arXiv:2603.00909v1 [cs.AR] 1 Mar 2026

Capstone：面向粗粒度可重构阵列编译器的功耗受限流水线

Sabrina Yarzada 和 Christopher Torng
南加州大学电气与计算机工程系，美国加利福尼亚州洛杉矶

摘要—粗粒度可重构阵列（CGRAs）日益受到关注，因为它们在保持类似 FPGA 的灵活性的同时，展现出可与 ASIC 相媲美的性能和能效。这些特性使 CGRA 在加速器及其他功耗受限系统中颇具吸引力。然而，现代 CGRA 编译器会积极进行流水线化，以提升频率和性能，往往因此违反严格的功耗预算。我们通过实证表明，在 Cascade 等最先进的 CGRA 编译器中，布局布线后的流水线化会使功耗单调增加，并最终在各种工作负载下超出固定功耗上限。为此，我们推出 *Capstone*，这是 Cascade 的一项功耗感知扩展，它将快速的编译器驻留功耗模型与用户可调的控制器相结合，引导比特流选择过程朝优化目标推进。Capstone 在布局布线后的编译循环中直接预测每次迭代的功耗，并选择一种或一小组布局布线配置，使其中至少一种符合用户指定的功耗上限。因此，我们将目标从不加区分地最大化频率，转变为在离散功耗上限下最大化安全频率。在涵盖基础密集型和稀疏应用的一组内核上，Capstone 在满足功耗上限的同时，尽可能减少剩余功率余量，并保持性能可行。我们的结果表明，上限感知编译既有必要，也切实可行，因为编译器能够主动找到符合上限的工作点，并在功耗约束下提供可预测的性能。

I. 引言

可重构空间计算是计算机体系结构中一个已被充分研究的领域，但我们想提请关注两个具体且新近出现的现代并行趋势。首先，粗粒度可重构阵列（CGRA）已成为专用集成电路与FPGA之间颇具吸引力的体系结构折中方案 [15, 18, 19, 23, 25, 27, 28, 35, 54, 69, 70]。它们将空间编程的灵活性与定制加速器的性能和效率结合起来，因而非常适合神经网络推理、图像处理流水线和传感器融合等快速演进的工作负载。其次，这些相同的工作负载正越来越多地部署在功耗受限的环境中，例如边缘推理、移动增强现实、电池供电相机流水线或热约束片上系统。在这些领域中，超出功耗预算可能导致节流或故障，而未能充分利用功耗预算则会浪费宝贵的余量，而这些余量本可用于换取速度。

尽管人们日益重视效率，大多数现代粗粒度可重构阵列编译器仍只针对性能进行优化。以往的粗粒度可重构阵列编译器流程通常通过传统的调度和布局策略实现高频运行，

Cascade [48] 提出了一种新颖的自动化布局布线后流水线化方法，可沿着已布线的互连和计算路径插入可配置寄存器，以缩短关键路径。该技术显著提升了可实现的时钟频率并改善了能量延迟积（EDP），但额外的寄存器和控制开销也会增加开关活动和静态功耗。随着流水线深度增加，总功耗会单调上升，最终超过实际可行的系统功率上限。在功耗受限的领域中，仅仅“最大化频率”的编译器可能会生成不可行的比特流。

因此，需要采用功耗封顶方案，以确保系统正确运行并提升可靠性。CGRA 功耗封顶的关键挑战在于后端编译器的可见性与控制能力。Cascade [48] 仅在静态时序分析（STA）的指导下执行迭代式布局布线后流水线化，却无法观察每次迭代对功耗的影响。我们的实证分析表明，在各种稠密型和稀疏型内核中，每次流水线化迭代都会使功耗持续上升，直至设计超过实际系统的功耗上限。图1展示了这一趋势：仅增加少量互连寄存器并提高频率，就可能使总功耗远超上限，而编译器却缺乏检测或防止违规的能力。现有的功耗封顶机制，例如运行时动态电压频率调节控制器
[4, 12, 30, 55, 62, 73, 86, 87]，只能在部署后被动响应，因此无法撤销比特流中固有的结构性过度流水线化。因此，编译时的决策本身必须具备功耗感知能力，才能在不影响性能的前提下确保符合功耗上限。

我们的关键洞见是，粗粒度可重构阵列编译器会公开丰富的结构信息，用以描述架构数据通路，例如活动计算瓦片的数量、布线密度和互连利用率。这些数据可用于快速估算功耗，满足内循环决策的时效要求，例如
Cascade [48] 引入的布局布线后流水线化循环。编译器无需调用商用 ASIC 签核工具（这类工具需要进行完整的寄生参数提取，而仿真又受运行时间限制），而是可以根据自身的中间表示学习一个轻量级模型，以预测布局布线后的功耗。将这类模型直接嵌入布局布线后的流水线化循环后，编译器便可将功耗与时序并列为首要优化目标。这一视角将优化目标从

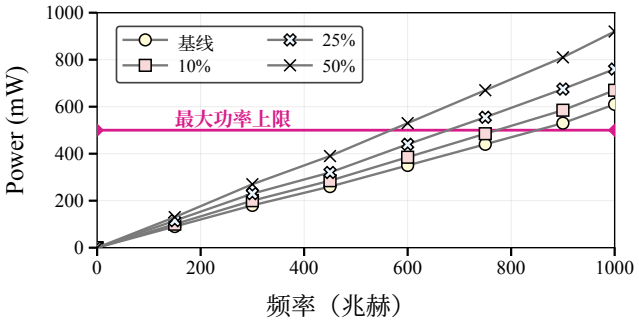


图1. 最先进的 CGRA 编译器为了提高时钟频率而进行流水线化时，可能会在不知不觉中超过功耗上限（水平线）。对于向量逐元素加法内核，Cascade 编译器未来增量性能优化（例如 10%）的预测结果同样都会超过功耗上限。

单纯追求频率最大化，转变为在功耗上限下最大化安全频率。

基于这一观察，我们提出了 Capstone——Cascade [48] 的功耗上限感知扩展。Capstone 集成了一个经过商业 ASIC 签核数据校准的快速、驻留于编译器的功耗模型，以及一个用户可调的控制器，用于引导布局布线后流水线化趋向符合功耗上限的工作点。在每次迭代期间，编译器预测每种配置的功耗，并选择一个或一小组布局布线配置，以确保至少有一个配置在运行时满足用户指定的功耗上限。控制器可根据所需的可靠性和运行时间以多种模式运行。该方法使 Capstone 能够主动落在接近功耗上限的工作点上，最大限度地减少浪费的功率余量，并避免代价高昂的过度流水线化位流。

我们的贡献如下：

- 我们开发了一种层次化的机器学习能耗模型，并使用门级功耗估算监督信号进行训练。该模型联合学习：(i) 编译器可见的能耗事件，(ii) 每个事件的能耗成本，以及 (iii) 到细粒度寄存器传输级功耗实例的映射，从而连接架构表示与物理表示，实现无需动态仿真的准确编译时功耗估算。

- 我们设计了功耗感知编译器控制器，能够明确考虑模型不确定性并提供可靠性界定保证，确保至少一个选定的比特流满足用户设定的功耗上限，同时最大限度地提高可实现频率。

- 我们在基本的密集型和稀疏型粗粒度可重构阵列内核上评估了 Capstone，展示了其准确的编译器级功耗预测、功耗上限合规性，以及相较于无约束编译的最小性能损失。

第二节回顾了 CGRA 架构、编译和功耗估算方面的相关工作与基础知识，

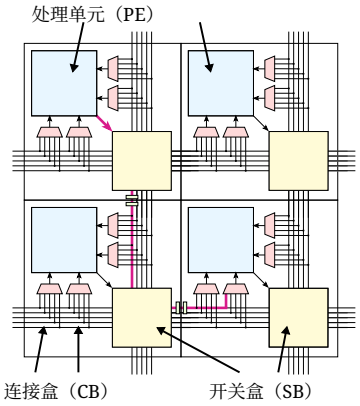


图 2. 粗粒度可重构阵列架构，包括处理单元（PE）瓦片和互连组件（交换盒、连接盒）。粉色路径标示一条具有代表性的布线关键路径；布局布线后流水线化通过沿互连插入绿色寄存器来打断该路径。沿北侧和西侧边缘的输入与输出通道连接到存储体。

由此说明了开发具备功率上限感知能力的编译器的必要性。第三节详细介绍了 Capstone 编译器级能耗模型的设计及其控制器机制。第四节展示了我们的实验结果，并与最先进的 CGRA 编译器进行了比较。结果表明，具备功率上限感知能力的编译既有必要，也切实可行，因为编译器能够主动找到符合功率上限约束的工作点，并在功耗约束下呈现可预测的性能。

II. 背景与相关工作

粗粒度可重构阵列（CGRAs） [15, 18, 19, 23, 25, 27, 28, 35, 54, 69, 70] 是一类可编程加速器，与 FPGA 等细粒度架构相比，兼具架构灵活性、更高的性能和能效 [14, 63]。如图 2 所示， [35] 中的一种典型粗粒度可重构阵列由处理单元（PE）瓦片、存储器（MEM）瓦片和输入/输出瓦片组成，这些瓦片通过带有可配置流水线寄存器的可重构布线互连相连。它们协同工作，在静态调度数据流图中执行加法、乘法和存储器访问等操作。

流水线化对于高吞吐量的粗粒度可重构阵列至关重要，因为较长的互连和计算链会限制时钟频率。Cascade [48]提出了一种布局布线后流水线化技术，利用可配置互连和静态时序分析反馈，沿关键路径迭代插入寄存器。尽管这能提高最高频率并改善能量延迟积，但该方法遵循性能优先策略，忽略了加深流水线所带来的功耗与能耗开销，因为每个阶段都会增加动态开关和静态开销。如图 3 所示，功耗随流水线深度单调上升。由于许多粗粒度可重构阵列的部署都受到严格的热设计或电池功耗预算限制， [1, 33, 34],编译器应在功耗上限内 优化性能，而不是一味追求不受限制的频率。然而，现有的粗粒度可重构阵列编译器 [17, 21, 32, 46, 47, 78, 81, 83], （包括 Cascade [48], ）均缺乏功耗感知。它们会激进地进行流水线化，却不估算或限制功耗。这一局限促使我们开发 Capstone：一种由功耗上限引导的编译器，它将快速能耗建模集成到 Cascade 的布局布线后循环中。

