计算机研究与发展
Journal of Computer Research and Development
https://crad.ict.ac.cn
62(5): 1108−1122, 2025

FireLink: 一种面向芯粒设计空间探索的评估框架

李 开 曾 坤 荣培涛 陈志强 张 甜 王永文
(国防科技大学计算机学院 长沙 410073)
(先进微处理器芯片与系统重点实验室(国防科技大学) 长沙 410073)
(klinudt@nudt.edu.cn)

FireLink: An Evaluation Framework for Chiplet Design Space Exploration

Li Kai, Zeng Kun, Rong Peitao, Chen Zhiqiang, Zhang Tian, and Wang Yongwen
(*College of Computer Science and Technology*, *National University of Defense Technology*, *Changsha* 410073)
(*Key Laboratory of Advanced Microprocessor Chips and Systems* (*National University of Defense Technology*), *Changsha* 410073)

Abstract The Chiplet-integrated chip based on advanced packaging technology offers a number of advantages in terms of manufacturing cost, design efficiency and special customization, etc. We represent a new and effective method of maintaining the performance growth of the chip in the post-Moore era. As an important method of quantitative analysis of architectural design, design space exploration (DSE) can assist designers in comprehending and evaluating the intricate interrelationships between design parameters. However, when applying the traditional DSE method directly to the Chiplet design, it gives rise to issues such as incomplete evaluation, inaccurate simulation, and low efficiency. The solution to these problems is FireLink, an evaluation framework for Chiplet design space exploration. FireLink supports the modelling and simulation of Chiplet microarchitectures and interconnection networks, and is capable of efficiently evaluating performance, power, area and cost metrics. Furthermore, experiments are conducted using ID3 (iterative dichotomiser 3) machine learning algorithm in this framework, which has been demonstrated to effectively improve the efficiency of DSE. In comparison with existing DSE methodologies, FireLink exhibits notable advantages in comprehensiveness of evaluating, completeness of modeling and efficiency of DSE, therefore designers can explore a wider range of design space in a shorter time, so as to select a better Chiplet design scheme.

Key words Chiplet; design space exploration; simulation; evaluation; microarchitecture

摘 要 基于先进封装技术的芯粒(Chiplet)集成芯片在制造成本、设计效率以及专用定制等方面更具优势,是延续芯片性能增长的有效途径.设计空间探索(design space exploration, DSE)作为体系结构量化分析的重要方法,能够帮助设计者理解并权衡设计参数间的复杂关系.但是将传统的芯片体系结构DSE方法直接应用于Chiplet设计时,存在评估不全面、模拟不精确以及探索效率低下等问题,针对这些问题提出了解决方案FireLink,作为一个面向Chiplet设计空间探索的评估框架,它支持Chiplet微架构以及互连网络的建模和模拟,具备高效评估性能、功耗、面积和成本指标的能力.此外,在该框架下采用了ID3(iterative dichotomiser 3)机器学习算法进行了实验,结果显示该框架能够有效提高DSE的效率.与现

收稿日期: 2024-02-02; **修回日期**: 2024-10-11
基金项目: 高层次科技创新人才工程人选自主科研项目(22-TDRCJH-02-006); 国家自然科学基金青年科学基金项目(NSFC-62202481); 国防科技大学科研计划项目(ZK22-05)
This work was supported by the TDRCJH Program (22-TDRCJH-02-006), the National Natural Science Foundation of China for Young Scientists (NSFC-62202481), and the Research Program of National University of Defense Technology (ZK22-05)
通信作者: 王永文(yongwen@nudt.edu.cn)

有的 DSE 框架和方法相比，FireLink 在评估全面性、建模完整性和高效性方面具有显著优势，使得设计者能够在更短时间内探索更广泛的设计空间，进而选定较优的 Chiplet 设计方案.

关键词 芯粒；设计空间探索；模拟；评估；微架构

中图法分类号 TP391.9

DOI: 10.7544/issn1000-1239.202440082 **CSTR:** 32373.14.issn1000-1239.202440082

集成电路(integrated circuit, IC)的制造工艺随着摩尔定律发展至今，以硅为基础的半导体技术发展正面临挑战[1]. 随着 IC 产业进入后摩尔时代[2]，芯片的晶体管集成规模增长速度放缓，传统的通过缩小 CMOS 集成电路尺寸实现芯片性能提升的方法亟需革新. 芯粒(Chiplet)技术被半导体产业视为一种可以延续摩尔定律的解决方案[3]，在单片晶体管数量受限的情况下，Chiplet 架构能够通过先进封装技术集成多个“小芯片”(即 Chiplet)，以支撑芯片整体晶体管数量继续提升. 国外的芯片设计厂商 Intel[4-5]，AMD[6-8] 等都在积极布局，而目前国内仍受限于先进工艺制造能力，因此芯粒技术路线的探索具有现实意义和迫切需求.

基于 Chiplet 架构的芯片其各个 Chiplet 通常只实现完整芯片的部分功能，尺寸相比于传统的单片片上系统(system on chip, SoC)更小，因而较小的面积使其在制造时具有更高的良率. 作为完整芯片的组成部分，Chiplet 可以被视为硬件“IP”，能够在不同芯片中实现复用，从而降低芯片定制化的设计门槛和制造成本，缩短芯片开发时间. Chiplet 架构在异构集成方面也具有天然优势，基于先进的集成封装技术，可以更好地应对物联网(Internet of things, IoT)、人工智能(artificial intelligence, AI)、5G 和大数据[9-12] 等新兴领域以及复杂应用场景对芯片的带宽、延时、算力以及功耗等指标的多样化、定制化需求.

相比于 SoC，基于 Chiplet 集成的系统具有更丰富的设计多样性，与此同时 Chiplet 的划分、布局以及互连等方面的问题也使得设计空间探索(design space exploration, DSE)更为复杂. 在传统的通用处理器设计流程中，通常以规范化的指标如能量延迟积(energy-delay product, EDP)、能量延迟面积积(energy-delay-area product, EDAP)等作为优化目标，使用合理的 DSE 算法搜索最优的微架构参数配置. Chiplet 架构的 DSE 问题由于面临着更多维度的综合考量，需要在性能、功耗、面积 (performance power area, PPA) 与成本等评估指标上进行综合权衡.

在芯片架构设计早期，设计者通常会采用体系结构模拟器进行设计空间探索，对目标设计进行建模与仿真，快速获取相关评估指标以支持进一步量化分析. 现有开源的体系结构模拟器(Gem5[13]，GPGPU[14]，Sniper[15] 等)在面向 Chiplet 架构模拟时存在诸多不足，例如缺少完整的 Chiplet 体系架构建模以及模拟过程缓慢等问题. 由于 Chiplet 架构引入了新的设计模块和设计参数，如 D2D(die-to-die)接口和互连延迟等因素，需要在建模和评估时考虑更多的设计细节. 此外设计空间探索是一个反复迭代的过程，包括设计、模拟、评估在内的完整流程，通常软件模拟器的模拟频率只能达到 kHz 级别[16]，在模拟环节的耗时极大制约了设计空间探索的效率.

针对以上问题，本文面向 Chiplet 设计空间探索，建立了一个快速的自动化设计、模拟和评估框架——FireLink，以应对 Chiplet 技术迅速发展所带来的芯片架构和技术变化. 旨在帮助相关领域研究者快速验证和评估 Chiplet 架构设计策略，更好地探索 Chiplet 架构的潜力和瓶颈.

本文的主要贡献包括 3 个方面：

1)搭建了一个面向 Chiplet 架构进行敏捷设计、快速模拟和全面评估的综合框架 FireLink，该框架通过自建脚本实现多种开源工具间输入输出适配，提供了从输入设计参数、生成寄存器传输级(register transfer level, RTL)硬件描述、软件模拟、硬件实现到设计评估输出的自动化工作流程；

2)构建了面向 Chiplet 架构的设计模型和一个全面的综合评估指标 PPAC(performance power area cost)，反映性能、功耗、面积、成本，实现了 Chiplet 架构设计的多层次模拟和全面评估；

3)基于机器学习中的决策树算法提出一种对历史设计参数建模、预测并生成符合规则的设计参数的方法，以此裁剪设计空间规模，提升设计空间探索的效率.

1 相关工作

1.1 Chiplet 设计评估

传统的 SoC 难以满足人工智能、高性能计算和

数据中心等应用领域不断增长的高算力需求[17], 基于 Chiplet 集成的芯片设计方法凭借其更高的性价比获得越来越多的关注. 但是基于传统的 EDP, EDAP, PPA 等指标的评估方法难以体现 Chiplet 架构的单位算力成本优势, 因此亟需一种能够准确评估 Chiplet 综合“价值”的指标, 进而凭借其更好地指导前期设计优化.

目前面向 Chiplet 架构评估的相关研究普遍聚焦于单一的评估指标, 缺乏完整的框架, 难以综合分析设计、性能和成本之间的相互关系. Chiplet Actuary[18] 提出了一个定量的成本模型, 以帮助设计者在 Chiplet 设计初期粗略评估成本. 而 Ahmad 等人[19] 在面向 Chiplet 异构集成的成本和良率分析中展示了更多细节, 包括材料类型、测试方法、操作流程等, 从而确定基于 SoC 分解的 Chiplet 实现的可行性. 虽然文献[20] 对 2.5 维的集成电路设计进行 PPA、信号完整性(signal integrity, SI)和电源完整性(power integrity, PI)的分析, 但是缺少对成本的量化评估. 这些工作侧重于 Chiplet 架构设计过程中的某一个阶段, 缺乏一个全面的评估框架来同时考虑性能、功耗、面积和成本等多个维度.

