要的信息;传输层位于中间层定义了SRIO地址空间和端点器件间传输包所需的路由信息;物理层在整个分级结构的最底部,主要规定了物理层方式^[5]。

SRIO 数据包由包头、有效的数据载荷和 16 位 CRC 校验组成^[6]。包头的长度最大的有效载荷长度为 256 B。由于包长度短,因此传输延时较小,硬件上也易于实现,适合数字信号处理场合对传输延时要求较高的应用^[7]。XILINX 的部分 FPGA 中^[4]已经集成了 GTP、GTX 或 GTH 等高速串行收发电路,这些是 FPGA 实现 SRIO 高速传输的物理层基础。

在 XILINX 的开发环境下,可直接调用 Serial RapidIO gen2 IP 来实现 SRIO 的 LOG、BUF 及 PHY 三层结

构^[8]。在本文所述的系统中,IP 设置为 Condensed I/O 模式,Messaging 接口设置为 Separate Messaging Port。

完成如上的设置,生成 IP 后,开放的主要接口为 4 组: MSGIREQ、MSGIRESP、MSGTREQ、MSGTRESP, 这些接口间的信号流向和功能分别如图 2 和表 1 所示。

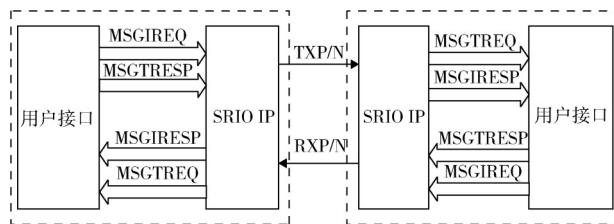


图2 SRIO IP 接口框图

表1 SRIO 接口功能表

SRIO 接口	方向	功能	HELLO 包类型
MSGIREQ	本地 SRIO->对端 SRIO	发送数据	MSG
MSGIRESP	对端 SRIO ->本地 SRIO	发送数据-确认	RESP
MSGTREQ	对端 SRIO ->本地 SRIO	接收数据	MSG
MSGTRESP	本地 SRIO->对端 SRIO	接收数据-反馈	RESP

本系统的包头使用 HELLO 包^[9]中的 MSG 和 RESP 包。包头格式见图 3。

在该系统中,为了适应任意数量的发送通道和接收通道,需要将通道ID加入到SRIO包中。本系统采用了MSG包的mailbox参数和RESP包中特有的TargetTID参数来表示当前发送通道和接收通道的通道数信息。

2 系统架构

SRIO IP 开放的接口均为 AXI4-stream 接口^[10]。其中 MSGIREQ 和 MSGIRESP 为发送控制处理逻辑模块所连接的接口,本地 SRIO 节点所有发送通道的数据均需要转换为这两组接口的 HELLO 包。MSGTREQ 和 MSGTRESP 接口为从对端 SRIO 接收到的接口,经过接收控制处理逻辑模块对 HELLO 包进行处理,转化为接收通道输出的数据。

