Journal of System Simulation

Volume 32 | Issue 5 Article 5

5-15-2020

Design and Implementation of Reconfigurable Video Array Processor Test Platform

Jiang Lin
1. College of Electronic Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, China; ;

Feilong He
2. College of Computer Science, Xi'an University of Posts and Telecommunications, Xi'an 710121, China;

Shan Rui
1. College of Electronic Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, China; ;

Wang Shuai
1. College of Electronic Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, China; ;

See next page for additional authors

Design and Implementation of Reconfigurable Video Array Processor Test Platform

Abstract

Abstract: Aiming at the design requirements of the reconfigurable video array processor and the problem of traditional method testing the video codec system with slow speed, low precision and poor observability. *A Qt-based user interface is developed, and a hardware-software co-testing platform based on FPGA is designed and implemented. The platform realizes the data transmission and image reproduction based on the software simulation on the PC side, and the parallel mapping of the video encoding and decoding algorithms based on the reconfigurable video array processor on the FPGA side. The experiment results show that the data can be transmitted correctly between FPGA and PC when the working frequency of the platform is 100 MHz, and the replacement demand of the different test cases can be satisfied when the algorithm is tested, and the experiment has a good observability.*

Keywords

testing platform, hardware and software collaboration, user interface, video codec, reconfigurable video array processor

Authors

Jiang Lin, Feilong He, Shan Rui, Wang Shuai, Haoyue Wu, and Wu Xin

Recommended Citation

Jiang Lin, He Feilong, Shan Rui, Wang Shuai, Wu Haoyue, Wu Xin. Design and Implementation of Reconfigurable Video Array Processor Test Platform[J]. Journal of System Simulation, 2020, 32(5): 792-800.

第 32 卷第 5 期 系统仿真学报© Vol. 32 No. 5
2020 年 5 月 Journal of System Simulation May, 2020

可重构视频阵列处理器测试平台设计与实现

蒋林[1]，贺飞龙[2]，山蕊[1]，王帅[1]，吴皓月[1]，武鑫[1]
(1. 西安邮电大学电子工程学院，陕西 西安 710121；2. 西安邮电大学计算机学院，陕西 西安 710121)

摘要：针对可重构视频阵列处理器的设计要求及传统测试方法测试视频编解码系统时速度慢、精度低和可观测性不强的问题。*开发了基于 Qt 的用户界面，设计实现了以现场可编程门阵列(Field programmable gate-array，FPGA)为核心的软硬件协同测试平台。在 PC 端实现以软件仿真为基础的数据传输与图像重现，在 FPGA 端实现以可重构视频阵列处理器为基础的视频编解码算法并行映射。实验结果表明，在工作频率为 100 MHz 时，FPGA 与 PC 之间可正确传输数据并满足算法测试时不同测试用例的更换需求，具有较好的可观测性。*

关键词：测试平台；软硬件协同；用户界面；视频编解码；可重构视频阵列处理器
中图分类号：TP391.9 文献标识码：A 文章编号：1004-731X (2020) 05-0792-09
DOI: 10.16182/j.issn1004731x.joss.18-0542

Design and Implementation of Reconfigurable Video Array Processor Test Platform

Jiang Lin[1], *He Feilong*[2], *Shan Rui*[1], *Wang Shuai*[1], *Wu Haoyue*[1], *Wu Xin*[1]
(1. College of Electronic Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710121, China;
2. College of Computer Science, Xi'an University of Posts and Telecommunications, Xi'an 710121, China)

Abstract: Aiming at the design requirements of the reconfigurable video array processor and the problem of traditional method testing the video codec system with slow speed, low precision and poor observability. *A Qt-based user interface is developed, and a hardware-software co-testing platform based on FPGA is designed and implemented. The platform realizes the data transmission and image reproduction based on the software simulation on the PC side, and the parallel mapping of the video encoding and decoding algorithms based on the reconfigurable video array processor on the FPGA side. The experiment results show that the data can be transmitted correctly between FPGA and PC when the working frequency of the platform is 100 MHz, and the replacement demand of the different test cases can be satisfied when the algorithm is tested, and the experiment has a good observability.*

Keywords: testing platform; hardware and software collaboration; user interface; video codec; reconfigurable video array processor

引言

随着集成电路工艺的不断进步，芯片的集成度不断提高，芯片设计的复杂度不断提升，为保证投片成功率，验证测试起着举足轻重的作用[1]。特别是视频编解码高复杂度系统[2]，通常必须支持各种系统参数和多种操作模式。为了满足用户高吞吐量，高数据带宽和实时应用等体验需求，视频标准变化很快，需要大量的相关测试用例进行功能测试[3]，并且需要很长的测试时间，几乎占

收稿日期：2018-08-07 修回日期：2018-11-26；
基金项目：国家自然科学基金(61772417, 61602377, 61634004)，陕西省科技统筹创新工程(2016KTZDGY02-04-02)，陕西省重点研发计划(2017GY-060)；
作者简介：蒋林(1970-)，男，陕西西安，博士，教授，硕导，研究方向为集成电路设计。

