DOI:10.3969/j.issn.1003-5060.2025.09.007

基于事务级协同仿真的 AXI4 交易器设计与验证

计润五[1], 黄正峰[1], 杨 滔[2], 孙 亮[2]
(1.合肥工业大学 微电子学院,安徽 合肥 230601; 2.无锡亚科鸿禹电子有限公司,江苏 无锡 214105)

摘 要:针对软硬件协同仿真过程中通信时间不同步和硬件仿真速度受限的难题,文章设计实现了一种高级可扩展接口(Advanced eXtensible Interface 4,AXI4)协议的交易器。基于事务级的软硬件协同仿真,结合硬件描述语言特性以及函数式编程的软件特性,使用 SpinalHDL 硬件描述语言设计一种 AXI4 接口协议的交易器,利用高级语言的敏捷特性对交易器生成流程进行高效处理,加速仿真验证阶段的编译流程。基于通用验证方法学(universal verification methodology,UVM)搭建验证平台对设计的交易器进行功能验证,结果显示交易器的代码覆盖率综合达到 99.17%,功能覆盖率达到 100%,符合交易器的功能要求。调用 AXI Interconnect IP 作为待测设计(design under test,DUT)在国产硬件仿真器-HyperSemu 上实现事务级传输,结果显示,相较于纯软件仿真加速比达 29.94,加快了协同仿真的验证速度,提升了仿真性能。

关键词:交易器;高级可扩展接口(AXI4);通用验证方法学(UVM);硬件仿真器;SpinalHDL 硬件描述语言
中图分类号:TN402 **文献标志码**:A **文章编号**:1003-5060(2025)09-1201-07

Design and verification of AXI4 transactor based on transaction-level co-emulation

JI Runwu[1], HUANG Zhengfeng[1], YANG Tao[2], SUN Liang[2]
(1. School of Microelectronics, Hefei University of Technology, Hefei 230601, China; 2. HyperSilicon Co., Ltd., Wuxi 214105, China)

Abstract: Aiming at the problems of unsynchronized communication time and limited hardware emulation speed in the process of software and hardware co-emulation, this paper designs and implements a transactor of Advanced eXtensible Interface 4 (AXI4) protocol. Based on transaction-level software and hardware co-emulation, combined with the characteristics of hardware description language (HDL) and software features of functional programming, a transactor of AXI4 protocol is designed using SpinalHDL, which uses the agile features of high-level language to efficiently process the transaction generation process, and accelerates the compilation process of the emulation verification stage. Based on universal verification methodology(UVM), a verification platform was built to conduct functional verification on the designed transactor, and the results showed that the code coverage of the transactor comprehensively reached 99.17%, and the functional coverage reached 100%, meeting the functional requirements of the transactor. Using AXI Interconnect IP as design under test(DUT), transaction-level transmission was achieved on the domestic hardware emulator HyperSemu. Compared with VCS, the pure software simulation, the acceleration ratio reached 29.94, accelerating the verification speed of co-emulation and improving emulation performance.

Key words: transactor; Advanced eXtensible Interface 4 (AXI4); universal verification methodology (UVM); emulator; SpinalHDL

收稿日期:2024-01-11;**修回日期**:2024-02-21
基金项目:国家自然科学基金资助项目(62274052;62374049);安徽省重点研究与开发计划资助项目(202304a05020003)
作者简介:计润五(1998—),男,安徽合肥人,合肥工业大学硕士生;
黄正峰(1978—),男,安徽无为人,博士,合肥工业大学教授,博士生导师,通信作者,E-mail:huangzhengfeng@139.com.