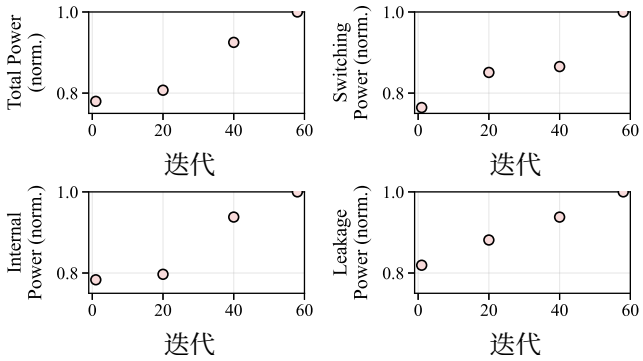


图 3. Cascade [48] 使用的向量逐元素加法内核在不同布局布线后流水线迭代中的归一化功耗。

在 ASIC 流程的多个阶段都可以估算功耗, [7, 29, 75], 但需要在保真度与速度之间权衡。采用布局布线后寄生参数进行门级签核最为准确，但速度太慢，不适用于迭代编译。McPAT 等解析框架 [42] 速度更快，但会忽略主导粗粒度可重构阵列功耗的领域特定组件开销。因此，编译时估算器必须兼具速度和足够的准确性。常见方法包括：（i）利用解析模型将活动事件映射为动态能量和漏电项；（ii）采用基于学习的估算器 [36,37,39,43,52,59,71,80,84]。Capstone 采用基于学习的方法，构建驻留于编译器、由门级数据监督的能耗模型，逐步且系统地逼近更精细、更准确的能耗开销。

编译器级功耗模型不可避免地会与布局布线后估算值存在偏差，因为它们将详细的位级切换和寄生效应压缩为少量集总事件；而在 Cascade 等粗粒度可重构阵列后端中，连续的流水线化迭代会进一步改变布线和有效电容。图 4 展示了商用 ASIC 工具给出的门级签核功耗估算与 Capstone 模型的一个手动校准实例之间的百分比误差，表明虽然提炼后的模型能够恢复正确的数量级，但显著误差仍不可避免。因此，Capstone 将估算不确定性纳入考量，而非假设存在单点的神谕式模型，并采用受鲁棒优化和机会约束优化启发的概率保证 [5, 6, 10, 13, 16, 26, 38, 51, 53, 57, 58,61, 64, 66, 76, 79] ，以确保运行时功耗上限合规，同时最大化频率并限制违规风险。

除了粗粒度可重构阵列之外，CPU、GPU 和加速器领域也开展了能量感知编译与功耗控制的研究 [44, 49, 50, 68, 74]。Simmani [30], 、NeuroMeter [67], 和 PowerProbe [85] 等布局布线后框架可提供准确的估算，但速度太慢，无法用于编译器内层循环；而运行时控制器 [4, 86, 87] 依赖反应式反馈，而非结构性预防。Capstone 与这些工作相辅相成，通过确保运行时符合功耗上限，生成无需采用部署节流等反应式修正措施、且运行功耗已接近目标上限的比特流，例如布局布线后

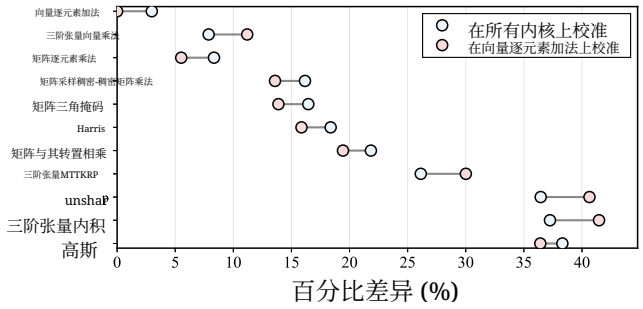


图 4. PTPX 与 Capstone模型功耗估算值对比。红色仅在 vec_elemadd 上校准；蓝色在所有内核上校准。

节流。此外，鲁棒优化、分布鲁棒优化（DRO） [3,41,72,82], 和保形预测 [8,9,20,45,77] 的理论基础也为 Capstone 的功耗感知控制器提供了依据。

总之，先前工作涵盖能量感知编译、布局布线后建模、运行时控制以及不确定性感知优化。然而，我们发现没有任何先前工作能够支持粗粒度可重构阵列编译器内部所需的快速、细粒度、功耗感知决策。Capstone 通过将快速能耗建模、不确定性处理与编译时功率封顶统一到一个集成流程中，填补了这一空白。

III. CAPSTONE：置信度保证下的功耗感知编译

在不增加任何额外基础设施的情况下，于编译时量化粗粒度可重构阵列功耗的一种合理方法是：在流水线处理遍中的选定点对布局布线图进行快照，然后离线构建相应的硬件网表，再运行签核级功率估算工具。这就是我们用于生成图 3 中数据的流程。该方法虽然可行，但立刻暴露出运行时间瓶颈。也就是说，在编译器内部反复调用签核级功率估算对于内层循环而言过于缓慢（需数小时），因而需要一个快速的预测模型。

Capstone 通过以下方式扩展 Cascade 的布局布线后流水线化循环：(i) 一种快速的编译器级能耗模型，可与静态时序分析同步运行；(ii) 一个控制器，用于选择一组K 个布局布线配置候选比特流，使其中至少一个在目标置信水平下满足用户指定的功耗上限，同时尽量减少剩余功率余量。

A. 基于门级监督的能耗模型学习

我们提出一种数据驱动的分层方法，用于学习编译器可见的能耗模型。在该方法中，采用 Synopsys PRIMETIME PX (PTPX) 等工具进行签核质量的门级功率分析，以监督能量事件的发现，以及这些事件到编译器级活动特征的映射。核心目标是在保留门级测量的物理依据的同时，提供适用于内循环优化的快速编译时估算器。图 5 概述了我们提出的流程。编译器和 PTPX 观察

表 I. CAPSTONE 能耗模型符号说明。

符号	说明	符号	说明
E	# 编译器事件数	R	# PRIMETIME PX 行
k	数据集索引	$X^{(k)}$	事件计数向量
$y^{(k)}$	PRIMETIME PX 行功率	W	事件→行权重
$W_{:e}$	列 e 的W	α	能量系数
α_e	事件系数e	$\hat{y}^{(k)}$	预测行功率
$\hat{P}^{(k)}$	预测总功耗	$\mathbf{1}$	全一向量
λ_W	ℓ_1 权重作用于W	λ_α	ℓ_2 权重作用于 α

同一次 执行，但采用不同的保真度；训练过程则学习一种既忠实于 PTPX、又仅凭编译器可见特征即可评估的表示。这种方法衔接了物理和体系结构功耗表示，无需动态仿真即可进行编译时功耗估算。由于学习得到的系数表示固定工作点下每事件能量，编译器便可通过估算事件速率（事件/周期 ×频率）将其转换为功耗，从而实现编译时功耗封顶。具体而言，$P \approx f \sum_e \alpha_e x_e + P_{\text{leak}}$。我们在表 I 中总结了符号说明。

通常，将编译器可见特征映射到门级功耗报告标签，是由人工参与完成的，但学习方法可以取代这一过程，使其实现自动化。将学习得到的映射与自动化回归流程相结合，以调整数值，便形成了一种总体上可自动化的方法，仅使用编译器可见特征，随着时间推移生成质量更高的功耗模型。

对于给定的输入、调度和配置，编译器会在抽象的体系结构数据通路中传播具体的数据流（展示事件／活动计数）。与此同时，PTPX 使用输入数据评估对应的门级数据通路，并报告实例级功耗，包括翻转活动和漏电功耗。因此，我们将编译器和 PTPX 视为观察同一执行轨迹的两种互补视角。我们的模型将 PTPX 层级结构重新组织为乘积和表示（事件计数 ×事件成本），其总和经过调整后，与 PTPX 在固定工作点报告的结果相匹配。核心思想是，学习一种在训练时具有物理依据（通过与 PTPX 对齐）、但在编译时间完全可评估（仅使用编译器可见特征）的表示。

对于每种活动实现 k，令$y^{(k)} \in \mathbb{R}^R_{\geq 0}$为 PTPX 逐行功耗向量（来自层次化功耗报告的行，这些行枚举了门级实例），并令$X^{(k)} \in \mathbb{R}^E_{\geq 0}$为编译器可见事件计数。我们学习一个非负权重矩阵$W \in \mathbb{R}^{R\times E}_{\geq 0}$，用于将每个事件分配到 PTPX 行，并学习非负逐事件系数 $\alpha \in \mathbb{R}^E_{\geq 0}$，以通过纳入逐事件能量大小来缩放平均单实例能耗。矩阵W 的列表示每个事件关注哪些 PTPX 行。这构成了行级预测器 $\hat{y}^{(k)}$ 和总预测功耗 $\hat{P}^{(k)}$的基础。

$$\hat{y}^{(k)} = W\,\mathrm{diag}(\alpha)\,X^{(k)}, \qquad \hat{P}^{(k)} = \mathbf{1}^\top \hat{y}^{(k)}. \tag{1}$$

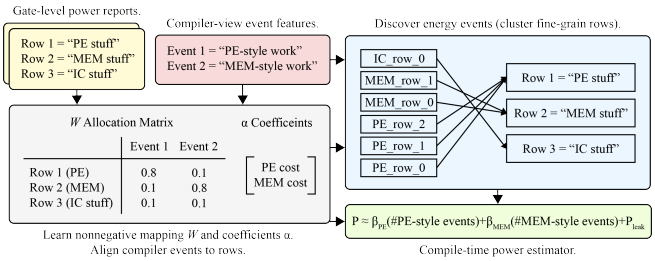


图 5. 层级能量模型学习。我们学习一个非负映射W 和逐事件系数 α，将编译器事件与门级行对齐；根据逐行功耗发现能量事件，然后通过对编译器可见特征进行乘积求和来执行编译时预测。

为方便起见，我们定义有效逐事件系数β_e，将学习得到的行分配（W）折叠为单一的编译时权重。在编译时，预测 $\hat{P}$可简化为对编译器可见事件进行乘积求和：

$$\hat{P}^{(k)} = \mathbf{1}^\top \hat{y}^{(k)} = \sum_{e=1}^{E} \beta_e X_e^{(k)}, \qquad \beta_e \triangleq \alpha_e \sum_{r=1}^{R} W_{re}. \tag{2}$$

学习参数（W α），以最小化与多个数据集上的 PTPX 估算值之间的差异，同时通过非负性和温和的正则化保持其物理意义：

$$\min_{W,\alpha\geq 0} \sum_k \left\| y^{(k)} - \hat{y}^{(k)} \right\|_2^2 + \lambda_W \|W\|_1 + \lambda_\alpha \|\alpha\|_2^2.$$