系统整体框架如图 4 所示。

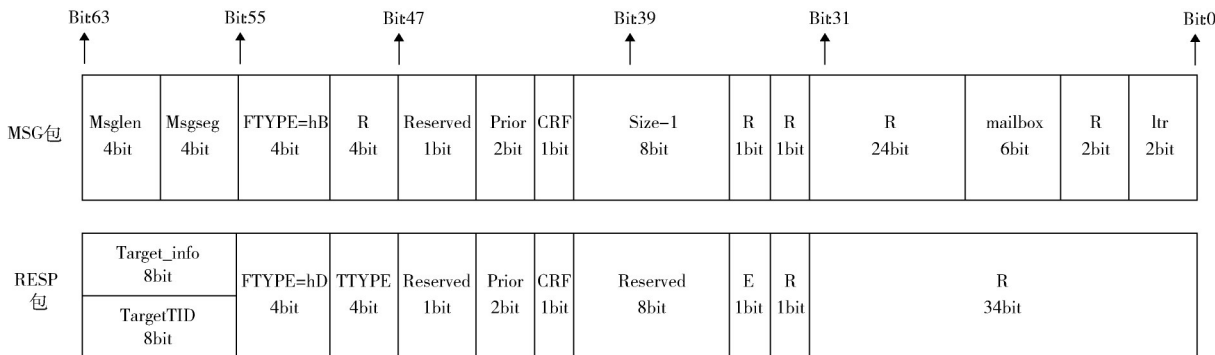


图3 HELLO 帧头格式

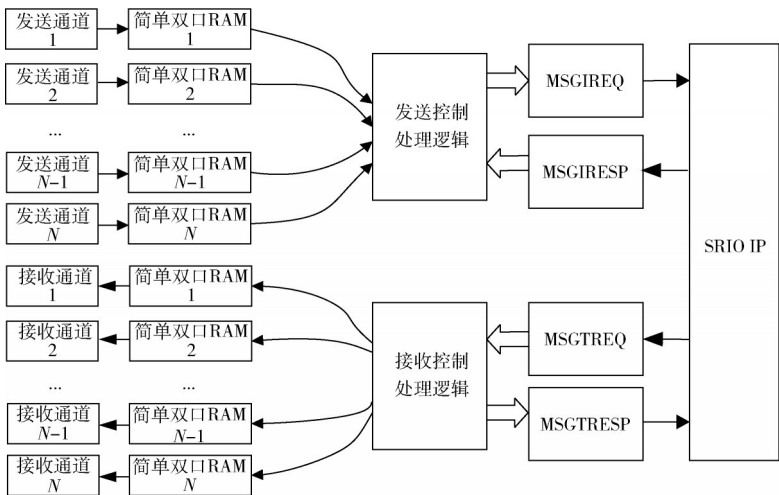


图4 系统整体框架

在发送控制处理逻辑,根据待发送的发送通道数信息(假设为 M 通道),将 M 填充到MSGIREQ包的mailbox参数中,实现本地SRIO的数据发送;发送完成后,发送通道(M)进入到等待ACK的状态。发送控制处理逻辑仍然可继续其他发送通道的数据发送。

当MSGIRESP收到RESP包(TargetTID信息等于 M)时,发送通道 M 从等待ACK状态进入到收到ACK状态。此时发送通道 M 认为对端SRIO已正确接收上一次的发送数据。

在接收控制处理逻辑中,其起始状态为接收到MSGTREQ接口的MSG包头。根据mailbox参数(假设为 N ,最大可支持到64),则该MSG包的接收通道为 N ,同时提取payload数据转换为接收通道 N 的输出时序,完成后,向MSGTRESP接口发送RESP包,其TargetTID参数填充为 N ,实现对端SRIO包的响应反馈。

基于以上架构,可实现多路不同时钟域、不同速度的数据在SRIO链路之间的传输。

2.1 通道接口时序

本系统采用一种通用的发送通道和接收通道时序,如图5所示。

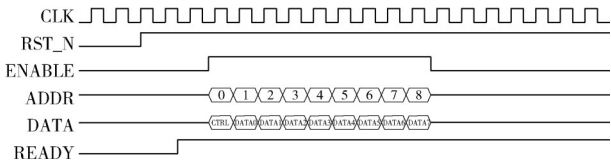


图5 通道接口时序

接口由6个信号组成,分别是时钟CLK、复位RST_N、数据有效标志ENABLE、地址ADDR、数据DATA、准备标志READY,数据位宽为16 bit。

所有的信号均为CLK时钟下的同步信号,RST_N低电平复位,ENABLE、ADDR、DATA必须在READY为高

时有效;ENABLE为高时,地址ADDR=0时的DATA值为CTRL值,CTRL值内含一次SRIO包的数据长度等信息,其中最高bit为1时意味着SRIO传输包有效,低位bit代表着包的包长;ADDR=1到ADDR= N 范围内的DATA值为一次SRIO传输包的有效数据。由于一个SRIO包最大支持256 B,则该地址位 N 最大为128。

当有多个不同时钟域、不同收发速度的数据接口接入到SRIO系统时,均可通过极简单的控制逻辑转换为该接口的时序,实现SRIO的快速应用。

2.2 发送控制模块

发送控制模块主要由双口RAM、Round-Robin模块和发送控制状态机三部分组成。发送接口数据首先按接口时序中的地址信号存入到发送双口RAM模块,等待后续模块读取。Round-Robin模块用于实现所有通道发送的优先级顺序处理;发送控制模块则完成发送双口RAM通道和SRIO IP接口(MSGIREQ和MSGIRESP)的转换。发送模块整体框架如图6所示。

