计算机研究与发展
Journal of Computer Research and Development
https://crad.ict.ac.cn
62(6): 1492−1512, 2025

映天湖:晶圆级通用异构多芯粒千万亿次计算机

董文阔[1] 殷春锁[1] 张志锰[1] 王鹏超[3] 沙 江[3] 王梦雅[2] 朱旻琦[2] 刘宏伟[1] 刘宇航[1] 郝沁汾[1]
[1](中国科学院计算技术研究所 北京 100190)
[2](中国电子科技集团公司第五十八研究所 江苏无锡 214035)
[3](无锡芯光互连技术研究院有限公司 江苏无锡 214131)
(dongwenkuo23@mails.ucas.ac.cn)

Yingtian-Lake: A Wafer-Scale General-Purpose Heterogeneous Multi-chiplet Petascale Computer

Dong Wenkuo[1], Yin Chunsuo[1], Zhang Zhimeng[1], Wang Pengchao[3], Sha Jiang[3], Wang Mengya[2], Zhu Minqi[2], Liu Hongwei[1], Liu Yuhang[1], and Hao Qinfen[1]
[1](*Institute of Computing Technology*, *Chinese Academy of Sciences*, *Beijing* 100190)
[2](*58th Research Institute*, *China Electronics Technology Group Corporation*, *Wuxi*, *Jiangsu*, 214035)
[3](*Wuxi Institute of Interconnect Technology*, *Wuxi*, *Jiangsu*, 214131)

Abstract Wafer-scale computer integrates multiple chiplets through advanced packaging technologies, overcoming traditional chip area limitations to achieve computational power scaling. However, existing domain-specific designs struggle to meet generalized computing requirements. In this study, we propose Yingtian-Lake, which is a wafer-scale general-purpose computer targeting workload characteristics of high-performance computing and intelligent computing scenarios. First, a decoupled computing module-interposer architecture design with standardized I/O interfaces enables multi-modal computing module compatibility. Second, a reconfigurable wafer-scale network employing dynamic topology adaptation technology accommodates diverse traffic patterns. Third, a fault-aware tolerant routing algorithm ensures service continuity during computing unit failures. Experimental results demonstrate that the proposed reconfigurable network achieves second-level topology switching latency. The prototyped 16-module system fabricated with TSMC 28 nm process shows 1.45 times and 1.78 times energy efficiency improvements in high-performance linear algebra computations and deep learning inference tasks respectively, while delivering petaflops-level performance on a single wafer. This breakthrough architecture validates the technical feasibility of universal wafer-scale systems, establishing a scalable hardware foundation for next-generation heterogeneous computing platforms.

Key words wafer-scale computer; high-performance computing; intelligent computing; standardized I/O design; reconfigurable wafer-scale network

摘 要 晶圆级计算机通过先进封装技术集成多芯粒,突破传统芯片面积限制实现算力扩展,但现存方案因领域专用化设计难以满足通用计算需求.面向高性能计算与智能计算场景的负载特征,提出一种新型通用化晶圆级系统架构——映天湖.首先通过解耦式计算模组-互连基板架构设计,结合标准化I/O接口

收稿日期: 2025-03-01; **修回日期**: 2025-04-14
基金项目: 国家重点研发计划项目(2022YFB4401501); 江苏省创新支撑计划(软科学研究)专项(BE2023006-4)
This work was supported by the National Key Research and Development Program of China (2022YFB4401501) and the Jiangsu Provincial Special Program for Innovation Support (Soft Science Research) (BE2023006-4).
通信作者: 郝沁汾(haoqinfen@ict.ac.cn)

支持多种计算模组；其次构建可重构晶上网络，采用动态拓扑重构技术适配不同业务流量模式；继而开发拓扑无关的容错控制，保障计算单元失效时的服务持续性. 实验结果表明，所设计的可重构晶上网络可实现秒级拓扑切换时延. 基于 TSMC 28 nm 工艺成功流片验证的 16 个计算模组的原型系统，在高性能线性代数计算任务中展现了约 1.45 倍的吞吐量提升，在深度学习推理任务中则展现约 1.78 倍的时延性能提升，单晶圆可实现千万亿次性能，证实该架构在实现晶圆级系统通用化方面的技术突破，为下一代异构计算平台提供了可扩展的硬件基础架构.

关键词 晶圆级计算机；高性能计算；智能计算；标准化 I/O 设计；可重构晶上网络

中图法分类号 TP302.2

DOI: 10.7544/issn1000-1239.202550163 **CSTR:** 32373.14.issn1000-1239.202550163

集成电路的计算能力可表征为晶体管总数，其由晶体管密度与芯片裸片面积的乘积决定. 这一双重参数依赖关系在当代半导体微缩中面临双重困境：晶体管密度受限于制造工艺的物理极限，在 3 nm 技术节点后，通过持续提升密度延续摩尔定律已愈发困难[1,2]；芯片裸片面积则面临光刻机曝光场尺寸的物理限制，以及因裸片面积增大导致良率下降引发的成本指数增长问题[3]. 这种工艺微缩效益递减与成本-面积平方关系的叠加效应，共同导致传统芯片扩展范式的关键瓶颈.

芯粒(chiplet)技术与先进封装体系正在重塑半导体产业格局. 芯粒是指一种具有完整功能并单独完成设计和流片的未封装芯片，用于和其他同类芯片通过先进封装技术集成后完成更完整、更强的功能. 芯粒技术通过将复杂芯片分解为独立功能模块，结合异构集成封装方案，突破传统单芯片架构的物理极限[4]. 在此技术框架下，先进封装已从单纯的芯片保护层演进为系统级互连平台，通过硅中介层、微凸块等创新结构实现亚微米级互连精度. 先进封装是指区别于传统引线类框架封装和单芯片封装的新型封装技术，其特点主要是采用新的互连材料和封装概念，如台积电的 CoWoS 工艺[5]，采用了硅转接板以提升芯粒间的互连密度，被称为 2.5D 封装技术[6]. 在先进封装技术快速发展的过程中，出现了一类晶圆级封装技术，如台积电的 InfoSOW[7-9]，其主要特点是不同于 CoWoS 等工艺来封装多个芯粒，晶圆级封装技术用于封装实现包含了几十到几百个芯粒的晶圆级计算机(wafer-scale computer)，这为通过提升芯片总面积从而继续提高计算能力指明了一条新的道路.

晶圆级计算机是基于晶圆级封装技术[10-11]构建的新型计算架构，其核心技术特征体现为：通过高密度互连架构与异质集成工艺将数百个功能化芯粒进行晶圆级系统集成，同时整合高能效供电系统、主动式散热方案等工程化设计，并依托定制化系统级软件栈协同驱动运行. 相较于传统单芯片架构，该系统通过突破芯片面积与互连带宽的双重限制，可实现数量级跃升的计算能力，为解决后摩尔时代单芯片性能提升瓶颈问题提供了突破性技术路径. 该架构的规模化实现不仅标志着集成电路制造与系统集成技术的跨越，更可能重构未来超算和/或智算系统的设计范式.

截止目前，全球学术界与产业界已涌现出若干具有代表性的晶圆级计算机原型，其技术特征集中体现在异构架构设计、晶上互连网络拓扑优化、高密度系统级封装方案以及能效热管理协同设计等关键维度，并已取得阶段性技术突破[10-13]. 典型系统通过创新性架构设计，如在硅中介层实现光电子混合互连网络以及采用分布式电压调节模块优化供电效率等，初步验证了晶圆级系统的工程可行性.

然而，由于该领域仍处于技术探索期，现有系统在技术成熟度与工程适用性方面仍存在显著瓶颈. 首先，面向特定计算领域(如智能计算、科学计算)的专用架构设计导致系统通用性受限；其次，晶上互连网络缺乏动态重构与容错机制，难以满足大规模并行计算可靠性需求；再者，当前晶圆级封装工艺在材料热膨胀系数匹配、微凸点高密度键合等环节仍存在技术瓶颈；此外，千瓦级功率密度引发的三维供电网络损耗与非线性热梯度分布问题，制约系统稳定运行. 这些多维度的技术难题亟待通过跨学科协同创新实现重点突破，进而推动晶圆级计算机从实验室原型向产业化应用转化.

1 相关工作

晶圆级计算机作为后摩尔时代新兴计算架构范式，凭借其异构集成潜力与系统级能效优化优势，已

成为高性能计算和智能计算领域的前沿研究方向[14], 兼具理论研究价值与产业战略意义. 当前技术路线呈现多元化演进特征: 在架构层面, 基于硅基光电子融合的异构集成架构、采用分布式内存层次的光互连方案, 以及面向存算一体的三维堆叠设计等差异化技术路线相继涌现[15]; 在工程实现维度, 不同系统在芯粒粒度划分、晶上互连拓扑优化、封装热机械应力控制等关键技术环节形成特色化解决方案. 以下系统梳理当前晶圆级计算机的架构与核心技术特点, 并剖析其面临的跨尺度协同设计、动态容错机制、大规模制造一致性等核心挑战, 以期为“映天湖”(Yingtian-Lake)设计提供技术路径参考和对比.

Cerebras WSE 系列晶圆级计算机通过不断迭代制程、增加晶体管数量和核心数量, 以及提升内存容量和带宽, 实现了显著的性能提升, 尤其在 AI 模型训练和推理的性能方面, 3 代 Cerebras WSE 的技术特点和性能参数如表 1 所示.

Cerebras WSE 的 AI 核心采用只支持乘加运算的极简核心设计, 使整个芯片的功能保持在较低的水平. 据推测, WSE[3] 系统采用了光罩拼接的封装方式以实现晶圆上所有的芯片互连, 由于缺少裸芯片的封测环节, 制造过程中的缺陷导致良率降低, 为此 Cerebras 专门设计了冗余核心与容错路由等措施以避开这个问题, 极简核心的设计在这方面也起到了一定作用. WSE 系统的另外一个特点是, 在软件方面, Cerebras 支持 TensorFlow 和 PyTorch 等流行智能计算编程框架, 通过其开发的 CGC(Cerebras graph compiler)编译软件将模型自动编译为优化的可执行文件并被映射到晶圆级计算机的计算资源中. 然而, 由于该系统设计采用了专用核心设计, 只能用于智能计算和某些领域的高性能计算应用[22-23], 限制了其在更广泛领域的应用.

Table 1 Comparison of the Key Specifications of Three Generations of Wafer-Scale Computer in Cerebras Company

表 1 Cerebras 公司 3 代晶圆级计算机主要技术规格对比

规格参数	WSE-1[16]	WSE-2[17-20]	WSE-3[21]
工艺制程/nm	16	7	5
晶体管数目	1.2 万亿	2.6 万亿	4 万亿
面积/mm^2	46 225	46 225	46 225
核心数目	400 000	850 000	900 000
系统性能/PF	9	75	125
系统功耗/kW	15	23	27
片上存储/GB	18	40	44
存储带宽/PBps	9	20	21
I/O 带宽/Tbps	1.2	1.2	1.2
片上网络带宽/Pbps	100	220	214
系统尺寸/RU	16	15	15
封装方式	光罩拼接	光罩拼接	光罩拼接
散热方式	液冷	液冷	液冷
供电方式	垂直	垂直	垂直

Tesla Dojo 系统[24-25] 的核心芯片 D1 集成了 354 个基于定制指令集开发的 RISC-V 核心, 可支持 BF16、CFP8、FP32 等多种不同的计算精度. 与 Cerebras WSE 系统不同的地方在于, Tesla 的 Dojo 系统采用了台积电的晶圆级集成扇出封装技术[23-24], 芯片流片并切割后通过晶圆级封装工艺进行重构, 将 25 个 D1 芯片集成为基本硬件单元, 12 个基本硬件单元再组成一个机柜, 10 个机柜进一步组成 ExaPOD 计算集群. 特斯拉 Dojo 系统的各个性能参数如表 2 所示, 这里不再赘述. 由于特斯拉 Dojo 系统采用晶圆级集成扇出封装技术进行基本硬件单元封装, 因此可以基于测试后的 D1 芯片进行集成, 这些芯片在制造过程中经过了严格的测试和筛选, 进一步提升了 D1 芯片的产出良率, 使特斯拉 Dojo 系统不需要太多考虑晶上网络的容错设计问题. 在软件支持层面, 特斯拉 Dojo 系统提供了分布式通信和同步机制, 包括计数信号量和屏障等, 在这点上更接近于通用系统, 如基于 MPC 编程模式的传统高性能计算系统. 然而, D1 芯片的设计同样高度定制化, 主要针对特斯拉自动驾驶模型的特定需求, 这使得其在通用计算场景中的适用性有限.