开发周期的 80%[4]，因此，如何在紧张的开发时间内保证设计的可靠性一直是设计者关心的问题。

为解决这些问题，国内外学者提出了各种测试方法，但这些测试方法中，软件仿真技术虽然仿真结果准确且容易发现设计中的毛刺电路，但是当算法的复杂度较高时，会导致仿真速度较慢，不适合规模较大的设计[5]；形式验证技术虽然在测试时覆盖率可达 100%，但是没有考虑时序问题[6]；静态时序分析技术可根据设计的电路结构来测试设计是否满足时序要求，并且不需要测试向量，但是它只适于电路的时序测试，没法保证电路功能的正确性[7]；FPGA 原型测试技术相比于软件仿真技术，速度要高出几个数量级，并且具有精度高及允许设计人员在平台上进行实际操作和可重复编程以及并行处理等优点，但是可观测性不足，不如软件仿真灵活[8]。

软硬件协同测试兼顾了软件仿真的灵活性、可观测性和 FPGA 原型测试的速度快与精度高的特点，为系统原型验证提供了新的测试方法。相比于传统测试方法，可以从模块级到系统级进行测试，可避免由于某一模块出错而推倒整个方案重新设计造成的损失。然而，对于视频编解码系统的测试，文献[9]提出具体性能指标的测试方法和原理，但重点针对编解码延时和图像参数分析，不能就编解码系统底层进行测试，无法统计硬件性能数据；而文献[10]采用自顶向下的方法设计了基准测试程序，虽然可对视频编解码系统底层硬件进行性能参数的统计，但可观测性不强，测试结果的全面性和可靠性有待提升；文献[11]设计并实现了一个视频编解码器性能评估系统，该系统可用于自动对视频编码器进行详细分析，但是该系统的输入参数较少，无法评估其他一些参数且目前还不稳定。

因此，针对可重构视频阵列处理器的设计要求及传统测试方法测试视频编解码系统时速度慢、精度低和可观测性不强的问题，采用软硬件协同设计的方法，设计实现了可重构视频阵列处理器的可视化测试平台。在 PC 端基于 Qt 技术采用多文档界面(MDI)、信号/槽机制和 QProcess 类完成用户界面的设计，实现以软件仿真为基础的数据传输与图像重现；通过对用户接口模块的设计在 FPGA 端实现以可重构视频阵列处理器为基础的视频编解码算法并行映射。为测试所设计平台及可重构视频阵列处理器的整体性能，分别选取标准的视频测试序列和项目组优化的运动补偿算法进行测试。结果表明，与软件仿真相比，加速比为 25.81 且精度明显提升，与 FPGA 原型测试相比，具有较好的可观测性，且灵活性更强，满足不同测试序列的更换需求。

1 测试平台总体设计方案

1.1 需求分析

可重构视频阵列处理器如图 1 所示。核心部分由 8×8 个处理元簇(PEG)组成，其中每个处理元簇包含 4×4 个轻核处理元(PE)，PEG 内部 16 个 PE 采用邻接互连共享寄存器的方式进行拓扑连接，可以对自身的寄存器、存储器进行访问，也可以对相邻处理元的寄存器进行访问。为解决邻接互连在远程数据访问时效率不高的问题，该结构通过网络适配器将每一个 PEG 与路由网络相连，以实现远程数据的访存。为有效提升处理器的性能，设计 H-Tree 型层次化配置网络，通过配置指令来实现多种模式的切换，以达到合理分配资源和避免重新编译程序的目的。

在对高效视频编码(High Efficiency Video Coding，HEVC)可重构实现的过程中需要在各个 PEG 之间进行任务分配与调度，同时由于视频算法之间的数据并行性很高，在测试时需要实时地输入几百万像素的原始视频数据[12]，在数据量庞大的情况下采用软件仿真、形式测试、时序分析和 FPGA 原型方法均不能满足在速度与精度方面的需求，并且可观测性不强。因此，需要设计兼顾软件仿真的灵活性、可观测性和 FPGA 原型测试速度快与精度高的测试平台。

https://dc-china-simulation.researchcommons.org/journal/vol32/iss5/5
DOI: 10.16182/j.issn1004731x.joss.18-0542

第 32 卷第 5 期 系统仿真学报 Vol. 32 No. 5
2020 年 5 月 Journal of System Simulation May, 2020