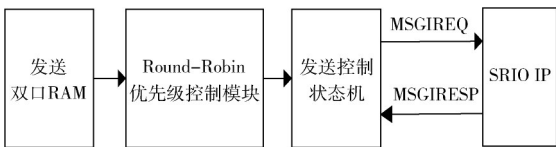


图6 发送模块整体框架

2.2.1 发送双口RAM

发送双口RAM模块核心为简单双口RAM,其中A口为16 bit数据;B口为64 bit数据。

写入部分将多通道接口时序发送部分的数据写入到双口RAM的A口中,当I_srio_tx_enable为高时I_srio_tx_addr[15:0]/I_srio_tx_datain[15:0]信号分别写入双口RAM的地址和数据部分。其中地址0存入的数据,最高位为状态位,低位为word个数。

读出部分为发送控制状态机控制。

2.2.2 Round-Robin 优先级控制模块

Round-Robin(轮询调度算法)优先级控制模块根据不同输入通道的优先级顺序,实现不同重要性通道的数据发送 SRIO 包的先后顺序处理。

Round-Robin 的原理是每一次将来自输入接口的请求轮流分配给内部中的发送处理逻辑,从优先级 1(优先级最高)开始,直到优先级 N (优先级最低),然后重新开始循环,来实现各个任务的顺序执行。同时,每个输入接口有一个 Aging(变化值),保证了等待过久的任务也依然能被执行到。

每个通道都被分配了一个优先级号码,作为 Aging CountDown(递减计数)的起始值。也就是说,优先级越高的进程,会更早地 CountDown to 0,即最高优先级,从而更早地被执行。

本模块所用的状态机如图 7 所示。

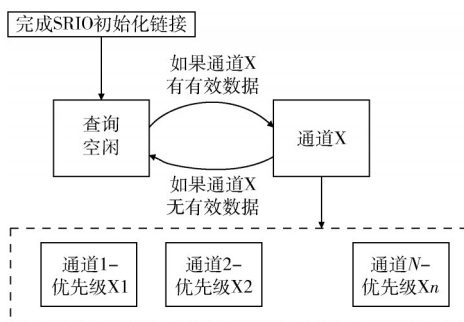


图7 Round-Robin 状态机图

每个通道均具有一个优先级,举例通道 1 为 X_1 优先级;通道 2 为 X_2 优先级...通道 N 为 X_n 优先级。

在完成 SRIO 链路初始化后($port_initialized$ 和 $link_initialized$ 均为 1),首先进入查询空闲的状态,接着进入到读取通道 X 的模式。其中查询空闲和读取通道 X 两个状态互相循环,即完成读取完某一个通道的数据后,先返回到查询空闲状态,再进入读取通道 X 状态。

首先读取最高优先级 X_{max} 的通道 MAX,同时 Aging 计数 $Aging(max) - 1$,查询状态(通过读取地址 $I_srio_tx_addr$ 为 0 时的 $I_srio_tx_datain$ 最高位)是否为 1,如果为 1,则开始通道 MAX 的数据读取,如果等于 0,则进入查询其他通道 X 的状态。读取完通道 MAX 的数据后,进入到查询空闲状态。