美国加利福尼亚大学洛杉矶分校(UCLA)的研究团队设计了一款晶圆级处理器系统[26-27], 与 Cerebras 公司的 WSE 系统及特斯拉公司的 Dojo 系统均不同, 在晶圆级计算机内部的芯粒之间互连采用了低成本的硅互连基板(silicon interconnect fabric, Si-IF)[26] 技术, 避免使用昂贵的光罩拼接或晶圆级集成扇出封装技

Table 2 Main Technical Specifications of Tesla Dojo Wafer-Scale Computer

表 2 特斯拉 Dojo 晶圆级计算机主要技术规格

规格参数	D1 芯片	最小硬件单元	单机柜
核心数目	354	8 850	106 200
片上存储	440 MB	11 GB	132 GB
计算性能	362 TFLOPS	9 PFLOPS	108 PFLOPS
读带宽	138 TBps	3.4 PBps	40.5 PBps
写带宽	93 TBps	2.3 PBps	27.3 PBps
功耗	600 W	15 kW	180 kW

术，通过 10 μm 间距的铜柱实现芯粒间 400 线/mm 超高密度互连，并通过片内交叉开关和晶圆级 Mesh 网络实现通信. 由于将 TSV 通孔集成到 Si-IF 晶圆上的技术仍在开发中，为解决晶圆级供电与散热难题，系统采用边缘供电和风冷散热方式，在一定程度上会影响系统的整体性能和可靠性，在大规模部署时可能需要更高效的供电和散热解决方案. 与 Cerebras 的 WSE 和 Tesla Dojo 系统不同的另一个方面是 UCLA 设计的晶圆级计算机既可以支持 CPU[26]，也可以支持 GPU[27]，适用于多种应用场景，这是世界上第一个面向通用应用场景设计的晶圆级计算系机. 然而，目前尚未看到 UCLA 设计的晶圆级计算机的产品发布.

在我国，信息工程大学与之江实验室的软件定义晶上系统（SDSoW）采用热压键合等晶圆级封装技术[28]，通过软件定义互连和软件定义节点[29]，实现了硬件资源的动态配置和灵活管理. SDSoW 的核心理念是通过软件定义的方式，打破传统硬件架构的限制，使系统能够根据不同的应用场景和需求，灵活地调整硬件资源的配置和管理方式. 然而，设计高度依赖软件定义技术，这意味着硬件和软件的开发之间需要高度的协同，增加依赖性，对硬件的实现技术挑战较大.

“之江大芯片”是中国科学院计算技术研究所和之江实验室研制的一款基于芯粒的晶圆级处理器[30]，旨在满足现代高度并行工作负载对大规模核心数量和内存带宽的需求. 该芯片采用 22 nm 工艺制造，由 16 个芯粒组成，通过 2.5D 中介层封装技术实现芯粒之间的互连. 每个芯粒内有 16 个 RISC-V 架构核心，总计达到 256 个核心，该设计能够扩展至 100 个芯粒，实现 1 600 个核心的集成. 然而，该系统在大规模扩展时可能面临供电和散热等挑战，且其软件生态和开发工具链尚需完善.

晶圆级计算机作为突破传统计算架构的新范式，虽然在系统能效比、集成度及并行计算能力方面取得突破性进展，但其技术生态仍面临关键性发展瓶颈. 当前系统的核心制约体现在领域通用性缺失与系统级适应性不足两大维度. 从架构设计层面看，现有系统多采用面向 AI 模型训练、自动驾驶等垂直领域的高度定制化架构，导致跨领域迁移能力受限，这种专用化设计模式不仅显著提高研发与制造成本，更阻碍了技术复用与规模化应用；从技术标准化视角分析，系统级设计框架尚未形成统一的架构规范，芯粒接口协议、互连拓扑及封装工艺的碎片化现状加剧了产业链协同难度. 此外，在工程实现层面，千瓦级功率密度引发的三维供电网络阻抗失衡与非线性热梯度分布问题、晶圆级封装中硅基中介层与有机基板的热机械应力失配挑战，以及面向超大规模并行的分布式软件栈适配性不足等复合型难题，共同制约着该技术从实验室原型向产业化应用的跨越. 这些系统性挑战亟待通过架构-工艺-软件协同创新范式实现突破，方能推动晶圆级计算机真正支撑通用计算场景.

为应对上述技术挑战，本文研究提出面向通用计算场景的晶圆级计算机——映天湖，其特点体现在支持多种工作负载（涵盖高性能计算、智能计算、数据通信及无线通信等），通用性强且包含大量异构芯粒. 本文主要贡献有 4 点：

1）提出基于“计算模组-互连基板”解耦架构的通用晶圆级计算机架构，支持多种不同的工作负载和应用场景.

2）提出同时支持多种异构计算芯粒的统一 I/O 设计，通过特定设计使 CPU 芯粒、AI 芯粒、DSP 芯粒、switch 芯粒等不同类型的计算芯粒能够利用同一种晶圆级计算机架构.

3）提出可重构的晶上网络，晶上网络能够动态调整网络拓扑配置，以适应不同的应用负载中的不同流量模式.

4）提出基于晶上网络可重构特性的容错机制，通过对可重构晶上网络进一步分层抽象，在逻辑互连与失效的物理互连之间增加一层无损互连层映射抽象，实现容错控制.

通过这些设计，所提出的晶圆级计算机能够满足多样化的计算需求，从而降低晶圆级计算机的研发和制造成本，提高系统的性能和能效，为晶圆级计算机领域的发展提供新的实际可行方案.

2 映天湖的总体结构

2.1 应用场景需求分析

本文面向多种应用场景如高性能计算、智能计算、信号处理、数据通信等，设计通用的晶圆级计算机架构，以较小的系统设计代价，满足多种应用场景的复杂需求.

在统一的晶圆级计算机体系结构下，晶上互连网络对业务特点的适应性成为主要的研究目标. 在各种业务场景中，高性能计算应用的负载特点是浮点计算量大、计算密集，在分布式系统实现环境下多个计算单元之间进行协作时，通信的模式比较规则

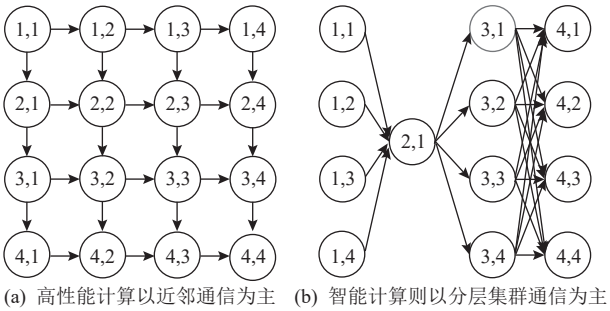
化，可以分为点对点、点对多点等. 智能计算应用的特点是定点(如 int8 精度)计算量大、计算密集，在分布式系统下进行协作时，通信的模式不完全与高性能计算相同，但也比较规则化，比如在训练中的前向和后向传播所带来的节点通信特点、在训练中交换权重的通信特点；无线通信领域如 5G 等，在基站控制器的业务处理中，多个 DSP 处理器之间的通信协作主要是流程式协作，在数据通信领域也类似，多个交换芯片之间很少针对同一业务协作，主要是业务上下游协作的关系. 在这多种应用场景中，多个业务或计算单元之间的协作模式有 2 种，一种是共同协作完成一个业务，因此之间的通信交互流量会比较大，另一种是计算单元之间协作通信较少，但是存在上下游的业务流量，因此在研究晶圆级计算机的组成原理时，需要结合多种应用场景进行晶上网络互连拓扑的设计考虑.

在晶圆级计算机这样的计算系统规模背景下，网络对负载的适配性造成的性能影响显著高于小规模计算系统，因此，晶圆级计算机的晶上网络设计需要考虑网络对于各种负载的适配. 本节首先对几种经典的应用负载进行简要分析.

高性能计算的一个重要应用场景是对偏微分方程的求解，求解过程一般采用网格迭代计算方式，通过将计算区域划分成一组网格单元，在每一轮迭代中，网格节点通过收集周围的数据来进行数据更新. 因此在计算过程中每一个节点都需要与近邻的节点进行通信，以求解偏微分方程的核心过程，以高斯-塞德尔算法为例，核心求解步骤为：

$$x_{i,j}(k)=\frac{x_{i,j-1}(k)+x_{i-1,j}(k)+x_{i+1,j}(k-1)+x_{i,j+1}(k-1)}{4}. \tag{1}$$

图 1(a)展示了算法的通信过程. 从左上角节点开始，得到计算结果之后向其相邻的网格发送数据，相邻网格得到数据之后即可展开计算，以此类推. 在迭代计算的中间过程中，网格相互交错的分成 2 组，每组可以在相同的时间点上实现计算. 算法将在网络中产生密集的近邻单点通信流量. 这种仅与周围节点进行通信的流量模式对网络的紧凑性提出了要求，即要求网络有较大的连通度以尽可能降低通信时延和减少路由跳数.



(a) 高性能计算以近邻通信为主 (b) 智能计算则以分层集群通信为主

Fig. 1 Significant differences between the two load communication modes

图 1 两种负载通信模式的显著差异

而在智能计算领域则以集群通信为主要通信方法，以 Transformer 的多头注意力机制为例：

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=Softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)V. \tag{2}$$

整个过程涉及 2 次矩阵乘法，根据计算过程展开即为频繁的向量求和操作. 这在分布式场景下将产生大量的聚合归约流量模式，以聚合归约为代表的集群通信是智能计算的经典模式，其由于短时间内在网络中注入大量的数据的流量特征对互连网络带宽提出了较高要求. 网络结构与流量的同构性也对负载性能有较高的影响. 图 1(b)展示了使用深度学习模型进行智能计算常见的几种通信模式以及分层特点，节点根据模型分成不同层次，层次之间以流量的聚合、广播以及全通信模式为主.

可见，不同的应用负载在晶圆级计算机的晶上网络中体现出不同的通信特点. 为了拓宽晶圆级计算机的应用场景，提升通用性以降低设计和制造成本，实现晶上网络对多种应用所需的通信特征的灵活支持是一个关键.

2.2 总体结构

如图 2 所示，映天湖晶圆级计算机共包含 3 个物理层次，自顶向下依次是整机层、功能与互连层和散热层. 其中，整机层负责提供电源、时钟和对外连接端口；功能与互连层主要包括由不同的计算芯粒和内存组成的计算模组及由互连芯粒组成的互连基板，计算模组间通信由互连芯粒支持，每个计算模组连接到 1 个或多个互连芯粒；散热层负责整个系统的物理散热和机械支撑，主要是散热冷板或液冷装置等.

当晶圆级计算机应用于多种不同的应用场景，可以采用以不同的计算模组加不同的 I/O 模组，基于统一晶圆互连基板架构，构成晶圆级计算机. 如在高

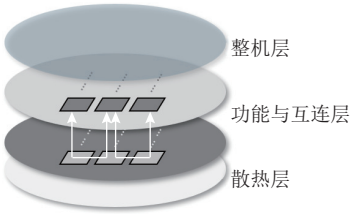


Fig. 2 Layered systematic architecture of Yingtian-Lake

图 2 “映天湖”的层次化系统架构

性能计算场景中，计算模组主要由浮点能力强的高性能 CPU 芯粒和存储芯粒构成，同时 I/O 模组分布在晶圆级计算机的周围，以实现与外部的数据交互. 在智能计算场景中，计算模组主要由能够进行高密度乘加计算的智能计算模组和存储芯粒组成. 在无线通信和数据通信场景中，计算模组内部组成换成了 DSP 数字信号处理芯粒与存储芯粒或交换芯粒与存储芯粒，同时 I/O 模组也可以独立更换，如在 5G 信号处理的场景下可以换成无线基带芯粒. 如果要实现这种目标，首先需要实现计算模组和互连基板的互连解耦设计，然后将互连基板标准化.

图 3 展示了映天湖通用晶圆级计算机的逻辑结构，位于边缘的功能模组负责系统对外的 I/O，称为 I/O 模组，位于内侧的模组则为计算功能模组，用于实现计算，图中“a”处的每一个计算模组由 2 个芯粒组成，一个是计算芯粒，另一个是存储芯粒，2 个芯粒间通过并行信号进行互连，而“b”处的 I/O 模组则由电 I/O 芯粒和光 I/O 组成，之间通过差分信号进行互连.

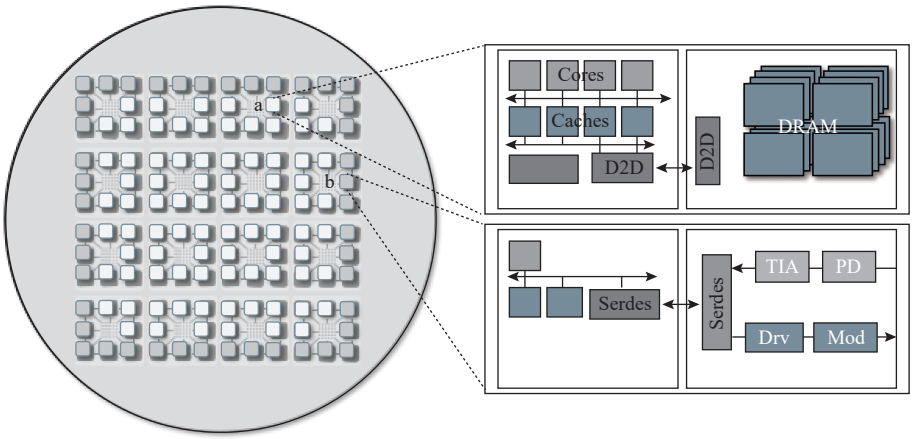


Fig. 3 Logical structure of Yingtian-Lake

图 3 “映天湖”的逻辑结构

图 4 则从剖面图视角揭示了映天湖系统的架构布局及各层次的紧密联系.