随着芯片规模和复杂度的急剧增长,大规模数字设计验证过程中存在综合时间长、代码迭代频率高的应用难题,给数字前端功能验证工作带来了极大的挑战,开发人员通常选择硬件仿真加速器实现大型设计的功能仿真加速验证。

软硬件协同仿真是一种将软件和硬件结合起来进行系统级仿真的有效方法。在软硬件协同仿真中,通常使用硬件描述语言(hardware design language,HDL)来设计硬件电路,使用C/C++来编写软件代码。为了使软件与硬件能够进行协同仿真,需要建立通信接口,使软件能够与硬件进行交互。在软硬件协同仿真中,交易器(transactor)是一个关键的组件。交易器负责在软件与硬件之间进行通信和数据传输,充当了软件与硬件之间的桥梁,将两者的交互进行协调和管理。

交易器的主要功能包括通信接口、数据转换、时序管理、错误处理。使用交易器,可以更好地模拟和验证软硬件系统的功能和性能,从而提高系统开发的效率和可靠性。交易器通常针对不同的通信协议和接口标准提供支持,目前支持高速外设部件互连(peripheral component interconnect express,PCIe)、通用异步收发传输器(universal asynchronous receiver/transmitter, UART)、Ethernet等多种协议,可以针对不同的协议需求设计对应的交易器。

综上所述,交易器设计是一项重要且具有挑战性的工作。本文结合硬件仿真平台的特点,在基于标准协同建模接口(standard co-emulation modeling interface,SCE-MI)标准的事务级协同仿真平台中使用SpinalHDL设计一种支持高级可扩展接口(Advanced eXtensible Interface 4,AXI4)协议的交易器组件,并进行通用验证方法学(universal verification methodology,UVM)验证和硬件仿真器测试。

1 事务级协同仿真

软硬件协同仿真系统中的通信方式主要分为基于周期精确级(cycle-based)仿真和基于事务级(transaction-based)仿真。事务级仿真可以实现软、硬件之间的高速通信,是软硬件协同仿真中最有效的仿真手段。事务级仿真原理如图1所示。

从图1可以看出,事务级仿真通过物理通道实现,与周期精确级仿真不同,其传输的是无时序事务。软件侧通过事务代理发送和接收无时序事务,硬件侧通过事务器将无时序事务转化为有时序的信号传送给待测设计(design under test,DUT),从而在传输过程中无需关注时序信号,加快了仿真速度[1]。

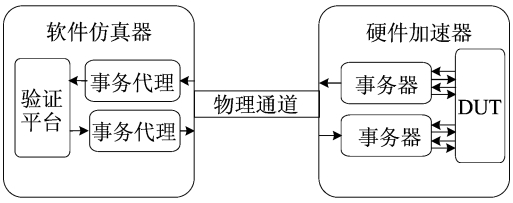

图1 事务级仿真原理图

2 交易器设计

2.1 AXI4总线

AXI4表示高级可扩展接口,它是由Arm定义的接口协议。AXI4接口主要分为AXI4(AXI4-Full)、AXI4-Lite、AXI4-Stream 3种类型。

本文针对AXI4-Full协议(下文称AXI4)进行研究和设计,该协议定义了5条通道,分别为读地址通道(read address channel,AR)、读数据通道(read data address channel,R)、写地址通道(write address channel,AW)、写数据通道(write data channel,W)、写响应通道(write response channel,B)[2]。AXI4协议如图2所示。

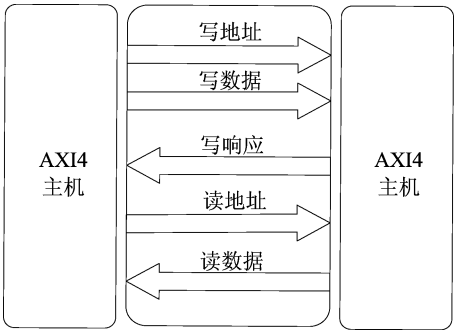

图2 AXI4协议

2.2 SpinalHDL语言

SpinalHDL是一种基于Scala编写的硬件描述语言,包含了Scala高级语言的所有特性,如面向对象编程、函数式编程、元编程,其本质上是Scala语言的类库,一种硬件描述语言生成器。相较于传统的Verilog、VHDL等硬件描述语言,SpinalHDL具有清晰的硬件概念,在RTL设计中具有大量高效方便的自定义基础组件(如Stream、Flow、State machine等)和高效的参数化

能力，可以提高设计的可重用性、可持续性等。用 SpinalHDL 语言进行寄存器传输级（register transfer level，RTL）设计的流程如图 3 所示。