依此方式循环,即可以兼顾优先级的次序,又可以实现低优先级长时间轮询失败的问题。

2.2.3 发送控制状态机

发送控制状态机实现从优先级控制模块输出的对应 RAM 读出数据,转换为 MSGIREQ 和 MSGIRESP 两个接口时序的模块。状态机如图 8 所示。

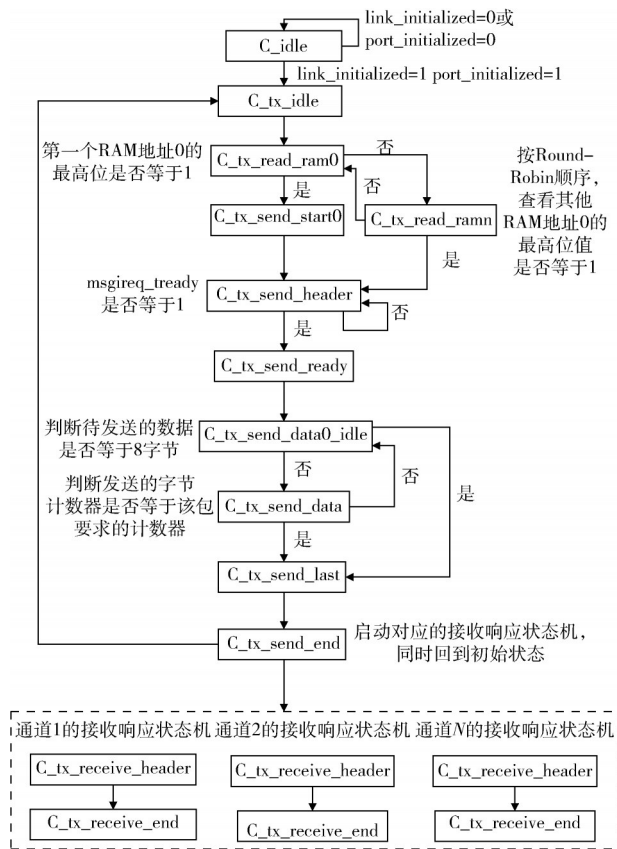


图8 发送控制状态机图

如图 8 所示:上电后 C_idle 状态下,在完成 SRIO 链路初始化后($port_initialized$ 和 $link_initialized$ 均为 1),发送状态机进入到 C_tx_idle 状态。

C_tx_idle 状态下,直接进入 $C_tx_read_ram0$ 状态。

$C_tx_read_ram0$ 状态下,读通道 0 的发送双口 RAM-0 中地址 0 的数据。如果地址 0 数据的最高位为 1,代表该通道有待发送的数据,则进入 $C_tx_send_start0$ 状态。

$C_tx_send_start0$ 状态下,如果最高位为 0,则进入到 $C_tx_read_ramn$ 状态。

$C_tx_send_start0$ 状态下,进入 $C_tx_send_header$ 状态。

$C_tx_read_ramn$ 状态下,按 Round-Robin 模块输出的读通道顺序读取该通道的 RAM 地址 0 中的数据。如果数据的最高位为 1,代表该通道有待发送的数据,则进入 $C_tx_send_header$ 状态;如果最高位为 0,则重新到 $C_tx_read_ram0$ 状态,回到读 RAM-0 状态。

$C_tx_send_header$ 状态下,判断 MSGIREQ 接口的 $msgirq_tready$ 值,如果等于 1,即可认为 MSGIREQ 接口可以发数据,此时进入到 $C_tx_send_ready$ 状态。

$C_tx_send_ready$ 状态,输出 $msgirq_tvalid$ 和 $msgirq_tdata[63:0] = MSG$ 包头的的数据。接着进入到 $C_tx_send_data0_idle$ 状态。

$C_tx_send_data0_idle$ 状态下,需要判断待发送的数

据包长度是否为8字节,如果小于或等于8字节,直接进入C_tx_send_last状态;如果大于8字节,则进入到C_tx_send_data状态。

C_tx_send_data状态下,输出msgireq_tvalid=1和msgireq_tdata[63:0]=payload数据给到MSGIREQ接口。

C_tx_send_last状态下,输出msgireq_tvalid=1、msgireq_tdata[63:0]=payload数据到MSGIREQ接口。此状态下,即完成一整个SRIO包的通道数据发送。完成后进入到C_tx_send_end状态。

C_tx_send_end状态下,重新回到C_tx_idle状态。同时相应通道的接收响应状态机会启动,等待对端SRIO的响应包,使用此方式,不影响通道数据的继续发送。

如当前发送的为第Y个通道,则通道Y的接收响应状态机会进入到C_tx_receive_header状态,此刻此通道需要等待MSGIRESP回应对应通道的响应包,如果接收到MSGIRESP接口的msgiresp_tvalid=1,同时msgiresp_tdata[63:54]值即TargetTID值等于Y,即接收到Y通道发送数据的响应。此时完成Y通道的所有发送流程。

基于以上方式,可实现N个通道接口的数据流水式发送,在完成一个通道的发送后,可以继续其他通道的数据发送。同时能收到RESP包,确保每个发送通道发送数据的可靠性。

2.3 接收控制模块

接收控制模块主要由接收双口RAM和接收控制状态机两部分组成。通过SRIO接口MSGTREQ和MSGTRESP的数据,首先进入到接收控制状态机,经过对MSG包的解析,分析确定mailbox位通道数,写入到对应的接收双口RAM。最后发出RESP包到MSGTRESP接口。完成该操作后,即可完成SRIO多通道数据的接收。接收模块整体框架如图9所示。

