arXiv:2601.13610v1 [cs.CR] 20 Jan 2026

面向片上网络架构的全有或全无变换安全多路径路由

Hansika Weerasena、Matthew Randall 和 Prabhat Mishra佛罗里达大学计算机与信息科学与工程系，美国佛罗里达州盖恩斯维尔

摘要——确保片上网络（NoC）的安全对于设计可信的片上网络系统级芯片（SoC）架构至关重要。虽然存在各种利用片上通信漏洞的威胁，但通过恶意节点实施的窃听攻击最为常见且隐蔽。尽管加密能够保障数据包的机密性，但可能会给资源受限的片上系统带来无法接受的开销。本文提出了一种轻量级机密性保护框架，将基于拟群的全有或全无变换（AONT）与片上网络片上系统中的安全多路径路由相结合。通过对每个数据包应用 AONT，并将变换后的数据块分发到多条互不重叠的路径上，我们确保任何中间路由器都无法在未获得所有数据块的情况下重构原始数据。大量实验评估表明，该方法能够有效缓解恶意路由器实施的窃听攻击，同时仅产生可忽略的面积和性能开销。结果还表明，与使用传统加密防御窃听攻击相比，基于 AONT 的多路径路由可将开销降低 7.3 倍。

I. 引言

现代异构片上系统（SoC）和多处理器片上系统（MPSoC）如今集成了数百个知识产权（IP）核，以满足日益增长的应用需求。片上网络（NoC）通过一组互连的路由器，实现这些核心之间可扩展、低延迟、高带宽且高效的通信。由于片上网络促进了所有知识产权核之间的通信，因此它构成了理想的威胁载体——攻击者可借此大肆利用，带来重大的安全挑战。受市场时机限制而集成第三方知识产权核，以及依赖不可信的晶圆代工厂，都增加了硬件木马和路由器遭到入侵的风险，进而可能损害系统机密性、完整性或可用性。

针对片上网络在不同安全目标方面的漏洞，已有大量研究 [1]。图1展示了窃听攻击的一个示例，它是

片上通信中最常见且隐蔽的威胁。窃听是一种被动攻击，遭入侵的路由器会在流控单元穿过网络时悄无声息地进行监视，可能泄露加密密钥、推理结果或内存地址等敏感信息。传统防御措施（包括密码加密，如 AES-CTR）可提供强有力的保密性保证，但可能带来显著的面积、性能和能耗开销，因此不适用于许多片上系统设计。这些局限凸显了对轻量级解决方案的需求，以提供强有力的保护

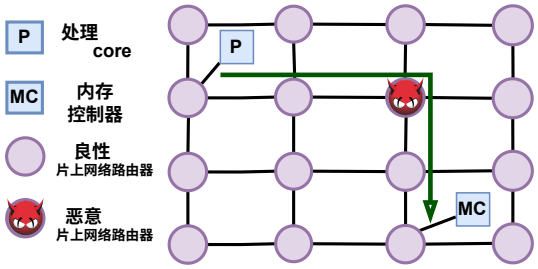

图1：4×4网格片上网络中的窃听攻击示例。恶意路由器可以窃听处理核心（P）与目录控制器（DC）之间的通信。

以抵御窃听等被动威胁。我们提出了一种轻量级的片上通信保密性保护框架，使其非常适用于资源受限的片上系统。本文的贡献如下。

- 我们提出了一种轻量级的全有或全无变换（AONT），将一条消息拆分为多个块，必须使用所有块才能重构原始消息。
- 我们提出一种用于网格片上网络的非重叠多路径路由策略，以分发变换后的数据块。
- 我们在多种场景下评估了我们的方法，结果表明，该方法在仅带来极小面积和性能开销的同时，实现了强大的机密性。

尽管基于填充与筛选的全有或全无变换已用于保护片上网络 [2]，，我们的工作将全有或全无变换与不相交多路径路由协同集成，消除了冗余流量并实现了更优的性能。

II. 威胁模型

在基于片上网络的多处理器片上系统中，片上网络在为基于目录的协议（如 MESI 或 MOESI）传输缓存一致性消息方面发挥着关键作用。当本地缓存未命中时，核心会向共享二级缓存或末级缓存（LLC）发送控制数据包（例如加载或存储请求）；如果数据未被缓存，末级缓存可能会将请求转发至内存。数据一旦从末级缓存或主内存中定位，就会以数据包的形式将响应发送回请求核心。这种往返通信完全通过片上网络进行，通常会经过多个中间路由器。在此，我们假设采用广泛使用的网格片上网络拓扑。图1展示了4×4网格片上网络中的一次窃听攻击，其中单个恶意路由器被策略性地部署用于窃听。单个恶意路由器被广泛用作一种现实的威胁模型，因为其面积和功耗开销极小，在设计阶段很难被检测到，