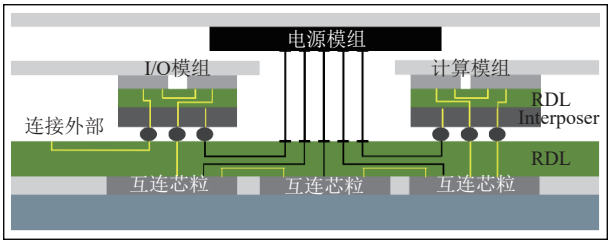


Fig. 4 Cross section of the general wafer-level computer Yingtian-Lake

图 4 映天湖通用晶圆级计算机的剖面图

图中位于最顶端的是系统整机层中的电源网络，为整个系统提供电力供应，电源网络为每一个模组群配备专用的电源模组用于供电管理. 在晶圆级计算机中，巨大的电路规模导致电力供应需采用垂直供电的方式以提高电力传输效率和减少因供电线路过长导致的能量损耗和信号干扰.

电源网络下是功能与互连层，图 4 刻画了功能层中 I/O 模组和计算模组的剖面结构，在每一个计算模组内部，芯粒与芯粒间的互连采用 2.5D 封装技术实现，而计算模组间的互连任务则由位于其下的互连子层负责.

互连子层构建起一个由大量互连芯粒（可以由 FPGA 芯片实现）组成的复杂互连网络，FPGA 互连芯粒之间通过 FPGA 芯片的部分 I/O 信号与相邻互连芯粒连接，并通过另外一部分 I/O 信号与上层的功能模组相连. 互连层通过 FPGA 互连网络实现可重构的路由.

“映天湖”具有很多晶圆级计算机共有的特征，即数量惊人的 I/O 端口带来的高基数交换能力[26-27]，这为“映天湖”的横向扩展提供了巨大的发展空间，如何针对晶圆级计算机进行横向扩展留待后续工作.

2.3 计算模组与互连芯粒之间的解耦设计

在本文的晶圆级计算机设计中，在计算模组内通过硅转接板实现芯粒之间的互连，模组间则必须通过互连层提供的互连网络来支持其实现互连，模组与互连芯粒之间通过先进封装方式互连，在计算模组 I/O 标准化的前提下，可以实现计算模组与由互连芯粒组成的互连子层的解耦. 一方面互连子层可以通过统一接口连接不同类型的功能模组，从而实现上层功能的多样化；另一方面互连子层通过互连芯粒的可重构性，可以针对不同的流量需求构建不同的互连拓扑结构，从而提高了晶上网络对负载的

适应性.

互连子层的互连芯粒与计算模组之间的互连关系具体根据模组的数量和大小来确定. 存在 2 种类型的关系，如图 5 中①所示，这种关系为 1 对 1 关系，例如使用 8 个互连芯粒实现 8 个计算模组互连. 如图 5 中②所示，这种关系为 1 对多关系，1 个模组对应多个互连芯粒，如 8 个互连芯粒只支持 4 个功能模组. 这是因为有一些模组的面积很大，其面积已经超过了一个互连芯粒所支持的面积范围.

图 6 展示了不同的互连逻辑关系在单一实际物理结构上支持的一个设计示例，其核心为通过 FPGA 互连芯粒的可重构特点实现逻辑连接关系的改变. 图 6(b)展示了一种二维网格的逻辑结构，图 6(a)展示了基于互连层可重构实现的另一种互连情形，模组群内将互连层重构为 3D 立方体结构以连接模组群内 8 个功能模组.

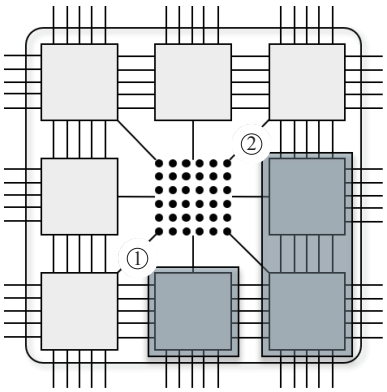


Fig. 5 Two connection relationships between functional modules and interconnection chips

图 5 功能模组与互连芯粒的 2 种连接关系

值得强调的是，虽然以上结构均基于 8 个互连芯粒的物理互连情形，但实际上无论是从系统结构的分析层面，还是从之后的算法设计层面，物理拓扑结构的组织方式仅仅受限于制造层面，互连芯粒之间不仅可以实现相邻的物理互连，还可以实现跨芯粒互连，且有些逻辑上的互连结构必须在物理层面有跨芯粒连接的支持. 对于跨芯粒实现逻辑连接的情形见第 4 节.

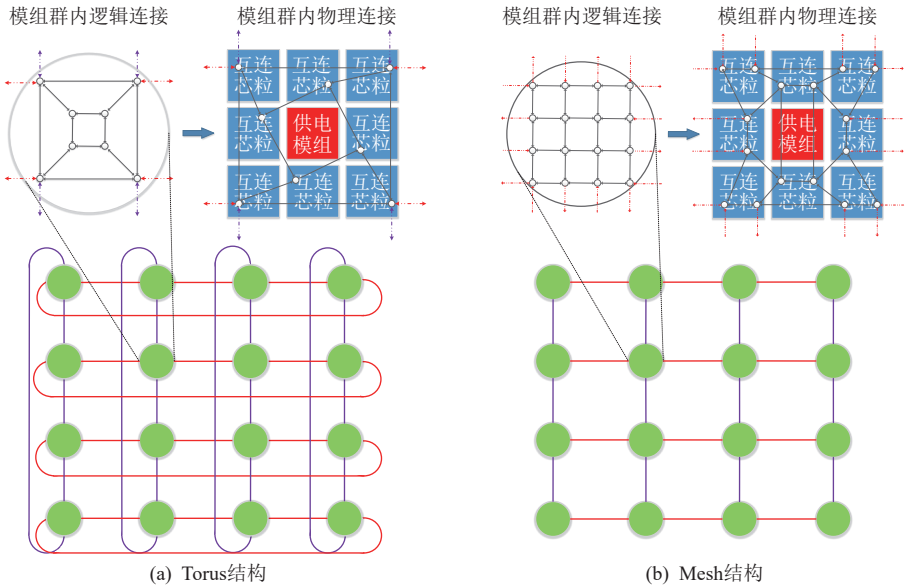
2.4 散热系统与翘曲问题

面对如此庞大的系统规模，势必会带来严重的功耗与散热限制，在如此大的芯片规模与温度变化条件下，系统发生翘曲问题也是一个必要的考虑因素. 本节针对这一系列问题做简要的介绍.

对于本系统而言，通过采用功能模组层的表面热流密度已经达到了 17.15 W/cm^2. 风冷散热已经无法满足系统的散热，因此系统选择采用浸没沸腾散热方式来解决散热问题. 如图 7 所示.

整个晶圆级计算机全部浸没于氟化液中，利用发热器件向液体传热，再由液体通过两相相变形式以最大功率向外输出，最终通过冷凝器将热量传导至室外. 散热系统工作时，循环泵启动，带动冷却液在散热装置的管路系统中循环流动，冷凝器处通过风机将管路中的热量输出，最终实现系统换热.

系统的温度应控制在 85 ℃ 以下，为达到此散热目的，需要结合系统仿真来对各个位置进行分析与选型. 系统经过各类仿真与数值计算方法对系统进



(a) Torus结构 (b) Mesh结构

Fig. 6 One physical interconnect supports two logical interconnections

图 6 1 种物理互连支持 2 种逻辑互连

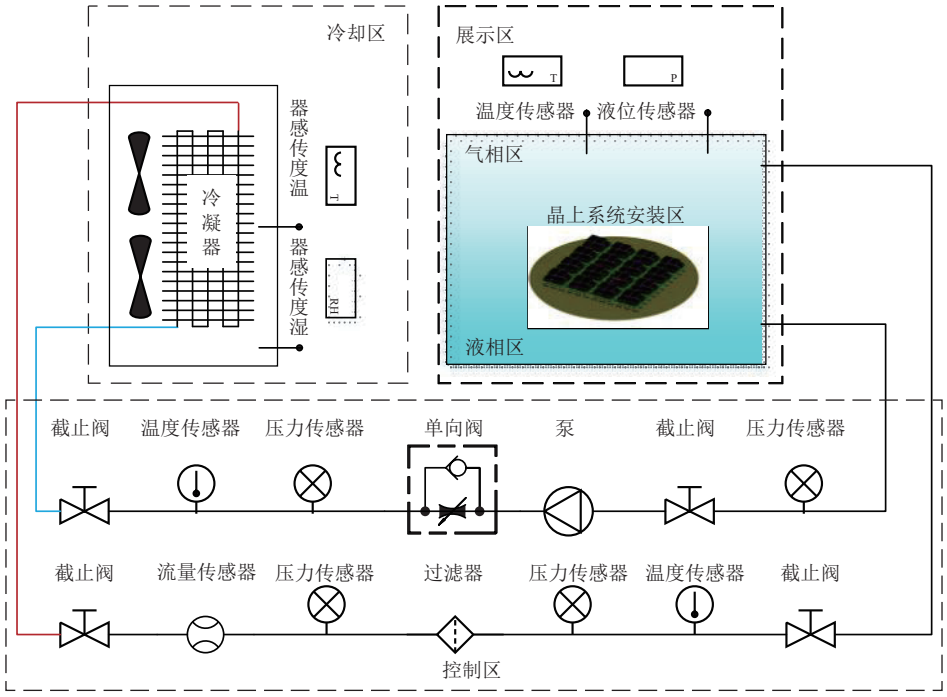


Fig. 7 Prototype heat removal system design of Yingtian-Lake

图 7 “映天湖”原型散热系统设计

行分析，包括但不限于流体体积法等，本文不详细介绍仿真过程，仅就理论结果结合工程实际得出的最终核心参数进行展示，如表 3 所示.

对于翘曲问题，其核心问题来源于各材料热膨胀系数不匹配、模塑工艺参数不当、结构设计缺陷 3 个方面，因此系统需分别从材料、工艺、结构 3 个方面来应对. 本文仅就实践结论进行总结.

在材料方面主要是对树脂进行选型，经过多轮比较，热膨胀系数相对较低（每 1 ℃ 下变化 7ppm）的树脂材料最终通过了塑封工艺验证，减少了与硅芯片的热失配.

Table 3 Key Parameters and Engineering Selection Guide for Heat Removal System

表 3 散热系统核心参数与工程选型

参数	数值
最高温度/℃	85
最低散热功率/kW	8.5
循环泵理论流量/($m^3 \cdot h^{-1}$)	2.9
循环泵选型	卧式多级离心泵
循环泵流量/($m^3 \cdot h^{-1}$)	4
循环泵扬程/m	30
循环泵功率/W	610
风机理论风量/($m^3 \cdot h^{-1}$)	4 000
风机选型	外转子轴流风机
风机风压/Pa	100
风机功率/W	349

模塑工艺中的参数设置主要是探究工艺温度与模塑时间，对芯片偏移情况有直接的影响；而芯片偏移则与树脂固化时的粘度有关，粘度越大，对芯片的偏移影响就越大. 基于系列仿真结果，“映天湖”的各个过程的最优参数如表 4 所示.

Table 4 Key Parameters of Molding Process

表 4 模塑工艺核心参数

参数	数值
预烘烤温度/℃	130
塑封成型烘烤温度/℃	125
后固化烘烤温度/℃	150
高温解胶温度/℃	180~210
成型压力/kN	220
压延速度/($mm \cdot s^{-1}$)	10
预期翘曲度/mm	≤3
模塑良率	≥85%

实际的 SoW 扇出晶圆的晶圆翘曲度为 2.357 mm，仍然较大，因此在最终的封装过程中需要增加 Dummy 芯片来改善翘曲.

3 计算模组 I/O 统一化

计算模组的 I/O 统一化工作可以分成 2 部分，如

图 8 所示, 一部分是计算模组内部的各种芯粒之间, 属于芯粒接口电路标准化的范畴, 另外一部分是计算模组和下面的互连芯粒之间的 I/O 标准化.

3.1 模组内芯粒互连标准化技术

在计算模组内部的各种芯粒之间标准化方面, 除国际标准 UCIe 外[31], 近年来我国完成了多项标准的立项和规格定义工作[32]. 目前, 围绕芯粒间的互连技术标准化工作主要集中在芯粒本身所带有的芯粒接口电路物理层的定义和开发上, 芯粒互连的应用场景方面主要包括 C2C, C2M 和 C2I/O, 目前标准中的定义主要集中在实现 C2C 和 C2I/O 场景, 较少涉及 C2M, 即计算芯粒和内存芯粒之间的接口标准化. 美国 ODSA 组织牵头定义了 OpenHBI 标准, 使芯粒间的接口标准能够兼容 HBM 芯粒的连接, 中国电子化标准协会在 2023 年 3 月也立项了类似标准. 因此, 晶圆级计算机的模组内芯粒互连标准化工作接近完成.