1.2 Chiplet 模拟技术

目前面向 Chiplet 架构的建模和模拟技术研究正在广泛开展, 一些研究团队利用商业 EDA 工具建立逻辑设计与物理实现的协同设计和仿真框架. 例如 Kim 等人[20] 面向异构 2.5 维设计开发了一个高度集成的设计仿真流程, Kabir 等人[21] 提出了类似的前后端协同的处理器设计方法. 这一类型的工作依赖于商业的 EDA 工具和数据集, 导致了其有限的适用范围. 虽然商业 EDA 工具和工艺库能够提供详细的芯片信息(功耗、面积、寄生效应、可靠性等), 但是由于这些工具和数据致力于商业用途, 存在许可证获取困难、使用规范严格、学习成本高等限制, 使得其用于学术研究的效果并不理想. 此外, 后端设计需要大量的计算资源和时间投入, 设计参数的任何微小修改都需要完成长链路耗时的串行流程. 这种高昂的时间花费和成本开销限制了专业 EDA 工具在芯片早期开发设计中的进一步推广和应用. 因此本文的研究选择基于开源项目进行拓展, 并且对 Chiplet 架构模拟过程进行了加速, 减少后端物理实现流程对设计空间探索的影响.

体系结构模拟器通过对目标架构建模、模拟和量化分析来指导设计, 能够缩短芯片设计时间、降低开发成本[22], 这些优势使其在芯片早期开发时具有突出地位. 周俊成[23] 通过将 RTL 软件模拟器 Sniper 与 Booksim 片上网络 (network-on-chip, NoC) 模拟器结合来对 Chiplet 架构设计进行联合模拟. Zhi 等人[24] 提出了一种使用开源模拟器为多 Chiplet 集成系统构建模拟器的通用方法, 能够支持对大规模系统的并行模拟, 实现了 Chiplet 间和 Chiplet 内互连的精确建模, 并开发了针对 Chiplet 集成系统的分布式和共享内存模型. 虽然软件模拟器在研究中被广泛应用, 但是对于复杂设计问题而言, 其 kHz 级别的模拟运行速度严重影响了设计空间探索的效率, 尤其需要对模拟过程进行监控和提取关键信息时, 其速度会变得更加缓慢.

软件的模拟速度难以满足 Chiplet 设计空间探索的需求, 尤其对于在设计初期需要对目标设计快速进行迭代优化的情况下, 模拟的速度是设计者必须考虑的关键问题. 基于 FPGA 加速的模拟方法能够以更高的频率运行模拟系统, 但需要编写相应的寄存器传输级硬件设计代码且对设计修改困难. 本文所做的工作很好地权衡了模拟速度与设计空间探索效率之间的关系, 基于 Chisel[25] 敏捷设计语言的设计框架能够快速实现 Chiplet 微架构和互连网络结构集成, 然后自动生成相应的寄存器传输级硬件描述并通过 FPGA 加速模拟过程.

1.3 Chiplet 设计空间探索

Chiplet 架构相较于 SoC 的设计空间更为复杂、模拟难度更大, 因此研究者们在面向 Chiplet 架构的设计空间探索的工作各有侧重. 例如只关注于 Chiplet 架构下基于中介层的先进封装技术对片上网络设计[26-29] 的影响, 或是 2.5 维集成设计的物理特性(热效应、电气性能、应力等)仿真[30-32]. Pal 等人[33] 开发了基于 Chiplet 组装的处理器设计空间探索框架, 目标是确定最优的 Chiplet 组合. 此前提到的 Kim 等人[20] 的研究领域只专注于 2.5 维集成封装的芯片设计空间探索, 通过完全自动化的架构、电路和封装全阶段 EDA 流程, 分析并权衡基于中介层的 Chiplet 集成方案. 这些工作聚焦于先进封装, 强调 2.5 维或 3 维集成电路设计的量化分析, 但是建模缺乏对 Chiplet 微架构细节的考量.

还有一些研究者以某一特定应用领域为方向探索 Chiplet 架构的设计空间, 例如 Puzzle[34] 面向深度学习领域, Chopin[35] 关注于成本效益, Lan 等人[36] 的研究工作以神经形态处理器为目标. 相比于这些研究者, 更专精于某一特定目标进行基于 Chiplet 的设计空间探索研究, 本文提出的面向 Chiplet 设计空间

探索的评估框架是更基础性的、广泛适用的，能够对基于 Chiplet 集成的芯片设计方案的可行性进行全方位的分析和评估.

2 FireLink

本节将介绍 FireLink 的整体框架，以及框架集成的各个开源工具发挥的作用，并描述如何面向 Chiplet 架构进行设计、模拟和评估. 为便于阐述，明确 3 个术语在本文中的具体含义：

1）Chiplet. 完整功能的芯片可以按照一定规则划分为多种类型的模块，然后由单个或多个模块集成的，具有特定功能的 Die.

2）Die.由晶圆切割的未经封装的物理器件.

3）模块. 具有某种特定功能的集成电路.

2.1 框架概述

图 1 展示了 FireLink 的框架示意图，相关的工具及其功能如表 1 所示. FireLink 基于开源平台 Chipyard[43] 进行拓展，Chipyard 是一个基于 Chisel 的硬件设计平台. 框架集成了多种开源工具和不同种类的面向 RISC-V 系统的 RTL 生成器，能够支持异构多核芯片开发的硬件设计、模拟和实现全流程. 这些部件间具有良好的互操作性，但是由于版本兼容性以及严格的输入限制问题，它们之间的互联并非天然适配，需要对传输信息进行专门处理. 例如 Verilator 并不支持“$finish()”和“$fatal()”等不可综合的语句以及逻辑信号“X”的编译，Yosys[44] 综合前也需要对代码进行语法检查. 通过创建一系列的脚本解决了这些问题，包括执行错误处理、输出数据采集和提取关键参数，整个流程能够自动化执行.

Chiplet 的微架构设计依托于 Rocket Chip 生成器实现建模，它是一个基于 Chisel 语言的参数化的硬件生成器. 通过 Chisel 包装的大量开源 IP 语料库，能够快速地构建一个 RISC-V 系统所需的所有功能单元. 与此同时，Constellation 作为一个多核互连网络的生成器已经被集成到 FireLink 框架中，为基于 Chipyard 构建的硬件设计建立互连网络. 该互连结构具有良好的灵活性和可配置性，能够支持任意有向图网络拓扑，通过物理结构、数据流方向和路由设置 3 个层次设计互连网络的方法极大简化了 Chiplet 之间的通信建立过程. 于是 FireLink 在构建分层的拓扑结构的基础上，可以实现精确的 Chiplet 内与 Chiplet 间的互连网络建模.

多核系统由于共享缓存的存在，为保证缓存一致性带来的死锁问题是体系结构设计的难题. 虚拟网络是解决死锁的一种有效方法，网络能够通过虚拟通道传输不同类型的报文，但是设计额外的虚拟通道会产生高功耗和大面积，尤其在核数目较多的体系结构下这种负面的影响尤为突出[45]. 在 FireLink 框架下支持配置虚拟通道的互连网络设计，能够确保网络是无死锁的，这有利于快速构建网络而无须考虑复杂的流关系.

为了兼顾准确性和模拟速度，FireLink 框架重新设计了建模与模拟过程，对原有固定的标准模拟流程与功能部件解耦，并把模拟区分为基于软件的模拟和基于 FPGA 加速的模拟. 在进行软件模拟时，编译

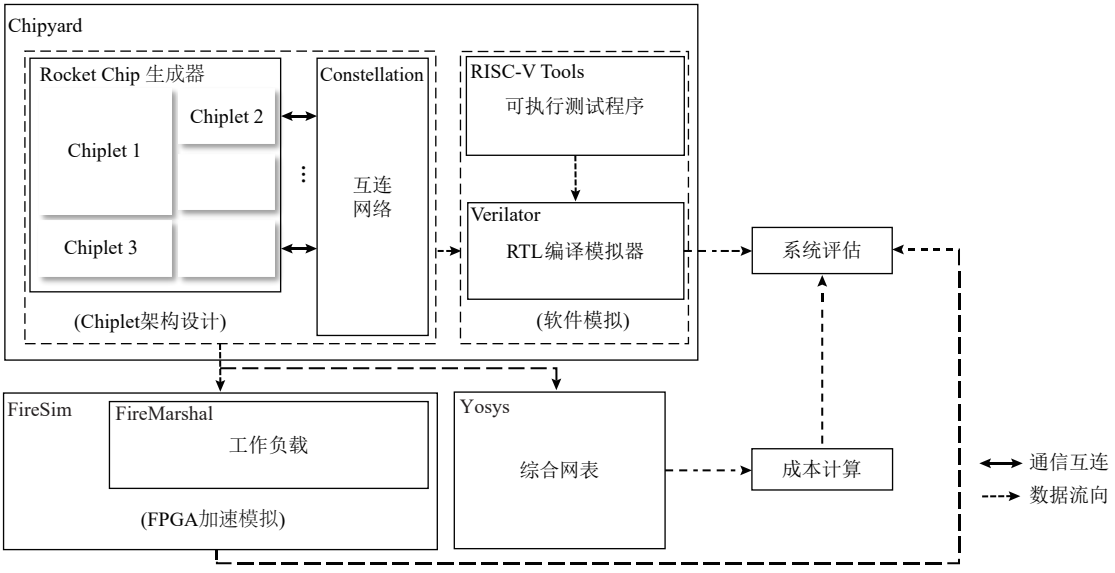


Fig. 1 Illustration of FireLink framework

图 1 FireLink 框架示意图

Table 1 Functions of the Tools Supported by Chipyard

表 1 Chipyard 支持工具的功能

工具	功能	目标
Rocket Chip[37]	RISC-V SoC 生成器	处理器微架构
Constellation[38]	NoC 生成器	互连网络
Verilator[39]	软件模拟器	软件模拟器
RISC-V Tools[40]	系统开发工具链	可执行测试程序
FireSim[41]	硬件模拟框架	硬件加速模拟
FireMarshal[42]	工作负载生成器	自定义工作负载

器 Verilator 根据预先的设计架构描述(Scala)将生成的 SystemVerilog 编译为可执行的模拟器代码, 通过运行 RISC-V Tools 提供的基准测试程序集进行模拟.

FireSim 是一个通过 FPGA 加速模拟过程的硬件模拟平台, 支持基于 Chipyad 平台建立的目标架构模型, 能够对其进行周期精确的全系统模拟. FireSim 对目标设计模型的寄存器传输级硬件描述进行综合并生成门级网表, 综合后的网表映射到 FPGA 内部资源实现具体的电路, 然后设计文件转换为比特流并被下载部署在本地 FPGA 上, 由其进一步执行加速模拟. FireMarshal 是面向 RISC-V 系统的工作负载生成工具, 支持在 FireSim 流程中生成自定义程序负载, 保证了测试程序的可拓展性. 工作负载由一系列“作业”组成, 每个“作业”能够对应分配到目标系统中的多个逻辑节点上并行处理.