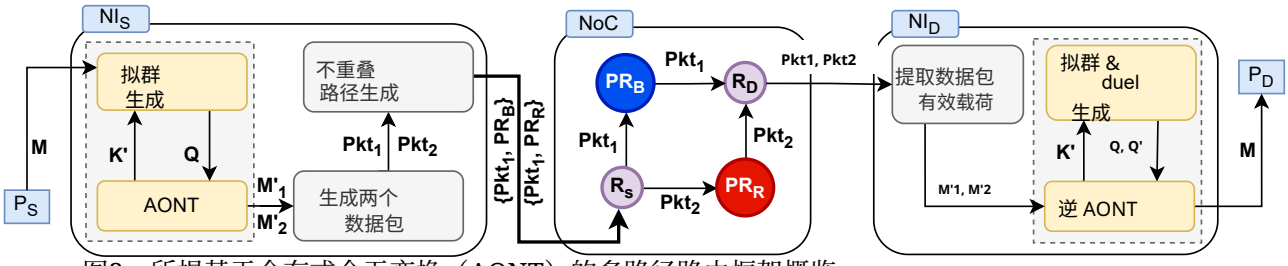

图2：所提基于全有或全无变换（AONT）的多路径路由框架概览。

恶意路由器可以拦截合法节点之间传输中的数据包。这使其能够提取敏感信息，例如加密密钥、专有的深度学习模型参数，或源自内存的用户隐私数据，从而对安全与安保关键系统构成严重的机密性威胁。

III. 基于全有或全无变换的安全多路径路由

为了增强片上通信的机密性，我们在网格片上网络中实现了一种轻量级多路径路由机制，并将其与全有或全无变换（AONT）相结合。我们的方法包含两个主要部分：(1) 基于拟群的全有或全无变换，以及 (2) 不重叠多路径路由。图2展示了从源节点 P_S 向目的节点 P_D发送有效载荷M 时的处理过程。源网络接口（NI_S）首先生成一个拟群 Q。AONT 使用 Q 将 M 变换为两个伪消息：M'_1和M'_2，它们分别作为数据包 Pkt_1和 Pkt_2的有效载荷。不重叠路径生成会选出两个枢纽路由器（PR_B和 PR_R）；Pkt_1经由 PR_B路由，Pkt_2经由 PR_R路由，随后两者均到达目的路由器（R_D）。不重叠路径确保没有任何单个恶意路由器能够访问所有 AONT 块，从而保证完全机密性，杜绝哪怕部分数据泄漏的可能。在目的地，网络接口提取两个数据包的有效载荷，即 M'_1和 M'_2。最后，逆 AONT 使用拟群及其对偶来恢复原始消息M。本节其余部分将介绍我们方法中的三个重要步骤：AONT 与分组化、多路径路由，以及数据包重组与逆 AONT。

A. 全有或全无变换与分组化

全有或全无变换是一种密码学原语，它将一条消息转换为多个块，必须集齐所有块才能恢复原始消息；缺少任意一个块都会导致消息甚至无法部分恢复。算法1描述了如何使用拟群和数据包生成来实现全有或全无变换。消息会重新排版，以支持在基数 n 下进行算术运算（第 2 行）。这里，消息被分为 s 个块 $B_1, B_2, \ldots, B_s$，其中每个块 $B_i = (h_{i1}, h_{i2}, \ldots, h_{in})$包含 n 个元素。每个元素 h_{ij} 以基数 n 表示，使用 $\log_2 n$位。例如，当 $n = 4$ 时，$(01)_2$ 映射为 1，$(11)_2$映射为 3。为降低进制转换的计算开销，n被选为 2 的幂，因此采用费马素数作为 p。费马素数的形式为 $2^{2^z}+1$，其中 z是整数，可通过基数为 n 的值高效地进行模运算。调用算法 2可构造拟群