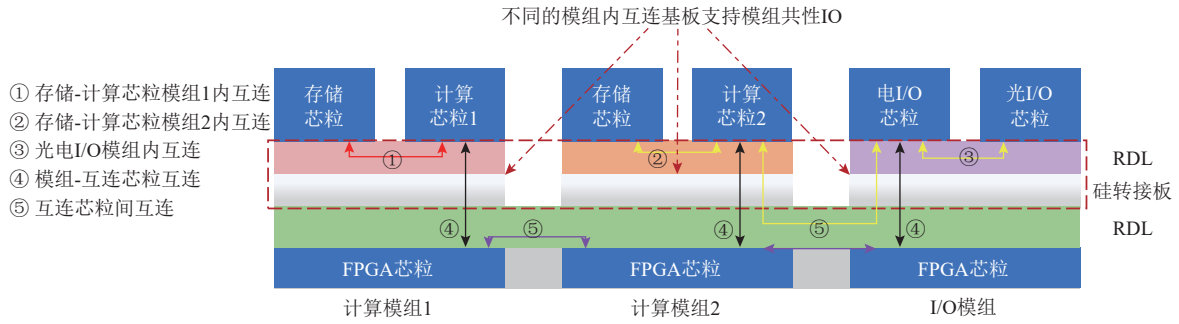


Fig. 8 I/O standardization between chiplets within a module and between modules and interconnect chiplets

图 8 模组内芯粒间及模组到互连芯粒的 I/O 标准化

3.2 计算模组 I/O 统一化

计算模组到互连芯粒的 I/O 统一化方面, 主要分为 2 种情况, 即 1 对 1 和 1 对多.

1)计算模组和互连芯粒 1 对 1 互连的情况

对于面积较小、接口数量较少的芯粒, 可以使用 1 对 1 的互连集成方案. 大多数芯粒以及由它们组成的功能模组属于此种类型.

对于像计算芯粒构成的计算模组, 模组内一般集成存储芯粒. 由于不同计算芯粒所需的存储芯粒数量有所不同, 在实施方案中可以适当地使用多层堆叠存储芯粒的方式支持多存储芯粒方案, 或者采用 HBM 方案.

在模组层面, 标准化输入输出的设计体现在不同的芯粒组成的计算模组可以实现替换, 如图 9 所示. 具体方案是: 计算模组采用硅转接板(interposer)设计各自的模组内芯粒间互连. 模组内互连基板实现了引脚重排布功能, 不同的模组内互连基板朝上连接不同的芯粒输入输出, 包括实现芯粒间互连; 朝下实现相同的标准化输入输出布局, 通过晶圆级互连基板与互连芯粒相连, 进而实现不同的计算模组替换.

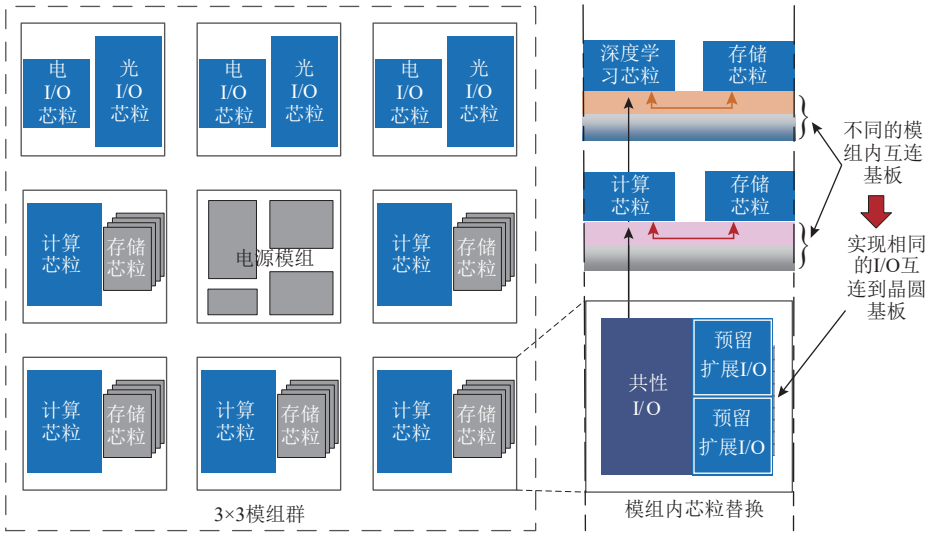
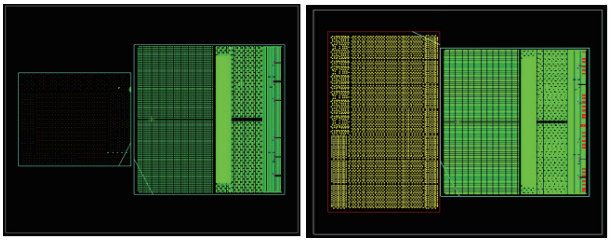


Fig. 9 Illustration of replacement of high-performance computing module and intelligent computing module

图 9 高性能计算模组和智能计算模组替换示意图

图 10 以一个具体的设计案例来说明. 图中有 2 个计算模组, 一个是包含高性能计算芯粒的计算模组, 另外一个是包含深度学习芯粒的计算模组. 其中高性能计算芯粒集成了 4 个 RISC-V 处理器核; 深度学习处理器集成了 4 Tops 智能计算算力. 这 2 种芯粒使用相同的存储芯粒作为存储器组成模组, 但是功能芯粒的形状和面积是不同的.



(a) 高性能计算模组 (b) 智能计算模组

Fig. 10 High performance computing module and intelligent computing module

图 10 高性能计算模组和智能计算模组

高性能计算芯粒和深度学习芯粒的芯粒间接口也是相同的, 是一组 16 对 12.5 Gbps 的接口. 所以芯粒间接口在模组转换板上可以放在完全相同的位置上.

在模组的边界上, 除了芯粒间互连的数据接口, 其他接口通常是慢速接口, 比如控制信号、配置信号和调试等接口. 这些信号接口不同于芯粒间接口, 因此称为扩展输入输出信号接口. 对这些扩展输入输出信号的处理方式是预留出 2 种芯粒慢速信号取并的信号数量的输入输出在模组内的转换板上. 这些信号直接跟互连芯粒低速接口相连.

这 2 个模组的转换板设计的对内接口是不同的, 分别连接着计算芯粒和深度学习芯粒, 但是对外输入输出是完全相同的, 从而实现了晶圆级计算机中的计算模组层面的可替换.

2)计算模组和互连芯粒 1 对多情况

对于面积较大的计算芯粒和模组, 由于其性能设计目标较高, 其面积和尺寸大于单个模组的物理尺寸限制且 I/O 数量也更丰富. 这种情况在实施中使用跨互连芯粒布局的方案, 如图 11 所示.

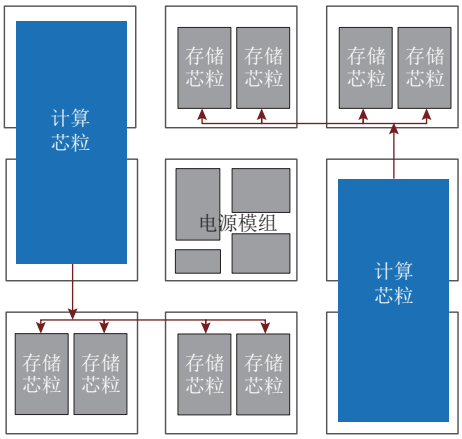


Fig. 11 Cross-interconnect chiplet layout of computing module

图 11 计算模组跨互连芯粒布局

以图 12 为例, 计算模组占用 2 个互连芯粒位置, 通过使用更大的模组内互连基板, 实现 2 个互连芯粒与计算芯粒的互连.

4 支持可重构的晶上网络

我们首先构建互连层配置物理基础, 然后探索互连层重构策略设计空间, 最后对该探索策略给出理论上的性能分析.

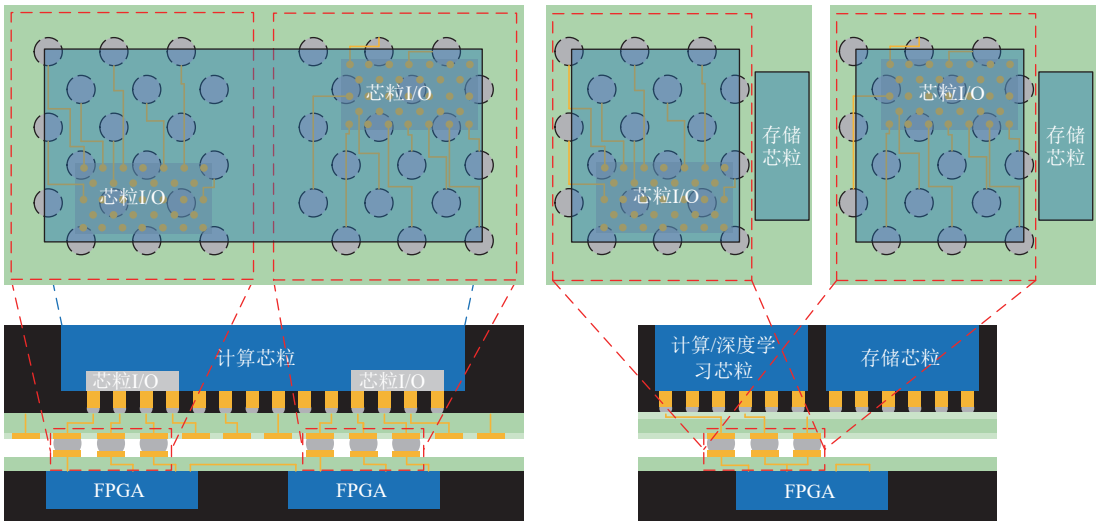


Fig. 12 Interconnection between two interconnected chiplets and a computing chiplet

图 12 2 个互连芯粒与 1 个计算芯粒的互连

4.1 构建互连层配置物理基础

由互连芯粒组成的互连层通过先进封装技术将多个互连芯粒互连组成晶上网络. 图 13 左图展示了一种功能模组群连接到互连网络上的详细实现. 一个互连芯粒可以例化 1 到多个路由节点，每一个计算模组连接互连芯粒中的 1 个至多个路由节点，路由节点之间可以通过物理链路和逻辑链路(通常在互连芯粒内部)的配置以及在互连芯粒上的路由功能实现不同的逻辑互连关系，图 13 中展示了使用互连层提供的线路在模组群内实现 3D 立方网络的实际互连情形，路由节点之间通过配置物理链路或逻辑链路实现逻辑连接. 具体的网络配置通过重构算法进行控制.

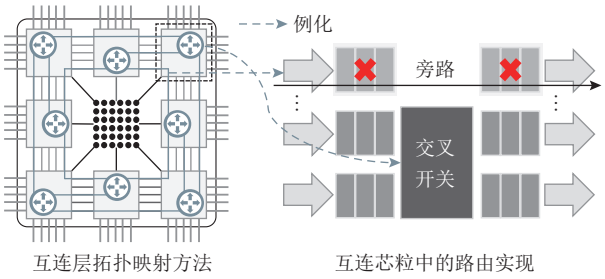


Fig. 13 Interconnect layer configuration and FPGA instantiation scheme

图 13 互连层配置与 FPGA 例化方案

本文为简化链路配置的叙述，对物理链路做了假设，每一条物理链路只能被配置为某一个方向的通信链路，即单工通信. 假设每一个互连芯粒共有 k 条物理链路连接，那么对于整个网络的重构就是规划各个通信链路的通信方向，以及每一个互连芯粒的路由策略.

每一个互连芯粒所需要支持的通信需求可以用一个通信图来表示. 通信图中的每一个节点代表芯粒所连接的链路，每一条有向边代表 2 条物理链路之间的通信关系，由于每一条链路只能被配置为某一个方向的通信，所以通信图一定呈现二部图的图论性质.

以图 14 所展示的二部图为例. 边 0-1 代表一条独立于其他链路的通信需求，其物理链路间的通信可直接通过直连旁路实现，不需要为此构造路由微结构，从而降低通信延迟和节省 FPGA 资源占用. 同样出于节省资源和延迟降低的目的，除 0-1 这类可直连的边之外的其他弱连通分量，FPGA 互连芯粒上将为每一组弱连通分量生成一个路由微结构，旁路与路由微结构如图 13 右图所示.

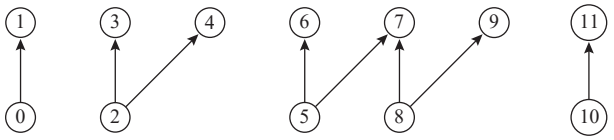


Fig. 14 Example of communication diagram

图 14 通信图示例

互连层的链路可重构为互连层的实现提供了巨大的设计空间，理论上可以将多种不同的连接关系部署到互连层中. 但实践表明，在如此巨大的设计空间中针对目标连接关系进行人工探索具有挑战. 需要一个简单、有效的设计空间探索方法来寻找部署目标连接关系的可行解甚至最优解.

4.2 探索互连层重构策略设计空间

本节介绍基于互连层物理实现提供的设计空间进行探索的基本策略，即功能模组间如何通过对互连芯粒及链路的配置实现模组间的目标连接关系问题，后文简称为拓扑映射问题. 晶上网络所连接的计算模组与互连芯粒之间，以及互连芯粒之间由固定数量的物理链路互连，由于功能模组最终通过路由节点实现互连，在互连层将只讨论路由节点，功能模组的互连关系则由路由节点间的互连体现.

表 5 列出了本文所用的数学符号的含义.

每一条物理链路可以被配置为一条单向物理链路，但在配置之前，每一条链路是无向的. 晶上网络整体被抽象为无向图 $\boldsymbol{C}_{N\times N}$，称为容量矩阵，$N$ 为互连芯粒的个数，$\boldsymbol{C}_{xy}$ 代表互连芯粒 x 和互连芯粒 y 之间存在的链路个数的一半，对角线元素为 0.