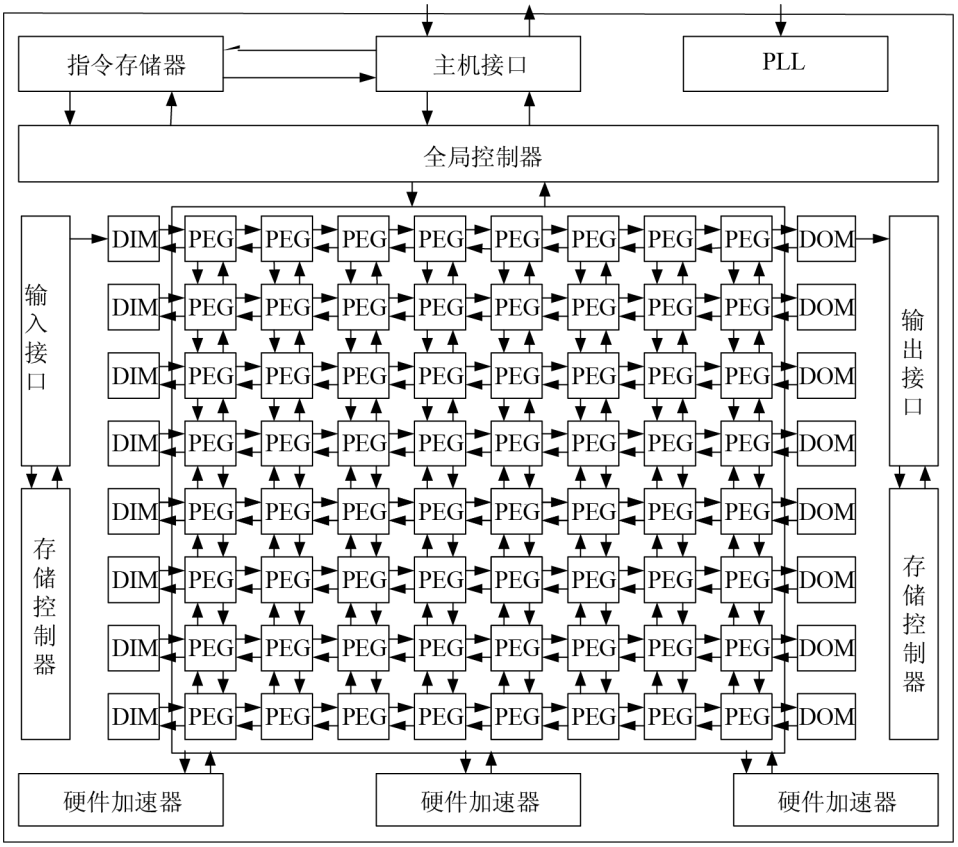


图 1　可重构视频阵列处理器结构图
Fig. 1　Array processor structure

1.2 平台框架设计

测试平台总体设计如图 2 所示。整个系统主要由上位机和 BEE4 开发平台组成，上位机通过以太网将相关数据和设计文件下发到 BEE4 平台；再通过 BEE4 显示终端激活设计；随后可对设计进行相应测试。测试平台的软件部分主要由用户界面和处理结果显示模块组成，其中用户界面主要向阵列处理器提供配置数据和配置指令；硬件部分主要包括存储模块、用户接口设计模块、可重构视频阵列处理器模块，其中利用开发平台支持使用 DDR3 DRAM 和 DDR3 FIFO 允许系统时钟在任何独立于存储器时钟的任意频率下运行的特性，通过 DDR3 FIFO 将 DDR3 DRAM 和用户接口设计部分相连接，构成存储部分；用户接口模块主要完成用户逻辑和系统的信息交互；可重构视频阵列处理器模块在用户界面完成数据和指令配置后，完成相应运算处理。

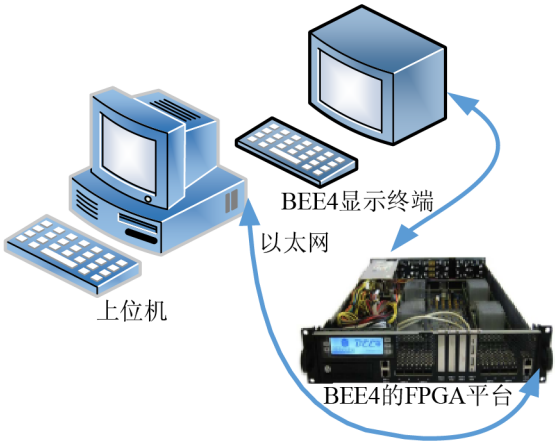


图 2　系统总体设计图
Fig. 2　Overall system design

2　测试平台设计

2.1 基于 QT 的用户界面设计

针对可重构视频阵列处理器并行处理数据的特性，本界面基于 Qt 技术采用多文档界面(MDI)

进行设计，即在一个主窗口中可对多个文档窗口进行管理。当用户单击某一窗口时，该窗口便被激活并且通过选择标记标识该窗口。该用户界面可很方便的将汇编指令翻译为可重构视频阵列处理器能够执行的二进制指令，并可直接点击图标 M 触发进程间通信，调用 Modelsim 对可重构视频阵列处理器进行仿真，整体结构如图 3 所示。

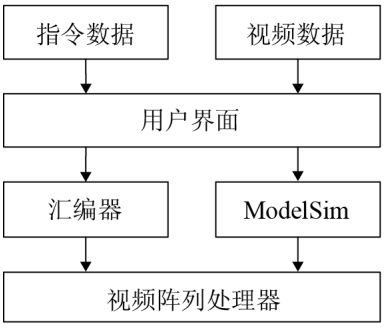


图 3　整体结构图
Fig. 3　Overall structure

仿真调试界面如图 4 所示，主要功能块的设计运用信号和槽机制。该机制可使编程人员将没有相关性的对象绑定在一起，实现对象之间的通信。汇编器的翻译功能可通过自定义槽函数和预定义信号连接来实现，汇编指令遍历过程如图 5 所示。调用 ModelSim 时运用 QProcess 类完成进程间的交互，允许程序执行外部程序并与其交互，当用户有需求时，点击相应菜单就可启用 ModelSim。

