基于 FPGA 网表结构和行为特征分析的硬件木马检测

李一玮[1], 宿　豪[1], 鲁逸晴[1], 武玲娟[2], 胡　伟[1]

[1] 西北工业大学 网络空间安全学院 西安 中国 710072
[2] 华中农业大学 信息学院 武汉 中国 430070

摘要　硬件木马是集成电路中未公开的功能或恶意设计修改, 可泄露敏感信息, 篡改关键存储器, 造成权限提升或拒绝服务, 是一种主要的硬件安全与可信威胁。木马通常具有轻量级、低活度和高隐蔽性的特点, 导致其难以检测。目前, 在寄存器传输级、门级, 甚至是晶体管级抽象层次上已有相当数量的木马检测研究, 然而, 现场可编程门阵列(Field Programmable Gate Array, FPGA)网表级的木马检测方法一直是一个被忽视的研究课题。现有木马检测方法主要包括功能测试、形式化验证、侧信道分析、翻转概率分析、逆向工程和基于人工智能的方法等, 尚面临着难以快速激活木马、依赖于高质量属性、对背景噪声敏感、易产生大量误报、对芯片造成物理损伤以及难以自动提取有效特征等不足。该文利用 FPGA 网表中显著的设计结构和行为特征, 开展查找表(Look-up-Table, LUT)级硬件木马特征分析, 进而提出基于深度学习的 LUT 级木马特征准确提取与智能匹配方法, 以及木马相关属性自动提取与形式化验证方法。实验结果表明该方法可以更精确地描述和匹配硬件木马特征, 平均真阴性率(True Negative Rate, TNR)、真阳性率(True Positive Rate, TPR)、准确率、ROC 曲线下面积(Area Under the ROC Curve, AUC)分别为 0.990、0.969、0.971 和 0.979, 该方法还可以通过对提取的属性进行形式化验证来自动搜索和恢复木马触发条件。该文工作提供了一种硬件木马检测研究的新思路, 是现有硬件木马检测方法体系的有效补充。结合近期在 FPGA 配置流解密和反向综合方面的研究进展, 论文也为配置流安全分析提供了一种技术途径。

关键词　硬件安全; 硬件木马检测; 查找表; 深度学习; 属性验证
中图法分类号　TP309　DOI 号 10.19363/J.cnki.cn10-1380/tn.2026.01.16

Hardware Trojan Detection Through Structural and Behavioral Feature Analysis of FPGA Netlist

LI Yiwei[1], SU Hao[1], LU Yiqing[1], WU Lingjuan[2], HU Wei[1]

[1] School of Cybersecurity, Northwestern Polytechnical University, Xi'an 710072, China
[2] College of Informatics, Huazhong Agricultural University, Wuhan 430070, China

Abstract　Hardware Trojans are unspecified functionality in or malicious design modifications to integrated circuits. They can leak sensitive information, overwrite critical memory, cause privilege escalation or denial of service and thus represent a major threat to hardware security and trust. Trojans are typically associated with the characteristics of light-weight, low-activation rate and highly stealthy, which make them hard-to-detect. While there are a considerable number of research works in hardware Trojan detection at the register transfer, gate and even transistor levels, Trojan detection in field programmable gate array (FPGA) netlist has long been an overlooked research vector. Existing hardware Trojan detection methods include functional testing, formal verification, side-channel analysis, switching probability analysis, reverse engineering and artificial intelligence (AI) based solutions. These methods are still seeing various drawbacks, such as hard to quickly activate the Trojan, relying on high-quality properties, sensitive to background noise, high false positives, causing physical damage to chips and difficult to automatically extract effective features. This paper performs Trojan feature analysis at the look-up-table (LUT) level, leveraging the significant structural and behavioral features in FPGA netlist. It proposes a method for precise extraction and intelligent matching of Trojan features through deep learning. In addition, it provides an approach for automated extraction and formal verification of Trojan related properties. Experimental results have demonstrated that the proposed method can achieve more precise depicting and matching of hardware Trojan features, with an average true negative rate (TNR), true positive rate (TPR), accuracy and area under the ROC curve (AUC) of 0.990, 0.969, 0.971 and 0.979 respectively. Our method can also automatically search for and recover Trojan trigger condition through formal verification of the extracted properties. This work provides a new perspective for hardware Trojan detection research and an effective complement for the spectrum of hardware Trojan detection methods. It