FireLink 框架能够对 Chiplet 架构设计的关键指标进行量化评估, 包括性能、面积、功耗和成本. 在模拟过程结束时能够从模拟的运行时间、周期数和频率等信息中量化性能. 对面积指标的推算, 凭借引入开源的 Verilog 综合和验证工具 Yosys, 对输入的 RTL 进行综合, 生成目标设计架构的网表信息, 以此来估算各模块的面积大小. FireSim 通过对设计网表的电路实现, 提供了功耗报告, 这将作为芯片设计功耗量化指标的参考值. 为全面评估 Chiplet 架构设计在性能、功耗、面积与成本上的综合“价值”, 框架引入了成本模型并提出了 PPAC 评估指标, 相关原理和可行性将在 2.3 节中详细介绍.

面向 Chiplet 架构设计空间探索领域, 提出的框架 FireLink 相比于其他 DSE 工具和研究工作的优势在于建模的完整性和更快的模拟速度, 关键指标对比如表 2 所示. 由于模拟目标、工作负载和系统配置等差异, 模拟速度与精度难以进行横向比较, 虽缺乏直接可比性, 但是 FireLink 框架能够支持 FPGA 加速, 这项功能确保了在对同等规模目标设计进行模拟时能够更高效.

Table 2 Comparison of DSE Tools and Research Work

表 2 DSE 工具和研究工作对比

工具/方法	NoC	速度	评估指标	评估精度
Gem5	无	很慢	性能	低
GPGPU	无	很慢	性能	低
HGBO-DSE[46]	无	快	PPA	低
BOOM-Explorer[47]	无	快	性能、功耗	高
ScaleHLS[48]	无	很快	性能	低
AutoDSE[49]	无	快	性能	低
SoC-Tuner[50]	无	慢	PPA	高
SODA[51]	有	很慢	PPA	低
MOOS[52]	有	很慢	性能、功耗	低
FireLink (本文)	有	快	PPAC	高

本文提出的框架支持了更多的建模细节, 包括功能部件、微架构、互连网络以及 Chiplet 层次的系统架构设计. FireLink 有效解决了传统的软件模拟速度慢导致的设计空间探索效率不高的问题, 加速模拟的机理是基于 Chisel 直接生成寄存器传输级的硬件描述的过程, 在框架下生成的是目标设计架构的模拟器而不是一个硬件原型, 对于每一个目标设计的时钟都将由动态可变的多个 FPGA 时钟来进行模拟[41]. 另一方面, FireLink 提出了对性能、功耗、面积和成本进行全面评估的指标 PPAC, 实现了对这些参数的自动化提取和准确量化, 这为快速迭代和验证架构设计提供了有力的支持.

2.2 Chiplet 架构建模

完整的 SoC 系统由一系列不可分割的基础模块构建, 从 SoC 分解到 Chiplet 的过程认为这些基础模块是不会更改的 (D2D 额外考虑). 这些模块相互组合或独立制造, 为单个 Die 构成了不同的 *chip*, 即 Chiplet, 再通过 Chiplet 互连集成组成 Chiplet 架构系统, 这一过程与 Chiplet 的分区划分方式密切相关. 以 k_j 和 k_i' 分别代表模块 j 和 Chiplet i 的数目, 公式描述为:

$$C_{\text{SoC}} \leftarrow module = \{m_1, m_2, \cdots, m_j\}, \tag{1}$$

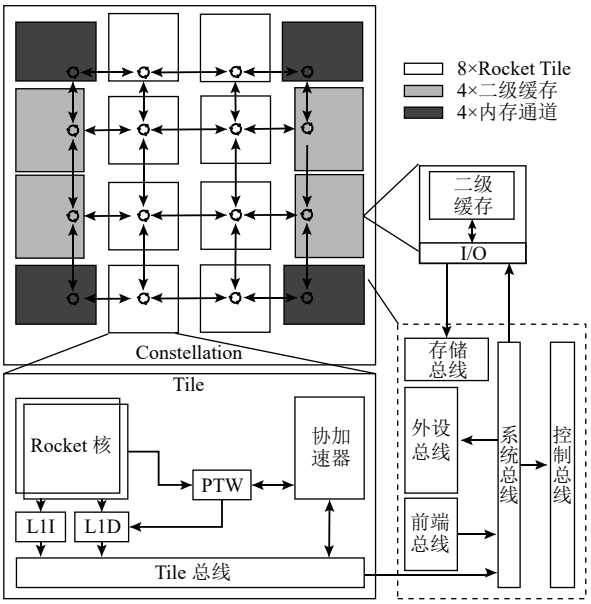
$$chip_i = \left\{m_2^{k_2}, m_3^{k_3}, \cdots, m_j^{k_j}\right\}, m_j \in module, \tag{2}$$

$$C_{\text{Chiplet}} = \left\{chip_1^{k_1'}, chip_2^{k_2'}, \cdots, chip_i^{k_i'}\right\}. \tag{3}$$

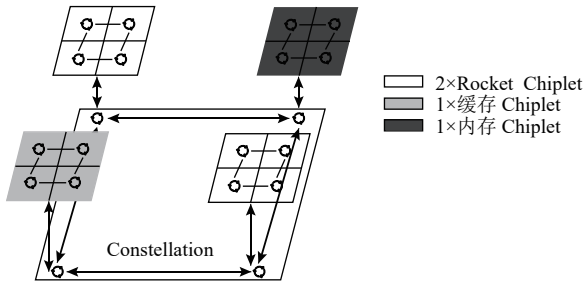
实验创建了一个 SoC 模型作为参照示例, 其由 8 个 Rocket Tile、4 个 L2 Bank 和 4 个内存访问模块组成, 通过 4×4 的 Mesh 网络进行互连集成. 其中每个 Rocket Tile 微架构被设计为 Rocket Core(Chisel 编写的五级有序标量处理器内核)、PTW (page-table walker)、

一级指令缓存(L1 instruction cache, L1I)和一级数据缓存(L1 data cache, L1D). Tile 架构内还可配置一个 Rocket 核定制协处理器 RoCC(Rocket custom co-processor)连接到内核, 以在特定应用领域进行加速.

如图 2 所示, 我们基于 SOC 划分对 Chiplet 架构进行建模. 考虑到 Chiplet 集成的复用特点, 将 SoC 可以划分为 Rocket Tile、L2 缓存和内存通道 3 种模块作为示例, 它们可以被分别设计为独立的 Chiplet, 也可以相互组合成为不同类型的 Chiplet. Chiplet 间的互连是 Chiplet 架构的关注点, 在先进的 2.5 维和 3 维封装中, 通常采用基于 NoC 的互连结构[53].



(a) 4×4 网格网络 SoC



(b) 层次化网络 Chiplet

Fig. 2 Illustration of chip modeling

图 2 芯片建模示意图

相较于单芯片的网络结构, Chiplet 间的互连网络面临着更复杂的互连挑战, 即:

1)层次化网络.Chiplet 架构下的互连不仅要处理 Chiplet 内的通信, 还要跨越 Chiplet 边界实现 Chiplet 间高效的数据交换.

2)各向异性互连. 由于 Chiplet 异构的可能性以及 Chiplet 尺寸的差异, 互连结构需要支持不同 Chiplet 之间的通信延迟和带宽可独立配置.

3)不规则拓扑. Chiplet 的放置布局会影响互连设计方案, 并且先进封装也一定程度上限制了互联网络的物理结构, 容易导致不规则的拓扑结构.

Chiplet 架构的互连网络设计关键在于联系 Chiplet 内与 Chiplet 间的数据传输, 实现这一目标依靠 Constellation 支持的丰富的可配置的物理规范, 包括网络通道、路由器配置和终端的关键参数等. FireLink 下互连网络建模支持的关键设计参数包括: 虚拟通道数目、有效负载宽度、终端延迟、互连传输速度和拓扑结构等, 基于这些参数的差异化配置, 可以细致描述 Chiplet 互连结构的关键特征, 使得互连结构能够适配 Chiplet 的特定需求. 例如 Chiplet 内/间的链路距离对传输速率的影响可以由通道速度与终端延迟表征, 有效负载宽度能够根据每个 Chiplet 的数据传输需求和内外互连场景进行特异性配置.

通过建立分层网络结构来描述 Chiplet 内部互连与 Chiplet 间互连的传输效率差异性, Chiplet 架构的互连结构建模问题得以解决. 除了可参数化的设计因素以外, FireLink 框架还支持路由算法、流量规范和虚拟通道分配策略等互连网络设计权衡. 能够精确到每个节点的配置功能为 Chiplet 的互连结构提供了充分自由的设计空间, 由此物理的封装实现能够有效映射到互连结构设计中.

所有的设计文件都使用 Chisel 语言构建, Chisel 是一种以 Scala 为基础开发的开源的硬件设计语言, 由于 Scala 中支持了丰富的包、类和预定义硬件描述, FireLink 框架下进行硬件设计是高度模块化、参数化的. 相对于 Verilog, VHDL, SystemVerilog 等传统的硬件设计语言, 使用 Chisel 进行开源处理器的敏捷化设计和开发探索十分高效, 能够将硬件模块设计的时间压缩 50% 以上, 进而减少芯片验证与实现的迭代周期(30% 以上)[54]. 当目标设计完成后, Chisel 格式的设计文件通过 Chisel 编译工具链转换为一种名为 FIRRTL 的中间表达, FIRRTL 进一步可以生成 SystemVerilog 格式的寄存器传输级硬件代码, 后续的模拟过程都是以此作为输入.

2.3 PPAC 评估

在基于 Chiplet 集成的芯片设计初期对其成本进行估计是有必要的, 设计者必须清晰了解芯片设计策略在成本上的潜在优势, 因为这将影响芯片架构设计决策. 在相同成本下的 PPA 表现劣于 SoC 集成的 Chiplet 集成设计方案是不可接受的.