对于连接关系，则由单向通信关系来进行表示，例如组织数据流图、神经网络等. 更一般地，想要建

Table 5 Mathematical Symbols Representation

表 5 数学符号表示

符号	含义
N	互连芯粒数
M	路由节点数
$\boldsymbol{C}_{N\times N}$	容量矩阵，描述互连芯粒网络
$\boldsymbol{\Lambda}_{M\times M}$	需求矩阵，描述通信需求
x,y	互连芯粒的序号
a,b	路由节点的序号
$\boldsymbol{f}_{M\times M\times N\times N}$	流量张量，描述链路映射
$\boldsymbol{\phi}_{M\times N}$	映射矩阵，描述路由节点映射
R	逻辑路由节点的集合
P	物理互连芯粒的集合
r	互连芯粒承载上限
$\boldsymbol{n}_{N\times N}$	互连芯粒物理近邻关系矩阵

立路由节点间相互双向通信的计算网络，也可以通过不同的 2 条单向连接实现. 连接关系在算法中即通信需求，使用 0-1 矩阵$\boldsymbol{\Lambda}_{M\times M}$来刻画每对路由节点间是否为通信关系，称为需求矩阵，M为参与互连的路由节点个数. $\boldsymbol{\Lambda}_{ab}=1$即路由节点$a$存在一条向 b 的单向通信关系.

互连芯粒可以配置无需路由的快速链路，逻辑上想要实现跨互连芯粒的连接关系，则需要以多余的链路消耗为代价的，将多条物理链路通过快速链路组合形成. 通信需求都使用了哪几条物理链路作为支持，则需通过四维 0-1 张量$\boldsymbol{f}_{M\times M\times N\times N}$表示，称为流量张量. $\boldsymbol{f}_{abxy}=1$ 则表示通信关系ab需要物理链路xy分配一条链路来支持通信，本文仅在链路层级实现非抢占式分配，每一个通信关系在每对互连芯粒间将分配到固定且最多 1 条物理链路.

除了链路规划的考量，功能模组的位置和相互物理关系的设计也是晶上网络规划的重要影响因素. 如果设计初期就将此纳入考量，功能模组与互连芯粒的对应关系也将成为设计空间的一部分. 本节仅就 2.2 节介绍的 2 种连接关系作为算法的设计范围，即 1 个或多个功能模组向 1 个互连芯粒上连接，在互连层则以 1 个互连芯粒可以例化多个路由节点呈现. 这种连接关系可通过一个 0-1 矩阵$\boldsymbol{\phi}_{M\times N}$来表示，称为映射矩阵. $\boldsymbol{\phi}_{ax}$表示路由节点a将由互连芯粒x例化.

拓扑映射问题可以被解释为多商品流问题的一个变种问题，在多商品流问题中的流量需求的源汇点都是固定的，该问题对此做了修正，修正思路如图 15 所示. 通过点与边的共同规划最终将逻辑拓扑映射到物理拓扑中去.

为简化描述，a,b符号单独出现时约定用于遍历路由节点下标，x,y符号单独出现时约定用于遍历互连芯粒下标，如$\forall a,b,x$即表示$\forall a\in R,b\in R,x\in P$，$R$为全体路由节点集合，$P$为全体互连芯粒集合. 如无特别说明，后文都按此约定简化公式描述.

首先分别对多商品流问题的 3 条约束进行修正：

1）容量约束. 即流量张量$\boldsymbol{f}$中所有通过无向边xy的流量总和不超过边的容量$\boldsymbol{C}_{xy}+\boldsymbol{C}_{yx}$.

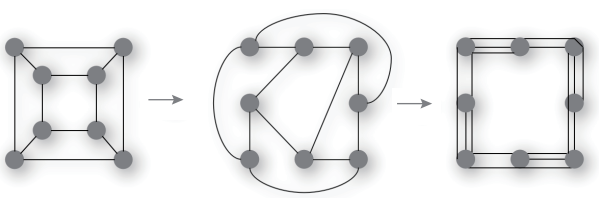


Fig. 15 Topology mapping problem

图 15 拓扑映射问题

$$\forall x,y,\sum_{ab}\boldsymbol{f}_{abxy}+\sum_{ab}\boldsymbol{f}_{abyx}\leqslant\boldsymbol{C}_{xy}+\boldsymbol{C}_{yx}. \tag{3}$$

这里暗含了$\forall a,b,x,\boldsymbol{f}_{abxx}=0$的约束条件.

2）流守恒. 某个互连芯粒如果对于某条连接关系来说非源非汇，那么支持该连接关系的进入该芯粒的链路与输出该芯粒的链路相等：

$$\forall a,b,x,(1-\boldsymbol{\phi}_{ax})(1-\boldsymbol{\phi}_{bx})\left(\sum_{y}\boldsymbol{f}_{abxy}-\sum_{y}\boldsymbol{f}_{abyx}\right)=0. \tag{4}$$

这在$\boldsymbol{\phi}$是常量时退化为多商品流问题的流守恒约束.

3）需求满足. 与流守恒对应，如果某个互连芯粒对于某一条连接关系来说是“源”，那么该芯粒的链路应该能够满足连接关系的输出需求，如果是“汇”则应满足连接关系的输入需求：

$$\forall a,b,x,\boldsymbol{\phi}_{ax}(1-\boldsymbol{\phi}_{bx})\left(\sum_{y}\boldsymbol{f}_{abxy}-\boldsymbol{\Lambda}_{ab}\right)=0, \tag{5}$$

$$\forall a,b,x,\boldsymbol{\phi}_{bx}(1-\boldsymbol{\phi}_{ax})\left(\sum_{y}\boldsymbol{f}_{abyx}-\boldsymbol{\Lambda}_{ab}\right)=0. \tag{6}$$

式(5)(6)同样在$\boldsymbol{\phi}$是常量时退化为多商品流问题的需求满足约束.

如果要对$\boldsymbol{\phi}$纳入考量，一个合理的$\boldsymbol{\phi}$则需要满足 2 点约束：

1）节点度限制. 一个互连芯粒能够承载的路由节点数存在上限：

$$\forall x,\sum_{a}\boldsymbol{\phi}_{ax}\leqslant r. \tag{7}$$

2）强制映射约束. 一个路由节点必须且只能映射到一个互连芯粒上：

$$\forall a,\sum_{x}\boldsymbol{\phi}_{ax}=1. \tag{8}$$

整个问题的设计空间即通过 5 条设计约束及其变体实现，目标函数选取最短和最少目标来实现多目标规划，即首先优化使规划中的最长路径最短，在此基础之上再优化使得整个规划占用的物理链路数最少.

$$\min\max_{ab}\sum_{xy}\boldsymbol{f}_{abxy}, \tag{9}$$

$$\min\sum_{abxy}\boldsymbol{f}_{abxy}. \tag{10}$$

上述所有的问题定义都属于抽象的整数线性规划问题，即可以对非线性约束与目标函数进行线性转化，从而使用整数线性规划求解器进行求解.

注意，上述模型是一个基础模型，针对不同的实

际场景，还需要做更多约束控制. 例如对于跨互连芯粒互连的情形，在互连层可以表示为某 2 个路由节点a,b必须严格处于相邻的互连芯粒上：

$$\forall x,y,\boldsymbol{\phi}_{ax}\boldsymbol{\phi}_{by}(\boldsymbol{n}_{xy}-1)=0, \tag{11}$$

其中，$\boldsymbol{n}$表示互连芯粒之间在物理上的近邻关系矩阵.

4.3 模型性能估算

上述所有的问题定义都属于抽象的整数线性规划问题，即可以将非线性约束与目标函数转化为线性，从而使用整数线性规划求解器进行求解. 但整数线性规划问题属于 NP 难问题，往往需要通过启发式算法获得问题的解，对于更庞大的模型甚至可能在能容忍的时间内找不到问题的解. 本文不对该问题的具体求解策略做讨论，转而从问题本身出发来降低问题求解规模.

因为无论是设计层面还是实际制造层面，同类型的模组群往往采用相同或对称的结构，实际上的网络配置的复杂度与单模组群规模和不同的模组群数量呈指数关系. 所以实际应用中通常通过求解单模组群的结果来决定整个晶上网络的最终设计. 我们结合 2 个应用场景来介绍这种考量的可行性.

1)网格网络拓扑映射. 网格网络由于其路由的简单性和健壮性成为诸多互连网络设计的首选拓扑，对于 16 个路由节点映射到 8 个互连芯粒上的情形，该问题的规模可以在单台计算机上以数十秒量级速度完成(在第 8 秒收敛到最优解，第 24.36 秒结束搜索返回求解结果).

2)存储-计算芯粒排布规划. 图 11 给出了跨芯粒排布的一个基础情形，在制造层面，所有的计算模组群都采用同一排布形式，引入各模组群配置全等约束后，问题的规模只有 1 个模组群的 8 个模组的大小，大芯粒的存在会进一步压缩求解空间. 整个问题求解可以在秒级时间内完成.

一些经典的求解问题的时间性能如表 6 所示. 这是在单台 8 核心及 16 线程的计算机上的计算结果. 可以看到对相对平凡的解，如将网格网络映射到同构的网格网络上时，模型可以在很大规模的网络下很快给出最优解，对于一般情形，系统则在几十秒内完成求解，而对于更大规模的极限情形，模型则可以在有限时间内给出可行解. 本节就该问题求解速度和常见的问题规模给出一个经验结论：当问题的路由节点、互连芯粒的数量均不超过 20 时，可以通过单计算机的启发式方法在最慢分钟量级时间内实现问题的求解. 本文称该规模满足单机求解条件.

Table 6 Model Solving Performance

表 6 模型求解性能

问题	求解时间/s	解的质量
8×8 Mesh 同构映射	3.37	最优
16 节点映射至 8 互连芯粒	24.36	最优
6 维立方网映射至 8×8 的 8 通道网格	60.00	可行

5 晶上网络容错机制

首先进行网络缺陷分析，然后设计容错机制.

5.1 网络缺陷分析

晶圆级计算机由于其封装工艺的限制，在封装过程中会产生封装缺陷，这使得即便在保障计算模组内计算芯粒与互连芯粒全部为良品裸片(known good die, KGD)的情形下，系统依旧有可能出现缺陷，对于一个巨大的晶上网络而言，错误无法完全避免.

从晶上网络可能出现的错误种类来看，晶上互连网络的错误主要分为 4 种情况：静态单根物理线路失效、静态互连芯粒失效、动态线路瞬时故障、动态互连芯粒瞬时故障. 其中静态故障主要由于封装工艺中的缺陷导致，比如封装加工导致的失效；动态故障则与运行时的干扰有关. 本文描述的工作主要聚焦静态缺陷导致的链路不可达问题.

无论是物理线路出现故障，还是互连芯粒失效，在抽象层面都属于网络实际的容量矩阵或连接关系与制造目标不符. 这首先对晶上网络本身的物理拓扑结构设计提出了容错要求，例如可能实现接近超立方体网络的物理拓扑结构来实现高容错. 本文不对互连网络拓扑做任何假设.

以互连芯粒封装错误为例，图 16 展示了拓扑无关的容错机制的基本概念，我们将看到在这种机制下，链路故障的容错控制方法是互连芯粒容错控制

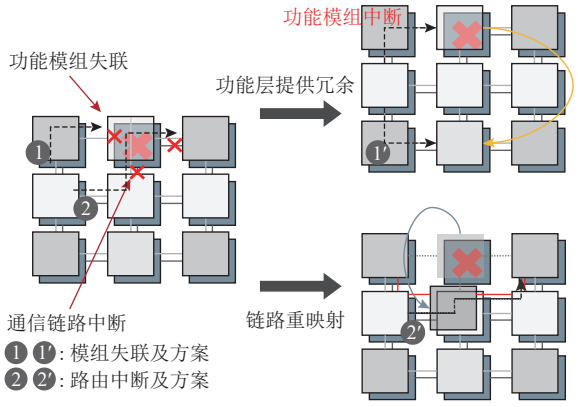


Fig. 16 Topology-independent fault tolerance mechanism

图 16 拓扑无关的容错机制

方法的一个很小的子集.

互连芯粒的失效将同时影响功能层和互连层2个层面，功能层的功能模组将由于互连芯粒与功能模组失联而与整个网络断开，而在互连层，则由于该互连芯粒不可达，导致原路由机制部分链路无法借助该芯粒例化的路由节点实现转发.

5.2 拓扑无关的容错机制

本节基于5.1节对网络缺陷的分析，提出一种拓扑无关的容错方案，以支持通用设计目标.

功能层的失联是不可挽回的，只能要求用户必须对每一种功能模组都做至少2份的冗余设计，无法连接到的功能模组的功能将由其他冗余模组代理，从而保证功能层上的容错.

而对于互连层则体现为实际容量矩阵与设计目标不符. 容错方案采取基于第4节介绍的拓扑映射问题的求解的方法，首先将虚拟的无损网络映射在实际缺陷网络上，再将目标连接关系映射到虚拟无损网络. 缺陷网络中原损坏的互连芯粒将寄生在相邻的互连芯粒上，被寄生的芯粒被称为寄主芯粒，寄主芯粒将承载所有发送给原寄生芯粒的网络流量. 该方案将保障原互连结构的无死锁性质.