其中， λ_W 和 λ_α 是超参数。此设置将每个事件分配到各行（W），并设置逐事件成本（α），以使模型的预测更好地与 PTPX 估算值对齐。 ℓ_1 正则化使每个事件（W的每一列）选择少量有意义的行组，从而更易于解释映射。对 α 进行的 ℓ_2 正则化可使学习得到的能量值在不同数据集之间保持平滑且一致。我们联合学习W 和 α，使 $\hat{y}^{(k)}$ 在多个活动样本中与测得的 PTPX 向量 $y^{(k)}$ 相匹配。由此得到的层次化能耗模型通过学习得到的映射保留了与门级功耗估计值之间基于物理原理的关系，同时足够轻量，适用于设计空间探索。

我们通过交替非负更新来求解。固定 α 后，使用带有较小 ℓ_1 惩罚项的非负最小二乘（NNLS）更新W 的每一列。固定W后，通过非负岭回归更新 α。每次更新W时，我们都会重新缩放其各列，并将相应的缩放比例转移到 α中。这样可以消除一种冗余：W 和 α可以沿相反方向缩放，而不改变最终模型。该过程有两个重要特性：（i）它能学习每个事件对应哪些行至关重要（即W中的“注意力”模式）；（ii）它能利用门级功耗层级提供的监督信号，学习每个事件的平均成本（即系数 α）。

我们通过将每个编译器级事件以均匀权重连接到W中的所有 PTPX 行来初始化模型，并且

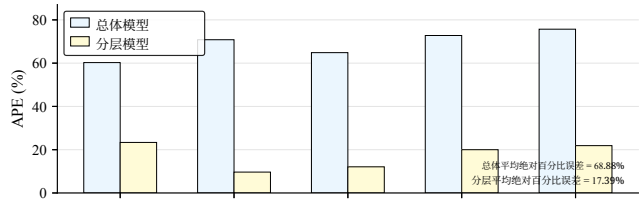


图 6. 总体模型与分层能耗模型在粗粒度可重构阵列内核上的留一交叉验证总功耗预测误差（平均绝对百分比误差）。

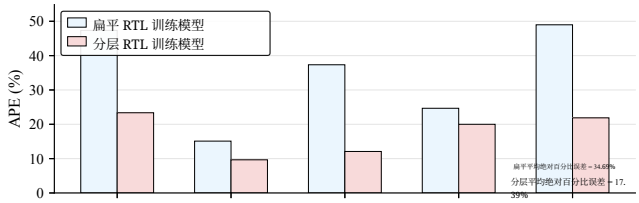


图7. 基于扁平与分层 RTL 功耗报告训练的能耗模型的留一交叉验证总功耗预测误差（平均绝对百分比误差）。

使用简单的非负最小二乘拟合总功耗来设置 α。随后，模型会在所有内核和活动模式上联合校准，因为对于固定的设计和工作点，从事件空间到 PTPX 行的映射应保持稳定。在交替更新期间，W 的每一列都会自然地向与相应事件持续协同变化的行组偏移，从而得到完全由数据驱动的非负映射。超参数（λ_W，λ_α）通过嵌套验证选定；当验证误差不再改善时，训练便通过早停终止。随着新数据集的到来，我们从上一次学习得到的结果（W α）开始，再执行几轮交替更新，使模型能够跟踪数据中的渐进变化，同时保留其底层结构。

我们使用多个活动数据集 进行训练（例如，不同的输入分布、调度方案或配置变体），这些数据集均针对同一抽象数据通路进行测试。每个数据集都会生成一对 $(X^{(k)},\ y^{(k)})$。由于编译器和门级功耗工具以不同的保真度捕获相同的底层执行过程，因此 PTPX 自然可作为利用编译器可见事件进行学习的监督信号。不一致的映射会受到残差 $\|y^{(k)} - \hat{}y^{(k)}\|_2$ 的惩罚。

尽管编译器会将具体数据推入数据通路，但对门级功耗有显著贡献的因素（例如细粒度翻转率、占空比以及跨事件变化的位宽）在预测时并未作为特征显式可用。因此，学习得到的参数表示这些隐藏因素在训练分布下的期望：α_e 是事件e的期望单实例能耗，而W_{re} 是将该事件的能耗分配到行r的期望分配。当部署活动与训练活动相似时，这种边缘化是有效的。事件特征足以捕获数据通路结构，从而使剩余变化在许多实例上被平均掉，并且工作条件 $\{V,f,$工艺角$\}$ 与训练中使用的条件相匹配（或通过单独的参数集处理）。这明确承认了抽象差距，同时使学习模型能够保持平均而言 准确，并足够稳健以驱动编译时决策。

构建的能耗模型的泛化通过以下途径被迭代评估并改进

留一内核法(留一交叉验证)分析。也就是说，在每一折中，我们留出一个内核（包括其所有输入/活动变体），并在其余数据上训练，再用该模型预测留出的内核。这样可以评估模型在不同应用程序之间的迁移能力，这反映了实际部署情况。我们报告样本内拟合和留一交叉验证预测的平均绝对百分比误差及R^2 。对于分层模型，我们还报告行级重建误差 $\|y^{(k)} - \hat{}y^{(k)}\|_2$，并评估 W 在各折之间的稳定性。所有评估均在匹配的工作点进行。如果存在多个 $\{V,f,$角点$\}$，则针对每个点分别训练一个能耗模型。

我们的分层能耗建模方法可灵活适用于不同架构，因为它只需要：(i) 编译器可见且在不同数据通路间保持稳定的活动特征；(ii) 任何具备逐实例分解功能的布局后门级功耗报告工具。通过在多个数据集上联合学习从事件空间到门级功耗报告行的映射，该模型捕捉到了可复用且基于物理原理的关系，从而支持快速设计空间探索。

我们支持两种学习范式。标量聚合基线省略W，并通过 NNLS 加上一个较小的岭正则项，对总量 α 进行拟合，即 $\hat{}P^{(k)} = \sum_e \alpha_e X_e^{(k)}$。这样可以获得具有可解释性的皮焦耳/事件系数，并保证其非负性。在层次化模型中，我们使用完整的 PTPX 向量$y^{(k)}$ ，并通过交替更新联合学习事件成本 α 和将事件映射到 PTPX 行的权重矩阵W ，具体方法如前文所述。在行 级别进行监督（拟合完整的门级层级功耗报告向量，而不仅仅是总功耗）为模型提供了更多信息丰富的约束。这减少了学习得到的因子存在的歧义，提升了对新内核的泛化能力，并生成与 PTPX 一致且清晰的逐行功耗分解。图 6 表明，与 PTPX 估算值相比，这种额外结构能够在所有内核上持续降低总功耗的平均绝对百分比误差（MAPE），从而收紧 Capstone 的功耗估算并减少浪费的余量。

用于训练的PTPX报告可以从层次化RTL网表或更扁平的网表中获得。层次化报告按架构模块（瓦片、互连等）对行进行分组，这使得W中学习得到的列更易于解释，并略微提高准确率，而扁平报告则产生更分散的分配。图7展示了这一权衡。因此，我们为Capstone配备了一个层次化的行监督能耗模型，该模型在层次化门

表II. Capstone 控制器符号说明。

符号	说明	符号	说明
x	布局布线配置	f	时钟频率（兆赫）
图	布局布线图	II	启动间隔
$\hat{P}(x)$	预测功耗（毫瓦）	$P^{\text{true}}(x)$	真实功耗
P^{PTPX}	PTPX参考功耗	μ	$\hat{P}(x)$ 简写
C	功耗上限（毫瓦）	K	# 返回的比特流
γ	保护带	γ_{anc}	锚点保护带
γ_{spec}	推测性保护带	U_{gb}	保护带上界
α	漏覆盖率	$q_\alpha(g)$	保形残差分位数
U_{conf}	保形上界	$\rho(f)$	频率缩放
$\mathscr{D}_{\text{cal}}$	校准集	s_i	单侧残差
g	校准组	$\phi(x)$	特征向量
λ	多样性权重	Δf	最小频率步长
M_i	误差有界模型	ε_k	事件误差界

级功耗报告上训练，同时在需要时仍兼容更扁平的寄存器传输级风格。

B. 编译器级功耗感知控制器

为了在布局布线后流水线化循环中考虑并限制功耗，必须将功耗控制器集成到编译器中。借助预测功耗模型如今提供的量化功耗信息，控制器负责引导优化框架，如图8所述。也就是说，给定用户定义的设计旋钮（包括功耗上限、置信度目标以及要生成的比特流数量 K），编译器必须返回 K 个比特流，在功耗上限内最大化性能，并确保返回的比特流中至少有一个在运行时确实满足功耗上限。我们在表II中总结了相关符号。

如前所述，我们必须正视编译器级功耗模型不可避免地存在不准确性。因此，我们考虑两类保证：概率性保证，旨在将违规概率控制在较小但非零的水平；以及硬性保证，只要真实模型处于指定的有界不确定性族内，就能成立。Capstone 在单个编译器级功耗控制器中提供三种规划器模式，用户可根据可用信息和校准工作量选择；接受的信息越多，模式提供的保真度越高。Capstone I介绍了一种保护带规划器，它依靠保守的静态安全裕量，以由乘性裕量确定的置信度近似实现我们的三个优化目标（图8）。这是最简单的规划器，无需校准数据，但同时也是最难以兼顾所有优化目标的规划器。Capstone II引入了一种由学习型安全裕度控制的共形包络规划器，它以校准数据集为代价，提供有限样本、分布无关的保证：以至少 $1-\alpha$ 的置信度获得安全比特流，且小 α。最后，Capstone III通过要求用户手动指定能耗模型的一组有界误差范围来提供硬性保证。在这些界限内，控制器会穷尽推理所有最坏情况模型误差。