为了对 Chiplet 架构进行准确的评估，验证设计空间探索的正确性和可靠性的切实需求. 本文面向 Chiplet 架构提出了一个 PPAC 评估函数，该函数基于芯片设计的性能、功耗、面积和成本 4 个关键参数. PPAC 是一个综合的评估指标，其包含的 4 个输入对结果的影响程度是不一样的，例如性能提升 10% 并不能和降低 10% 的功耗相等. 所以基于加权平均法对性能、功耗和面积因素分别赋予权重.

$$I_{\mathrm{PPAC}}=\frac{\alpha\times performance+\beta\times power^{-1}+\gamma\times area^{-1}}{cost}, \tag{4}$$

$$\alpha+\beta+\gamma=1. \tag{5}$$

式(4)中 α, β, γ是权重系数，权重值随设计目标和应用需求调整以控制设计优化方向. 例如面向低功耗的设计，从体系结构的层次考虑能够使芯片功耗优化幅度更显著[55]，β 的取值将适当增加. 其余 4 个参数 *performance*, *power*, *area*, *cost* 分别是目标 Chiplet 架构设计的性能、功耗和面积的量化指标以及成本模型的输出，它们的值在 FireLink 的流程中获取，并且获取这些值的方法是解耦的.

性能作为评价体系结构设计的关键参数，在设计验证中，将目标设计的硬件描述实例化并映射到 FPGA，通过运行真实的应用程序以获取精确的性能指标. Vivado 软件的综合工具将目标设计描述的 SystemVerilog 代码综合成硬件，在此过程中可以从输出的功耗报告中获取功耗参数 *power* 的相关信息.

面积参数的获取通常需要基于商业 EDA 工具(如 Cadence[56], Synopsys[57] 等)完成，但对于前期设计而言这是一个非常耗时的过程，严重影响设计空间探索过程中反复迭代优化的效率. FireLink 框架通过引入了快速 RTL 综合工具 Yosys，在设计后端将 SystemVerilog 文件综合为门级网表，获取总体设计的晶体管规模大小，以估算面积. 与此同时，在 FPGA 加速仿真流程中，当通过 Vivado 对硬件设计进行综合和实现时可以得到利用报告作为面积参数的辅助参考.

成本模型参考了 Chiplet Actuary[15] 的工作，在此基础上依托 FireLink 和设计流程重新解释了成本模型，以更好地指导该框架下对 Chiplet 架构的设计空间探索. 主要变化在于成本模型中的设计成本，如式(6)所示.

$$cost_{\mathrm{design}}=\sum_{m_i\in M}\left(K_{m_i}N_{m_i}\delta+C_{\mathrm{module}}\right)+\sum_{c_i\in C}\left(K_{c_i}N_{c_i}\delta+C_{\mathrm{chip}}\right). \tag{6}$$

如图 3 所示，Chiplet 集成系统在设计成本上的特征表现在 3 个方面：

1)额外的 D2D 功能模块的设计成本；

2)Chiplet 集成芯片需要支付与 SoC 相同的功能模块设计成本，除此之外其基本的集成单元——Chiplet 是由模块集合 M 的子集实现的 Die，每一种类型的 Chiplet 会产生额外的架构设计成本；

3)Chiplet 集成具有一个显著的特点，这种设计使 Chiplet 可以在广泛的系统中重复使用，即 Chiplet

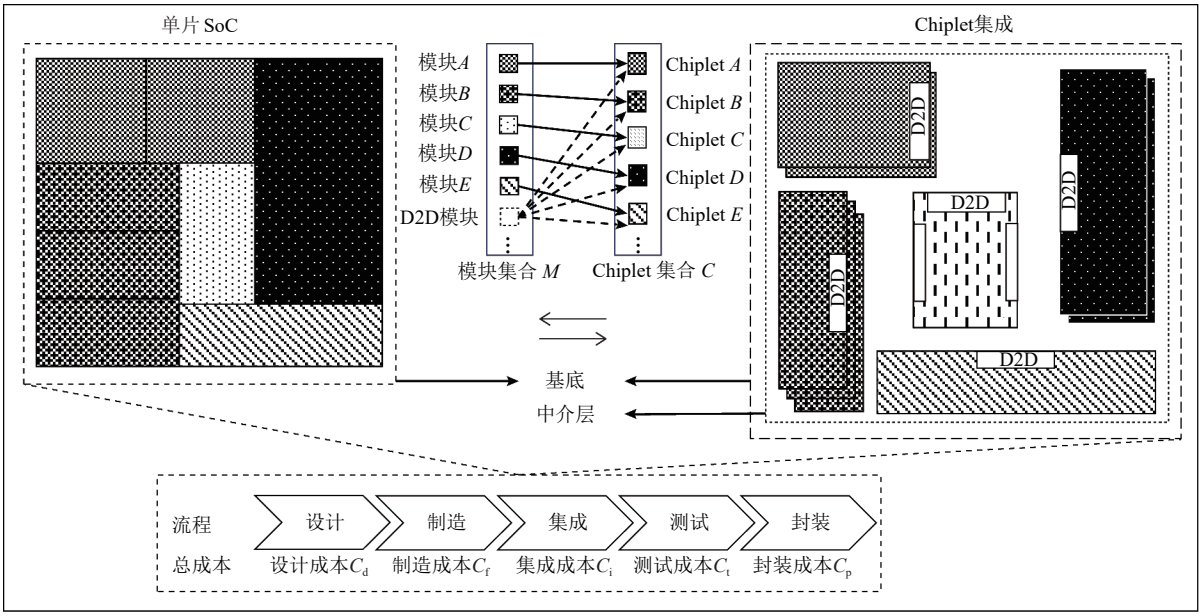


Fig. 3 Illustration of cost model

图 3 成本模型示意图

被视为成本可摊销的“硬件 IP”.

为解决面向 Chiplet 架构的芯片设计评估问题，尤其是面对基于 SoC 分解的 Chiplet 集成系统是否具备实现的经济可行性，本文基于提出的 Chiplet 架构设计和模拟框架 FireLink 以及可获取的数据信息重新对成本模型进行了描述. 成本模型中各个公式中的参数、特征数据均来自于公共数据和相关工作所开放的信息，推荐使用商业信息以及 EDA 后端真实数据以进一步获得更好的可靠性和准确性.

2.4 DSE 流程

基于 RTL 生成器的模块化硬件设计方法以及敏捷设计语言带来的快速可配置性，芯片的微架构设计被抽象和简化为设计参数的选择. 通过参数配置能够快速地描述目标设计架构并搭建完整的系统模型，这有助于设计空间探索快速迭代设计方案. 基于前文介绍的 FireLink 框架和工具链，本文设计了一个自动化的芯片设计优化和评估流程，以支持面向 Chiplet 架构的设计空间探索. 如图 4 所示，评估是整个流程的关键，它由基于设计参数优化的设计空间探索算法以及以 PPAC 指标为参考的设计验证方法 2 个部分组成.

Chiplet 架构的设计空间 Φ 是各 Chiplet 设计空间与 Chiplet 集成相关的设计参数 P' 的笛卡儿积，是一个复杂的高维空间. 其中 P' 不会作为设计空间探索算法的输入，将在算法运行过程前确定. P' 定义为多 Chiplet 互连层次的设计参数，主要集中在 Chiplet 间互连结构上，由于层次化的互连网络作为 Chiplet 间通信的基础架构，其设计决策直接关联到整个系统的物理实现，所以互连结构的设计参数需要在进行详细的设计空间探索之前被初步确认.

$$\Phi = \prod_{i=1}^{n} chip_i \times P'. \tag{7}$$

在图 4 中芯片架构所有的设计参数被包装为一个集合，它会确定一个完整的芯片架构. 如式 (8) 所示，每个设计参数 p 由多个离散的候选值 v 组成，这些候选值构成了设计参数的取值范围. 通常情况下，每种设计参数的候选值范围是不同的，并且分布是不均匀的.

$$Chiplet = \{p_1, p_2, \cdots, p_j\},\ p_j = \{v_1, v_2, \cdots, v_i\}. \tag{8}$$

Chiplet 架构的设计空间探索问题实质上就是对所有影响设计的因素 x 进行选择权衡，通过评估架构设计的 PPAC 指标，衡量设计方案的优劣. 设计空间探索就是一个不断迭代搜索最佳设计的过程，可以表述为

$$\boldsymbol{design} = (x_1, x_2, \cdots, x_n), \tag{9}$$

$$Y: \boldsymbol{design} \rightarrow Evaluation,$$
$$Y(\boldsymbol{design}) = E(performance, power, area, cost), \tag{10}$$

$$\boldsymbol{design}^* = \arg\max_{\boldsymbol{design} \in \Phi} E(\boldsymbol{design}). \tag{11}$$

丰富的设计参数在具备优异的设计灵活性的同时，也使得错误的配置更容易发生. 对于经验丰富的架构设计者而言，基于先验知识可以很好地避免这种情况. 但 Chiplet 集成的系统作为一种新的设计架构，并不能确保传统的多核架构设计“经验”总能发挥作用. 在面对设计参数集合这种离散数据时，ID3[58] 算法具有很好的适用性，并且不需要对数据进

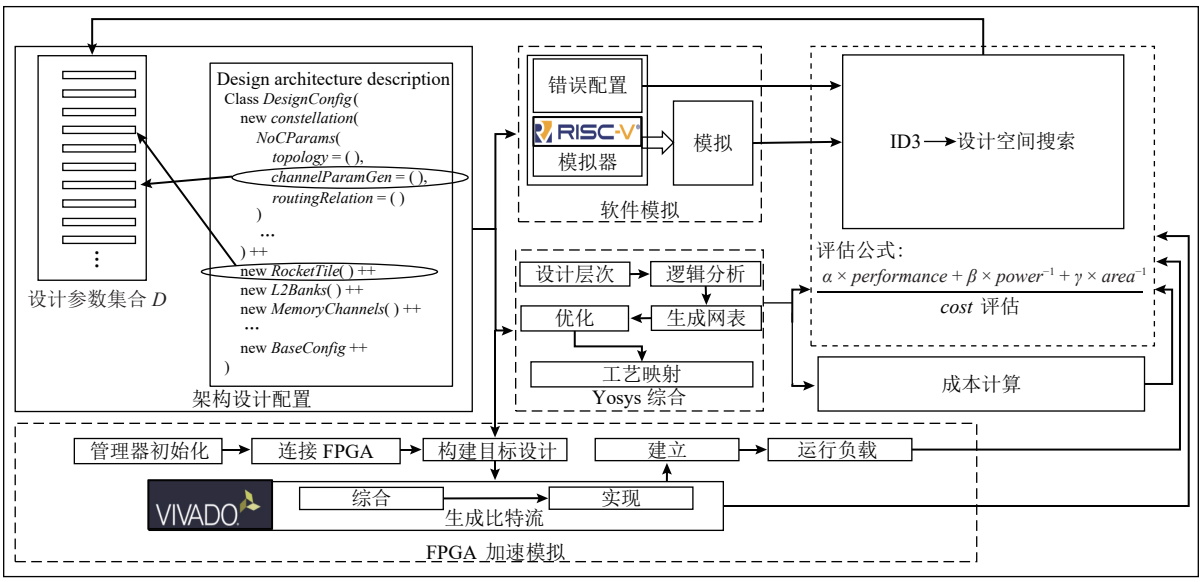


Fig. 4 Illustration of DSE

图 4 设计空间探索流程示意图

行规范化和标准化处理. 为此在选择设计参数时通过采用 ID3 算法为历史设计参数选择构建决策树分类器, 进而预测并生成合理的设计参数集, 以尽可能避免“违法”的参数配置.