算法1 全有或全无变换与分组化

1: **函数** 全有或全无变换与分组化(M)
2: $M \leftarrow B_1 \| \ldots \| B_s$其中 $B_i = (h_{i1}, \ldots, h_{in})$
3: $K' \leftarrow$ 随机排列
4: $\langle Q, \bullet \rangle, _ \leftarrow Quasi(K')$
5: $l_1 \leftarrow k_1, l_2 \leftarrow k_2 \bullet l_1, \ldots, l_n \leftarrow k_n \bullet l_{n-1}$和 $l \leftarrow l_n$
6: **对于** $i = 1$到 s，**执行**
7: $I(i) \leftarrow n$ 的基表示，其中 i 与 $|I(i)| = n$
8: $R(i) \leftarrow (r_{i1}, \ldots, r_{in})$其中 $r_{in} \leftarrow l \bullet i_{in}, r_{in-1} \leftarrow r_{in} \bullet i_{in-1}, \ldots, r_{i1} \leftarrow r_{i2} \bullet i_{i1}$
9: $B'_i \leftarrow (h'_{i1}, \ldots, h'_{in})$其中 $h'_{ij} \leftarrow r_{ij} \bullet h_{ij}$
10: $C_1 \leftarrow B'_1$
11: **对于** $i = 2$至 s，**执行**
12: $C_i \leftarrow C_{i-1} * B_i$
13: $B'_{s+1} \leftarrow C_s * K'$
14: $M'_1, M'_2 \leftarrow B'_1 \| \ldots \| B'_{s/2}, B'_{s/2+1} \| \ldots \| B'_{s+1}$
15: $Pkt_1, Pkt_2 \leftarrow$头部(Pkt_1)$\|M'_1$,头部(Pkt_2) $\|M'_2$
16: **返回** $Pkt_1, \quad Pkt_2$

$\langle Q, \bullet \rangle$ 使用随机生成的密钥 K'（第 4 行）。阶为 n的有限拟群在集合 Q 上定义了二元运算 $\bullet$，使得对于任意 $a, b \in Q$，方程 $a \bullet x = b$和$y \bullet a = b$关于 x 和 y均有唯一解。主元 l 通过对 K' 中的元素迭代应用 $\bullet$得出（第 5 行）。

第 6–9 行将每个消息块 B_i 转换为伪消息块 B'_i，具体如下：第 7 行将整数 i表示为以 n 为底的长度为 n 的向量 $I(i)$（例如，$i = 1 = (0001)_4$ 以 4 为底时变为 (4, 4, 4, 1)，其中字符 4 表示 0）。在第 8 行，通过对 $I(i)$中的元素应用二元运算符 $\bullet$ 来计算中间块 $R(i)$，并使用主元l 作为 n^{th} 元素。最后，第 9 行通过对 B_i 和$E(i)$逐元素应用 $\bullet$，生成伪块 B'_i。第 10–13 行通过迭代组合块 $C_i = C_{i-1} * B_i$来计算最终伪块 B'_{s+1}，其中 $C_1 = B'_1$，且对每个元素，$*$ 定义为 $(a_i \cdot b_i) \bmod p$。然后，将 B'_i连接到 $B'_{s/2}$，并通过连接其余 AONT 块来形成伪消息 M'_1（第 14 行）。最后，以 M'_1为有效载荷创建 Pkt_1，并以 M'_2 为有效载荷创建 Pkt_2（第 15 行）。*head()* 函数将为相应的数据包生成报头。

1) 拟群生成：我们的方法采用 Marnes 等人提出的拟群生成技术 [3]，因为该技术能够快速、随机地完成构造过程（算法 2）。拉丁方（最小二乘）是一个由 n 个符号构成的 $n \times n$ 矩阵，每个符号在每一行和每一列中都恰好出现一次

。阶为n 的拟群的乘法表是一个大小相同的拉丁方。在我们的情形中，$n = p - 1$，其中 p 是素数。拉丁方的第一行初始化为一个随机排列，并作为输入提供给算法（第 2 行）。对于 $i = 2, \ldots, n$，i 行中的每个元素都使用公式 $(i \times q_{1j}) \bmod p$ 计算（第 5 行）。生成的拉丁方用于 AONT 变换。算法 2 还会同时生成该拟群的对偶，不过只有接收端的逆 AONT 需要用到它。对于第 i 行、第 j 列中的每个元素 q_{ij}，相应的对偶元素 q'_{ix} 设为 j，其中 $x = q_{ij}$ （第 6 行和第 7 行）。