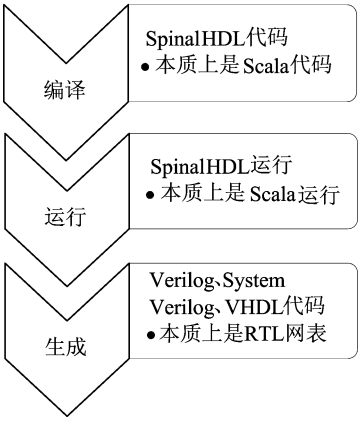

图 3　SpinalHDL 设计流程

2.3　AXI4 交易器设计

本文在基于 SCE-MI 标准的事务级协同仿真平台中设计了一种支持 AXI4 协议的交易器。SCE-MI 协议是用来搭建软硬件协同仿真交互的通信接口，是软硬件交互的桥梁[3]。

基于 SCE-MI 标准的软硬件协同仿真平台[4]如图 4 所示。

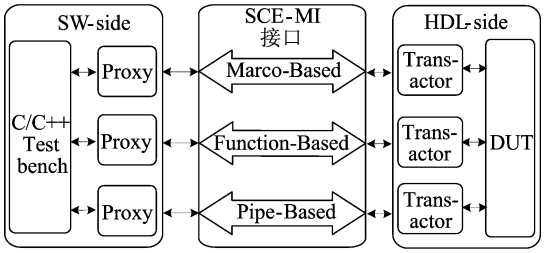

图 4　基于 SCE-MI 2.0 协同仿真平台

图 4 中，事务消息通过软件侧（SW-side）C/C++产生，由事务代理（Proxy）经过基于宏（Marco-Based）、函数（Function-Based）、管道（Pipe-Based）的 SCE-MI 接口传送至硬件侧（HDL-side）的交易器，经过交易器处理后将时序消息送入 DUT，进行测试。