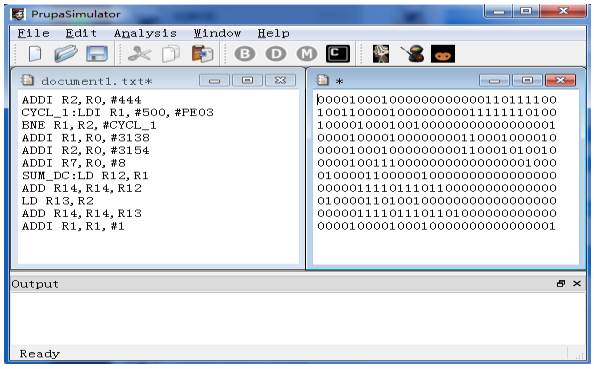


图 4　仿真调试界面
Fig. 4　Simulation debugging interface

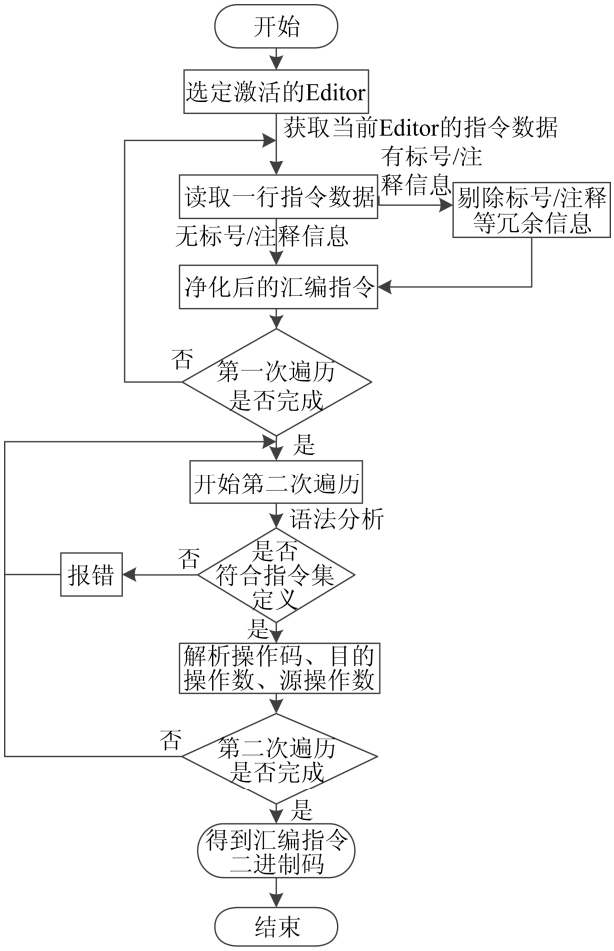


图 5　指令遍历过程
Fig. 5　Instruction traversal process

2.2 数据通路设计

系统运行时首先通过用户界面的汇编功能把对应算法的汇编指令翻译为可重构视频阵列处理器能够执行的二进制指令，接着将对应的二进制指令映射到所选 PE 上，完成阵列处理器的功能配置。再用 PC 端 MATLAB 将测试序列分离保存成 txt 文件，用 BEE4 自带的 b4d_bram_read 功能将数据通过远程连接发送到 Shared_BRAM 中，然后通过用户接口设计传输到 DDR3 FIFO，由 DDR3 FIFO 传送到片外 DDR3 中。存储完毕后，阵列处理器通过用户接口设计开始读取 DDR3 中的测试序列。接着，阵列处理器将处理完的数据通过用户接口设计存入 Shared_BRAM 中，最后 MATLAB 通过 b4d_bram_read 功能将处理完的数据恢复成可视化的图像。

https://dc-china-simulation.researchcommons.org/journal/vol32/iss5/5
DOI: 10.16182/j.issn1004731x.joss.18-0542

第 32 卷第 5 期 系统仿真学报 Vol. 32 No. 5
2020 年 5 月 Journal of System Simulation May, 2020

2.3 用户接口模块设计

用户接口模块分为数据模块和请求模块两个部分，整体框图及连接如图 6 所示。

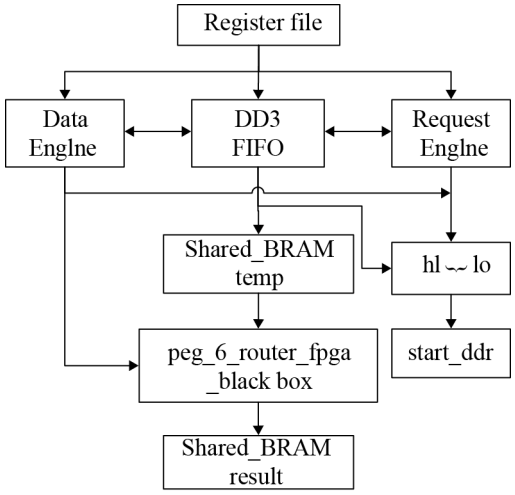


图 6 用户接口模块设计总体框图
Fig. 6 Overall block diagram of user interface module design

整体设计需要使用 6 个 Shared_BRAM 参与，其中 4 个用于向 FIFO 写入数据，temp 用于缓存 FIFO 出栈的数据，阵列处理器通过 DIM 读取缓存中的数据后通过 DOM 写入 result 中。BPS BEE4 模块用于指定系统时钟与目标器件。ChipScope Configuration 模块的放置可以在设计中使用 ChipScope 探针来观测信号。以上使用寄存器模块的信号可通过 MATLAB 使用 b4d 功能远程写入数据控制。