设计文件首先会进行软件模拟以验证设计参数的有效性, 因为在编译生成模拟器的寄存器传输级硬件代码过程中可能会出现“illegal argument exception”等类似错误. 如果当前编译的设计参数属于“illegal”, 那么这一组设计参数将被打上“erro”的标签, 然后写入一个数据集. 而成功编译并进行 RISC-V 基准测试的设计参数集合, 作为成功的案例被标记为“success”. 以设计参数是否能成功编译作为分类特征, 基于 ID3 算法对历史设计参数构建的决策树模型预测并生成合法的设计参数配置, 并以此作为新的设计架构描述, 这一过程变相实现了对设计空间的裁剪. 具体算法如算法 1 所示.

算法 1. 基于 ID3 生成设计参数算法.

输入: 历史设计参数数据集 *Data*;

输出: 预测的有效设计参数集合 *Parameters*.

① *Train_set*, *Test_set*←*Split_dataset* (*Data*); /*根据目标特征划分训练集与测试集*/

② *base_entropy*←*Calculate_entropy*(*Train_set*); /*计算数据集信息熵*/

③ for each *attribute* in *Train_set*

④ *info_gain*←*Gain*(*base_entropy*, *attribute*, *Train_set*); /*计算当前属性信息增益*/

⑤ *best_attribut*←*Judge*(*attribute*, *info_gain*); /*评判信息增益获取最佳属性*/

⑥ end for

⑦ *Tree.root*←*best_attribute*; /*最佳属性作为决策树根节点*/

⑧ for each *value* in *tree.root*

⑨ *subset*←*Get_subset*(*Tree.root*, *value*); /*获取最佳属性的每个值子集*/

⑩ end for

⑪ if *subset* is empty

⑫ *Tree.Add_leaf*(*value*);

⑬ else if *subset* is not empty

⑭ *subtree*←*ID3_Subtree*(*subset*); /*递归构建子树*/

⑮ *Tree.add_subtree*(*value*, *subtree*);

⑯ end if

⑰ while *node* is not None /*选择一个叶节点开始直至无下一节点*/

⑱ *Parameter*[*node.attribute*]←*node.value*;

⑲ *node*←*Tree.get_next*(*node*); /*根据决策树的规则选择下一个节点*/

⑳ end while

PPAC 作为一个评价目标 Chiplet 架构设计方案是否优异的关键判断因素, 它受诸多设计因素影响, 不仅是设计参数, 也包括 FireLink 框架下各个工具本身的误差和程序运行的误差(即使在相同设计下测试程序每次的结果都不是完全一致的). 并且性能、功耗、面积和成本也并非真实芯片的数据, 只是基于当前设计的模型的特征信息的一个量化的值. 但是对于早期的实际空间探索而言, 这些指标足以反映芯片设计的“趋势”. 因为 FireLink 框架下 Chiplet 设计的架构不同于软件模拟器中的模型, 所有的设计单元都会生成相应的硬件代码, 并且能够基于 FPGA 进行电路实现. 所以在 FireLink 下的模拟数据具有较强的可信度, 能够为 Chiplet 架构设计者提供可靠的参考和决策意见.

3 实验及结果

在本节中将分析 3 个案例以证明提出的面向 Chiplet 架构设计空间探索的评估框架的可行性. 实验以 8 核 RISC-V 系统为目标, 以 2.2 节中建模的 SoC 设计模型为基础. 实验相关参数如表 3 所示, 实验构建了多个 Chiplet 集成芯片架构设计模型以探索不同维度的设计方法, 并基于定量的综合评估帮助深入理解其背后的原理.

Table 3 Experimental Relative Parameters

表 3 实验相关参数

配置	参数	取值
处理器核	类型	Rocket Core
	数目	8
L1 缓存	关联度	4 路组相联
	大小/KB	32
	延迟	2
L2 缓存	关联度	8 路组相连
	大小/KB	4×512
内存通道	数目	4
互联网络	虚拟通道数	4
	Chiplet 内/间报文传输速度	4/1
	Chiplet 内/间有效负载宽度	128/32
FPGA	型号	U250
	运行频率/MHz	80

3.1 分区划分

分区划分、Chiplet尺寸与制造成本、Chiplet的复用性等多个因素影响下的设计权衡是一项困难的挑战，如何将单Die形式的SoC设计转换为Chiplet集成的系统是架构设计师首先需要进行思考的问题. 在FireLink框架下，芯片架构的设计被简化为一个功能部件堆叠配置的过程，这帮助设计者能够很容易地以功能作为划分标准实现SoC解构. 如第2节中所展示的那样，在此展示一个范例，简单地将SoC划分为处理器核心区、高速共享缓存区以及存储控制区，总线、接口及其余部件纳入互连网络区域中. 实际上在FireLink框架下设计空间的定义是更加灵活和广泛的，支持更广泛的划分方式. 由于Chisel语言的高度模块化特点，Core, MMIO, DMA, I/O等系统组件能够作为更细粒度的模块在系统中进行集成，并允许将这些模块合并或分离为单独的Chiplet.

通过控制设计变量，每个案例设计仅对Chiplet的划分方式进行修改，在固定的硬件配置参数下设置了8种Chiplet架构功能划分和布局的设计范例，如图5所示. 根据分区划分方案建立Chiplet模型，其中Chiplet *A*将Tile和其他部件分别划分为2个独立的Chiplet，Chiplet *B*将Tile分别与Bank和DRAM模块组合形成2个不同的Chiplet. 与此同时还建立了Chiplet *C*和Chiplet *D*这2种由同种Chiplet集成的架构设计，其余4种分区划分的方案旨在探索Chiplet架构划分粒度对PPAC的影响. 构建完成的模型经编译生成对应的寄存器传输级设计文件，分别输入Yosys和Firesim流程中获取的面积、性能和功耗参数，然后估算成本.

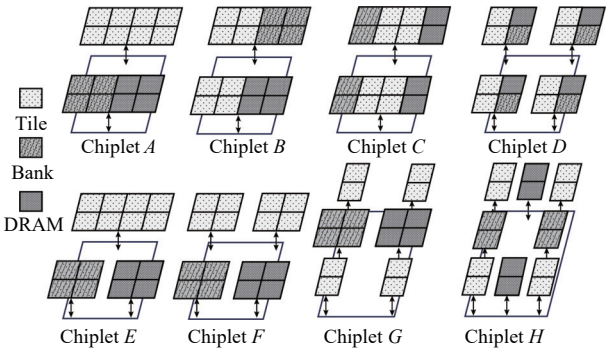


Fig. 5 Illustration of Chiplet partitions

图5 Chiplet分区示意图

更进一步，需要考虑如何确定各功能分区的划分粒度、如何设计Chiplet的模块组成结构. 由于良率模型是一个负指数幂函数[18]，显而易见随着单片的尺寸增加，其制造良率下降的速度会越来越快，极大地阻碍了大规模、大面积的单片SoC进一步提高集成度. 因此需要在完整的设计、模拟和评估框架下衡量Chiplet尺寸对设计方案总成本的显著影响，以快速确认设计方向.

这项实验的目的是探讨SoC划分的经济效益，为此，对SoC的总成本和总的硅面积进行了归一化处理，以更直观地体现Chiplet集成方案与SoC的差异. 如图6所示，基于Chiplet集成的架构设计在总成本上普遍具有优势. 促成这一现象的原因一方面在于开发Chiplet的设计成本由于更广泛的应用性获得了更大程度的摊销，另一方面则是小尺寸Chiplet具有更高的良率从而降低了制造成本. 可以预见的是，随着市场需求(Chiplet生产的规模)增大，设计成本在芯片总成本中的占比会越来越小，工艺、材料和那些不可摊销的成本会变得越来越重要.