Capstone I：保护带规划器（静态安全裕量）。保护带方法（如算法1所示）通过固定的乘性裕量增强

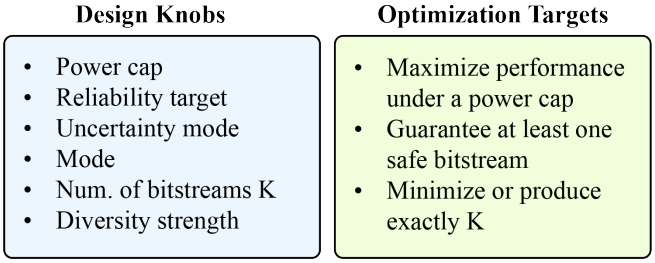


图8. Capstone优化框架。

编译器的功耗预测。令 $\hat{P}(x)$ 表示功耗模型对布局布线配置 x （图，f_{MHz}，II）的估计值（单位：毫瓦）。我们通过保护带构造一个保守的上界，记为$U_{\text{gb}}(x)$：

$$U_{\text{gb}}(x) = (1+\gamma)\hat{P}(x), \qquad \gamma \geq 0,$$

并在$U_{\text{gb}}(x) \leq C$时，将x 声明为安全，其中C 为用户指定的功耗上限。

由于 γ 是静态用户输入，而非从校准数据中学习得到，该方法不提供形式上的统计覆盖保证。相反，它隐式假设未来设计上的最坏情况相对低估 误差不超过 γ。在实践中，保护带参数 γ 可被视为所有建模来源上（未知的）聚合误差分布的紧凑替代，这些来源包括工艺技术失配、PVT变化、工作负载建模不当以及能耗模型中的抽象误差。因此，要在运行时满足功率上限，需要选择足够保守的 γ （例如0.15–0.30），以覆盖建模偏差和分布偏移。在本工作中，我们将 γ 视为用户可控的安全旋钮，并将系统性的保护带选择策略留待未来工作。该方法的关键优势在于其实用性：它不需要校准数据集，部署极为简单，因而构成一个自然的比较基线。

在布局布线后迭代流水线循环中，编译器会生成一系列候选配置 $\{x_t\}$，其频率 $\{f_t\}$ 会随着流水线化程度加深而单调递增。在每次迭代 t 中，我们计算 $\hat{P}(x_t)$ 和$U_{\text{gb}}(x_t)$：

- **安全锚点**。 我们使用一个较大的保护带 γ锚点来维护锚点候选项。每当 Uanchorgb $(x_t) \leq C$ 时，就将锚点更新为 x_t。只要建模误差在 γ锚点范围内，该锚点就是我们所依赖的配置；单独部署它即可满足功率上限。

- **推测集**。 与此同时，我们使用较不保守的保护带γ规格 $< \gamma$锚点，并行跟踪最多$K - 1$个推测候选方案，以便在仍强制执行U规格 gb（x_t） $\leq C$的同时允许更高频率。我们在轻量级特征空间中加入较小的多样性惩罚（例如互连流数量、所用处理单元端口总数、取整后的频率），从而使返回的 K 个位流集合不包含近重复项。这减少了相关误差模式，并

经验上提高了满足“至少有一个是安全的”目标的概率。

- **停止规则**。当最佳推测界仍低于功耗上限且仍有更多关键路径可以断开时，我们继续进行流水线化。当 Uspec gb (x_t) $> C$ 时，我们停止，并返回锚点和排名靠前的推测设计，总计 K 个位流。

由于$\hat{}P$ 和保护带都会随频率变化，Ugb 会随着f 的变化自然收紧或放宽，无需额外调优。用户可通过以下设计旋钮控制权衡：γ_{anchor}、 γ_{spec}、K、功耗上限C以及多样性强度。

算法1保护带规划器

要求： 功耗上限 cap；输出限制 K；保护带 γ_{anc}，γ_{spec}；多样性权重 λ；最小频率步长 Δf；功耗模型 $P(\cdot)$；特征 $\phi(\cdot)$

确保： 最多 K 个位流，其中包含一个安全锚点

初始化： 候选项 $\leftarrow$ /0
锚点 $\leftarrow \perp$
$f_{\text{prev}} \leftarrow 0$
函数 SCORE(x) ▷带多样性惩罚的频率
    返回 $\hat{P}(x) - \lambda \sum_{y \in candidates} \frac{\phi(x)\cdot\phi(y)}{\|\phi(x)\|\|\phi(y)\|}$
对每个频率递增的流水线设计x **执行**
    如果 $\hat{P}(x) < f_{\text{prev}} + \Delta f$ **则**
        继续
    $\mu \leftarrow \hat{P}(x)$
    $U_{anc} \leftarrow (1+\gamma_{anc})\mu$
    $U_{spec} \leftarrow (1+\gamma_{spec})\mu$
    如果$U_{anc} \le$ 功率上限 **则**
        锚点 $\leftarrow x$
    如果$U_{spec} \le$ 功率上限 **则**
        如果 |候选项| $< K - 1$ **则**
            添加 x 到候选项
        else
            $y \leftarrow$ 评分最低的元素，在候选项中
            如果 评分（x）$>$ 评分（y） **则**
                将y替换为x
    else
        中断
    $f_{\text{prev}} \leftarrow f(x)$
$S \leftarrow$ {锚点}$\cup$ 评分候选项中的前 24 个元素
返回S

由于 Capstone I 无法正式保证会返回安全比特流，因此可以自然地为其添加一种简单的运行时故障机制。现代商用片上系统已经提供片上功率计，可定期将总功率测量值写入内存映射寄存器。如果检测到已部署的比特流超过功耗上限，系统便可触发粗粒度的架构级节流（例如大幅降低动态电压频率调节水平，或限制频率），以便及时使功耗回到预算范围内 [2, 31, 40, 60]。实际上，保护带宽度是在稳态性能与此类运行时降频事件的预期发生率之间进行权衡：较宽的保护带可降低触发节流的可能性，而较窄的保护带可提升性能，但会更加依赖回退机制。全面探索此类运行时机制的设计空间不属于本研究的范畴。

Capstone II：共形包络规划器（学习型安全裕度）。 当校准数据稀缺时，保护带方法部署起来非常简单；而共形包络则可以利用校准数据学习有限样本、分布无关误差裕度 [8, 9, 20, 45, 77]，从而在不牺牲保证的前提下收紧保守性。如果正在进行功耗分析的内核与 PTPX 估算训练集中的示例非常相似，则功率估算很可能是准确的。因此，我们可以在其周围施加较小的保护带。如果内核差异很大，并且需要较高程度的组合才能产生能量估算，则其准确性可能较低，我们需要更大的保护带。算法 2 概述了我们的共形方法。

与之前一样，令 $\hat{}P(x)$ 表示功耗模型对布局布线配置 x (图, f_{MHz}, II) 的估算（单位为毫瓦）。来自一组校准元组

$$\mathscr{D}_{\text{cal}} = \{(x_i, P_i^{\text{PTPX}}, \hat{P}_i, g_i)\}_{i=1}^{n},$$

我们构造单侧非一致性分数

$$s_i \;=\; \max\bigl(0,\, P_i^{\text{PTPX}} - \hat{P}_i\bigr),$$

用于量化低估（即可能违反功耗上限的失效模式）。对于用户选择的漏覆盖率$\alpha \in (0,1)$，以及（可选的）分组键 g（例如按内核分组），我们计算经验上分位数

$$q_\alpha(g) \;=\; \text{Quantile}_{1-\alpha}\bigl(\{s_i :\, g_i = g\}\bigr)$$

$$q_\alpha(\text{global}) \;=\; \text{Quantile}_{1-\alpha}\bigl(\{s_i\}_{i=1}^{n}\bigr),$$

并定义保形上界

$$U_{\text{conf}}(x) \;=\; \hat{P}(x) \;+\; \rho(f_{\text{MHz}})\cdot q_\alpha(g(x)),$$

$$\rho(f) \;=\; \max\Bigl(1,\, \frac{f}{f_{\text{ref}}}\Bigr),$$

其中， $\rho(\cdot)$ 可选地根据相对于参考频率 f_{ref} （例如 100 兆赫）的频率缩放裕量，以处理残差随 f近似线性增长的情况。当组 g有足够的校准点时，我们使用 $q\alpha(g)$；否则回退到$q\alpha$（全局）。在校准点与未来样本点满足标准可交换性假设的情况下，单侧保形界满足以下概率

$$\Pr\Bigl(P^{\text{true}}(x) \le U_{\text{conf}}(x)\Bigr) \;\ge\; 1-\alpha$$

如果 $U_{\text{conf}}(x) \le C$，则可确保锚点设计在运行时满足功耗上限，置信度至少为 $1-\alpha$。如果用户要求集合级置信度 $1-\varepsilon$，以确保返回的比特流中至少有一个低于功耗上限，我们可以通过联合界（$K\alpha \le \varepsilon$）将 (α, K) 映射为 ε，而无需独立性假设；如果认为独立性假设合理，也可以通过乘积界 $\prod_{i=1}^{K}(1-\alpha_i) \ge 1-\varepsilon$ 进行映射。

除安全锚点、推测集和停止规则的定义外，规划器逻辑和选择机制与保护带情形（第 III-B 节）完全相同。我们只需将 U_{conf} 替换为 U_{gb}。

算法 2 共形包络规划器

要求： 共形水平 $\alpha_{anc}, \alpha_{spec}$；校准 data $\{(P_j^{\text{PTPX}}, \hat{P}_j, g_j, f_j)\}$；可选的频率缩放标志 s_f；
1:（其他输入同算法 1）
确保： 最多 K 个比特流具有保形功耗保证 tees

2: **离线校准：**
3: **对于**每个校准样本 j **执行**
4: $r_j \leftarrow P_j^{\text{PTPX}} - \hat{P}_j$ ▷模型残差
5: **如果** $s_f = 1$ **则**
6: $r_j \leftarrow r_j / \max(1, f_j/100)$
7: **针对**每个组 g（或单个全局组）**执行**
8: $q_{anc}(g) \leftarrow (1-\alpha_{anc})$ 分位数：$\{r_j : g_j = g\}$
9: $q_{spec}(g) \leftarrow (1-\alpha_{spec})$ 分位数：$\{r_j : g_j = g\}$
10: **函数**上界(μ，f，g，模式)
11: **如果**模式 = 锚点**则**
12: $q \leftarrow q_{anc}(g)$
13: **else**
14: $q \leftarrow q_{spec}(g)$
15: 如果 $s_f = 1$，则缩放为 $\leftarrow \max(1, f/100)$；否则为 1
16: **返回** $\mu +$ 缩放 $\cdot q$
17: **在线选择：**
18: 运行算法1，并按如下方式替换功耗检查：
19: **对**每个流水线设计 x **执行**
20: $\mu \leftarrow P(x)$
21: $g \leftarrow$ x 的组标签（或全局）
22: $U_{anc} \leftarrow$ 上界($\mu, f(x), g$，锚点)
23: $U_{spec} \leftarrow$ 上界($\mu, f(x), g$，规格) ▷所有其他步骤与算法1相同

Capstone III：有界不准确性保证。尽管 Capstone I 和 Capstone II 的保护带规划器和共形包络规划器在经过保守调优后，能够以较高概率满足用户指定的功耗上限，但它们只能在选定的置信水平下提供概率性保证。这源于我们先前的观察：功耗估算，尤其是在计算栈较高层级进行时，必然会产生一定的不准确性。尤其是，共形方法的置信度建立在以下