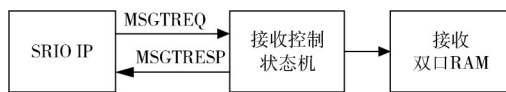


图9 接收模块整体框架

2.3.1 接收控制状态机

接收控制状态机实现SRIO IP的MSGTREQ和MSGTRESP两个接口处理的逻辑。该模块接收MSGTREQ接口的数据,解析MSG包中帧头部分的mailbox值,确定接收到的SRIO包对应的接收通道,同时完成该SRIO包的payload部分接收并写入到后续接收双口RAM模块,完成该操作后,再发送RESP包给MSGTRESP接口。整个接收控制状态机流程如图10所示。

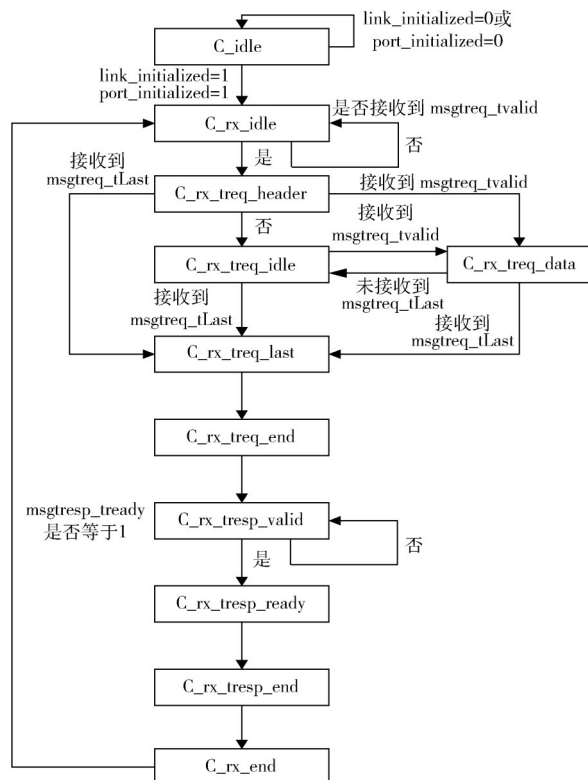


图10 接收控制状态机

如图10所示,上电后接收状态机进入到C_idle状态。完成SRIO链路初始化后,发送状态机进入到C_rx_idle状态。

C_rx_idle状态下,判断是否接收msgtreq_tvalid=1,如果接收到,直接进入C_rx_treq_header状态。

C_rx_treq_header状态下,如果接收到msgtreq_tvalid=1,则认为接收到MSG包的帧头,此时进入到C_rx_treq_data状态;如果接收到msgtreq_tlast=1,则认为接收到MSG的最后一个64 bit payload数据,此时进入到C_rx_treq_data状态;如果未接收到以上两种状态,则进入到C_rx_treq_idle状态。

C_rx_treq_data状态下,如果接收到msgtreq_tlast=1,则接收到MSG的最后一个64 bit payload数据,此时进入到C_rx_treq_last状态;如果未接收到,则进入到C_rx_treq_idle状态。

C_rx_treq_idle状态下,如果接收到msgtreq_tvalid=1,则认为接收到MSG包的payload数据,此时进入到C_rx_treq_data状态;如果接收到msgtreq_tlast=1,则认为接收到MSG的最后一个64 bit payload数据,此时进入到C_rx_treq_last状态;如果未接收到以上两种状态,则继续保持本状态。

C_rx_treq_last状态下,则认为完成一个SRIO的接收处理,SRIO payload中的数据需要存入到接收RAM模块中。状态进入到C_rx_treq_end状态。

C_rx_treq_end 状态下,进入 C_rx_resp_valid 状态,后续为 MSGTRESP 接口的处理。

C_rx_resp_valid 状态下,判断 msgtresp_tready, 如果为 1,则进入到 C_rx_resp_ready 状态;如果为 0,则继续保持本状态。

C_rx_resp_ready 状态完成响应包 RESP 的发送。

2.3.2 接收控制状态机

在 C_rx_treq_header 状态时,从接收到的 SRIO 包头 TargetTID 位,确定收到的包的 SRIO 通道。同时根据 C_rx_treq_data 和 C_rx_treq_last 状态下收到的 msg_treq_data,解析出 payload 部分,写入到双口 RAM 的 A 口。