2.3.1　设计框架

AXI4 交易器设计分为 AXI4 从机交易器和 AXI4 主机交易器 2 个部分，2 种交易器设计框架图如图 5 和图 6 所示。

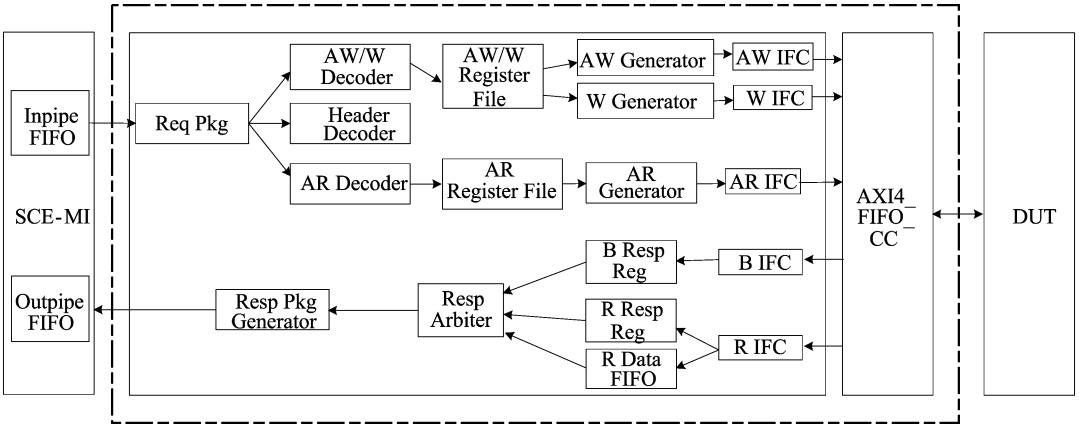

图 5　AXI4 从机交易器设计框架

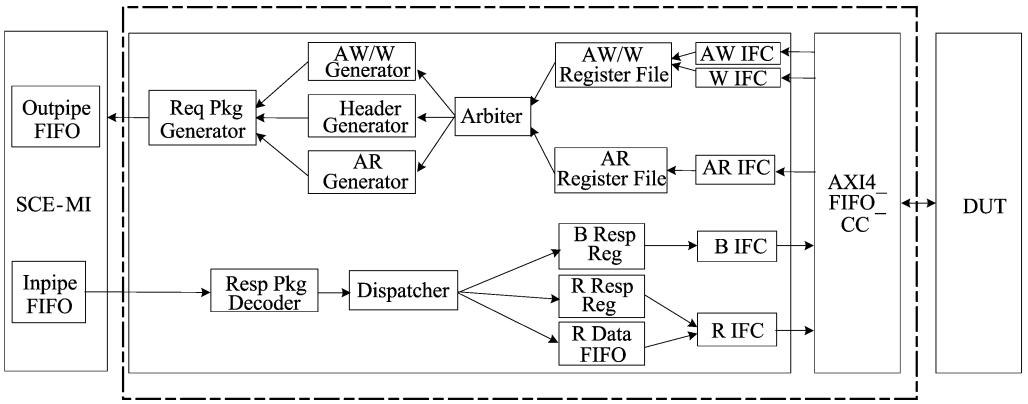

图 6　AXI4 主机交易器设计框架

主机(master)和从机(slave)是相对 DUT 的行为来界定的,当 DUT 作为主机发起命令,软件侧作为从机接受命令时,使用主机交易器来进行事务级传输,反之则使用从机交易器。

AXI4 从机交易器从 SCE-MI 总线出来的 Inpipe 先入先出(first in first out,FIFO)中读取源数据包,并经译码模块解析包(包括写地址/写数据译码(AW/W Decoder)、包头译码(Header Decoder)、读地址译码(AR Decoder))中信息后将其存入寄存器中,根据寄存器中信息生成写地址、写数据、读地址通道的 AXI4 读写时序接口(interface,IFC)并经过跨时钟域模块(AXI4_FIFO_CC)输出至 DUT。DUT 根据写地址、写数据、读地址通道的时序生成写响应、读数据通道时序。AXI4 从机交易器根据跨时钟域后的 DUT 响应时序接收响应信息和读数据,并将其存入寄存器和 FIFO 中,数据仲裁逻辑将响应信息和数据传给响应包生成模块(Resp Pkg Generator)组成数据包输出给 Outpipe FIFO,然后送入 SCE-MI 总线。

AXI4 主机交易器从跨时钟域模块接收到 DUT 传出的数据源,先将时序信息存入寄存器中,然后根据寄存器中写地址、写数据、读地址通道的 AXI4 读写时序按照 Round-Robin(轮询调度)算法仲裁。请求包生成模块(Rep Pkg Generator)根据写地址、写数据、读地址通道的数据组成不同数据包后送入 Outpipe FIFO,最后进入 SCE-MI 总线逻辑。软件侧根据 DUT 发出的请求,生成写响应、读数据通道的响应时序包由 Inpipe FIFO 送入交易器中,并由译码模块将响应数据解析后再由调度模块(Dispatcher)分配好响应信息和数据存入寄存器和 FIFO 中,然后生成写响应、读数据通道的时序后经过跨时钟域处理再输入 DUT。

本文设计中的从机交易器与主机交易器是完全可综合的,由 DUT 实例化使用。

2.3.2 关键处理

1) 跨 4 KiB 边界处理。AXI4 协议中规定 1 次突发(burst)传输不能跨越 4 KiB 边界,主要是为了避免 1 次传输访问 2 个从机导致信息的处理错误。实际验证过程中为了提高验证性能,在软件侧编写的测试激励中可以突破突发长度(burst length)小于 256 的限制,硬件侧接收到数据进行突发长度的多次拆分后传输,这样在 4 KiB 边界问题上的处理就显得尤为重要。本文设计主要通过设计状态机进行处理,原理是将 1 次突发传输拆分为多个突发来处理,如图 7 所示。

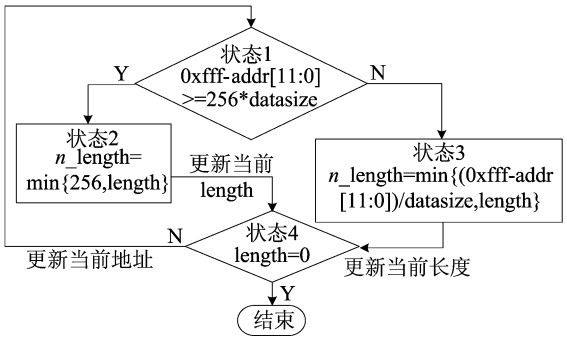

图 7 跨 4 KiB 处理流程

状态 1 首先判定当前首地址与 4 KiB 即 0xfff 的距离在最大 256 的突发长度情况下能否占满,若不够 256 的突发长度来传输则需要拆分多个突发传输,进入状态 2,否则进入状态 3。