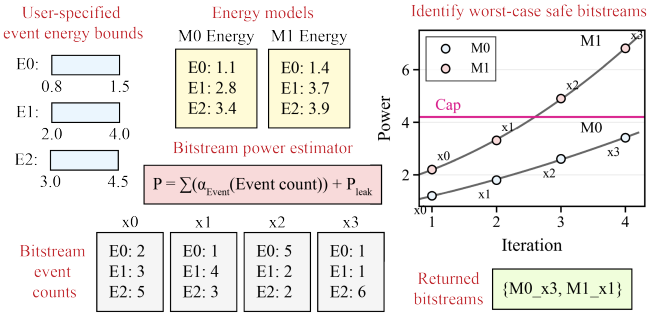


图 9. 能耗建模的变化可能导致在功耗上限下选择不同的可行比特流。

假设之上：其训练集与真实值相关。与此相对，我们也考虑了需要完全保证的情形：编译器返回的布局布线配置比特流中，至少应有一个在实际硅片部署时满足功耗上限，即使与训练数据中观察到的条件存在显著差异的情形下也是如此。

Capstone I 将所有模型不确定性汇总为单一的标量保护带 γ，而不明确建模任何误差范围；Capstone III 则在能耗事件级分解并界定每个误差源。处理无界误差并提供任何形式的保证都是不可行的。不过，如果用户愿意提供能耗模型误差界限的完整规格（即为每个编译器级能耗事件的成本单独指定误差界限），那么就可以对所有模型不准确性情形进行穷尽分析。

因此，我们定义了第三种规划器模式，在该模式下，Capstone 会在用户指定的范围内进行穷举误差搜索。具体而言，我们以标注了用户指定误差界限的能耗模型为基础，实例化多个能耗模型，并以离散步长遍历各种可能评估值的笛卡尔积。误差可以反映建模策略、工艺技术、工艺变化及其他因素所造成的差异，具体由用户决定。例如，图 9 展示了一个基础模型 M_0 和第二个模型 M_1，其中事件 *E0, E1, E2* 的成本估算均在用户指定的误差界限内缩放为其他离散事件成本。对于每个模型 M_i，最终功率估算相对于功耗上限会形成不同的曲线。在所有可能功耗模型构成的穷举集合中，我们只需为每个模型选出唯一一个低于功耗上限且剩余余量最小的比特流。Capstone 会将这些比特流的并集作为一个集合输出，去除相同配置的重复项，并可选择进一步合并。如果用户确保该系列中至少有一个模型 M_i 能够反映真实功耗与编译器基础模型之间的差异，那么为该 M_i 选出的比特流便能保证在运行时满足功耗上限。

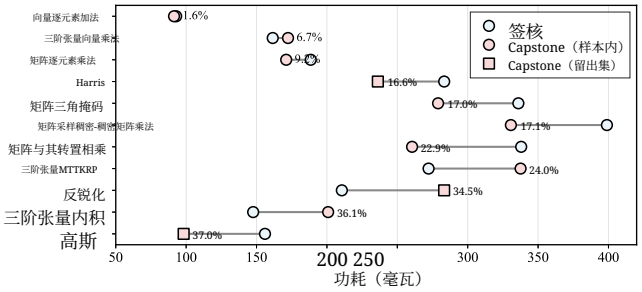


图 10. 100 兆赫下签核结果与 Capstone模型功耗对比。

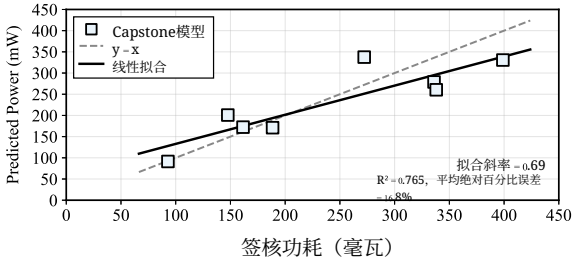


图 11. 门级结果与 Capstone模型偏差对比。每个点代表一个内核。Capstone模型：决定系数 = 0.765，斜率 = 0.69，平均绝对百分比误差 = 16.8%。

IV. 评估

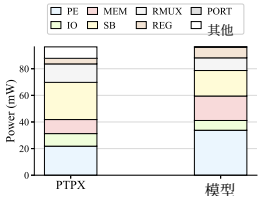
我们的实验针对由 Garnet [65], 生成、采用英特尔 16 纳米工艺的 32×16 粗粒度可重构阵列结构，其中处理单元、存储器和输入/输出瓦片通过可编程路由互连连接。互连段支持可选的流水线寄存器，编译器可在布局布线后迭代流水线循环期间插入这些寄存器。所有实验中，我们固定结构参数（阵列尺寸、交换盒拓扑和布线宽度）。我们评估 Cascade 使用的一组基础密集型和稀疏型内核。密集型内核使用 Halide [56] 和 Clockwork [24] 生成，并表示为 CoreIR [11]数据流图；稀疏型内核则由 Custard 生成，并转换为 SAM [22]数据流图表示。

我们采用基于 Cascade 的 [48] 编译器流程进行调度、布局和布线，并结合 Capstone 的能耗模型和三种控制器模式（第 III 节）。所有实验均使用相同的 ASIC 工具链（Cadence 和 Synopsys）。Cascade 的静态时序分析会报告每次布局布线后流水线迭代可达到的频率，而 Capstone 的能耗模型则在循环中内联运行，无需波形仿真或活动提取。

由于我们无法使用实际的粗粒度可重构阵列平台，因此通过将 Cascade 生成的最终布局布线后应用图转换为寄存器传输级，并对其运行完整的 ASIC 流程（综合、布局布线和签核），来模拟逐内核重配置。我们使用最终的流水线化后图，因为其中包含编译器插入的最多流水线寄存器，从而增加了可用于学习功耗系数的互连和寄存器原语数量，这些系数取自分层报告。最后，

原语	事件	系数	顶部 PTPX 行
处理单元（PE）瓦片	处理单元_瓦片	$\beta_{\text{pe_tiles}}$	处理单元_瓦片
存储器（MEM）瓦片	存储器_瓦片	$\beta_{\text{mem_tiles}}$	存储器_瓦片
REG	寄存器	$\beta_{\text{registers}}$	REG
输入/输出（IO）瓦片	输入/输出_瓦片	$\beta_{\text{IO_tiles}}$	输入/输出_瓦片
IC（存储体）	ic_存储体	$\beta_{\text{ic_sb}}$	存储体_路由
IC（RMUX）ic_rmux		$\beta_{\text{ic_rmux}}$	RMUX_路由
IC（端口）ic_端口		$\beta_{\text{ic_port}}$	端口_路由

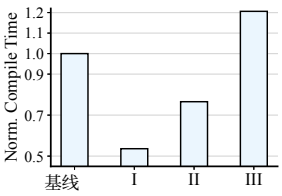
（a）事件到 PTPX 行的映射。



（b）功耗细分。

图12。学习得到的原语系数和模型细分。面板（a）展示了从编译器事件到 PRIMETIME PX 原语所形成的分组。面板（b）比较了 Capstone vec-elemadd 的模型预测和 PTPX 签核分解。

组件	每次迭代时间（秒）	占比
静态时序分析	2.79	56.82%
流水线化	0.36	7.33%
Capstone预测器	0.01	$<1\%$
布局布线后迭代总数	4.91	100%
流水线循环	486.27	–
签核功耗	1.07×10^5 秒	–



(a) 运行时间分解。 (b) 归一化编译时间。

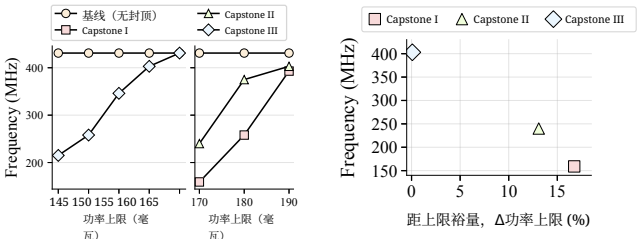
图13. Capstone对tensor3-ttv的运行时间影响。

我们在布局布线后进行签核功耗分析，使用寄生参数和来自编译后的测试平台的门级开关活动，生成用于训练 Capstone 能耗模型的 PTPX报告。由于所有内核都使用相同的校准原语模型和测试平台方法，且各内核之间只有布线后的应用图不同，因此这种方法不会影响结论的有效性。

A. 能耗模型准确性、可解释性与开销

我们从三个维度评估 Capstone 的分层学习能耗模型，这些维度是编译时功率上限优化所必需的：(i) 与签核门级功耗相比的预测准确性，(ii) 学习得到的事件到硬件分解的可解释性，以及 (iii) 在布局布线后优化循环中使用时的运行时间开销。

图 10 在固定工作点下，将 Capstone 对各基准测试的预测功耗与 PTPX 进行比较。每个哑铃图报告签核 PTPX 总量和 Capstone模型估算值，其标注标签表示相对于签核的百分比误差。在各内核上，Capstone 对签核的跟踪足够接近，从而保留了内核按功耗的排序。这对内循环使用十分重要，因为控制器不仅依赖绝对精度，还依赖邻近配置之间稳定的排名，以便决定哪种布局布线方案的功率上限效率最高。图 11 将同一实验汇总为预测功耗相对 PTPX 的校准图，并附有一条 $y=x$ 参考线。Capstone 的预测贴近对角线，实现了高相关性（$R^2 = 0.765$）和低误差（平均绝对百分比误差 $= 16.8\%$），表明单一模型无需任何内核特定调优即可泛化到多样化的内核。拟合直线的斜率为 0.69，截距为



(a) 最大可达频率。 (b) 裕量-频率权衡。

图14. 针对 tensor3-ttv 的 Capstone 控制器评估。(a) 最高频率随功率上限的变化。(b) 在C=170毫瓦（Capstone I/II）、C=160 毫瓦（Capstone III）下，余量与上限之比随频率的变化。

控制器	成功率	中位数 Δ功率上限 (%)	第95百分位 Δ功率上限(%)	平均归一化频率	K
基线	0%	–	–	1.0	4
Capstone I	100%	25.99	26.04	0.46	4
Capstone II	100%	23.14	23.29	0.65	4
Capstone III	100%	16.45	19.93	1.0	90