该容错方案存在容错极限，即在功能层，如果所有的同类型模组都损坏，那么网络将失去原有的系统功能从而彻底失效；在互连层由于拓扑映射问题本身上存在极限，当问题模型给出了无解的计算结果时，网络将宣告失效.

下文对基于拓扑映射问题的拓扑无关容错控制给出详细操作步骤：

1)划分子网络. 通过4.3节中介绍的模型求解时间的大致估计可知，完整的晶上网络拓扑映射问题的求解时间是无法容忍的. 但是对于容错机制而言，其核心只是为了将损坏的互连芯粒寄生到一个合理的寄主芯粒上. 因此只涉及晶上网络的局部重构. 例如通过广度优先搜索的方法，搜索以损坏芯粒为中心的外围2层的节点组合作为求解子网络进行重构. 网络的其余部分保持原有结构. 以网格网络为例，广度优先搜索2轮将最多选中13个互连芯粒用于问题求解，满足4.3节给出的单机求解条件(12个备选寄主芯粒和1个损坏芯粒). 该方法对于大多数连通度不高的网络成立.

2)求解规划问题. 划分出子网络后，即可通过拓扑映射问题求解模型将该子网络的虚拟无损网络部署在缺陷网络上. 而逻辑互连关系对于互连层拓扑的损坏是无感知的，因此依然可以通过拓扑映射问题的求解来给出问题的解.

3)映射综合. 2个拓扑映射结果是独立完成的，对于实际的网络部署则需要对2个计算结果进行综合. 即将映射到无损网络的结果与无损网络映射到缺陷网络的结果做逻辑演算得到最终的映射结果.

对于逻辑互连关系映射至无损网络的拓扑映射问题求解结果，使用$\boldsymbol{f}^{\mathrm{L}}_{abx_1y_1}$和$\boldsymbol{\phi}^{\mathrm{L}}_{ax_1}$表示，而对无损网络映射至缺陷网络的求解结果则用$\boldsymbol{f}^{\mathrm{P}}_{x_1y_1x_2y_2}$和$\boldsymbol{\phi}^{\mathrm{P}}_{x_1x_2}$来表示. 对2个模型的结果做如下综合即可得到逻辑互连关系映射至实际缺陷网络的求解值：

$$\boldsymbol{f}_{abx_2y_2}=\bigvee\nolimits_{x_1y_1}\boldsymbol{f}^{\mathrm{L}}_{abx_1y_1}\boldsymbol{f}^{\mathrm{P}}_{x_1y_1x_2y_2},\tag{12}$$

$$\boldsymbol{\phi}_{ax_2}=\boldsymbol{\phi}^{\mathrm{L}}_{ax_1}\boldsymbol{\phi}^{\mathrm{P}}_{x_1x_2}.\tag{13}$$

$\boldsymbol{\phi}^{\mathrm{L}}_{ax_1}$中的$x_1$由于强制映射约束，是存在且唯一的. 以无损网络为网格结构为例，单互连芯粒的错误将影响至多4条链路的绕路，在链路资源相对充足的条件下，损坏的芯粒将寄生在周围某一个寄主上，借助冗余链路实现互连.

该容错机制利用互连层可重构性特征实现容错设计. 与此同时，由于FPGA资源相对丰富，在链路资源同样充足的条件下，该容错机制能够在一定程度上保障系统的性能.

6 实验评估

6.1 模拟实验

本节对晶圆级计算机的可重构网络性能进行评估，评估平台采用基于Booksim2模拟器. Booksim2以时钟周期为基本单位，通过细粒度的片上网络模拟来仿真系统性能[33]，本文在Booksim2的基础上加入了经典的流量模式和适配的路由算法. 在这个模拟器上本节评估了16×16节点的通用晶圆级计算机中晶上网络的理论性能. 其中每一个路由节点的基本参数为每端口包含4个虚拟通道(virtual channel, VC)，缓冲区大小为32个flit，单个flit位宽256 b.

本节着重探讨特定流量模式的延迟-吞吐量曲线，以探讨系统重构对单包平均延迟的影响. 为分析单包平均延迟，实验选择了3种经典的流量模式：以全局归约(All Reduce)和全全通信(All to All)作为人工智能计算场景中集群通信的代表性模式，以近邻(Neighbor)通信模式为高性能计算场景中的代表性模式，前文已经介绍过这些流量模式的应用场景，本节仅对3种流量模式的特征做一个简单的介绍.

1)全局归约. 全局归约流量模式是单点广播或

聚合的代表性流量模式,过程为首先在主节点向节点组广播一个聚合请求,每一个节点在接收到聚合请求后,将聚合数据发送到主节点.本文使用全局归约作为一系列广播流量模式的代表来测试系统性能.

2)全全通信.全全通信流量模式是全通信的代表性流量模式,每一个节点都需要对节点组其他节点发送数据,同时又需要接受其他节点发送来的所有数据.这是一种对通信带宽要求甚高的一种通信模式,其代表了一系列的涉及节点组内全通信流量模式,例如全局分散(All Scatter)等.

3)近邻通信.近邻通信流量模式即网格计算中的经典模式,每一个网格节点都与自己在逻辑网格上相邻的其他节点交换数据.在网络并不拥塞的情形下,近邻通信模式与网络对逻辑互连结构的亲和性高度相关,当网络与逻辑互连结构明显异构时,将存在大量节点的数据包通过多条路由到达目的节点,从而对网络性能造成影响.该通信模式大量存在于高性能计算领域,用于偏微分方程的计算.

基于对上述3种流量模式的分析,我们在晶上网络中构建2种逻辑拓扑来探讨不同的连接关系对上述3种流量模式的适应性,如图17所示.我们的实验使用网格(Mesh)和树(Tree)来互连256个功能模组,并支持上述3种不同的流量模式.

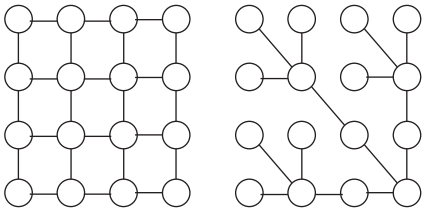


Fig. 17 Two examples of interconnection network structures

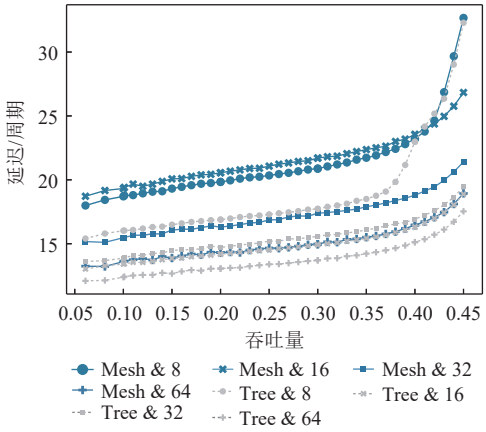
图17 2种互连网络结构示例

网格拓扑为16×16方块状拓扑结构,其中每一个路由节点都有上下左右4个方向以及向功能模组转发的方向共5个端口,其被用于代表网格亲和性较强的网络.

树状拓扑则更为复杂,采用递归定义方式进行定义,16×16首先确定右下角为根节点,按方块等分成4个区域,然后每一个带根节点所在区域外的3个区域的右下角作为子树的根节点,将其连接在最近邻的根节点所在区域的树的物理最近邻的叶子节点上,以此递归定义直至树只有1个节点为止.其被用于代表优化树状集群通信的网络.

在实验中,为了规避频率的影响,我们选择以网络时钟周期为单包平均延迟的时间单位以讨论该类系统的网络性能,而吞吐量则定义为每一个参与流量模式的网络节点向网络注入数据包的时间所服从的伯努利过程的强度,即每个时钟周期发包的概率(如果该周期节点需要进行发包).延迟-吞吐量曲线所要呈现的2个指标,一个指标是吞吐量较小时网络没有造成拥塞时的单包平均延迟,另一个指标则是网络最大吞吐量,即延迟明显提高时对应的吞吐量.

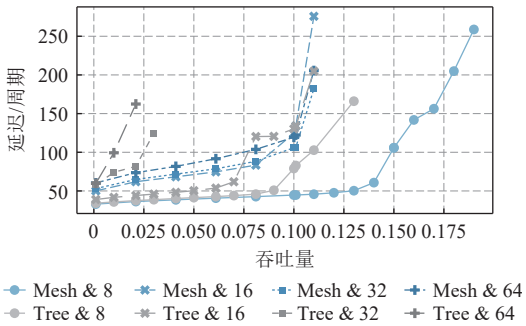
图18代表按照不同节点组划分的全局归约流量模式在Mesh网络和Tree网络2种网络的延迟-吞吐量曲线.可以看到Tree网络对于全局归约具有平均1.45倍的平均单包延迟性能提升,这验证了将Mesh重构为Tree结构更有助于提升全局归约类型流量模式的性能.而图19展示了全全通信流量模式的延迟-吞吐量曲线,可以看到Tree拓扑对于系统延迟依旧有很大改观,但是系统的最大吞吐量却有相比于Mesh结构0.78倍的性能劣势,这是由Tree结构在接近树根的位置通信带宽不足导致的,这也反映了在拓扑



注:“&”后数字表示节点组大小.

Fig. 18 Latency-throughput curves of running All Reduce traffic patterns

图18 运行全局归约流量模式的延迟-吞吐量曲线



注:“&”后数字表示节点组大小.

Fig. 19 Latency-throughput curves of running All to All traffic patterns

图19 运行全全通信流量模式的延迟-吞吐量曲线

重构过程中存在的性能取舍问题.

而在近邻通信中，Tree 结构的性能明显较之于 Mesh 严重劣势，如图 20 所示，Tree 结构在近邻模式下的最大吞吐量明显弱于 Mesh 结构，Mesh 结构约有 1.7 倍于 Tree 结构的性能优势，而在延迟上也没有明显的提升. 这说明了 Mesh 结构由于其良好的网格亲和性，能够更好地支持网格计算这类通信负载.

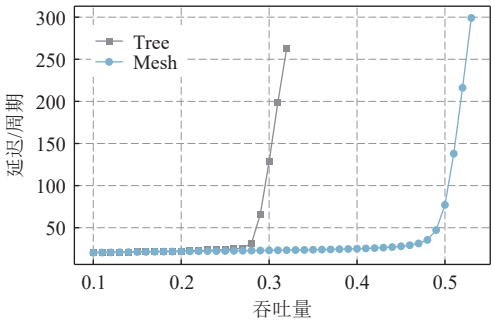


Fig. 20 Latency-throughput curves of running neighbor traffic patterns

图 20 运行近邻流量模式的延迟-吞吐量曲线

基于 2 种拓扑结构对 3 种流量模式的性能评估，可以看到不同拓扑结构对不同应用负载支持的明显性能差异，而本文所设计的晶上网络由于其良好的可重构特性，可以根据不同的负载特点进行重构，从而实现对多种业务负载的支持.

6.2 FPGA 原型验证

6.2.1 实验平台简介

该 FPGA 验证板基于 YX4F300T900I FPGA 芯片设计和自行开发，如图 21 所示，单块原型验证板内板载 4 块 FPGA. 单块原型验证板可通过 FMC 排线连接，用作更大规模的原型验证. 该工程使用 vivado EDA 工具完成代码编译、bit 流生成以及代码烧录.

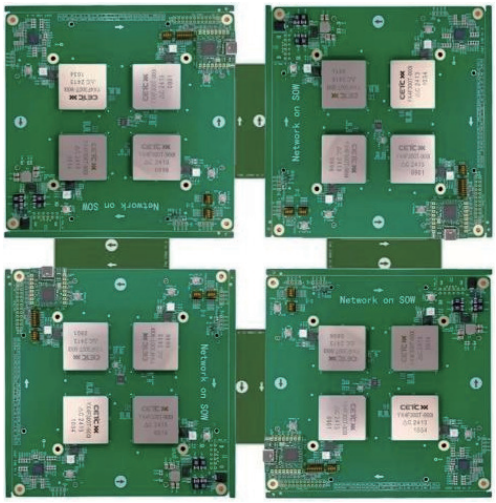


Fig. 21 FPGA prototype verification system

图 21 FPGA 原型验证系统

6.2.2 验证思路

该实验共实现 2 种拓扑，一种为 256 个路由器组成的网格拓扑，另一种为 256 个路由器组成的 5D 环绕拓扑. 其网格拓扑，经评估后在 8 块 FPGA 上完成，每块 FPGA 上运行 8×4=32 个路由器，如图 22 所示. 另一种为 256 个路由器组成的 5D 环绕拓扑，每块 FPGA 上运行 32 个路由器，如图 23 所示.