状态 2 计算当前突发长度与最大突发长度 256 的最小值来进行下一次突发传输,同时更新突发长度大小,进入判断状态 4。

状态 3 计算当前地址距离 4 KiB 边界的地址范围需要传输的突发长度与当前突发长度的最小值来进行下一次突发传输,同时更新突发长度大小,进入判断状态 4。

状态 4 判断突发长度是否为 0,若是则传输结束,否则进入状态 1 继续拆分突发长度。

2) SpinalHDL 编译流程处理。在 AXI4 交易器设计过程中使用 SpinalHDL 进行设计,考虑到集成在 EDA 工具中的敏捷性,提高参数化配置和流程规范性。本文利用高级语言的特性,设计一套实例化交易器的设计流程,如图 8 所示。

首先在 json 文件(一种按照 JavaScript 对象语法的数据格式)中定义名称、类型、突发长度、数据位宽、地址位宽、用户数据位宽等参数,用来配置生成 Verilog;jar 包(java 的文件包格式)是交易器 SpinalHDL 的源码打包而成,包括 slave 和 master 2 个源设计文件。由于代码设计充分利用 Spinal 参数配置的优势,通过运行 jar 包程序并读取 json 文件中的配置参数,传输参数进入设计源码会生成参数对应的 Verilog 代码,如 Axi4_slave_32.v 表示位宽是 32 位的从机交易器。同样也支持 VHDL、System Verilog 代码。最后在 DUT 代码中例化生成 Verilog 代码,即可完成交易器的实例化应用。

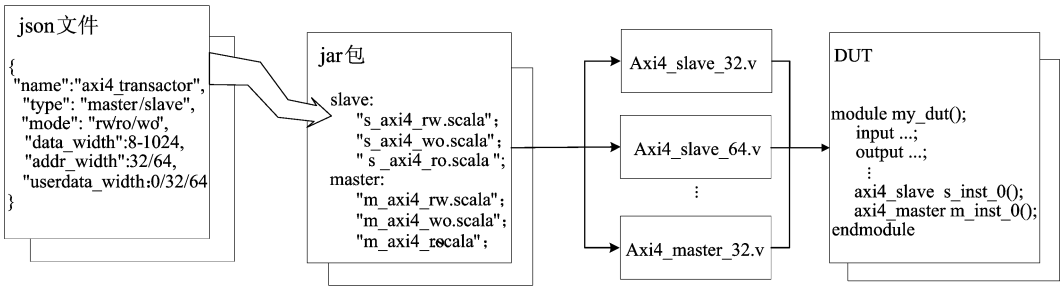

图 8　SpinalHDL 编译流程处理

3　实验仿真和分析

3.1　功能验证

UVM 是基于 System Verilog 的高级验证方法学,继承了 System Verilog 验证语言面向对象的思想,在搭建验证环境中具有高效和便捷的特性[5-10]。本文设计在功能验证阶段采用基于 UVM 的验证方法学来搭建 AXI4 从机交易器和 AXI4 主机交易器的验证平台,验证框架如图 9 和图 10 所示。