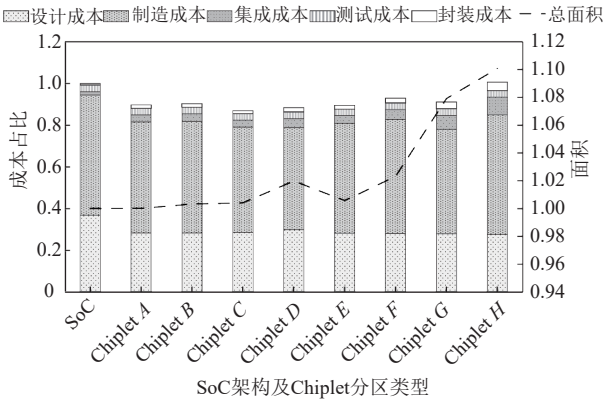


Fig. 6 Difference in the cost of Chiplet partition

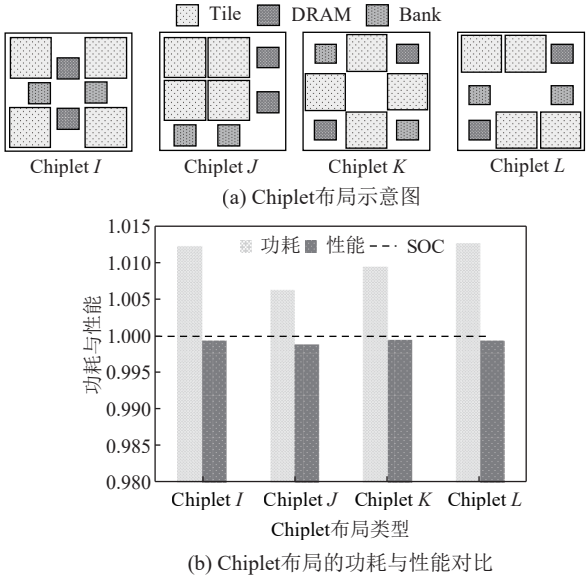
图6 Chiplet划分在成本上的差异

如图6所展示的，Chiplet架构在制造成本上相比于SoC的下降并不明显，这是因为在应用成本模型时，中介层的成本的权重重要性相比其他因素有所提高，以体现先进集成技术的高成本. 与此同时，SoC划分出的Chiplet并非越小越好. 因为分裂为更多数量Chiplet的同时，也引入了更多的D2D模块和更复杂的互连网络. 可以分析得出，更微小的划分会增加集成和封装成本，而总硅面积是增长的，并且当集成(例如有源中介层或3维集成技术)成本进一步提高时，SoC与基于Chiplet集成的设计之间的成本差距就越小. 事实上，如今Chiplet集成设计实现大规模量产面临的关键问题就在于如何降低先进封装的成本，否则基于Chiplet集成的芯片方案可能比SoC更昂贵. 随着时间的推移和技术逐渐成熟，SoC的成本会呈下降趋势，但是集成规模依旧难以提升. 因此，Chiplet集成作为能够在封装内持续提升晶体管集成

数目的有效方法, 随着先进封装技术的发展趋势向好, 基于 Chiplet 集成的芯片架构将会有更好的发展前景.

3.2 Chiplet 架构布局

采用与 3.1 节 Chiplet 分区划分实验相同的硬件配置, 表 3 中的设计参数不做改变, 进行同样的实验流程. 如图 7 所示, 实验建立了 4 种不同的 Chiplet 布局方案. 通过 Constellation 生成固定的 3×3 的 Mesh 拓扑结构, 将 Chiplet 分布到不同的逻辑节点上. 由于 Constellation 是一个高度解耦的规范系统, 物理网络资源规范, 逻辑端点行为和流控制规范相互分开. 路由节点间会自动生成匹配的逻辑规范和流控制, 并且可以确保是无死锁的. 互连网络将和 Chiplet 的设计一起集成生成寄存器传输级硬件代码, 最终映射到 FPGA 上, 这一过程相当于自动完成了 Chiplet 之间的布局布线流程.



(a) Chiplet布局示意图

(b) Chiplet布局的功耗与性能对比

Fig. 7 Illustration of Chiplet layout

图 7 Chiplet 布局示意图

实验在本地 FPGA 上运行了 GAP benchmark suite[59], 以探索面向加速图形处理的 Chiplet 架构. 在图 7 中展示了 Chiplet 架构 4 种布局方式在功耗和性能上的差异, 以相同配置的 SoC 功耗与性能作为参考. 可以发现一个现象: Chiplet 集成的架构在功耗和性能上并不占有优势. 这是因为 Chiplet 间的互连延迟增加了内存访问的压力, 数据处理的速度受到影响, 并且额外的 D2D 通信开销会引入额外的功耗. 之所以 4 种布局的 Chiplet 集成架构在功耗和性能上的表现存在差异, 主要原因在于通信效率. 其中 Chiplet *J* 和 Chiplet *L* 总有一个处理单元存在“远访问”, 尤其对于 Chiplet *J* 中处于最边缘位置的 Chiplet 而言, 访问缓存和内存的代价更高. 而 Chiplet *I* 和 Chiplet *K* 中的运算 Chiplet 总能就近完成访存动作, 但 Chiplet *I* 的布局降低了数据在处理单元之间的传输效率, 因此相比于 Chiplet *K* 产生了更高的功耗.

由于相同的 Chiplet 尺寸, 4 种布局方案在面积上的差异可以忽略, 此时传统的 PPA 指标难以全面评估它们的优劣. 基于 Chiplet 布局的拓扑逻辑复杂性分析布局方式对成本的影响[60], 本文提出的 PPAC 指标计算能够实现对 Chiplet 布局方式进行评估. 必须注意的是, 式(4)中的参数 α, β, γ 的选取对最终的评估结果存在直接影响, 如图 8 所示.

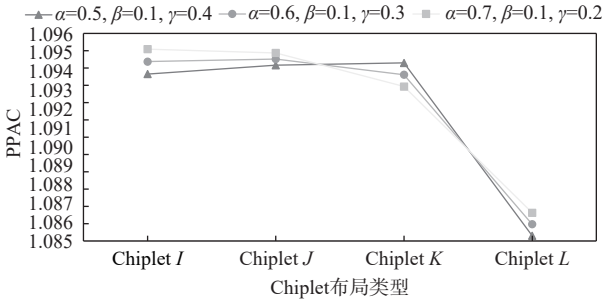


Fig. 8 Influence of PPAC parameter selection on Chiplet layout evaluation

图 8 PPAC 参数选取对 Chiplet 布局评估的影响

Chiplet 的布局规划是芯片设计的关键步骤, 事实上这也是一个极其复杂的问题, 这一过程需要考虑布线密度、传输延迟、信号干扰、散热性等多方面因素. 本文的关注点并不会深入到这些细节中去, 而是进行一个更高维度的、抽象程度更高的探索. Chiplet 放置在中介层上通过 NoC 实现互连接, 通过改变 Chiplet 之间的位置关系, 进一步研究 Chiplet 的逻辑布局对系统整体性能的影响.

3.3 Chiplet 微架构

在微架构层次, Rocket Core 公开了许多参数以支持灵活的定制化设计, 包括浮点管道阶段的数量、缓存大小和转换旁路缓冲器(translation look-aside buffer, TLB)大小等. 实验选取了 11 个具有代表性的设计参数作为输入, 设计空间规模约为 7×10^9, 如表 4 所示. 并且选择了来自于开源测试套件 RISCV-Tests[61] 的 12 种基准测试程序来评估微架构的性能.

实验探索了基于 Rocket 核的 Chiplet 架构设计, 其中不满足设计规则的设计参数数量大约占搜索空间的 85% 以上, 一个合法微架构(8 核规模下)完整的模拟评估时间为 2~3 h, 在 100 组错误数据训练下 ID3 算法对“合法”设计参数的预测准确率能够达到

Table 4 Microarchitecture Design Parameter Space of Rocket Core

表 4 Rocket 核微架构设计参数空间

模块	设计参数	取值
Mul	mulUnroll	1,2,4,6,8,10,12,14,16
	sfmaLatency	1,2,3,4,5,6,7,8
	dfmaLatency	1,2,3,4,5,6,7,8
	nPMPs	1,2,4,6,8, 10,12,14,16
Cache	nSets	2,4,8,16,32
	nWays	1,2,4,6,8,10,12,14,16
	nTLBSets	1,2,4,6,8,10,12,14,16
	nTLBWays	2,4,8,16,32
	nMSHRs	1,2,3,4,5,6,7,8,9
BTB	nEntires	4,8,12,16,24,28
	nPages	1,2,4,6,8,10,12,14,16

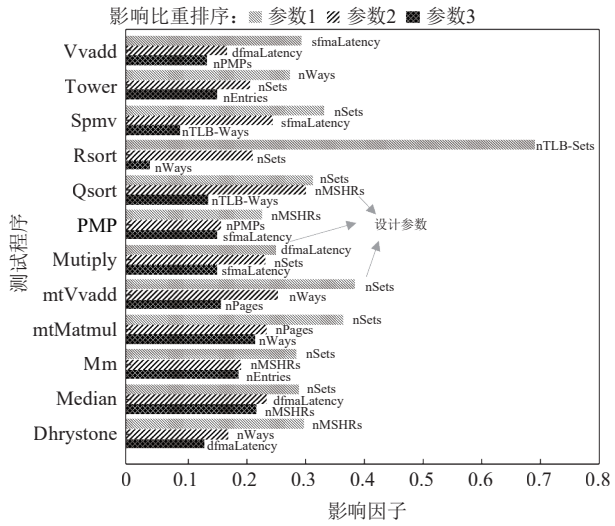


Fig. 9 Impact of design parameters on the performance of the test program

图 9 设计参数对测试程序性能的影响

95% 以上. 另一方面, 通过 ID3 决策树模型分析, 可以获取到每一种设计参数对于各类测试程序的影响程度. 每一种设计参数在不同类型的测试程序下对芯片性能产生的影响是不同的, 图 9 展示了对于每种测试程序性能影响最大的 3 个设计参数.

处理器核的微架构中物理内存保护单元(physical memory protection unit, PMP)的数目对于 RISCV-Tests 下的 PMP-benchmark 测试程序的性能而言具有特殊的影响. 缺失状态保持寄存器(miss-status handling registers, MSHR)是用于处理缓存失效(cache miss)的功能单元, 记录未完成的访存请求, 展现了与运算性能显著的关联度. 建议在设计开始之前首先基于先验知识确定设计目标以及那些未知的和感兴趣的芯片参数, 这将有效减少设计空间规模以及建模的复杂度, 当然更大规模的设计空间探索在本文提出的 FireLink 下是可行的.

表 5 展示了一种局部最优的设计方案, Chiplet 架构自动 DSE 的结果与工程师手动设计的 Rocket 核集成 SoC 相比, 除成本优势外, 在 PPA 上并没有明显的优势. 这是由于设置了较高的互连“惩罚”, 突出 Chiplet 架构对先进封装技术的依赖以及相应的高昂代价. 但是以相同的成本为约束时, Chiplet 架构相对于 SoC 在成本上的冗余可以被投入在设计中, 这意味着设计者会拥有更高的设计权自由度和更广阔的优化空间.