路由器设计包含 5 个输入通道（5D 环绕拓扑为

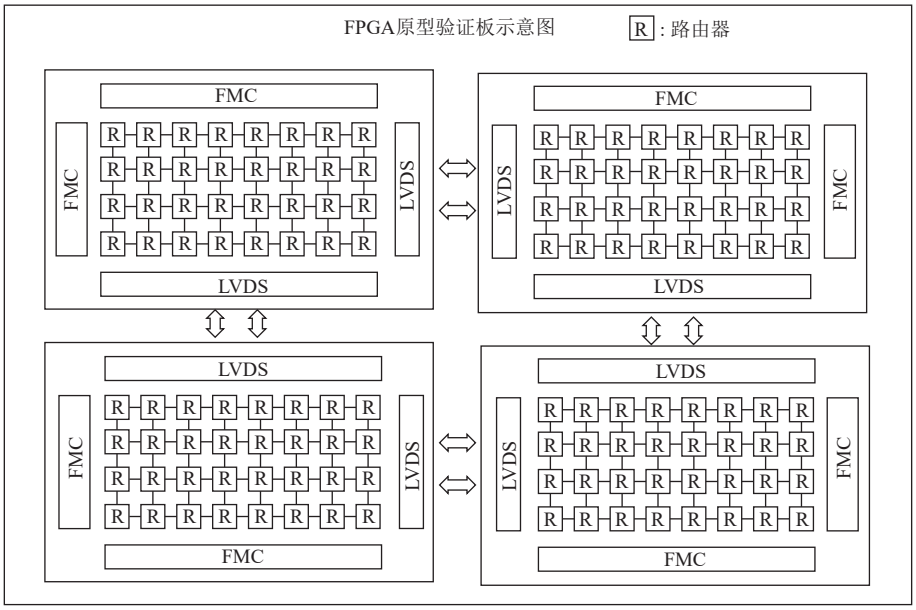


Fig. 22 16×8 Mesh topology

图 22 16×8 Mesh 拓扑

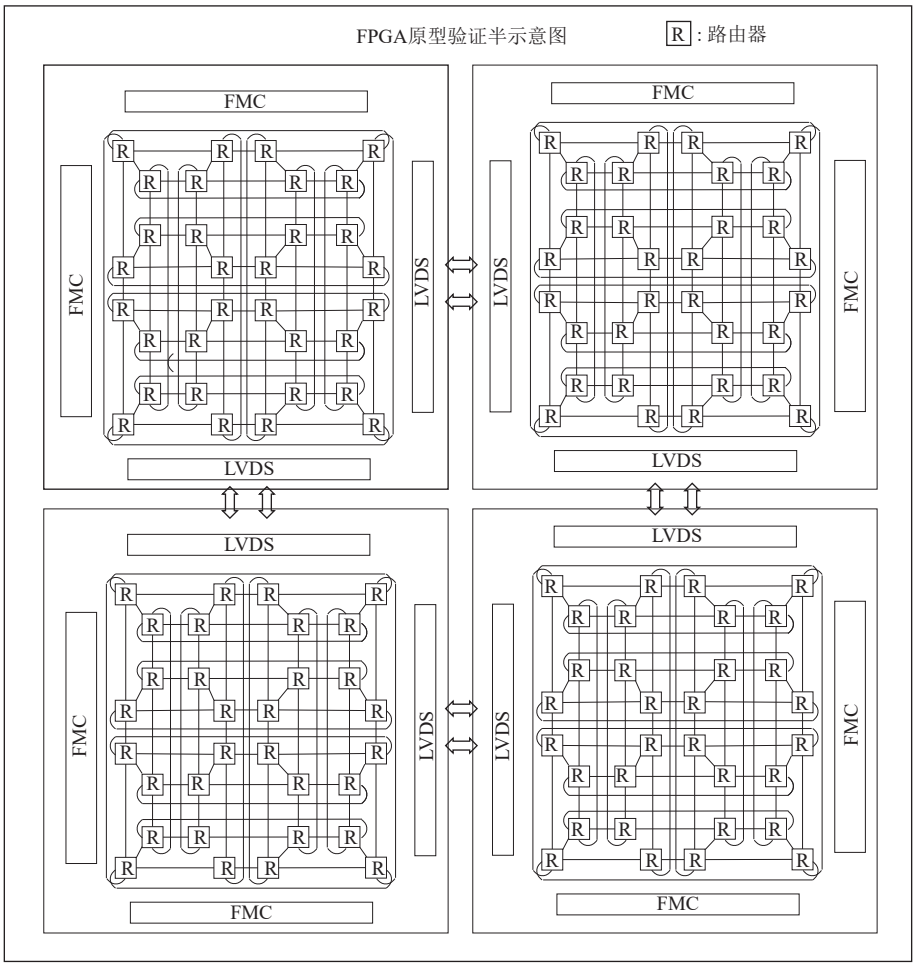


Fig. 23 5D Torus topology

图 23 5D Torus 拓扑

6 个)、5×5 的交叉开关(5D 环绕拓扑为 6×6)和 5 个输出通道(5D 环绕拓扑为 6 个). 每个输入通道还包含 4 个虚拟通道，当输入通道接收到数据包时，根据数据包的目的地址计算要使用的输出通道. 它向仲裁器断言请求信号，以便为所请求的输出通道分配一条数据通路，当请求赢得仲裁时，仲裁者将授予信道访问权限，然后通过交叉开关传输到输出通道.

6.2.3 拓扑结构代码实现

采用 Verilog 语言，完成 2 种拓扑的设计. 一种为网格拓扑，另一种为 5D 环绕拓扑. 其网格拓扑结构实现如图 24 所示，每个路由有 5 个方向，东南西北分别与相邻路由相连接，边缘上的路由没有与之相连接的路由其对应方向为空接. 剩余 1 个方向与本地节点相连接.

5D 环绕拓扑结构实现如图 25 所示，每个路由有 6 个方向，结构分为上下 2 个平面，每个平面边缘上最北边路由经过一跳与最南边路由连接，最西边路由经过一跳与最东边路由连接，同一平面内其余

```
实现mesh拓扑结构算法
输入参数: ROUTER_X, /* 表示路由器网格的 X 方向数量*/
          ROUTER_Y, /* 表示路由器网格的 Y 方向数量*/
          DATA_WIDTH, /* 表示数据宽度*/
          FIFO_DEPTH, /* 表示 FIFO 深度*/
1  for (i = 0; i < ROUTER_X; i = i + 1) begin
2    for (j = 0; j < ROUTER_Y; j = j + 1) begin
3      Router #(.DATA_WIDTH(DATA_WIDTH),.FIFO_DEPTH
       (FIFO_DEPTH));
4       ( .时钟(时钟),.复位(复位);
5         .北方向(j = 0 ? 空连接: 与j−1排南方向连接);
6         .南方向(j = ROUTER_Y−1 ? 空连接: 与j−1排北方向连接);
7         .西方向(i = 0 ? 空连接: 与i−1列东方向连接);
8         .东方向(i = ROUTER_X−1 ? 空连接: 与i+1列西方向连接);
9         .本地(本地));
10     endfor
11 endfor
```

Fig. 24 Pseudocode of Mesh topology structure

图 24 Mesh 拓扑结构伪代码

路由东南西北方向与相邻路由相连，其中 1 个方向连接上下 2 个平面，剩余 1 个方向与本地节点相连.

6.2.4 实验结果

基于 6.1 节中描述的 3 种流量模式，选择 2 种流

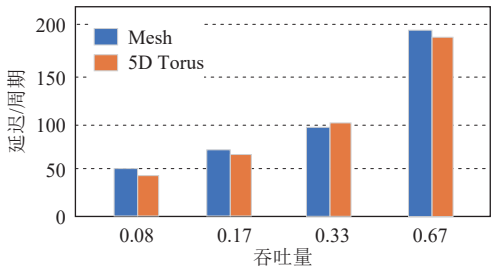
```
实现5D torus拓扑结构算法
输入参数: ROUTER_X, /*表示路由器网格的 X 方向数量*/
          ROUTER_Y, /*表示路由器网格的 Y 方向数量*/
          DATA_WIDTH, /*表示数据宽度*/
          FIFO_DEPTH, /*表示FIFO 深度*/
1  for (i = 0; i < ROUTER_X; i = i + 1) begin
2    for (j = 0; j < ROUTER_Y; j = j + 1) begin
3      Router #(.DATA_WIDTH(DATA_WIDTH),.FIFO_DEPTH
       (FIFO_DEPTH).LAYER(0));
4        (.时钟(时钟), .复位(复位);
5        .北方向(j=0 ? 与ROUTER−1 排南方向连接:与j−1排南方向连接);
6        .南方向(j=ROUTER_Y−1? 与1排北方向连接:与j−1排北方向连接);
7        .西方向(i=0?与ROUTER_X−1 列东方向连接:与i−1列东方向连接);
8        .东方向(i=ROUTER_X−1?与1 列西方向连接:与i+1列西方向连接);
9        .上下方向(上下方向), .本地(本地));
10     Router#(.DATA_WIDTH(DATA_WIDTH),.FIFO_DEPTH
       (FIFO_DEPTH).LAYER(1));
11       (.时钟(时钟), .复位(复位),
12       .北方向(j=0 ? 与ROUTER−1排南方向连接: 与j−1排南方向连接);
13       .南方向(j=ROUTER_Y−1?与1排北方向连接: 与j−1排北方向连接);
14       .西方向(i=0?与ROUTER_X−1列东方向连接: 与i−1列东方向连接);
15       .东方向(i=ROUTER_X−1?与1列西方向连接: 与i+1列西方向连接);
16       .上下方向(上下方向),.本地(本地));
17     endfor
18   endfor
```

Fig. 25 Pseudocode of 5D Torus topology structure

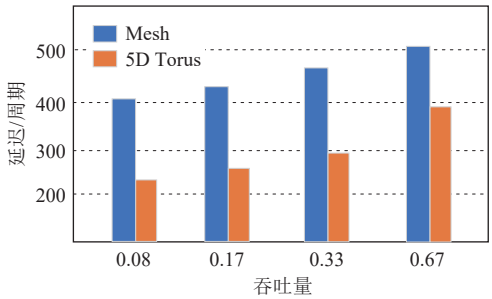
图 25 5D Torus 拓扑结构伪代码

量模式全局归约和全全通信. 我们在原型验证中构建 2 种逻辑拓扑来探讨不同的连接关系对这 2 种流量的适应性. 实验使用 Mesh 网络和带有层次的 5D Torus 拓扑结构来互连, 分别支持 2 种不同的流量模式.

基于整个拓扑结构, 在不同的吞吐量下, 分析从开始发包到所有包都接收完毕总共的延时信息, 以周期为单位. 经过数据对比, 如图 26 可看出, 对于全局归约的流量模式, 使用 Mesh 拓扑和 5D Torus 拓扑的适应性相当; 对于全全通信流量模式, 5D Torus 拓扑的适应性则优于 Mesh 拓扑.



(a) 全局归约流量模式不同逻辑拓扑的吞吐量-延迟曲线



(b) 全全通信流量模式不同逻辑拓扑的吞吐量-延迟曲线

Fig. 26 Throughput-latency curves for different logical topologies with different traffic patterns

图 26 不同流量模式不同逻辑拓扑的吞吐量-延迟曲线

6.3 系统原型实现

在以上分析和实验的基础上, 本文开发了一个小规模的晶圆级计算机原型, 主要包括 2 颗电源模组、2 颗 I/O 模组、14 颗计算模组和内部 16 颗互连芯粒. 该晶圆级计算机的晶圆级互连基板为 12 寸晶圆, 以 8 个功能模组为一个模组群, 通过互连芯粒实现了模组群内的相连. 组装完成的晶圆级系统原型样机如图 26 所示, 表面的电源模组层和功能模组层采用风冷散热方式, 内部的逻辑互连层采用液冷散热方式.

此外, 针对本文所提出的系统架构进行了验证. 将原型样机上的 2 组 I/O 模组通过光纤分别连接到 2 块 PCIe 转接板上, 同时 2 块 PCIe 转接板通过 PCIe 插槽插入服务器, 如图 27 所示, 一侧作为发送端, 另一侧作为接收端. 下面以渲染运算为例对晶圆级系统的通用计算能力进行验证.

Fig. 27 Prototype of the wafer-scale system

图 27 晶圆级系统原型样机

首先, 由发送端生成数据, 其上位机软件界面如图 28 所示. 数据经过光纤、I/O 模组、互连芯粒传送到计算模组, 在计算模组中进行图像渲染运算; 然后, 运算完成后再经过互连系统, 将运算结果传输到接收端一侧的 I/O 模组; 最终, 在接收端上位机将计算结果显示出来, 软件界面如图 29、图 30 所示, 表明晶圆级系统内的计算模组运算正确.