图15. 所有（内核，功率上限）组合的控制器指标汇总。

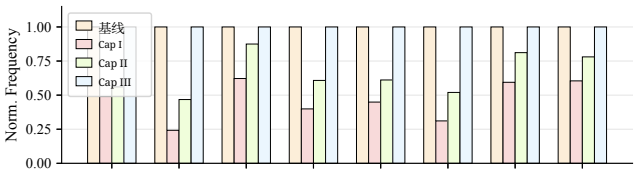


图16. 在固定的逐内核功率上限下，比较无功耗限制与 Capstone I、II 和 III 控制器的各内核归一化频率。

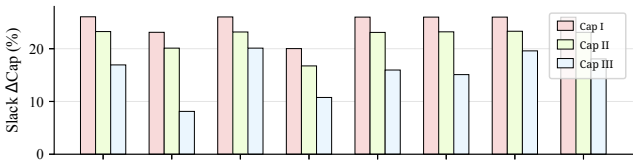


图17. 在固定的逐内核功率上限下，比较无功耗限制基线与 Capstone I、II 和 III 控制器的各内核距上限裕量。

64.12毫瓦，表明这是带有适度加性偏移的系统性缩放偏差，而非纯粹的随机散布。

除了匹配总功耗外，我们还评估学习模型是否能建立从编译器可见事件到门级功耗贡献项的有意义映射。图12(a)报告了学习得到的原语级系数，以及为每个编译器事件组筛选出的主导PTPX 行。学习得到的映射符合架构直觉：与处理单元（PE）相关的事件集中于处理元件瓦片实例，存储器事件与存储器结构相对应，而互连事件则映射到布线和寄存器资源。这表明，该模型并非只是在拟合总量，而是发现了编译器活动与物理硬件功耗之间一致的结构对应关系。图12(b)进一步通过比较一个代表性内核的Capstone原语级分解与PTPX汇总数据，对此进行了验证。两种分解结果非常接近；由于跨内核校准，存在少量预期内的残差。这表明Capstone的乘积和估算器保留了处理单元、存储器和互连功耗之间的物理分布。这种分解还带来了诊断优势。当预测出现偏差时，通常可以将不匹配归因于特定的原语类别（例如处理单元、存储器或互连），而非仅仅表现为难以区分的总功耗误差。总体而言，图12表明Capstone兼具准确性和可解释性，这对于可信赖的编译时优化至关重要。

最后，我们测量了在 Cascade 的布局布线后优化循环中使用 Capstone 预测器所带来的开销。图 13(a) 展示了一个典型内核每次迭代的运行时间分解。静态时序分析占据了迭代时间的主要部分（56.82%），而 Cascade 的流水线化占 7.33%。Capstone 预测器增加了

占每次迭代运行时间的 $<1\%$，因此相较于现有编译流程，其成本几乎可以忽略不计，同时相较于签核功耗估算可提供 $10^7\times$ 的加速。

B. 编译器级功耗感知控制器

我们评估了 Capstone 的三个编译时控制器：Capstone I（保护带）、Capstone II（共形包络）和 Capstone III（有界不准确性）。三种模式均搜索相同的布局布线后流水线设计空间，区别仅在于选择满足功耗上限的解时如何考虑能量模型不确定性。对于 Capstone I，我们使用 $\gamma_{\text{spec}}=0.30$ 和 $\gamma_{\text{anchor}}=0.45$ ；对于 Capstone II，我们使用 $\alpha_{\text{anchor}}=0.005$ 和 $\alpha_{\text{spec}}=0.05$ ，从而在接近功耗上限时提供稳定运行。Capstone III 覆盖 I 和 II 的组合误差空间，因此是最激进的控制器。

我们报告四项指标。成功表示在签核时是否满足硬性功率上限，而比特流数量 K 表示返回的布局布线配置数量。距上限裕量量化了剩余余量。令C 为功率上限（毫瓦），P 为该配置的真实功耗，我们定义

$$\Delta\text{Cap} \triangleq \frac{C-P}{C}\times 100\%,$$

因此，可行性要求 Δ功率上限 ≥ 0，数值越大表示未使用的余量越多。最后，我们报告归一化频率，其定义为同一（内核，C）组合的实现频率除以基线（未限制功率）频率。

为建立直观理解，图14研究了控制器在单个代表性内核上的行为。图14(a)绘制了最大可实现频率与功耗上限C的关系。与无功耗限制基线不同，三种Capstone模式均跟踪该约束。Capstone I最为保守，Capstone II居中，Capstone III实现最高频率。我们报告从其候选集中选出的最激进的Capstone III比特流，它与基线一致，因为

表 III. 在目标功耗上限下与先前 CGRA 编译器的比较。

编译器	工艺	互连结构	工作负载	功率上限（mW）	频率（兆赫）	功耗（毫瓦）	Δ功率上限（%）	成功
RipTide	22FFL	6×6	FFT	365	50/25/12.5	0.24/0.12/0.06	99.93/99.97/99.98	Y/Y/Y
Snafu	22FFL	6×6	FFT	365	50/25/12.5	0.54/0.27/0.135	99.85/99.93/99.97	Y/Y/Y
UE-CGRA	28nm	8×8	FFT	365	750/325/162.5	14/7/3.5	96.16/98.1/99.04	Y/Y/Y
Plasticine	28nm	–	内积	225	280/140/70	18900/9450/4725	-8300/-4100/-2000	Y/Y/Y
Cascade	12nm	32×16	内积	225	481/240.5/120.25	189.1/94.6/47.3	15.96/7.98/3.99	N/N/N
Capstone I	16nm	32×16	内积	225	216/108/54	166.5/83.3/41.6	26/13/6.5	Y/Y/Y
Capstone II	16nm	32×16	内积	225	294/147/73.5	173/86.5/43.3	23.1/11.6/5.8	Y/Y/Y
Capstone III	16nm	32×16	内积	225	481/240.5/120.25	189.1/94.6/47.3	15.96/7.98/3.99	Y/Y/Y
Cascade	12nm	32×16	SDDMM	365	479/239.5/119.75	310.2/115.1/57.55	15.1/7.6/3.8	Y/Y/Y
Capstone I	16nm	32×16	SDDMM	365	149/74.5/37.25	270.9/135.5/67.7	26/13/6.5	Y/Y/Y
Capstone II	16nm	32×16	SDDMM	365	249/124.5/62.25	281.1/140.6/70.3	23.2/11.6/5.8	Y/Y/Y
Capstone III	16nm	32×16	SDDMM	365	479/239.5/119.75	310.2/115.1/57.55	15.1/7.6/3.8	是/是/是

最大程度的流水线化仍低于该上限。图14(b)绘制了余量与上限之比（Δ功率上限 ≥ 0）与所选频率的关系，展示了安全性与性能权衡。Capstone I对保守上界施加功率上限，因此最早停止流水线化，留下最多余量。Capstone II在保持鲁棒性的同时收紧余量。Capstone III仅对预测施加功率上限，进一步进行流水线化，并且运行时最接近该上限。

跨工作负载的结果证实了这些趋势。图16显示，Capstone III 在几乎所有内核上都实现了最高的归一化频率，而 Capstone II 则获得了大部分增益。与此相辅相成，图17显示了保守程度上的相同排序：Capstone I 留下的余量最大，Capstone II 大幅减少了余量，而 Capstone III 在保持可行的同时以最接近功率上限的状态运行。

图15总结了所有（内核，功率上限）组合的可行性和余量。基线的成功率为 0%，因为它不强制执行功耗约束；而在 Capstone I/II 所假设的保守参数下，所有 Capstone 模式均通过构造实现了 100% 的成功率。中位数余量与尾部余量量化了预期的范围：从保守的 Capstone I，到均衡的 Capstone II，再到激进但仍确保不超功率上限的 Capstone III。对于每个控制器，我们评估锚点和推测候选方案，并报告在相应功率上限约束下仍保持可行的最高频率设计。平均而言，推测方案带来4.1×的频率提升，同时相较于锚点将余量降低了 24.6%（较低的 Δ功率上限）。

我们注意到，尽管 Capstone III 的性能最高，但其大量位流可能会在K 不受约束时增加运行时部署的复杂性。在实际系统中，可以通过选择较小且固定的K （如 Capstone I/II），或在必要时采用体系结构节流机制来降低这项开销。在较长的部署周期内，尝试少量候选比特流的成本会逐渐摊薄，因此我们不再进一步研究大K 值下的运行时选择。

最后，图13(b)表明，启用 Capstone 会使编译时间相对于基线最多增加 20%，并且在某些情况下，当提前功率上限检测停止流水线化时，还会减少总编译时间，同时取代了原本代价高昂到难以承受的签核功耗调用。

C. 与最先进的粗粒度可重构阵列的比较

为了将 Capstone 的独特能力与最先进的 CGRA 编译器进行比较，我们针对一类计算密度相近的内核，评估了全部三种优化目标。Capstone 是唯一一种通过学习型能量模型和控制器执行编译器级功耗封顶的方法。其他现有编译器均假定不具备功耗感知能力，但我们考虑了乐观基线：在这些基线中，以固定占空比 [2, 31, 40] 的节流方式实施体系结构级功率封顶。我们利用各编译器的绝对频率与功耗数据来模拟这一行为，不对数据进行重新缩放以归一化节点（工艺技术、可重构阵列尺寸等），从而尽可能准确地刻画这些先前的设计。随后，我们通过连续将这些数值减半，构造激进的 $2\times$ 和 $4\times$ 节流点，这相当于在体系结构层面以一半速率输入数据以降低功耗。表 III 每个单元格中列出的三个数值（原始值、$2\times$、$4\times$ 节流值）反映了这种缩放。

为研究在类似真实部署场景的运行状态下具备功耗上限感知的行为，我们针对每个内核评估了多个绝对功耗上限设置 C （毫瓦）（$C = 365$），其中仅有一组呈现在表 III 中。在所有设置下，Capstone 是唯一始终满足功耗上限并最小化剩余余量 ΔCap 的流程，但代价是生成K 个候选比特流，而不是单一设计点。其他编译器可能只是偶然满足部分功耗上限。相比之下，Capstone 从设计上就以实现功耗上限合规为目标：Capstone I/II 使用保守的保护带，而 Capstone III 则在K 个候选项中进行搜索，以确保至少有一个设计满足功耗上限。在某些情况下，例如报告中的情况，Capstone III 与无功耗限制基线一致，因为最大程度流水线化的设计本身已经低于所选功耗上限。