算法 2 生成带对偶的拟群

1: **函数** $Quasi(K')$
2: $q_{11} \| q_{12} \| \ldots \| q_{1n} \leftarrow K'$
3: **对于** $i = 1$ 到 n，**执行**
4: **对于** $j = 1$ 到 n，**执行**
5: $q_{ij} \leftarrow (i \times q_{1j})$ 模 p
6: $x \leftarrow q_{ij}$
7: $q'_{ix} \leftarrow j$
8: $\langle Q, \bullet \rangle, \langle Q, \circ \rangle \leftarrow q_{ij}$ 的最小二乘、q'_{ix} 的最小二乘
9: **返回** $\langle Q, \bullet \rangle$，$\langle Q, \circ \rangle$

B. 不重叠路径生成

Pkt_1 和 Pkt_2 分别沿两条互不相交的路径进行路由，以确保没有单个路由器能够截获所有 AONT 块。片上网络根据源节点相对于目的节点的位置划分为蓝色和红色两个区域。蓝色区域按照表 I 中的规则选定，其余区域则构成红色区域。图 3 展示了目的节点位于源节点右下方的情况。红色区域采用 XY 路由（先 X 后 Y），蓝色区域采用 YX 路由（先 Y 后 X）。请注意，XY 和 YX 路由均分虚拟通道，以避免死锁。每个区域随机选取一个枢纽路由器（PR）。源节点将消息的一部分发送至红色区域的 PR，另一部分发送至蓝色区域的 PR，随后两者都将各自的部分转发至实际目的节点。为支持经由中间节点进行两跳路由，每个数据包（Pkt_1和 Pkt_2）都会在数据包头中将枢纽路由器和最终目的地（$fin\ \ id$）作为元数据。数据包到达枢纽路由器后，数据包头会更新为指向最终目的地。

表 I：蓝色区域选择规则

情况/场景	蓝色区域正方形
目的地位于源的右上方	位于源的上方且在目的地左侧
目的地位于源的右下方	位于源下方且位于目的地左侧
目的地位于源的左上方	位于源上方且位于目的地右侧
目的地位于源的左下方	位于源下方且位于目的地右侧

当源节点与目的节点位于同一行或同一列时，会出现两种特殊情况。在这些情况下，片上网络分别被划分为上下两半或左右两半，并且每个数据包经由不同半区中的一个枢纽路由器进行路由。

然而，在这些场景中使用默认路由策略可能会导致路径重叠。为此，我们引入了一个可选的翻转路由标志，使蓝色路径能够在蓝色枢纽路由器处切换路由策略（从 YX 切换为 XY）。

当源节点和目的节点位于同一行或同一列时，会触发这种切换，因为空间约束可能导致路径重叠。切换蓝色路径的路由策略可确保这两种特殊情况下的路由彼此不相交。这种方法保证了两条路径在空间上分离，从而确保机密性，不受窃听恶意路由器的威胁。

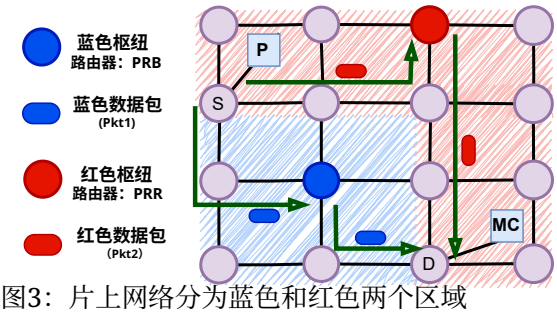

图3：片上网络分为蓝色和红色两个区域

C. 数据包重组与逆 *AONT*

算法3概述了数据包重组和逆$AONT$ 过程。在目的节点，首先根据数据包序列号对Pkt_1 和 Pkt_2 的有效载荷进行重组，并提取其有效载荷（第2行）。合并后，应用逆 AONT 来重构原始消息，与全有或全无变换 过程相比，存在以下主要区别。首先，在第6行，通过根据关系式$(a \cdot k') \bmod p = m' \iff a/m' = k'$，对 C_s 和 B'_{s+1} 进行逐元素除法，恢复密钥 K'。然后，在第7行，重构拟群及其对应的对偶。最后，在第12行，使用对偶算子 $\circ$ 从伪块中恢复原始消息块