InitDone 信号有效，DDR3 可以正常读写。ctrl 寄存器写入 1，通过位宽匹配模块将 32 位寄存器改为 1 位，寄存两拍后与 DDR3 FIFO 的 Reset 信号连接。start_en 寄存器写入 1 时控制电路地址开始自增，通过位宽匹配模块将 32 位寄存器改为 1 位，寄存两拍后与数据模块和请求模块的 start 端口连接。base_addr 写入 256 作为 DDR3 物理内存的基地址。RNW 写入 0 进入写数据状态，通过位宽匹配模块将 32 位寄存器改为 1 位，与数据模块的 RNW 相连，表示读或者写。数据模块和请求模块的 busy 信号同时为 0 时表示可以进行读数据，此信号与 EccError、InitDone 拼接为一个三位的数据寄存器两拍后与输出寄存器相连。

待 DDR3 加载数据完毕后，开始进入读数据状态。RdPop 信号有效，RdData 开始输出数据给阵列处理器 IPCore。阵列处理通过 PEG00 中 PE00 的 R10 号寄存器与 DIM 相连进行读取缓存中的数据。读取数据通过在 PE00 的指令存储器中存取相关读取数据指令来控制，设计为上电复位后，一只读取 temp 中 60000 号地址的数据，检测到 60000 号地址写入 256 时握手成功阵列处理器开始工作。读取数据时需给出使能和地址两个数据，使能为低时进入读数据状态，此时将阵列处理器输出的 rd_addrDIM 地址中的数据写入阵列处理器。待阵列处理器工作完毕后通过 DOM 将数据使能和地址输出到 result 模块中，待 MATLAB 读取数据恢复成可视化的图像。

2.3.1 数据模块设计

数据模块设计总体框图如图 7 所示。

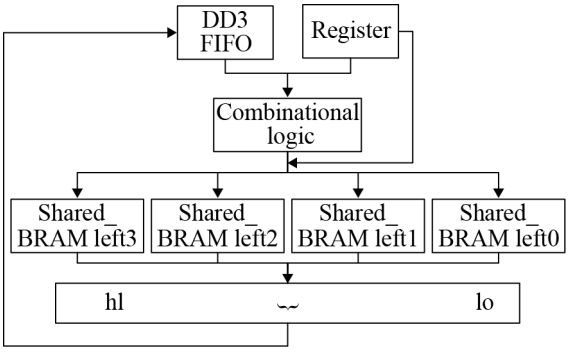


图 7 数据模块设计框图
Fig. 7 Data module design block diagram

由于 DDR3 FIFO 数据位宽为 128，所以写入 FIFO 的数据通过 4 个数据位宽为 32 地址位宽为 16 的 Shared BRAM 拼接而成，命名为 left0、left1、left2、left3。拼接后的数据寄存三拍后输出给 DDR3 FIFO 的 WrData。start 为高时电路开始工作，bram_addr 计数器进入初始化状态。RNW 为低电平时且电路不在 busy 状态时为写数据，寄存五拍之后 WrPush 信号输出给 DDR3 FIFO。RNW 为高

电平、电路非 busy 状态且 RdEmpty 信号为低电平时为读数据，为 RdPop 引脚与 DDR3 FIFO 相连。读信号或写信号有效时计数器开始工作，计数器作为 Shared_BRAM 的地址寄存一拍后分别与 left0、left1、left2、left3 的 addr 端口连接。由于 Shared_BRAM 的地址位宽为 16 位所以设置当计数到 32 767 时电路设置为 busy 状态，电路暂停工作。RNW 与 start 信号由 MATLAB 通过远程连接写入寄存器控制，RdEmpty 为 DDR3 FIFO 反馈信号，表示读取 FIFO 为空。