Table 5 Chiplet Architecture Design Schemes and Evaluation Results

表 5 Chiplet 架构设计方案及评估结果

架构设计	目标	参数选择		
配置	布局类型	Chiplet *E*		
	网络微架构	4,4:1,4,80,64,128, Mesh		
	核微架构	4,2,4,4,16,4,4,4,2,28,8		
评估	性能	功耗	面积	成本
	×0.93	×1.03	×1.13	×0.86

Chiplet 架构基于一种自上而下的设计方法进行探索, 首先需要确定 Chiplet 架构在多 Chiplet 集成层次的策略, 基于先验知识和仿真探索其分区划分方式以及 Chiplet 间互连结构. 并且考虑各设计参数对不同类型应用程序的影响, 能够倾向性地提高设计架构面向某一特定领域的性能.

4 结 论

目前集成电路发展进入“后摩尔”时代, 芯片设计制造面临着收益困境, Chiplet 集成技术开始受到广泛关注. 面向新的处理器体系结构, 传统的研究工具以及量化分析方法并不适用. 本文在许多不同的开源设计和开发工具之间进行了权衡与抉择, 并通过控制版本、确保环境一致性和构建自动化部署脚本实现了不同工具间更高水平的协调和同步, 从而创建一个完整的设计、模拟与评估全流程的设计空间探索框架——FireLink. 在该框架下探索了将单片 SoC 划分 Chiplet 的方法、集成这些 Chiplet 的布局方案以及单个 Chiplet 的微架构设计, 并且提出了 PPAC 用于探索 PPA 与芯片成本的权衡. 基于敏捷设计与

机器学习方法, FireLink 提升了面向 Chiplet 架构设计的设计空间探索效率, 为 Chiplet 体系结构提供辅助设计, 能够有效解决芯片设计门槛高、周期长的现实难题.

作者贡献声明: 李开为论文主要完成人; 曾坤指导实验; 荣培涛、陈志强、张甜辅助完成实验并修改论文; 王永文提出了指导意见.

参 考 文 献

[1] Wang Zhanguo. Discussion on microelectronics technology in the post-Moore era[C/OL]//Proc of the 14th National Conf on Compound Semiconductor Materials, Microwave Devices and Optoelectronic Devices. Beijing: Chinese Institute of Electronics, 2006[2024-02-01]. https://d.wanfangdata.com.cn/conference/6370041 (in Chinese)
(王占国. 后摩尔时代的微电子技术探讨[C/OL]//第十四届全国化合物半导体材料、微波器件和光电器件学术会议论文集. 北京: 中国电子学会, 2006[2024-02-01]. https://d.wanfangdata.com.cn/conference/6370041)

[2] DeBenedictis E P. It's time to redefine Moore's law again[J]. Computer, 2017, 50(2): 72−75

[3] Munoz R. Furthering Moore's law integration benefits in the Chiplet era[J]. IEEE Design & Test, 2024, 41(1): 81−90

[4] Psaila N, Nekkanty S, Shia D, et al. Detachable optical Chiplet connector for co-packaged photonics[J]. Journal of Lightwave Technology, 2023, 41(19): 6315−6323

[5] Sharma D D, Pasdast G, Qian Zhiguo, et al. Universal Chiplet interconnect express (UCIe): An open industry standard for innovations with Chiplets at package level[J]. IEEE Transactions on Components, Packaging and Manufacturing Technology, 2022, 12(9): 1423−1431

[6] Yoshida J. AMD, TSMC & IMEC show their Chiplet playbooks at ISSCC[EB/OL]. 2021[2024-02-01]. https://www.eetimes.com/amd-tsmc-imec-show-their-chiplet-playbooks-at-isscc/

[7] Gaitonde D. AMD next-generation FPGA built from Chiplets [C/OL]//Proc of the 35th IEEE Hot Chips Symp (HCS). Piscataway, NJ: IEEE, 2023 [2024-02-01]. https://doi.org/10.1109/HCS59251.2023.10254722s

[8] Suggs D, Subramony M, Bouvier D. The AMD "Zen 2" processor[J]. IEEE Micro, 2020, 40(2): 45−52

[9] Liu Xinyang. Thoughts on the development and innovation trend of 5G semiconductor industry[J]. ZTE Technology Journal, 2021, 27(4): 51−52 (in Chinese)
(刘新阳. 5G 半导体产业发展和创新趋势思考[J]. 中兴通讯技术, 2021, 27(4): 51−52)

[10] Kenji S, Zhang Xiaobo. Semiconductor optical amplifier and gain chip used in wavelength tunable lasers[J]. ZTE Communications, 2021, 19(3): 81−87

[11] Cai Jifan, Xu Zengyi, Chi Nan. Visible light communication constellation shaping and artificial intelligence demodulation technology[J]. ZTE Technology Journal, 2022, 28(4): 19−24 (in Chinese)
(蔡济帆, 徐增熠, 迟楠. 可见光通信星座整形与人工智能解调技术[J]. 中兴通讯技术, 2022, 28(4): 19−24)

[12] Zhang Chenchen, Zhang Nan, Cao Wei, et al. AI-based optimization of handover strategy in non-terrestrial networks[J]. ZTE Communications, 2021, 19(4): 98−104

[13] Binkert N, Beckmann B, Black G, et al. The Gem5 simulator[J/OL]. ACM SIGARCH Computer Architecture News, 2011[2024-02-01]. https://doi.org/10.1145/2024716.2024718

[14] Bakhoda A, Yuan G L, Fung W W L, et al. Analyzing CUDA workloads using a detailed GPU simulator[C]//Proc of the IEEE Int Symp on Performance Analysis of Systems and Software. Piscataway, NJ: IEEE, 2009: 163−174

[15] Carlson T E, Heirman W, Eeckhout L. Sniper: Exploring the level of abstraction for scalable and accurate parallel multi-core simulation [C/OL]//Proc of the Int Conf for High Performance Computing, Networking, Storage and Analysis (SC'11). Piscataway, NJ: IEEE, 2021 [2024-02-01]. https://doi.org/10.1145/2063384.2063454

[16] Xu Yinan, Yu Zihao, Tang Dan, et al. Towards developing high performance RISC-V processors using agile methodology[J]. IEEE Micro, 2023, 43(4): 1−9

[17] Xu Changyu. Chiplet and new microarchitecture approach Intel's new Xeon processor[J]. Micro Computers, 2023(18): 93−97 (in Chinese)
(徐昌宇. Chiplet 和全新微架构走近英特尔全新至强处理器[J]. 微型计算机, 2023(18): 93−97)

[18] Feng Yinxiao, Ma Kaisheng. Chiplet Actuary: A quantitative cost model and multi-Chiplet architecture exploration[C]//Proc of the 59th ACM/IEEE Design Automation Conf (DAC '22). New York: ACM, 2022: 121−126

[19] Ahmad M, DeLaCruz J, Ramamurthy A. Heterogeneous integration of Chiplets: Cost and yield tradeoff analysis[C/OL]//Proc of the 23rd Int Conf on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE). Piscataway, NJ: IEEE, 2022 [2024-02-01]. https://doi.org/10.1109/EuroSimE54907.2022.9758914

[20] Kim J, Murali G, Park H, et al. Architecture, chip, and package codesign flow for interposer-based 2.5-D Chiplet integration enabling heterogeneous IP reuse[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2020, 28(11): 2424−2437

[21] Kabir M A, Peng Yarui. Chiplet-package co-design for 2.5 D systems using standard ASIC CAD tools[C]//Proc of the 25th Asia and South Pacific Design Automation Conf (ASP-DAC). Piscataway, NJ: IEEE, 2020: 351−356

[22] Zhang Gianlong, Hou Rui, Yang Sibo, et al. The role of architecture simulators in processor design[J]. Journal of Computer Research and Development, 2019, 56(12): 2702−2719 (in Chinese)
(张乾龙, 侯锐, 杨思博, 等. 体系结构模拟器在处理器设计过程中的作用[J]. 计算机研究与发展, 2019, 56(12): 2702−2719)

[23] Zhou Juncheng. Research on joint simulation technology for CPU

cores [D]. Wuhan: Huazhong University of Science and Technology, 2022 (in Chinese)
(周俊成. 面向 CPUChiplet 的联合模拟技术研究[D]. 武汉：华中科技大学, 2022)

[24] Zhi Haocong, Xu Xianuo, Han Weijian, et al. A methodology for simulating multi-Chiplet systems using open-source simulators [C/OL]//Proc of the 8th Annual ACM Int Conf on Nanoscale Computing and Communication (NANOCOM'21). New York: ACM, 2021[2024-02-01]. https://doi.org/10.1145/3477206.3477459

[25] Bachrach J, Vo H, Richards B, et al. Chisel: Constructing hardware in a scala embedded language[C]//Proc of the Design Automation Conf. New York: ACM, 2012: 1212−1221

[26] Bharadwaj S, Yin Jieming, Beckmann B, et al. Kite: A family of heterogeneous interposer topologies enabled via accurate interconnect modeling[C/OL]//Proc of the 57th ACM/EDAC/IEEE Design Automation Conf (DAC'20). Piscataway, NJ: IEEE, 2020[2024-02-01]. https://doi.org/10.1109/DAC18072.2020.9218539

[27] Li Fuping, Wang Ying, Cheng Yuanqing, et al. Gia: A reusable general interposer architecture for agile Chiplet integration [C/OL]//Proc of the IEEE/ACM Int Conf on Computer Aided Design (ICCAD). Piscataway, NJ: IEEE, 2022[2024-02-01]. https://ieeexplore.ieee.org/document/10069926

[28] Wang Mengdi, Wang Ying, Liu Cheng, et al. Network-on-interposer design for agile neural-network processor chip customization [C]//Proc of the 58th ACM/IEEE Design Automation Conf (DAC). Piscataway, NJ: IEEE, 2021: 49−54

[29] Yin Jieming, Lin Zhifeng, Kayiran O, et al. Modular routing design for Chiplet-based system[C]//Proc of the 45th ACM/IEEE Annual Int Symp on Computer Architecture (ISCA). Piscataway, NJ: IEEE, 2018: 726−738