通讯作者：胡伟, 长聘教授, 博士生导师, Email: weihu@nwpu.edu.cn。
本课题得到国家重点研发计划项目(No. 2021YFB3100901); 国家自然科学基金项目(No. U23B2041、No. 62074131、No. 62404082)资助。
收稿日期: 2024-04-21; 修改日期: 2024-09-07; 定稿日期: 2026-01-07

also paves the way for FPGA bitstream security analysis considering the recent research advances in bitstream file decryption and reverse synthesis.

Key words hardware security; hardware Trojan detection; look-up-table; deep learning; property verification

1 引言

现代集成电路(Integrated Circuit, IC)的设计规模和集成度持续提升。为降低设计成本和缩短上市时间, IC 设计商通常会集成来自不可信第三方的知识产权(Third Party Intellectual Property, 3PIP)核。这些 3PIP 中可能包含了极其隐蔽的未公开功能甚至是恶意逻辑修改, 即硬件木马(Hardware Trojan, HT)。一旦集成电路设计流片后部署, 这些木马设计就犹如逻辑定时炸弹。攻击者即可利用这些隐藏的硬件木马, 发起强有力的硬件攻击。硬件木马通常均为精心设计, 具有轻量级、低活度和高隐蔽性的特点, 因此木马检测仍然是一项极具挑战性的开放性研究课题。

现有硬件木马检测技术主要包括逆向工程、逻辑测试、侧信道分析、翻转率分析、安全验证等[1]。逆向工程[2]通常需要经过去封装、逐层还原版图、提取网表、对比验证等步骤, 通过将还原后的文件与设计规范比较, 以确定芯片中是否存在可疑的电路结构或设计功能, 并进一步确认其是否属于硬件木马。逆向工程是一种破坏性的检测技术, 会给芯片带来不可逆的损害。对于批量芯片产品, 只能抽样分析, 难以实现全面检测。逻辑测试[3]方法对芯片施加测试向量, 通过捕捉不符合设计规范的异常电路行为, 实现硬件木马检测。该方法的难点在于如何生成高覆盖率的测试向量以激活硬件木马。研究者提出了基于机器学习和深度神经网络的测试向量生成方法[4], 以提高硬件木马的激活概率, 最小化测试时间和开销, 但是, 仍难以保证在海量测试输入组合中能够快速找到特定的测试向量。侧信道分析[5]方法利用了硬件木马的植入会使得芯片的延迟和功耗等侧信道参数发生改变的特点, 通过统计和特征分析等手段判断待测芯片中是否含有硬件木马。然而, 侧信道分析方法对工艺偏差和正常模块产生的背景噪声较为敏感, 从而影响其检测的准确性。翻转率分析[6]方法利用硬件木马电路的活度显著低于正常电路的特点, 通过估算或电路仿真提取低翻转率的设计结构来实现木马检测, 然而, 其检测准确率受仿真覆盖率的制约, 容易发生误报。安全验证[7]方法通过构建芯片设计安全模型, 形式化验证芯片设计需满足的安全属性(如机密性、完整性和隔离性), 实现违反安全属性类硬件木马的检测。然而, 安全验证结果严重依赖于安全属性的质量与完备性, 如何自动生成能够有效覆盖木马设计的安全属性仍属于亟待研究和解决的问题, 对具有数百万的逻辑门的电路系统进行完备的模型建立与验证也十分具有挑战性[8]。

本文以综合后的 FPGA 网表为研究对象, 利用 FPGA 网表中显著的设计结构和行为特征, 从 LUT 级提取硬件木马特征, 提出了基于深度学习以及基于属性自动提取与形式化验证的硬件木马检测方法, 为识别 3PIP 的硬件木马安全威胁提供了一种有效的途径, 本文主要贡献如下:

- 提出一种基于 FPGA 网表结构和行为特征分析的硬件木马检测方法, 首次将信息熵用于 LUT 单元木马特征静态分析, 迅速定位可疑模块, 缩小木马检测范围, 提供了一种硬件木马检测研究的新思路。
- 提出一种基于深度学习的 LUT 级木马特征准确提取与智能匹配方法, 基于有向图将硬件木马的行为描述转化为可量化的数据指标, 无需依赖专家知识, 可实现木马设计的精准识别。
- 提出一种基于 LUT 级别的低翻转和低覆盖属性自动提取与形式化验证的木马检测方法, 可搜索和恢复木马触发条件, 是现有硬件木马检测方法体系的有效补充。

本文以下部分组织结构如下: 第 2 节介绍了相关工作。第 3、4、5 节分别详细阐述了本文提出的 LUT 级硬件木马特征分析方法、基于图神经网络的 LUT 级硬件木马特征提取与匹配方法, 以及 LUT 级木马相关属性提取与验证方法。第 6 节给出了相关实验结果。最后, 第 7 节对本文工作进行了总结。

2 相关工作

2.1 木马特征相关研究

研究人员针对集成电路生命周期的不同阶段提出了多种硬件木马特征提取与检测方法, 其中侧信道分析(Side Channel Analysis, SCA)是有效的硅后解决方案。Sabri 等[9]提出了一种集成的硬件木马检测和定位方法, 该方法采用了基于可满足性分析的测试模式生成方案和基于选择器的调试技术。Pearce 等[10]分析时间、电磁信号、电源和硬件性能计数器等多模态侧信道信息, 开发了一种硬件木马检测框架。然而基于 SCA 的硬件木马检测方法通常依赖于

难以获取的黄金芯片, 同时对木马的规模和工艺偏差非常敏感。

Salmani[11]基于门级网表的可控性和可观测性特征, 提出了一种不需要黄金参考模型的硬件木马检测方法, 通过训练无监督聚类机器学习模型, 实现了逻辑门级硬件木马检测。考虑到硬件木马只有在电路的主输入和/或内部状态满足特定触发条件时才会被激活, Hasegawa 等[12]认为门级网表中硬件木马具有较大的扇入和扇出, 基于门级网表电路结构分析提取了 5 个硬件木马特征, 即与节点相隔两级的逻辑门的输入信号数量(Logic Gate Fanins, LGFi)、从节点到最近的触发器输入的逻辑门级数(FlipFlop Input, FFi)、从节点到最近的触发器输出的逻辑门级数(FlipFlop Output, FFo)、从输入到节点的最小逻辑门级数(Primary Input, PI)和从节点到输出的最小逻辑门级数(Primary Output, PO)。Piccolboni 等[13]在寄存器传输级(Register Transfer Level, RTL)提取了电路结构的控制流图, 利用其拓扑结构和子图同构算法把待测电路与硬件木马库中的木马结构进行匹配, 实现硬件木马的检测。Huang 等[14]在 RTL 基于硬件木马触发器控制流信息提取了 5 个硬件木马特征, 其中包括直接执行概率、间接执行概率和间接执行相对性 3 个定量特征, 以及不平衡程度和有限状态机相对性两个定性特征。Cheng 等[15]深入研究了 RTL 硬件木马的特性, 建立了木马检测的可信模型, 并提出了利用 Perl 语言进行硬件木马检测的方法。现有的木马特征提取工作主要集中在逻辑门级和寄存器传输级, 在 LUT 级别提取木马特征的相关工作尚不多见。