V. 结论

现代空间可重构加速器编译器对功耗无感知，也没有机制来满足系统——

和组件级功耗预算。我们提出了 Capstone，这是一种编译器级功耗封顶框架，将快速、自动化的能耗建模方法与三种功耗感知编译时控制器模式相结合：(I) 保护带、(II) 共形包络和 (III) 有界误差。我们的方法解决了活动估计和运行时瓶颈问题，因此可以在内部流水线化循环的每次迭代中评估功耗；Capstone 还会明确考虑建模误差可能带来的影响。据我们所知，Capstone 是首个实现粗粒度可重构阵列编译器级功耗封顶的方法，能够针对特定功耗目标调整设计配置，同时优化以保持性能。

参考文献

[1] E. aliagha、N. Charaf、N. K. Venkatesan 和 D. Göhringer。DA-CGRA：面向边缘的领域感知异构粗粒度可重构架构。欧洲微电子数字系统设计会议（*DSD*），2024年8月。[2] J. Aragon、J. Gonzalez 和 A. Gonzalez。通过选择性节流实现功耗感知控制推测。国际高性能计算机体系结构研讨会（*HPCA*），2003年2月。[3] Y. Cao、D. Li、Y. Zhang、Q. Tang、A. Khodaei、H. Zhang 和 Z. Han。在分布鲁棒优化交易能源框架下的多微电网最优能量管理。*IEEE*智能电网汇刊，2022年1月。[4] T. Chen、A. Rucker 和 G. E. Suh。能效硬件加速器的执行时间预测。国际微体系结构研讨会（*MICRO*），2015年10月。[5] Y. Cho 和 I. Yang。基于数据驱动的置信上界随机优化：性能保证与分布鲁棒性。*IEEE*决策与控制会议（*CDC*），2023年3月。[6] M. Chu、D. Allstot、J. Huard 和 K. Wong。基于NSGA的5GHz低噪声压控振荡器寄生效应感知优化。亚洲及南太平洋设计自动化会议（*ASP-DAC*），2004年1月。[7] J. Coburn、S. Ravi 和 A. Raghunathan。功耗仿真：功耗估算的新范式。设计自动化会议（*DAC*），2005年6月。[8] M. Cocheteux、J. Moreau 和 F. Davoine。不确定性感知在线外参标定：一种共形预测方法。*IEEE/CVF*冬季计算机视觉应用会议（*WACV*），2025年2月。[9] K. M. Cohen、S. Park、O. Simeone 和 S. Shamai Shitz。通过共形预测校准无线通信人工智能模型。*IEEE*通信与网络机器学习汇刊，2023年9月。[10] C. Dai、L. Wu 和 H. Wu。具有时空预算约束的基于多频段不确定集的鲁棒SCUC。*IEEE*电力系统汇刊，2016年11月。[11] R. Daly、L. Truong 和 P. Hanrahan。使用CoreIR调用和链接来自多种硬件语言的生成器。开源电子设计自动化技术研讨会（*WOSET*），2018年9月。[12] D. Y. Deng 和 G. E. Suh。用于灵活高效运行时监控的高性能并行加速器。*IEEE/IFIP*可靠系统与网络国际会议，2012年。[13] C. Duan、W. Fang、L. Jiang、L. Yao 和 J. Liu。基于Wasserstein度量的分布鲁棒机会约束近似交流最优潮流。*IEEE*电力系统汇刊，2018年9月。[14] C. Ebeling、D. C. Cronquist 和 P. Franklin。RaPiD：可重构流水线数据通路。现场可编程逻辑国际会议（*FPL*），1996年9月。[15] X. Fan、D. Wu、W. Cao、W. Luk和L. Wang。面向目标推理的流处理双轨粗粒度可重构阵列。*IEEE* 超大规模集成系统汇刊（*TVLSI*），2018年2月。[16] M. J. Feizollahi、S. Ahmed和M. Modarres。具有预算不确定性的串联-并联系统中的稳健冗余分配问题。*IEEE*可靠性汇刊，2014年3月。[17] X. Gao、Y. Qiu、Y. Dai、W. Yin和L. Wang。支持提取通用控制与专用算子的CGRA 前端编译器。亚洲及南太平洋设计自动化会议（*ASP-DAC*），2024年1月。[18] G. Gobieski、A. O. Atli、K. Mai、B. Lucia和N. Beckmann。Snafu：一种超低功耗、能耗极低的 CGRA 生成框架与体系结构。国际计算机体系结构研讨会（*ISCA*），2021年6月。[19] G. Gobieski、S. Ghosh、M. Heule、T. Mowry、T. Nowatzki、N. Beckmann和B. Lucia。RipTide：一种可编程、能耗极低的数据流编译器与体系结构。国际微体系结构研讨会（*MICRO*），2022年10月。[20] V. Gopakumar、A. Gray、J. Oskarsson、L. Zanisi、S. Pamela、D. Giles、M. Kusner和M. P. Deisenroth。使用保形预测对代理模型进行不确定性量化。*arXiv*预印本 *arXiv:2408.09881*，2024年8月。[21] M. Hamzeh、A. Shrivastava和S. Vrudhula。EPIMap：使用满态射将应用程序映射到粗粒度可重构阵列。设计自动化会议（*DAC*），2012年6月。[22] O. Hsu、M. Strange、R. Sharma、J. Won、K. Olukotun、J. S. Emer、M. A. Horowitz和F. Kjølstad。稀疏抽象机。编程语言与操作系统体系结构支持国际会议（*ASPLOS*），2023年3月。[23] Y. Huang、P. Ienne、O. Temam、Y. Chen和C. Wu。弹性 CGRA。国际现场可编程门阵列研讨会（*FPGA*），2013年2月。[24] D. Huff、S. Dai和P. Hanrahan。Clockwork：面向现场可编程门阵列上多速率图像处理应用的资源高效静态调度。国际现场可编程定制计算机研讨会，2021年6月。[25] S. M. A. H. Jafri、O. Bag、A. Hemani、N. Farahini、K. Paul、J. Plosila和H. Tenhunen。使用动态可重构隔离单元的能效感知粗粒度可重构体系结构。国际高质量电子设计研讨会（*ISQED*），2013年3月。[26] X. Jiao、A. Rahimi、B. Narayanaswamy、H. Fatemi、J. P. de Gyvez和R. K. Gupta。用于预测变异性导致的时序错误的基于监督学习的模型。*IEEE*国际新电路与系统会议（*NEWCAS*），2015年6月。[27] M. Karunaratne、A. K. Mohite、T. Mitra和L.-S. Peh。HyCUBE：具有可重构单周期多跳互连的粗粒度可重构阵列。设计自动化会议（*DAC*），2017年6月。[28] C. Kim、M. Chung、Y. Cho、M. Konijnenburg、S. Ryu和J. Kim。ULP-SRP：面向生物医学应用程序的超低功耗三星可重构处理器。国际现场可编程技术会议（*FPT*），2012年12月。[29] D. Kim、A. Izraelevitz、C. Celio、H. Kim、B. Zimmer、Y. Lee、J. Bachrach和K. Asanovi´c。Strober：面向任意寄存器传输级的快速准确基于样本的能耗仿真。*SIGARCH* 计算机体系结构新闻，2016年6月。[30] D. Kim、J. Zhao、J. Bachrach和K. Asanovi´c。Simmani：结合自动信号选择的任意寄存器传输级运行时功耗建模。国际微体系结构研讨会（*MICRO*），2019年10月。[31] Y. Kim、L. K. John、I. Paul、S. Manne和M. Schulte。在可靠性与功耗约束下提升性能。国际计算机辅助设计会议（*ICCAD*），2013年11月。