[30] Wang Chenghan, Xu Qinzhi, Nie Chuanjun, et al. A multiscale anisotropic thermal model of Chiplet heterogeneous integration system[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2024, 32(1): 1−12

[31] Liu Kaiyuan, Qin Houyun, Guo Jiarui, et al. Stress and fatigue life studies of solder joints in an advanced packaging with Chiplet [C]//Proc of the Int Conf on Mechatronics, IoT and Industrial Informatics (ICMIII). Piscataway, NJ: IEEE, 2023:187−191

[32] Zhang Zhong, Qiao Xinyu, Long Xinjiang, et al. Research on thermal stress of 2.5D Chiplet package structure[J]. Journal of Sensing Technology, 2023, 36(7): 1024−1031 (in Chinese)
(张中, 乔新宇, 龙欣江, 等. 2.5D Chiplet 封装结构的热应力研究[J]. 传感技术学报, 2023, 36(7): 1024−1031)

[33] Pal S, Petrisko D, Kumar R, et al. Design space exploration for Chiplet-assembly-based processors[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2020, 28(4): 1062−1073

[34] Wang Mengdi, Wang Ying, Liu Cheng, et al. Puzzle: A scalable framework for deep learning integrated chips[J]. Journal of Computer Research and Development, 2023, 60(6): 1216−1231(in Chinses)
(王梦迪, 王颖, 刘成, 等. Puzzle: 面向深度学习集成芯片的可扩展框架[J]. 计算机研究与发展, 2023, 60(6): 1216−1231)

[35] Ehrett P, Austin T, Bertacco V. Chopin: Composing cost-effective custom chips with algorithmic Chiplet[C]//Proc of the 39th IEEE Int Conf on Computer Design (ICCD). Piscataway, NJ: IEEE, 2021: 395−399

[36] Lan Jingjing, Nambiar V P, Sabapathy R, et al. An automatic chip-package co-design flow for multi-core neuromorphic computing SiPs[C]//Proc of the 22nd IEEE Electronics Packaging Technology Conf (EPTC). Piscataway, NJ: IEEE, 2020: 77−80

[37] Asanović K, Avižienis R, Bachrach J, et al. The rocket chip generator[R/OL]. 2016[2024-02-01]. http://www.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-17.html

[38] Zhao J, Agrawal A, Nikolic B, et al. Constellation: An open-source SoC-capable NoC generator[C/OL]//Proc of the 15th IEEE/ACM Int Workshop on Network on Chip Architectures (NoCArc). Piscataway, NJ: IEEE, 2022 [2024-02-01]. https://doi.org/10.1109/NoCArc57472.2022.9911299

[39] Snyder W, Wasson P, Galbi D, et al. Verilator[CP/OL]. [2024-02-01]. https://github.com/verilator/verilator

[40] Gonzalez A, Zhao J, Snyder T, et al. riscv-tools[CP/OL]. [2024-02-01]. https://github.com/ucb-bar/riscv-tools-feedstock

[41] Karandikar S, Mao H, Kim D, et al. FireSim: FPGA-accelerated cycle-exact scale-out system simulation in the public cloud[J]. IEEE Micro, 2018, 39(3): 29−42

[42] Pemberton N, Amid A. FireMarshal: Making HW/SW co-design reproducible and reliable[C]//Proc of the IEEE Int Symp on Performance Analysis of Systems and Software (ISPASS). Piscataway, NJ: IEEE,2021: 299−309

[43] Amid A, Biancolin D, Gonzalez A, et al. Chipyard: Integrated design, simulation, and implementation framework for custom SoCs[J]. IEEE Micro, 2020, 40(4): 10−21

[44] Shah D, Hung E, Wolf C, et al. Yosys+nextpnr: An open source framework from verilog to bitstream for commercial FPGAs [C/OL]//Proc of the 27th IEEE Annual Int Symp on Field-Programmable Custom Computing Machines (FCCM). Piscataway, NJ: IEEE, 2019[2024-02-01]. https://doi.org/10.1109/FCCM.2019.00010

[45] Chen Zhiqiang, Zhou Hongwei, Feng Quanyou, et al. Design and implementation of configurable cache coherence protocol for multi-core processors[J]. Journal of Computer Research and Development, 2021, 58(6): 1166−1175 (in Chinese)
(陈志强, 周宏伟, 冯权友, 等. 面向多核处理器的可配置缓存一致性协议设计与实现[J]. 计算机研究与发展, 2021, 58(6): 1166−1175)

[46] Kuang Huizhen, Cao Xianfeng, Li Jingyuan, et al. HGBO-DSE: Hierarchical GNN and Bayesian optimization based HLS design space exploration[C]// Proc of the Int Conf on Field Programmable Technology (ICFPT). Piscataway, NJ: IEEE, 2023: 106−114

[47] Bai Chen, Sun Qi, Zhai Jianwang, et al. BOOM-Explorer: RISC-V BOOM microarchitecture design space exploration framework [C/OL]//Proc of the IEEE/ACM Int Conf on Computer Aided Design (ICCAD). Piscataway, NJ: IEEE, 2021[2024-02-01]. https://doi.org/10.1109/ICCAD51958.2021.9643455

[48] Ye Hanchen, Jun H, Jeong H, et al. ScaleHLS: A scalable high-level synthesis framework with multi-level transformations and optimizations: invited[C]//Proc of the 59th ACM/IEEE Design Automation Conf (DAC'22). New York: ACM, 2022: 1355–1358

[49] Sohrabizadeh A, Yu C H, Gao Min, et al. AutoDSE: Enabling software programmers design efficient FPGA accelerators[J]. ACM Transactions on Design Automation of Electronic Systems, 2022, 27(4): 1−27

[50] Chen Shixin, Zheng Su, Bai Chen, et al. SoC-Tuner: An importance-guided exploration framework for DNN-targeting SoC design[C]// Proc of the 29th Asia and South Pacific Design Automation Conf (ASP-DAC). Piscataway, NJ: IEEE, 2024: 207−212

[51] Minutoli M, Castellana V G, Tan Cheng, et al. SODA: A new synthesis infrastructure for agile hardware design of machine learning accelerators[C/OL]//Proc of the IEEE/ACM Int Conf on Computer Aided Design (ICCAD). Piscataway, NJ: IEEE, 2020[2024-02-01]. https://doi.org/10.1145/3400302.3415781

[52] Deshwal A, Jayakodi A K, Joardar B K, et al. MOOS: A multi-objective design space exploration and optimization framework for NoC enabled manycore systems[J]. ACM Transactions on Embedded Computing Systems, 2019, 18(5): 1−23

[53] Chen Guilin, Wang Guanwu, Hu Jian, et al. Survey on Chiplet packaging structure and communication structure[J]. Journal of Computer Research and Development, 2022, 59(1): 22−30 (in Chinese)
(陈桂林, 王观武, 胡健, 等. Chiplet 封装结构与通信结构综述[J]. 计算机研究与发展, 2022, 59(1): 22−30)

[54] Lou Mian, Zhang Haijin, Yang Liang, et al. RISC-V processor design technology based on Chisel language[J]. Microelectronics and Computers, 2021, 38(3): 51−55 (in Chinses)
(娄冕, 张海金, 杨靓, 等. 基于 Chisel 语言的 RISC-V 处理器设计技术[J]. 微电子学与计算机, 2021, 38(3): 51−55)

[55] Xing Zuocheng, Ma Anguo, Zhang Chengyi, et al. SoC architecture-level power simulation and optimization techniques[C]//Proc of the 13th Computer Engineering and Technology Conf (NCCET'09). Beijing: CCF, 2009: 161−166 (in Chinses)
(邢座程, 马安国, 张承义, 等. SoC 体系结构级功耗模拟与优化技术[C]// 第十三届计算机工程与工艺会议论文集(NCCET'09). 北京: 中国计算机学会, 2009: 161−166)

[56] Cadence. 3D-IC design solutions[EB/OL]. [2024-02-01]. https://www.cadence.com/en_US/home/solutions/3dic-design-solutions.html

[57] Synopsys. 3D-IC compiler[EB/OL]. [2024-02-01]. https://www.synopsys.com/implementation-and-signoff/3dic-design.html

[58] Quinlan J R. Induction of decision trees[J]. Machine Learning, 1986, 1(1): 81−106

[59] Beamer S, Asanović K, Patterson D. The GAP benchmark suite [CP/OL]. [2024-02-01]. https://github.com/sbeamer/gapbs

[60] Kurihara T, Li Yamin. A cost and performance analytical model for large-scale on-chip interconnection networks[C]//Proc of the 4th Int Symp on Computing and Networking (CANDAR). Piscataway, NJ: IEEE, 2016: 447−450

[61] UC Berkeley. riscv-tests[CP/OL]. [2024-02-01]. https://github.com/riscv-software-src/riscv-tests



Li Kai, born in 2000. PhD candidate. His main research interests include microprocessor architecture design and chip design space exploration.

李 开, 2000 年生. 博士研究生. 主要研究方向为微处理器体系结构设计、芯片设计空间探索.



Zeng Kun, born in 1983. PhD, associate professor. Member of CCF. His main research interests include microprocessor architecture and parallel computing.

曾 坤, 1983 年生. 博士, 副研究员. CCF 会员. 主要研究方向为微处理器体系结构、并行计算.



Rong Peitao, born in 2000. Master candidate. His main research interests include microprocessor architecture design and FPGA accelerated simulation.

荣培涛, 2000 年生. 硕士研究生. 主要研究方向为微处理器体系结构设计、FPGA 加速仿真.



Chen Zhiqiang, born in 1998. PhD candidate. His main research interests include microprocessor architecture design and network on chip.

陈志强, 1998 年生. 博士研究生. 主要研究方向为微处理器体系结构设计、片上网络.



Zhang Tian, born in 1999. Master candidate. Her main research interest includes general sparse matrix multiplication.

张 甜, 1999 年生. 硕士研究生. 主要研究方向为通用稀疏矩阵乘法.



Wang Yongwen, born in 1977. PhD, professor. Senior member of CCF. His main research interests include processor microarchitecture, and high performance computing.

王永文, 1977 年生. 博士, 教授. CCF 高级会员. 主要研究方向为处理器微体系结构、高性能计算.