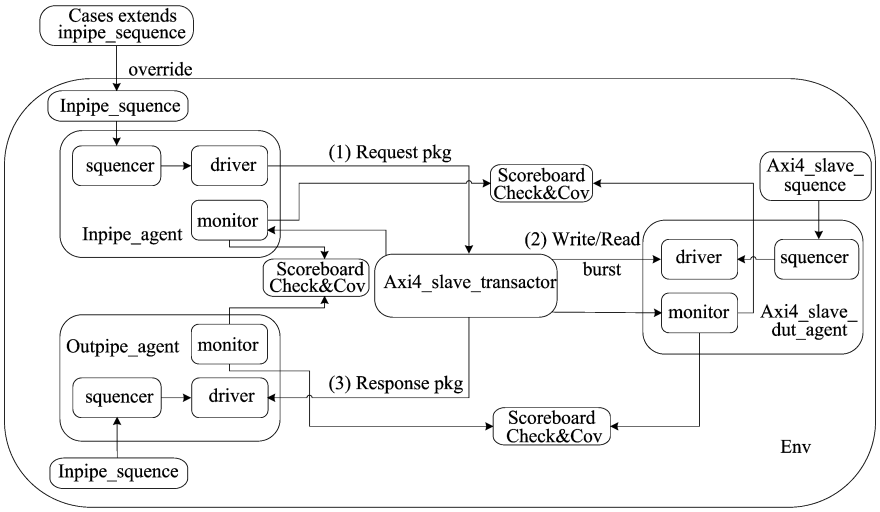

图 9　AXI4 从机交易器验证平台

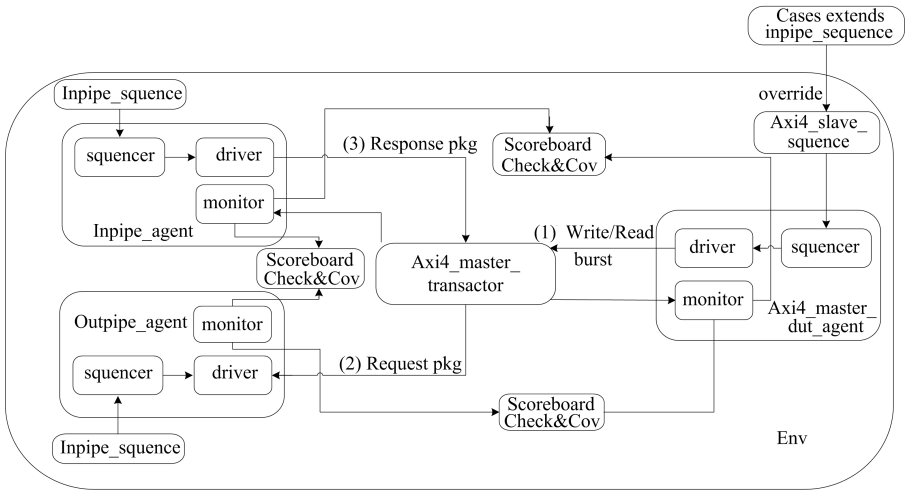

图 10　AXI4 主机交易器验证平台

图 9 和图 10 中的验证平台由代理(agent)、驱动器(driver)、激励(sequencer)、监测器(monitor)、计分板(scoreboard)、环境类(Env)等组成。其中:Inpipe_agent 模拟 Inpipe FIFO 的功能,在

从机交易器模块中负责生成请求(Request)数据包,包含读写控制信息和数据信息,在主机交易器模块中负责生成响应(Response)包,包含用于回应 DUT 的响应信息;Outpipe_agent 模拟 Outpipe FIFO 的功能,在从机交易器模块中负责接收 DUT 生成并经交易器处理后的响应包,包含来自 DUT 经交易器处理后的请求之后发出的响应信息,在主机交易器模块中负责接收 DUT 生成并经交易器处理后的请求包,包含 DUT 发出的控制信息和数据信息;Axi4_master_dut_agent 模拟 DUT 的功能,Axi4_master_dut_agent 为本文设计模块。

最终在该测试平台(test bench,TB)对 AXI4 交易器进行多次完全随机测试和定向测试。测试环境在 hs_clock=50 MHz、dut_clock=25 MHz 两时钟同频下完成;地址位宽范围可选 32 位或 64 位;数据位宽动态范围为 8～1 024 位;突发模式可选递增传输(Increment)、固定传输(Fix)、回环传输(Wrap)模式;突发长度设置 0～256 随机值和大于 256 随机值 2 种情况。测试覆盖范围包括协议规定的特性以及突发长度大于 256 时跨 4 KiB边界处理等。

测试结果符合交易器设计要求。通过 VCS+DVE查看覆盖率报告可知,AXI4 交易器代码覆盖率综合达到 99.17%。代码覆盖率收集结果如图 11 所示。从图 11 可以看出,行覆盖率、翻转覆盖率、状态机覆盖率、条件覆盖率、分支覆盖率分别为 99.70%、98.42%、100.00%、97.72%、100.00%。