[32] T. Kojima、B. Adhi、C. Cortes、Y. Tan和K. Sano。一种支持OpenMP应用程序的体系结构无关粗粒度可重构阵列编译器。*IEEE*国际并行与分布式处理研讨会（*IPDPSW*），2022年5月。[33] G. Korol、M. Jordan、M. Brandalero、M. B. Rutzig和A. C. S. Beck。功耗感知的面向阶段可重构架构。*IEEE*国际电子、电路与系统会议（*ICECS*），2019年11月。[34] G. Korol、M. Jordan、R. S. Silva、M. M. Pereira、M. Brandalero、M. B. Rutzig和A. C. S. Beck。面向粗粒度可重构阵列的运行时功耗感知阶段预测器。*IEEE*国际可重构计算与*FPGA*会议（*ReConFig*），2019年12月。[35] K. Koul、J. Melchert、K. Sreedhar、L. Truong、G. Nyengele、K. Zhang、Q. Liu、J. Setter、P.-H. Chen、Y. Mei、M. Strange、R. Daly、C. Donovick、A. Carsello、T. Kong、K. Feng、D. Huff、A. Nayak、R. Setaluri、J. Thomas、N. Bhagdikar、D. Durst、Z. Myers、N. Tsiskaridze、S. Richardson、R. Bahr、K. Fatahalian、P. Hanrahan、C. Barrett、M. Horowitz、C. Torng、F. Kjolstad和P. Raina。AHA：一种面向粗粒度可重构加速器与编译器设计的敏捷方法。*IEEE*嵌入式计算系统汇刊（*TECS*），2023年1月。[36] A. K. A. Kumar、S. Al-Salamin、H. Amrouch和A. Gerstlauer。基于机器学习的CPU微体系结构级功耗建模。*IEEE*计算机汇刊，2023年4月。[37] A. K. A. Kumar和A. Gerstlauer。基于学习的CPU功耗建模。*ACM/IEEE*第一届面向计算机辅助设计的机器学习研讨会（*MLCAD*），2019年。[38] T. A. Le、Q.-T. Vien、H. X. Nguyen、D. W. K. Ng和R. Schober。面向高能效安全SWIPT系统的鲁棒机会约束优化。*IEEE*绿色通信与网络汇刊，2017年9月。[39] D. Lee和A. Gerstlauer。系统级硬件IP的基于学习的细粒度功耗建模。*ACM*电子系统设计自动化汇刊（*TODAES*），2018年2月。[40] S.-W. Lee和J.-L. Gaudiot。高性能多线程架构中基于节流的资源管理。*IEEE*计算机汇刊，2006年9月。[41] B. Li、Y. Tan、A.-G. Wu和G.-R. Duan。一种基于分布鲁棒优化的随机模型预测控制方法。*IEEE*自动控制汇刊，2022年11月。[42] S. Li、J. H. Ahn、R. D. Strong、J. B. Brockman、D. M. Tullsen和N. P. Jouppi。面向多核与众核架构的McPAT框架：同时建模功耗、面积与时序。*ACM*体系结构与代码优化汇刊（*TACO*），10(1):5:1–5:29，2013年4月。[43] Z. Lin、J. Zhao、S. Sinha和W. Zhang。HL-Pow：一种面向高级综合的基于学习的功耗建模框架。*IEEE*亚洲及南太平洋设计自动化会议（*ASP-DAC*），2020年1月。[44] S. Liu、R. N. Pittman、A. Forin和J.-L. Gaudiot。通过可重构系统上的运行时部分重配置实现能效。*ACM*嵌入式计算系统汇刊，2013年3月。[45] C. Lu、Y. Yu、S. P. Karimireddy、M. Jordan和R. Raskar。面向分布式不确定性量化的联邦保形预测器。第*40*届国际机器学习会议论文集（*ICML*），*PMLR 202:22942–22964*，2023年7月。[46] Y. Luo、C. Tan、N. B. Agostini、A. Li、A. Tumeo、N. Dave和T. Geng。ML-CGRA：一种在粗粒度可重构阵列上实现高效机器学习加速的集成编译框架。*IEEE*国际并行与分布式处理研讨会（*IPDPSW*），2023年7月。[47] B. Mei、S. Vernalde、D. Verkest、H. De Man和R. Lauwereins。DRESC：一种面向粗粒度可重构架构的可重定向编译器。*IEEE*国际现场可编程技术会议，2002年12月。[48] J. Melchert、Y. Mei、K. Koul、Q. Liu、M. Horowitz 和 P. Raina。Cascade：面向粗粒度可重构阵列的应用流水线工具包。*IEEE*集成电路与系统计算机辅助设计汇刊（*TCAD*），2024年10月。[49] S. M. Nabavinejad、S. Reda 和 M. Ebrahimi。BatchSizer：DNN推理的功耗-性能权衡。第*26*届亚洲及南太平洋设计自动化会议（*ASPDAC*）论文集，2021年1月。[50] S. M. Nabavinejad、S. Reda 和 M. Ebrahimi。面向 GPU 加速器上 DNN 推理的协同批处理与动态电压频率调节。*IEEE*并行与分布式系统汇刊，2022年1月。[51] D. V. C. d. Nascimento、K. Georgiou、K. I. Eder 和 S. Xavier-de Souza。评估降低电压裕量对多处理器片上系统能效运行的影响。*IEEE*嵌入式系统快报，2024年3月。[52] Y. Nasser、J. Lorandel、J.-C. Prévotet 和 M. Hélard。FPGA和ASIC从RTL到晶体管级的功耗建模与估算技术综述。*IEEE*集成电路与系统计算机辅助设计汇刊，2021年6月。[53] M. Perry、J. Xu、E. Huang 和 C.-H. Chen。深度不确定性下的稳健采样预算分配。*IEEE*系统、人类与控制论：系统汇刊，2022年10月。[54] R. Prabhakar、Y. Zhang、D. Koeplinger、M. Feldman、T. Zhao、S. Hadjis、A. Pedram、C. Kozyrakis 和 K. Olukotun。Plasticine：面向并行模式的可重构架构。国际计算机体系结构研讨会（*ISCA*），2017年6月。[55] K. Prabhu、R. M. Radway、J. Yu、K. Bartolone、M. Giordano、F. Peddinghaus、Y. Urman、W.-S. Khwa、Y.-D. Chih、M.-F. Chang、S. Mitra 和 P. Raina。MINOTAUR：一种边缘端 Transformer 推理与训练加速器，配备 12 MBytes 片上电阻式 RAM和细粒度时空功耗门控。*IEEE*超大规模集成电路技术与电路研讨会（*VLSI*技术与电路），2024年6月。[56] J. Ragan-Kelley、C. Barnes、A. Adams、S. Paris、F. Durand 和 S. Amarasinghe。Halide：一种用于优化图像处理流水线中并行性、局部性和重复计算的语言与编译器。*ACMSigplanNot.48,6*，2013年6月。[57] A. Rahimi、L. Benini 和 R. K. Gupta。分层聚焦护带：一种缓解PVT变化与老化的自适应方法。欧洲设计、自动化与测试会议（*DATE*），2013年3月。[58] A. Rahimi、L. Benini 和 R. K. Gupta。用于缓解静态与动态可变性的应用自适应护带。*IEEE*计算机汇刊，2014年9月。[59] H. V. Ranjitha、S. Hiremath 和 S. G. Langadi。RTL功耗估计：面向SoC功耗签核的设计早期阶段方法。*IEEE*近期电子、信息与通信技术趋势国际会议（*RTEICT*），2018年5月。[60] V. J. Reddi、M. S. Gupta、G. Holloway、G.-Y. Wei、M. D. Smith 和 D. Brooks。电压紧急事件预测：利用特征签名缩小工作裕量。国际高性能计算机体系结构研讨会（*HPCA*），2009年2月。[61] P. K. Rout、D. P. Acharya 和 G. Panda。一种基于多目标优化的快速鲁棒设计方法：用于低功耗、低相位噪声的电流饥饿型压控振荡器。*IEEE*半导体制造汇刊，2014年2月。[62] Y. S. Shao、B. Reagen、G.-Y. Wei 和 D. Brooks。Aladdin：一种预RTL功耗-性能加速器模拟器，可支持定制化架构的大规模设计空间探索。国际计算机体系结构研讨会（*ISCA*），2014年6月。[63] H. Singh、M.-H. Lee、G. Lu、N. Bagherzadeh、F. J. Kurdahi 和 E. M. C. Filho。MorphoSys：一种面向数据并行与计算密集型应用的集成式可重构系统。*IEEE*计算机汇刊，2000年5月。

[64] A. Srivastava、T. Kachru 和 D. Sylvester。考虑工艺变化并采用鲁棒优化的低功耗设计空间探索。*IEEE*集成电路与系统计算机辅助设计汇刊，2007年1月。[65] 斯坦福AHA!敏捷硬件中心。garnet：新一代CGRA生成器。GitHub代码库，2019年。[66] J. Sun、C. Talarico、P. Gupta 和 J. Roveda。一种用于鲁棒模拟/混合信号设计的新型基于不确定性预算的方法。*ACM*电子系统设计自动化汇刊（*TODAES*），2015年11月。[67] T. Tang、S. Li、L. Nai、N. Jouppi 和 Y. Xie。NeuroMeter：一种面向机器学习加速器的功耗、面积与时序综合建模框架（工业专题论文）。国际高性能计算机体系结构研讨会（*HPCA*），2021年2月。[68] H. Tann、S. Hashemi、R. I. Bahar 和 S. Reda。可在运行时配置的深度神经网络：实现能效与准确度权衡。*IEEE/ACM/IFIP*硬件/软件协同设计与系统综合国际会议（*CODES*），2016年10月。[69] C. Torng、P. Pan、Y. Ou、C. Tan 和 C. Batten。面向不规则循环专用化的超弹性CGRA。国际高性能计算机体系结构研讨会（*HPCA*），2021年2月。[70] A. Vasilyev、N. Bhagdikar、A. Pedram、S. Richardson、S. Kvatinsky 和 M. Horowitz。评估面向成像与视觉应用的可编程体系结构。国际微体系结构研讨会（*MICRO*），2016年10月。[71] J. Wang、K. Li、J. Chen、R. Shi、L. Chen 和 W. Chen。利用机器学习预测RTL级电路的功耗。*IEEE*电子设计自动化国际研讨会（*ISEDA*），2023年5月。[72] W. Wei、F. Liu 和 S. Mei。能源与备用调度的分布鲁棒协同优化。*IEEE*可持续能源汇刊，2016年1月。[73] C. Xu、F. X. Lin、Y. Wang 和 L. Zhong。操作系统级设备运行时功耗管理自动化。编程语言与操作系统体系结构支持国际会议（*ASPLOS*），2015年3月。[74] S. Xu 和 B. C. Schafer。近似可重构硬件加速器：使微体系结构适应动态工作负载。国际计算机设计会议（*ICCD*），2017年11月。[75] J. Yang、L. Ma、K. Zhao、Y. Cai 和 T.-F. Ngai。利用RTL插桩进行早期实时SoC功耗估算。亚洲及南太平洋设计自动化会议（*ASPDAC*），2015年1月。

[76] L. Yang、Y. Xu、H. Sun 和 W. Wu。《不确定性下分布鲁棒联合机会约束最优潮流的易处理凸近似》。*IEEE*电力系统汇刊，2022年5月。[77] C. Yeh、N. Christianson、A. Wu、A. Wierman 和 Y. Yue。《不确定性优化的端到端共形校准》。*arXiv*预印本 *arXiv:2409.20534*，2024年9月。[78] T. Yu、O. Ragheb、S. Wicklund 和 J. Anderson。《MLIR-to-CGRA：面向粗粒度可重构阵列的通用MLIR编译器框架》。专用系统、体系结构与处理器国际会议 *(ASAP)*，2024年7月。[79] H. Zhang 和 P. Li。《不确定性下最优潮流的机会约束规划》。*IEEE*电力系统汇刊，2011年11月。[80] Y. Zhang、H. Ren 和 B. Khailany。《GRANNITE：用于可迁移功耗估算的图神经网络推理》。设计自动化会议 *(DAC)*，2020年7月。[81] Y. Zhang、N. Zhang、T. Zhao、M. Vilim、M. Shahbaz 和 K. Olukotun。《SARA：扩展可重构数据流加速器》。国际计算机体系结构研讨会 *(ISCA)*，2021年6月。[82] Z. Zhang、S. Lin、M. Dedeoglu、K. Ding 和 J. Zhang。《面向边缘智能的数据驱动分布鲁棒优化》。*IEEE*计算机通信会议，2020年7月。[83] Z. Zhao、W. Sheng、Q. Wang、W. Yin、P. Ye、J. Li 和 Z. Mao。《迈向更高性能和更鲁棒的粗粒度可重构阵列模调度编译》。*IEEE*并行与分布式系统汇刊，2020年4月。[84] Y. Zhou、H. Ren、Y. Zhang、B. Keller、B. Khailany 和 Z. Zhang。《PRIMAL：利用机器学习进行功耗推断》。设计自动化会议 *(DAC)*，2019年6月。[85] D. Zoni、L. Cremona 和 W. Fornaciari。《PowerProbe：通过自动寄存器传输级插桩实现运行时功耗建模》。欧洲设计、自动化与测试会议 *(DATE)*，2018年3月。[86] D. Zoni、L. Cremona 和 W. Fornaciari。《面向通用加速器和CPU的全数字能量约束控制器》。*IEEE*嵌入式系统快报，2020年3月。[87] D. Zoni、A. Galimberti 和 W. Fornaciari。《边缘运行时功耗监测器综述》。*ACM*计算调查，2023年7月。