$\bullet$。这里，对于集合 Q 上给定的二元运算符 $\bullet$，可以定义一个对偶二元运算符 $\circ$，使得 $a \circ b = c$ 当且仅当 $a \bullet c = b$。这一关系定义了一个新的有限拟群$\langle Q, \circ \rangle$，它是 $\langle Q, \bullet \rangle$ 的对偶。以下恒等式成立：$a \circ (a \bullet b) = b$ 和 $a \bullet (a \circ b) = b$。

算法3 数据包重组与逆 A ONT

1: **函数** 重新组装并 $AONT^{-1}(Pkt_1, Pkt_2)$
2: $M'_1, M'_2 \leftarrow$从 Pkt_1和 Pkt_2中提取有效载荷
3: $B'_1 \| \ldots \| B'_s \leftarrow M'_1 \| M'_2$其中 $B'_i = (h'_{i1}, \ldots, h'_{in})$
4: $C_1 \leftarrow B'_1$
5: **对于** $i = 2$ 到 s **执行**
6: $C_i \leftarrow C_{i-1} * B'_i$
7: $K' \leftarrow C_s / B'_{s+1}$
8: $\langle Q, \bullet \rangle, \langle Q, \circ \rangle \leftarrow Quasi(K')$
9: $l_1 \leftarrow k_1, l_2 \leftarrow k_2 \bullet l_1, \ldots, l_n \leftarrow k_n \bullet l_{n-1}$ 和 $l \leftarrow l_n$
10: **对** $i = 1$ 到 s **执行**
11: $I(i) \leftarrow$ 的基-n 表示，其中 i 包含 $|I(i)| = n$
12: $R(i) \leftarrow (r_{i1}, \ldots, r_{in})$其中 $e_{in} \leftarrow l \bullet i_{in}, r_{in-1} \leftarrow r_{in} \bullet i_{in-1}, \ldots, r_{i1} \leftarrow r_{i2} \bullet i_{i1}$
13: $B(i) \leftarrow (h_{i1}, \ldots, h_{in})$，其中 $h_{ij} \leftarrow r_{ij} \circ h'_{ij}$
14: $M \leftarrow B_i \| B_2 \| \ldots \| B_{s+1}$
15: **返回** M

IV. 实验

我们采用了结合周期精确 gem5 [4]和 Noxim [5]的联合仿真设置。gem5 全系统仿真使用七个 SPLASH-2 [6] 和 PARSEC [7]基准测试生成网络跟踪数据，随后将其用于 Noxim 中由跟踪驱动的仿真。所仿真的系统采用两级 MESI 缓存一致性协议，x86 核心的运行频率为 2GHz，并使用包含 64 个节点的 8×8 网格互连。每个核心配备 32KB 私有 L1 指令/数据缓存，所有核心共享一个 512KB L2 缓存。我们扩展了 Noxim，通过在网络接口处集成 AONT、逆 AONT 和多路径路由生成，并支持路由器处的目的节点交换，从而支持所提安全机制。我们将方案与广泛采用的标准 AES-128 进行比较，因为 Hummingbird 等轻量级密码尚未标准化且存在已知漏洞。由于我们的设计支持并行 AONT 变换，为确保公平比较，我们将其与并行 CTR 模式下的 AES-128 进行比较。评估包含三种场景：（1）无安全机制；（2）我们提出的方法；（3）CTR 模式下的 AES-128。

A. 实验结果

表 II 展示了三种配置下，七种基准测试在 8×8 片上网络上的平均延迟：无安全机制、AES-128 和我们的方法。对于每种基准测试，我们针对 20 种随机的应用程序到核心的映射进行了实验，报告的数值为这些实验的平均延迟。与无安全机制基线（12.9 个周期）相比，AES-128 使平均延迟增加了 20 倍以上（261.5 个周期），而我们的方法仅增加了 2.8 倍（35.9 个周期）。这表明，与传统加密相比，我们基于全有或全无变换的方案在提供强机密性的同时，显著降低了性能开销（降低 7.3 倍）。我们还将我们的方法与 [2] 中基于填充与筛选的 AONT 方案进行了比较；我们的方法性能更优，平均分组时延降低了 83.9%。这是因为填充过程会显著增大数据包大小（$\approx 4x$），原因是其中包含多个计数器和认证标签。

表 II：8x8 网格片上网络的性能比较

基准测试	延迟：无安全机制		延迟：AES-128		延迟：所提方案	
	Avg	Max	Avg	Max	Avg	Max
FFT	13.2	31	259.1	630	36.5	97
OCEAN	13.1	33	258.3	626	35.1	95
RADIX	14.1	33	259.5	628	34.2	99
FMM	10.9	33	266.9	630	35.5	101
LU	13.6	31	258.2	628	37.0	99
BARNES	11.2	31	270.5	630	36.2	101
黑色	13.9	31	257.9	626	36.6	99
平均值	12.9	33	261.5	630	35.9	101