功能覆盖率收集结果如图 12 所示。从图 12 可以看出,覆盖率点中含有 2 个交叉覆盖点 ab、ce 分别负责观测突发模式和突发大小以及突发长度和地址,结果显示功能覆盖率综合达到 100%,满足验证需求。

Name	Score	Line	Toggle	FSM	Condition	Branch
axi4_transactor	99.17%	99.70%	98.42%	100.00%	97.72%	100.00%
s_axi4ctr_rw	99.02%	99.60%	98.68%	100.00%	96.81%	100.00%
datain_buf	100.00%	100.00%	100.00%		100.00%	100.00%
dataout_buf	99.15%	100.00%	99.73%		96.88%	100.00%
rd_data_buf	99.19%	100.00%	100.00%		96.77%	100.00%
wr_req_buf	100.00%	100.00%	100.00%		100.00%	100.00%
axi4CC	99.48%	100.00%	97.90%		100.00%	100.00%
io_input_ar_queue	100.00%	100.00%	100.00%		100.00%	100.00%
io_input_aw_queue	99.79%	100.00%	99.17%		100.00%	100.00%
io_input_w_queue	99.96%	100.00%	99.83%		100.00%	100.00%
io_output_b_queue	99.74%	100.00%	98.95%		100.00%	100.00%
io_output_r_queue	99.59%	100.00%	98.36%		100.00%	100.00%

图 11 代码覆盖率收集结果

Cover Group Item	Score	Goal	Weight	At Least	Per Instance
$unit::seq_16::iptr_cov					
$unit::seq_16::iptr_cov	100.00%	100%	1	1	
ab	100.00%	100%	1	1	
addr	100.00%	100%	1	1	
blength	100.00%	100%	1	1	
burst_id	100.00%	100%	1	1	
burst_mode	100.00%	100%	1	1	
burst_size	100.00%	100%	1	1	
ce	100.00%	100%	1	1	

图 12 功能覆盖率收集结果

3.2 硬件仿真器验证

3.2.1 验证环境搭建

为了充分验证 AXI4 交易器的可行性,保证 DUT 的准确性,选择 Xilinx AXI4 Interconnect IP 作为 DUT 进行验证。设置 IP 为 1 个主机接口和 1 个从机接口,即需要实例化 1 个 AXI4 主机交易器和 1 个 AXI4 从机交易器,这里设置数据位宽 64 bit,地址位宽 32 bit,读写通道为 Read/Write。测试激励用 C++代码通过 SCEMI 软硬件协同平台实现。测试框架如图 13 所示。

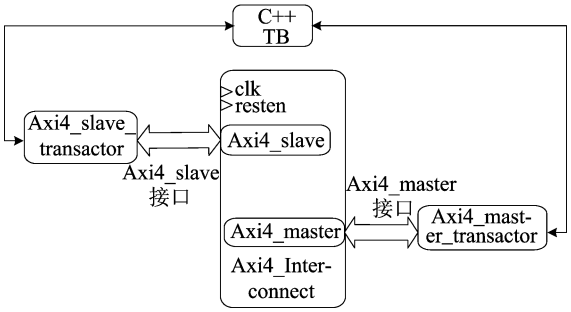

图 13 AXI4 Interconnect 板级测试框架

调用的 AXI4 Interconnect IP 有 Axi4_slave 接口和 Axi4_master 接口，连接本文设计的 Axi4_slave_transactor 和 Axi4_master_transactor，通过 TB 侧的 C++代码生成测试用例，经过交易器传输至 DUT 侧完成测试。

3.2.2　验证结果分析

本文采用无锡亚科鸿禹公司的国产硬件仿真器产品 HyperSemu V1.0 实现交易器的应用。在硬件端采用 Xilinx 的 Virtex UltraScale+系列产品 XCVU13P 的 FPGA 板卡。软件端通过硬件仿真器配套 EDA 工具将 C++测试代码传输至硬件侧。测试用例包括软件侧向从机交易器端发送 1 次写 1 次读，再通过主机交易器端传输至软件侧，形成闭环；采用突发传输递增模式，突发长度为 1 500；发起共 100 次这样的传输，充分验证跨 4 KiB 的情况；并且仿真时间上相较于 VCS 纯软件仿真提速达 29.94 倍，大大提升了仿真验证的速度。