B 口实时监测地址 0 的标志位,当标志位=1 时,读取存入到 B 口的数据,并转换为通道通用时序。

3 时序仿真

如图 11 所示,为 SRIO 3 通道系统的发送时序仿真。

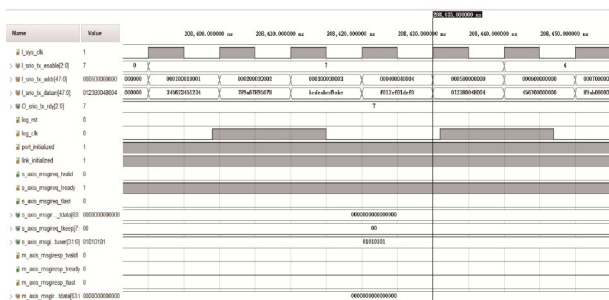


图 11 发送接口时序仿真图

该仿真设置的优先级顺序为:通道 2>通道 1>通道 3。

在 I_sys_clk 时钟下 I_srio_tx_enable[0]/I_srio_tx_addr[15:0]/I_srio_tx_datain[15:0]为第一通道,前 4 个时钟下,分别向地址 1、2、3、4 写入了 16'h1234、16'h5678、16'h9ABC、16'hDEF0 数据;第 5 个时钟下,向地址 0 写入了控制数据 16'h8004,最高 bit 位为 1 代表着标志位,4 代表着存有 4 个 word。

I_srio_tx_enable[1]/I_srio_tx_addr[31:16]/I_srio_tx_datain[31:16]为第二通道,前 4 个时钟下,分别向地址 1、2、3、4 写入了 16'h2345、16'h6789、16'hABCD、16'hEF10 数据;第 5 个时钟下,向地址 0 写入了控制数据 16'h8004,最高 bit 位为 1 代表着标志位,4 代表着存有 4 个 word。

发送通道转换为 MSGIREQ 接口的仿真时序如图 12 所示。从图中可以看出,帧头数据为 64'h00B0207000000012,发送的数据为 64'hEF10ABCD67892345。mailbox 位为 1,代表着通道 2。可以看出通道 2 最先发出,实现了优先级的发送顺序控制。

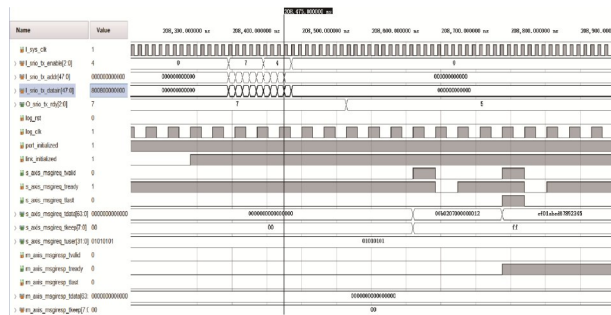


图 12 MSGIREQ 仿真图

发送通道接收到对端 SRIO 的响应包的时序如图 13 所示。

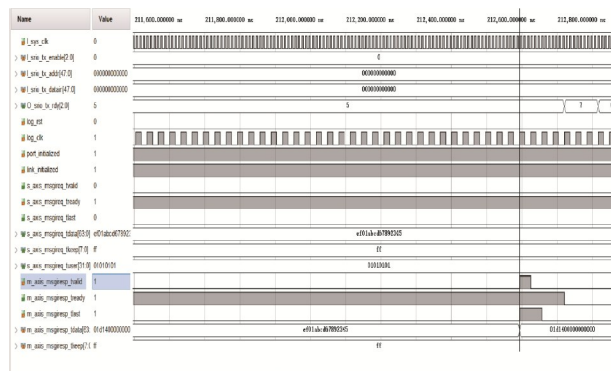


图 13 MSGIRESP 接口仿真图

在图 13 中,212 600 ns 时刻左右,MSGIRESP 接口接收到 RESP 包,其包头为 64'h01D1400000000000。RESP 包的 TargetTID=1,即为 MSG 通道 2 的响应包。与发送通道 MSGIREQ 接口比较,其延时为 212600-208475=4 125 ns,即 4.125 μ s(SRIO 线速率为 2.5 Gb/s)。