表 III展示了在 8×8 片上网络上同时运行两个基准测试时的平均延迟。与单基准测试情况类似，我们对 20 次随机应用映射的结果取平均值。如果关注表中的某个条目（例如，快速傅里叶变换 + *1*），每个主要基准测试（例如，快速傅里叶变换）都会在 20 次运行中分别与随机选取的第二个基准测试配对。与无安全机制基线（13.8 个周期）相比，AES-128 使平均延迟增加了超过 19×（266.5 个周期），而我们的

表 III：8x8 的性能比较：2 个基准测试

基准测试	延迟：无安全机制		延迟：AES-128		延迟：所提方案	
	Avg	Max	Avg	Max	Avg	Max
快速傅里叶变换 + 1	14.1	60	263.9	658	40.2	95
OCEAN + 1	14.9	50	261.7	654	39.9	95
RADIX + 1	15.1	57	263.1	640	38.7	97
快速多极子法 + 1	11.3	57	272.4	652	39.0	91
LU分解 + 1	14.3	50	262.6	654	39.4	95
BARNES + 1	12.2	56	272.2	665	39.2	95
黑色 + 1	14.7	54	265.3	654	40.1	105
平均值	13.8	60	266.5	665	39.5	105

所提方案的延迟仅增加了 2.9×（39.5 个周期），证实了其在多应用工作负载下的高效性。

B. 讨论

表 IV 展示了基线方案和我们的方法在单路由器及双路由器威胁模型下的窃听概率。通过评估所有可能的源-目的地对排列组合和恶意路由器部署位置，计算得出这些概率；如果任一恶意路由器位于数据包路径上，则将该情形标记为窃听事件。在单个恶意路由器的情况下，我们的方法将两种网格规模下的窃听概率均降至 0%（例如，在 8×8 中从 6.85% 降至 0%）。即使在更悲观的双恶意路由器模型下，我们的方法也将窃听概率限制在仅 0.47%。与 AES-128 CTR 方法相比，根据使用 28纳米标准单元库进行综合后得出的门等效（GE）估算，我们基于AONT的方法在 64节点片上网络中实现了约 79% 的面积缩减。与 AES 相比，我们的方法还消除了密钥共享和安全密钥存储的开销，因为源节点与目的节点之间无需预共享密钥。

表 IV：不同场景下的窃听概率

片上网络规模	威胁模型	防御	窃听概率
4×4	1 个恶意路由器	无防御 我们的方法	10.71% 0%
	2 个恶意路由器	无防御 我们的方法	41。76% 4.41%
8×8	1 个恶意路由器	无防御 我们的方法	6。85% 0%
	2 个恶意路由器	无防御 我们的方法	13.21% 0.47%

V. 结论

本文提出了一种轻量级保密框架，用于在NoC架构中实现安全片上通信。通过将基于拟群的AONT与不相交多路径路由相结合，我们的方法确保中间路由器在未获取所有数据包块的情况下无法重构原始消息。真实工作负载上的实验结果证实，与AES相比，我们的方法在提供保密性保证的同时，将性能开销降低了7.3倍，彰显了其抵御窃听攻击的有效性。

参考文献

[1] H. Weerasena 等，“电气、光学和无线片上互连的安全性：综述”，*ACM TODAES*，第29卷，第2期，第1–41页，2024年。[2] ——，“面向片上网络架构的基于填充与筛选和全有或全无变换的轻量级加密”，载于*HOST*，2021年。[3] S. I. Marnas 等，“使用拟群的全有或全无变换”，载于巴尔干信息学会议论文集，2003年，第183–191页。[4] N. Binkert 等，“gem5模拟器”，*SIGARCH*计算机体系结构新闻，2011年。[5] V. Catania 等，“使用Noxim进行周期精确的片上网络仿真”，*ACM TOMACS*，第27卷，第1期，第1–25页，2016年。[6] S. C. Woo 等，“splash-2程序：特征分析与方法论考量”，计算机体系结构新闻，第23卷，第2期，第24–36页，1995年。[7] C. Bienia 等，“parsec基准测试套件：特征分析与架构影响”，载于*PACT*，2008年，第72–81页。