硬件仿真器与 VCS 的仿真时间开销对比见表 1 所列，可以看出，实际硬件仿真时间与 VCS 纯软件仿真时间相比加速明显。

表 1　硬件仿真器与 VCS 仿真时间开销对比

DUT 类型	时间/s		加速比
	VCS	硬件仿真器	
AXI4 Interconnect	255.256 63	8.526 18	29.94

根据输出的 FPGA 资源利用报告显示，硬件部分的各资源占比见表 2 所列，可以看出，硬件仿真中最高工作频率达到了 1.14 MHz。

表 2　FPGA 各资源占用

资源	总资源数	占用数	占用率/%	交易器占用率/%
LUT	1 728 000	41 374	2.39	0.15
LUTRAM	791 040	4 709	0.60	0.02
FF	3 456 000	45 446	1.31	0.05
BRAM	2 688	915	3.40	0.30
DSP	12 288	12	0.10	0.02

4　结　论

本文基于事务级协同仿真平台，运用 SpinalHDL 语言设计了一种 AXI4 协议的交易器，详细介绍了设计框架以及关键的跨 4 KiB 传输处理和 Verilog 代码生成的流程处理，加速了仿真验证阶段的编译流程；并在 UVM 验证平台上对交易器进行了充分验证，结果显示完全符合交易器的功能要求。在板级验证阶段，通过搭建 AXI4 Interconnect DUT 测试环境，在国产硬件仿真器工具上实现事务级协同仿真通信，验证本文设计的交易器的可行性，结果显示，相较于纯软件仿真加速比达到 29.94，最高工作频率达到 1.14 MHz，实现了硬件仿真加速。

[参　考　文　献]

[1] 汪浩. 基于事务级的软硬件协同仿真验证系统的设计与实现[D]. 西安：西安电子科技大学，2021.

[2] 谭磊. 基于软硬件协同仿真的 AXI4 VIP 的设计与验证[D]. 西安：西安电子科技大学，2022.

[3] 刘凯，李平，廖永波. 基于 SCE-MI 标准的事务级 SoC 协同仿效平台设计[J]. 微电子学，2007，37(5)：624-627.

[4] SAVLA J. Getting started on co-emulation：transition your design and testbench to an emulator[C]//2018 19th International Workshop on Microprocessor and SOC Test and Verification (MTV). [S. l.]：IEEE，2018：46-51.

[5] 陈琳娜，孟建熠，林志涛. 面向串行总线的层次化 UVM 验证平台设计[J]. 传感器与微系统，2018，37(9)：84-89.

[6] SINHA ROY S，BASU A，SANKAR DAS T，et al. Implementation of image copyright protection tool using hardware-software co-simulation[J]. Multimedia Tools and Applications，2021，80(5)：1-15.

[7] NOAMI A，ALAHDAL A，KUMAR B P，et al. High-speed data transactions for memory controller based on AXI4 interface protocol SoC[C]//2021 International Conference on Advances in Electrical，Computing，Communication and Sustainable Technologies (ICAECT). [S. l.]：IEEE，2021：1-7.

[8] CHEN Y，WANG S，LI Z，et al. Reliable physical-layer cross-technology communication with emulation error correction[J]. IEEE ACM Transactions on Networking，2020，28(2)：612-624.

[9] DUAN T，DINAVAHI V. Adaptive time-stepping universal line and machine models for real time and faster-than-real-time hardware emulation[J]. IEEE Transactions on Industrial Electronics，2019，67(8)：6173-6182.

[10] KHAMIS M，EL-ASHRY S，SHALABY A，et al. A configurable RISC-V for NoC-based MPSoCs：a framework for hardware emulation [C]//2018 11th International Workshop on Network on Chip Architectures (NoCArc). [S. l.]：IEEE，2018：1-6.

（责任编辑　胡亚敏）