7 结束语

本文提出和设计了可以支持多种负载的通用晶圆级计算机——映天湖. 在系统结构设计方面, 采用了计算模组与互连芯粒设计解耦, 并针对计算模组内部芯粒间以及计算模组与互连芯粒之间的连接进行了标准化设计, 以实现晶圆级计算机对各种代表不同业务的计算模组的支持; 在晶上网络方面, 通过

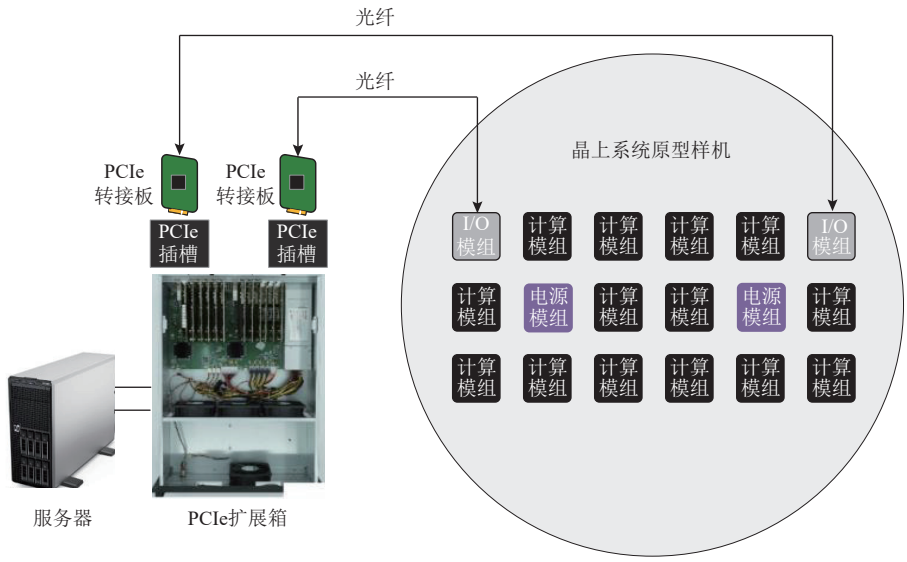


Fig. 28 Illustrtion of the prototype demonstration system

图 28 原型样机演示系统示意图

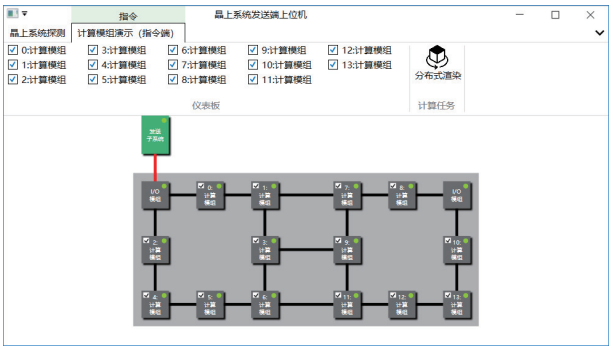


Fig. 29 Host computer software interface at transmitter

图 29 发送端上位机软件界面

Fig. 30 Host computer software interface at receiver

图 30 接收端上位机软件界面

互连芯粒中的可重构设计, 实现了对多种互连网络结构的支持, 针对封装过程可能导致的计算模组实效问题, 提出了拓扑无关的容错控制策略, 以规避晶上网络失效导致的系统故障; 通过模拟仿真、FPGA验证的方式验证了可重构晶上网络, 并开发了一个能够支持 8 个计算模组互连的 12 寸晶圆级原型机进行了测试验证. 映天湖晶圆级计算机有望在高性能计算、智能计算等领域发挥作用, 为解决传统芯片制造技术面临的挑战提供新的实际可行路径.

作者贡献声明: 董文阁提出和完成了晶上网络有关的理论分析; 殷春锁、张志锰、王鹏超、沙江负责晶圆级计算机组件的工作和论文撰写; 王梦雅、朱旻琦负责散热与翘曲问题的分析与设计; 刘宏伟负责早期版本、标准化 I/O 设计工作; 刘宇航分配论文撰写工作并给出重要修改意见; 郝沁汾负责论文撰写和审阅.

参 考 文 献

[1] Theis T N, Wong H S P. The end of Moore's law: A new beginning for information technology[J]. Computing in Science & Engineering, 2017, 19(2): 41−50

[2] Esmaeilzadeh H, Blem E, St. Amant R, et al. Dark silicon and the end of multicore scaling[C]//Proc of the 38th Int Symp on Computer Architecture. San Jose, California: ACM, 2011: 365–376

[3] Lauterbach G. The path to successful wafer-scale integration: The Cerebras story[J]. IEEE Micro, 2021, 41(6): 52−57

[4] Ma Xiaohan, Wang Ying, Wang Yujie, et al. Survey on chiplets: Interface, interconnect and integration methodology[J]. CCF Transactions on High Performance Computing, 2022, 4(1): 43−52

[5] Chuang Yilin, Yuan Chungsheng, Chen Jijan, et al. Unified methodology for heterogeneous integration with CoWoS technology[C]//Proc of 2013 IEEE 63rd Electronic Components and Technology Conf. Piscataway, NJ: IEEE, 2013: 852−859

[6] Xie J Y, Shi Hong, Li Yuan, et al. Enabling the 2.5D

integration[C]//Proc of Int Symp on Microelectronics. San Diego: International Microelectronics Assembly and Packaging Society, 2012: 254−267

[7] Tseng Chienfu, Liu Chungshi, Wu Chihsi, et al. InFO (wafer level integrated fan-out) Technology[C]//Proc of IEEE Electronic Components and Technology Conf (ECTC). Piscataway, NJ: IEEE, 2016, 1−6

[8] Chun Shurong, Kuo Tinhao, Tsai H Y, et al. Info_SoW (System-on-Wafer) for high performance computing[C]//Proc of IEEE Electronic Components and Technology Conf (ECTC). Piscataway, NJ: IEEE, 2020, 1−6

[9] Yu D. TSMC packaging technologies for chiplets and 3D[C]//Proc of Hot chips: A Symp on High Performance Chips (HCS). Piscataway, NJ: IEEE, 2021: 1−47

[10] Lau J H. Semiconductor Advanced Packaging[M]. Singapore: Springer Singapore, 2021

[11] Flack W W, Flores G E. Lithographic manufacturing techniques for wafer scale integration[C]//Proc of Int Conf on Wafer Scale Integration. Piscataway, NJ: IEEE, 1992: 4−13

[12] Li Peijie, Liu Qinrang, Chen Tingyi, et al. Research on the heterogeneous integrated interconnect interface[J]. Integrated Circuits and Embedded Systems, 2024, 24(2): 31−40 (in Chinese)
(李沛杰, 刘勤让, 陈艇异, 等. 异构集成互连接口研究综述[J]. 集成电路与嵌入式系统, 2024, 24(2): 31−40)

[13] Sean Lie. Multi-million core, multi-wafer AI cluster[C]//Proc of Hot Chips: A Symp on High Performance Chips (HCS). Los Alamitos, CA: IEEE Computer Society, 2021: 1–41

[14] Pal S, Petrisko D, Tomei M. Architecting wafer-scale processors-a GPU case study[C]//Proc of 2019 IEEE Int Symp on High Performance Computer Architecture (HPCA). Piscataway, NJ: IEEE, 2019: 250–263

[15] Hu Yang, Lin Xinhan, Wang Huizheng, et al. Wafer-scale computing: Advancements, challenges, and future perspectives[J]. IEEE Circuits and Systems Magazine, 2024, 33(1): 52−81

[16] Lie S. Wafer-scale deep learning[R/OL]. Sunnyvale, CA: Cerebras Systems, 2019[2025-03-01]. https://old.hotchips.org/hc31/HC31_1.13_Cerebras.SeanLie.v02.pdf

[17] Lie S. Cerebras architecture deep dive: First look inside the hardware/software co-design for deep learning[J]. IEEE Micro, 2023, 43(3): 18−30

[18] Lie S. Cerebras architecture deep dive: First look inside the HW/SW co-design for deep learning[C]//Proc of Hot Chips: A Symp on High Performance Chips (HCS). Piscataway, NJ: IEEE, 2022: 1−34

[19] Lie S. Inside the Cerebras wafer-scale cluster[J]. IEEE Micro, 2024, 44(3): 49−57

[20] Lie S. Inside the Cerebras wafer-scale cluster: Cerebras systems[C]// Proc of Hot Chips: A Symp on High Performance Chips (HCS). Piscataway, NJ: IEEE, 2023: 1−41

[21] Lie S. Wafer-Scale engine 3: The largest chip ever built[R/OL]. Sunnyvale, CA: Cerebras Systems, 2024[2025-03-01]. https://8968533.fs1.hubspotusercontent-na1.net/hubfs/8968533/Datasheets/WSE-3%20Datasheet.pdf

[22] Lie S. Cerebras systems: Achieving industry best AI performance through a systems approach[R/OL]. Sunnyvale, CA: Cerebras Systems, 2021[2025-03-01]. https://cerebras.ai/wp-content/uploads/2021/04/Cerebras-CS-2-Whitepaper.pdf

[23] Lie S. Wafer-scale AI: GPU impossible performance[C]//Proc of Hot Chips: A Symp on High Performance Chips (HCS). Piscataway, NJ: IEEE, 2024: 1−71

[24] Talpes E, Williams D, Das Sarma D, et al. The microarchitecture of DOJO, Tesla's exa-scale computer[J]. IEEE Micro, 2023, 43(3): 31−39

[25] Talpes E, Williams D, Das Sarma D. DOJO: The Microarchitecture of Tesla's exa-scale computer[C]//Proc of Hot Chips: A Symp on High Performance Chips (HCS). Piscataway, NJ: IEEE, 2022: 1−28. Components and Technology Conf (ECTC). Piscataway, NJ: IEEE, 2020, 1−6

[26] Pal S, Liu Jingyang, Alam I, et al. Designing a 2048-chiplet, 14336-core wafer-scale processor[C]//Proc of ACM/IEEE Design Automation Conf (DAC). Piscataway, NJ: IEEE, 2021: 1183−1188

[27] Pal S, Petrisko D, Tomei M, et al. Architecting wafer-scale processors-A GPU case study[C]//Proc of IEEE Int Symp on High Performance Computer Architecture (HPCA). Piscataway, NJ: IEEE, 2019: 250−263

[28] Wu Jiangxing, Liu Qinrang, Shen Jianliang, et al. From SoC to SDSoW: A new paradigm for microelectronics development[J]. SCIENTIA SINICA Informationis, 2024, 54(1): 1−19
(邬江兴, 刘勤让, 沈剑良, 等. 从 SoC 到 SDSoW: 微电子发展的新范式[J]. 中国科学: 信息科学, 2024, 54(1): 1−19)

[29] Lü Ping, Liu Qinrang, Wu Jiangxing, et al. New generation software-defined architecture[J]. SCIENTIA SINICA Informationis, 2018, 48(3): 315−328
(吕平, 刘勤让, 邬江兴, 等. 新一代软件定义体系结构[J]. 中国科学: 信息科学, 2018, 48(3): 315−328)

[30] Han Yinhe, Xu Haobo, Lu Meixuan, et al. The big chip: Challenge, model and architecture[J]. Fundamental Research, 2024, 4(6): 1431−1441

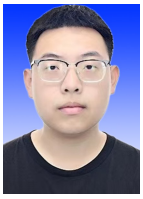
[31] Das Sharma D, Pasdast G, Qian Zhiguo, et al. Universal chiplet interconnect express(UCIe): An open industry standard for innovations with chiplets at package level[J]. IEEE Transactions on Components, Packaging and Manufacturing Technology, 2022, 12(9): 1423−1431

[32] China Electronics Standardization Association. T/CESA 1248—2023 Techical Requirement for Chiplet Interface Bus[S]. Beijing: China Electronics Standardization Association, 2023 (in Chinese)
(中国电子工业标准化技术协会. T/CESA 1248—2023 小芯片接口总线技术要求[S]. 北京: 中国电子工业标准化技术协会, 2023)

[33] Dally W J, Towles B P. Principles and Practices of Interconnection Networks[M]. San Francisco: Elsevier, 2004: 500−536

Dong Wenkuo, born in 2002. Master candidate. His main research interest includes interconnection network optimization.

董文阔, 2002 年生. 硕士研究生. 主要研究方向为互连网络优化.



Yin Chunsuo, born in 2000. PhD candidate. His main research interest includes interconnect technologies for wafer-scale systems.

殷春锁, 2000 年生. 博士研究生. 主要研究方向为晶上系统互连技术.

Zhang Zhimeng, born in 1999. PhD candidate. His main research interest includes neural networks and wafer-scale systems.

张志锰, 1999 年生. 博士研究生. 主要研究方向为神经网络、晶上系统.

Wang Pengchao, born in 1995. Bachelor, engineering. His main research interests include network-on-chip and physical layer controller design.

王鹏超, 1995 年生. 学士，工程师. 主要研究方向为片上网络、物理层控制器设计.

Sha Jiang, born in 1988. PhD. His main research interests include computer architecture, workload characterization, and SoC design.

沙　江, 1988 年生. 博士. 主要研究方向为计算机体系结构、负载特征分析、SoC 设计.

Wang Mengya, born in 1992. Master. Her main research interests include multi-chiplet system design and simulation technologies.

王梦雅, 1992 年生. 硕士. 主要研究方向为多芯粒集成系统设计、仿真技术.

Zhu Minqi, born in 1992. Master. Her main research interests include thermal management, thermal design, and thermal simulation technologies.

朱旻琦, 1992 年生. 硕士. 主要研究方向为热管理、热设计、热仿真技术.

Liu Hongwei, born in 1984. PhD, senior engineer. His main research interests include design of high-performance processors, heterogeneous computing, hardware security and cipher chips, and software-designed chips.

刘宏伟, 1984 年生. 博士，高级工程师. 主要研究方向为高性能处理器设计、异构计算、硬件安全与密码芯片、软件定义芯片.

Liu Yuhang, born in 1985. PhD, associate professor. His main research interests include computer architecture and high-performance computing.

刘宇航, 1985 年生. 博士，副研究员. 主要研究方向为计算机体系结构、高性能计算.

Hao Qinfen, Born in 1969. PhD, professor. Distinguished member of CCF, senior member of IEEE. His main research interests include computer architecture, Chiplet-based processor design, and wafer-scale computing systems.

郝沁汾, 1969 年生. 博士，教授. CCF 杰出会员，IEEE 高级会员. 主要研究方向为计算机体系结构、基于 Chiplet 的处理器设计、晶圆级计算系统.