2.3.2 请求模块设计

请求模块用于和 DDR3 的内存地址交互，具体实现方式如图 8 所示。

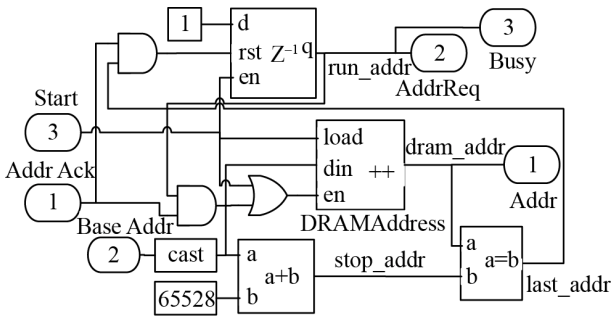


图 8 请求模块设计框图
Fig. 8 Request module design block diagram

Start=1 表示电路开始正常工作。此时 AddrReq 信号有效(与 DDR3 FIFO 上的 AddrReq 信号连接)，将 Addr 和 AddrRW 上的 DDR3 中的值发送到 DDR3 FIFO 中。同时 DRAMAdress 模块开始计数，此计数器的值与 DDR3 FIFO 的 Addr 信号相连，用于 DDR3 的物理 DRAM 地址。DRAM 地址是 DIMM 上的物理地址，每个地址位置都包含 64 位。本设计中的存储器控制器被设置为 8 的突发长度，因此每次地址都会增加此量。所以当基地址 256 与 65528 的和与 DRAM 的地址相等且 AddrAck 信号有效时电路进入 busy 状态。Start 与 base_addr 信号由 matlab 通过远程连接写入寄存器控制。AddrAck 为 DDR3 FIFO 反馈信号表示当前内存事务请求已被接受到命令 FIFO 中。

3 仿真测试

3.1 Qt 用户界面测试

图 9 所示为测试结果。图 9 中左侧为待翻译的源指令文件，右侧为翻译完成的可重构视频阵列处理器能够执行的二进制指令。以第一条指令为例，指令集中 ADDI 编码为 000010，R2 编码为 0010，R0 编码为 0000，立即数 111 用 16 位二进制数表示为 0000000001101111，故第一条指令翻译结果为 00001000100000000000000001101111，与图 9 中右侧第一条翻译结果相同，翻译结果正确。相应地可验证其他的指令翻译结果正确。

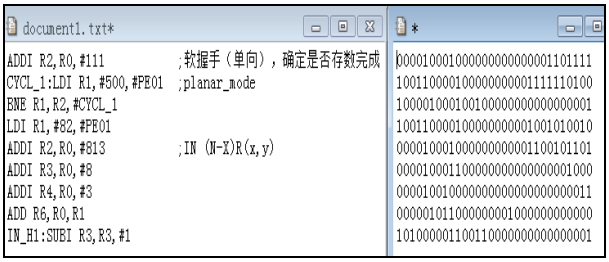


图 9 指令测试结果
Fig. 9 Instruction test results

3.2 簇间通信机制测试

为测试所设计的基于 FPGA 的可重构视频阵列处理器测试平台的工作性能，利用实验室现有的 BeeCube 公司的 BEE4 搭建了硬件仿真平台，图 10 所示为该测试平台实物图。

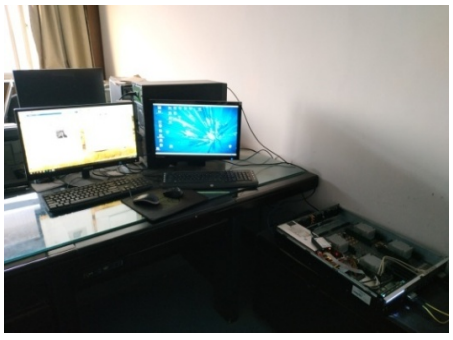


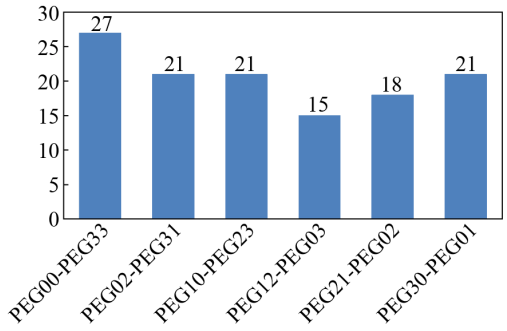
图 10 原型测试环境
Fig. 10 Prototype testing environment

基于上述平台，通过执行 ST 指令来测试可重构视频阵列处理器簇间通信机制的正确性。执行

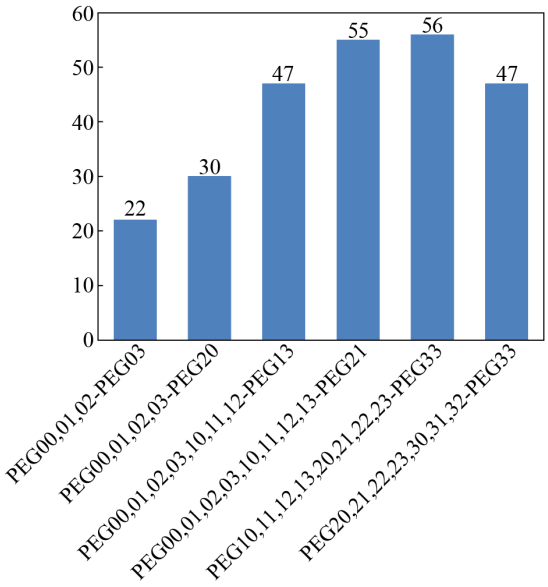
https://dc-china-simulation.researchcommons.org/journal/vol32/iss5/5
DOI: 10.16182/j.issn1004731x.joss.18-0542

第 32 卷第 5 期 系统仿真学报 Vol. 32 No. 5
2020 年 5 月 Journal of System Simulation May, 2020