接收通道接收到数据的时序仿真如图 14 所示。

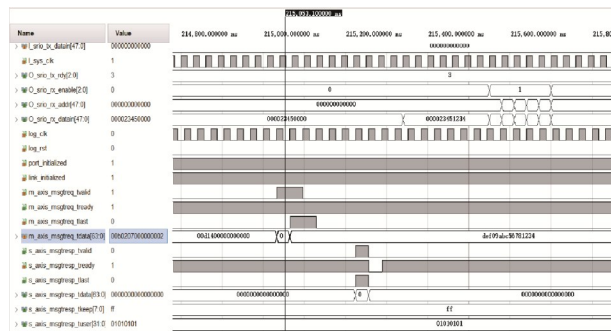


图 14 MSGTREQ 和 MSGTRESP 接口仿真

从图 14 中可见,接收通道的 MSGTREQ 接收到 64'h00B0207000000002,从包头可以分析 MSG 包为通道 1。接收通道输出接口同样为通道 1 有效(O_srio_rx_enable[0]=1),接收时序与设计一致。

4 结论

本文设计了一种 SRIO 应用于多通道发送和接收的控制系统,可降低 SRIO IP 使用的复杂度,同时兼容最大 64 个发送接收通道的直接扩展。通过代码实现与仿真,同一通道发包到确认收包,实际延时可到 4 μ s 左右(2.5 Gb/s 线速率),具有较好的可靠性。

该系统已经应用到某型号项目通信系统,可大幅节约通讯线缆。系统运行稳定可靠,同时对用户来说,接口使用方便,可极大地加速了 SRIO 的应用开发。后续可以通过改进优化,移植到 Nwrite/Swrite/Nread 包^[11]方式,实现更多的 SRIO 通信应用。

参考文献

- [1] LI X W, LIU X Z, LI M, et al. Realization and simulation of high speed flexible transmission system between FPGA chips based on SRIO[J]. Applied Mechanics & Materials, 2014, 543-547:1967-1971.
- [2] 陈刚, 康林, 陈航, 等. 基于 FPGA 的 SRIO 端点设计与实现[J]. 兵工自动化, 2021, 40(2): 49-52.
- [3] 陈婷, 岳强, 汪洋. FPGA 和 DSP 间基于 SRIO 的高速通信系统设计[J]. 单片机与嵌入式系统应用, 2016, 16(3): 37-40.
- [4] 季雷, 李斌, 李大鹏, 等. 一种 SRIO 多接口互联系统的 FPGA 实现方法[J]. 电子技术, 2015(6):6-9.
- [5] 孙发. 基于 FPGA 的综合航电互连体系结构设计[J]. 机械设计, 2018(S1):37-39.
- [6] 张清亮. 基于 FPGA 高速通用串行接口的设计与应用[D]. 西安:西安电子科技大学, 2015.
- [7] 冼友伦, 纪小明, 王显跃. 基于 TS1578 的 Rapid IO 互连技术[J]. 电子与封装, 2014(11):26-30.
- [8] 于东英, 陈俊, 康令州. 基于国产 FPGA 的高速 SRIO 接口设计与实现[J]. 通信技术, 2019, 52(1):255-258.
- [9] 多卉枫, 任勇峰, 武慧军. 基于 SRIO 的数据传输设计[J]. 电子测量技术, 2021, 44(21):7-11.
- [10] 李铁峰, 李鸥, 李红波. 协作多点传输中板间串行高速通信的 FPGA 设计[J]. 微型机与应用, 2013 (18):23-25.
- [11] 刘云鹏. 基于 FPGA 的 RapidIO 总线接口设计,验证与实现[D]. 西安:西安电子科技大学, 2013.

(收稿日期:2022-04-13)

作者简介:

薛培(1984-),男,硕士,工程师,主要研究方向:嵌入式系统开发、FPGA 视频图像处理等。

官剑(1986-),男,硕士,工程师,主要研究方向:光纤通信、无线通信等。

邵春伟(1982-),男,硕士,工程师,主要研究方向:嵌入式系统、图像处理等。



扫码下载电子文档

版权声明

经作者授权，本论文版权和信息网络传播权归属于《电子技术应用》杂志，凡未经本刊书面同意任何机构、组织和个人不得擅自复印、汇编、翻译和进行信息网络传播。未经本刊书面同意，禁止一切互联网论文资源平台非法上传、收录本论文。

截至目前，本论文已经授权被中国期刊全文数据库（CNKI）、万方数据知识服务平台、中文科技期刊数据库（维普网）、DOAJ、美国《乌利希期刊指南》、JST 日本科技技术振兴机构数据库等数据库全文收录。

对于违反上述禁止行为并违法使用本论文的机构、组织和个人，本刊将采取一切必要法律行动来维护正当权益。

特此声明！

《电子技术应用》编辑部

中国电子信息产业集团有限公司第六研究所