ST 指令给适配器输入激励，经路由网络传输到目的 PEG，通过 ChipScope 观测到数据传输成功。如图 11(a)所示，为无竞争情况下部分访存延迟统计结果；图 11(b)为竞争情况下即多个 PEG 访问同一个 PEG 部分访存延迟统计结果。在此通信机制中，路由器和适配器每一跳共需 4 个周期，而 Intel 80 核处理器 8×10 的 2D Mesh 网络[13]需要 6 个周期，通信延迟降低了 1/3，通信效率得以提升。



(a) 无竞争访存延迟统计图

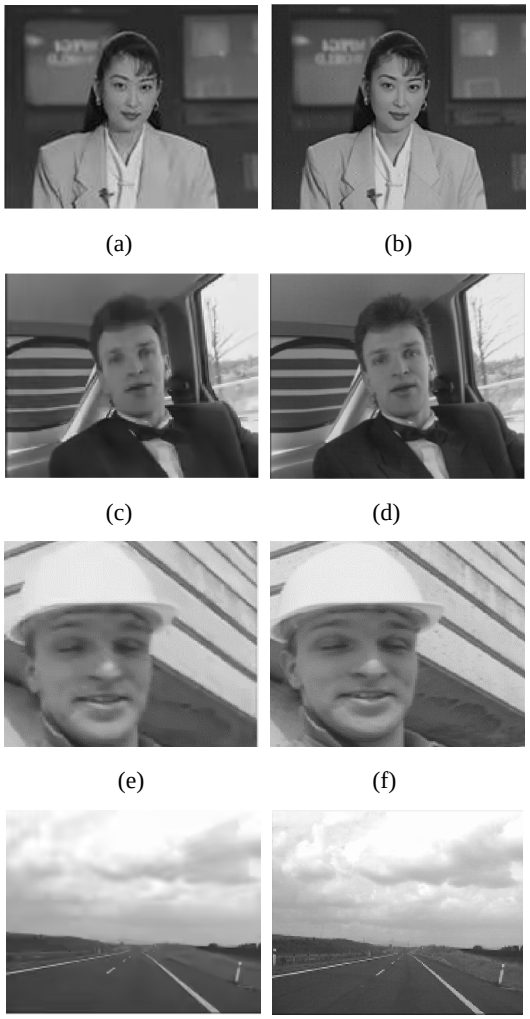


(b) 有竞争访存延迟统计图

图 11 内存访问延迟统计图
Fig. 11 Memory access latency chart

3.3 HEVC 算法硬件测试

为满足不同测试序列的更换需求，分别以标准测试序列 akiyo_qcif_176×144.yuv、carphone144×176.yuv、foreman_qcif.yuv 和 highway_qcif.yuv 为例测试 H.265/HEVC 中算法映射的可行性。如图 12 所示为测试结果，其中，图 12(a)和图 12(b)分别为 akiyo_qcif_176×144.yuv 测试序列的软件测试结果和可重构视频阵列处理器测试结果；图 12(c)和图 12(d)为 carphone144×176.yuv 序列的测试结果；图 12(e)和图 12(f)为 foreman_qcif.yuv 序列的测试结果；图 12(g)和图 12(h)为 highway_qcif.yuv 序列的测试结果；可以看出平台重构的图 12(b)、图 12(d)、图 12(f)和图 12(h)图的质量远远优于软件测试的图 12(a)、图 12(c)、图 12(e)和图 12(g)图。在软件上执行上述测试序列帧内帧间各一帧，需要 12.732 s，在该平台上需要 49320000 周期，根据硬件频率 100 MHz，总共用时 0.493 2 s，执行时间如表 1 所示。



(a) (b)

(c) (d)

(e) (f)

(g) (h)

图 12 测试结果
Fig. 12 Testing results

表 1 执行时间对比
Tab. 1 Execution time comparison /s

软件执行时间	文中平台执行时间	加速比
12.732	0.493	25.81

为测试可重构视频阵列处理器的工作性能，基于项目组前期优化的运动补偿算法，在该平台上进行了并行化实现，运动补偿时间如表 2 所示，串行执行运动补偿算法，需要 2.196 s，而并行执行只需要 0.257 s，加速比达到 8.54。性能分析如表 3 所示，文献[14]设计了可重构的滤波器，但在 90 nm 工艺下其资源占用远远高于本文，频率远远低于本文，且并行度仅为本文的 1/2。文献[15]虽然可通过扩展硬件体系结构来处理可变块的大小，在 180 nm 工艺下其资源占用略低于本文，但其频率远低于本文且并行度也仅为本文的 1/2。由于工艺不同，因此，引入面积效率参数进行对比，可从表 3 中得到，本文更优。文献[15]的资源占用低于本文是由于文献中只对算法中的部分进行了并行化设计而本文是对整个算法的设计。

表 2 运动补偿时间对比
Tab. 2 Motion compensation time comparison /s

串行时间	并行时间	加速比
2.196	0.257	8.54

表 3 性能对比
Tab. 3 Performance comparison

Related literature	[14]	[15]	This work	
Technology/nm	90	180	90	180
Frequency/MHz	171	185	357	200
Resources/(k gate)	32.49	41.97	25.25	47.78
parallelism	8	8	16	16
Area efficiency	0.045	0.009	0.154	0.081

4 结论

本文分析了传统测试方法测试视频编解码高复杂度系统时的局限性与软硬协同测试的优越性，结合实时仿真的需求开发了基于 Qt 的用户界面，采用软硬件协同设计的方法，设计实现了可重构视频阵列处理器的可视化测试平台，既能够从系统级出发对阵列处理器进行比较全面的测试，又可以通过原型测试结果对视频编解码算法的并行实现进行优化与验证。降低了测试复杂度，缩短了测试时间，与软件测试方法相比，精度明显提高，加速比为 25.81，与纯硬件相比具有较好的可观测性，且灵活性更强，满足不同测试序列的更换需求。但是存在工作频率高于 156 MHz 时重构图像不稳定及硬件平台之间的数据传输始终受制于传输通道带宽的问题。这也将是未来对可编程动态自重构三维阵列芯片体系结构研究中重点要解决的问题之一。

参考文献:

[1] Devika K N, Bhakthavatchalu R. Design of reconfigurable LFSR for VLSI IC testing in ASIC and FPGA[C]. 2017 International Conference on Communication and Signal Processing (ICCSP), Chennai, India: IEEE, 2017: 0928-0932.

[2] Zrida H K, Jemai A, Ammari A C, et al. System-Level Performance Evaluation of Very High Complexity Media Applications: A H264/AVC Encoder Case Study[J]. Int'l J. of Communications, Network and System Sciences (S1913-3715), 2011, 4(7): 436.

[3] Sutisna N, Lanante L, Nagao Y, et al. Unified HW/SW framework for efficient system level simulation[C]. Circuits and Systems (APCCAS), 2016 IEEE Asia Pacific Conference on. Jeju, South Korea: IEEE, 2016: 518-521.

[4] Teich J. Hardware/software codesign: The past, the present, and predicting the future[J]. Proceedings of the IEEE, 2012, 100(Special Centennial): 1411-1430.

[5] Yang Y, Xu J, Shi G, et al. Evaluation Test of Software and Hardware Co-simulation[M]. 5G Wireless Systems. Switzerland: Springer, Cham, 2018: 235-300.

[6] Liu S. Testing-Based Formal Verification for Algorithmic Function Theorems and Its Application to Software Verification and Validation[C]. International Symposium on System and Software Reliability. Shanghai, China: IEEE, 2017: 112-129.

[7] Kacou M A, Ghaffari F, Romain O, et al. FPGA static timing analysis enhancement based on real operating conditions[C]. Industrial Electronics Society, IECON 2017-43rd Annual Conference of the IEEE. Beijing, China: IEEE, 2017: 3556-3561.

[8] Eslami F, Wilton S J E. Incremental distributed trigger

https://dc-china-simulation.researchcommons.org/journal/vol32/iss5/5
DOI: 10.16182/j.issn1004731x.joss.18-0542

第 32 卷第 5 期 系统仿真学报 Vol. 32 No. 5
2020 年 5 月 Journal of System Simulation May, 2020

insertion for efficient FPGA debug[C]. International Conference on Field Programmable Logic and Applications. Munich, Germany: IEEE, 2014:1-4.

[9] 王斌. 数字监控系统视频编解码器性能测试方法[J]. 中国测试, 2012, 38(3): 101-104.
Wang Bin. Performance testing methods of video codec for digital monitor system[J]. China Measurement & Test, 2012, 38(3): 101-104.

[10] 汤旭龙, 安虹, 范东睿. 主流视频编解码软件的硬件性能分析与设计[J]. 计算机工程, 2014, 40(6): 300-305.
Tang Xulong, An Hong, Fan Dongrui. Hardware Performance Analysis and Design of Mainstream Video Codec Software[J]. Computer Engineering, 2014, 40(6): 300-305.

[11] Lei X, Jiang X. Design and implementation of a video codec performance evaluation system[C]. International Congress on Image and Signal Processing. Shenyang, China: IEEE, 2016: 142-147.

[12] Schmitz J A, Gharzai M K, Balkir S, et al. A 1000 frames/s Vision Chip Using Scalable Pixel-Neighborhood-Level Paralle l Processing[J]. IEEE Journal of Solid-State Circuits (S0018-9200), 2017, 52(2): 556-568.

[13] Kongetira P, Aingaran K, Olukotun K. Niagara: A 32-way Multithreaded SPARC processor[J]. IEEE Micro (S1937-4143), 2005, 25(2): 21-29.

[14] Guo Z, Zhou D, Goto S. An optimized MC interpolation architecture for HEVC[C]. IEEE International Conference on Acoustics, Speech and Signal Processing. Kyoto, Japan: IEEE, 2012: 1117-1120.

[15] Kammoun M, Atitallah A B, Masmoudi N. An efficient hardware architecture for interpolation filter of HEVC decoder[C]. International Multi-Conference on Systems, Signals & Devices. Mahdia, Tunisia: IEEE, 2015: 1-5.