2.2 基于深度学习的木马检测方法

近年来, 大量研究将机器学习方法应用于集成电路设计安全性分析, 其中硬件木马检测问题被视为二元分类问题, 即将被测集成电路设计分类为硬件木马和正常电路两种。

Yang 等[16]通过分析可疑电路设计的结构和信号特征, 针对寄存器传输级别提出了一种基于随机森林算法的硬件木马检测方法。Hasegawa 等[17]提出了一种基于多层神经网络的硬件木马检测方法, 通过提取门级网表中的五维特征, 将门级网表中的节点分类为木马节点和正常节点。Yan 等[18]基于最近邻不平衡数据分类算法扩展了硬件木马的特征, 有效解决了木马基准特征集较少的问题。Zhao 等[19]利用无监督的基于结构特征的聚类方法来检测硬件木马, 无需为每个特征手动设置阈值, 从而提高了检测精度。Han 等[20]从 RTL 源代码的抽象语法树中提取电路特征, 并提出了一种基于梯度增强算法的木马检测方法。Demrozi 等[21]基于图的特征和概率神经网络, 在 RTL 提出了一种硬件木马识别和分类方法。虽然上述检测方法可以成功检测出特定类型的木马, 但仍具有严重依赖专家知识进行木马特征提取、需要参考硬件设计以及无法检测新木马等不足。最近一项研究[22]利用图神经网络(Graph Neural Network, GNN)自动提取电路结构特征, 在门级和 RTL 实现了硬件木马检测。然而, 这种方法仅能判断硬件设计中是否包含木马, 无法定位到硬件木马所在位置。Yasaei 等[23]提出了在 RTL 利用图卷积网络对节点进行自动特征提取的方法, 将硬件设计转换为图对其进行节点分类, 并输出木马节点对应的恶意电路, 无需手动进行特征提取或代码检查。Lu 等[24]首次提出了一种基于信息熵聚类的硬件木马检测方法 HTDet, 该方法采用信息熵对待测电路设计 ASIC 网表仿真结果中信号的翻转概率进行量化, 从而识别翻转率极低的信号作为木马信号。

2.3 基于属性验证的木马检测方法

随着硬件木马检测技术的快速发展, 研究者提出基于属性验证的木马检测方法。Love 等[25]提出了一种基于携带证明的硬件安全属性验证框架, 进行集成电路设计 IP 核中的木马检测。Rajendran 等[26]扩展了该项工作, 基于有界模型检验(Bounded Model Checking, BMC)提出了一种木马检测方法, 该方法通过检测第三方 IP 核中关键数据是否被恶意修改来检测硬件木马。Hu 等[27]提出了一种属性驱动的硬件安全验证与木马检测方法, 通过将高级别的安全规范转换为低级别的安全策略、属性、断言和约束, 在信息流与统计模型上进行形式化验证。Liu 等[28]在寄存器传输级提出了一种静态安全分析框架, 可以自动从片上系统的硬件设计中提取数据流模型, 基于模型分析给定污染源和目标节点之间的污染传播条件, 得到对应的安全属性。Wu 等[29]结合自动测试向量生成(Automatic Test Pattern Generation, ATPG)、布尔可满足性问题(Satisfiability Problem, SAT)和有限状态机(Finite-state Machine, FSM), 提出了一种可以检测信息流违规行为的分析框架, 该框架详尽地分析违规行为并报告对应的输入模式。Zhao 等[30]从 RTL 源代码语义角度出发, 提出了一种基于信号流向的多叉树分层递归搜索方法来分析 RTL 源文件中的疑似硬件木马语句, 该方法可有效检测通过端口信号触发的硬件木马。Wang 等[31]提出了一种用于硬件木马检测的自动安全属性生成框架 ASPG, 借助粗粒度控制和数据流图来描述 RTL 的木

马特征, 将其与预定义的特征库进行匹配生成目标安全属性。Li 等[32]提出了一种基于 FSM 的模型构建方法和形式化验证框架, 该方法将木马的负载和触发条件分别映射为有限状态机中的状态和状态的转移条件, 可以有效地检测引起信息泄露和拒绝服务的硬件木马。然而, 现有基于属性验证的硬件木马检测方法, 存在状态空间爆炸问题, 同时难以实现安全属性的自动提取。

3 LUT 级硬件木马特征分析

硬件木马通常保持不活跃状态, 仅在特定条件下才会触发以实施恶意攻击, 从而避免其被激活和检测到。因此, 木马设计中通常都具有控制木马激活的触发逻辑和触发信号, 并且木马触发信号往往具有极低的活跃度, 即硬件木马的显著行为特征, 而用于产生触发信号的触发逻辑也一般具有典型的结构特征。图 1 以 Trust-Hub AES-T1000 木马测试向量为例, 对该木马设计触发逻辑在 RTL、门级和 LUT 级的特征进行对比。

该木马设计在 RTL 的特征为 if-else 分支结构和用于检测小概率事件的比较器。门级则具有相对分散的条件逻辑与选择器输出, 相对不显著。LUT 级具有多扇入锥的结构, 且 LUT 的初始化向量值具有不均匀的 0、1 分布。

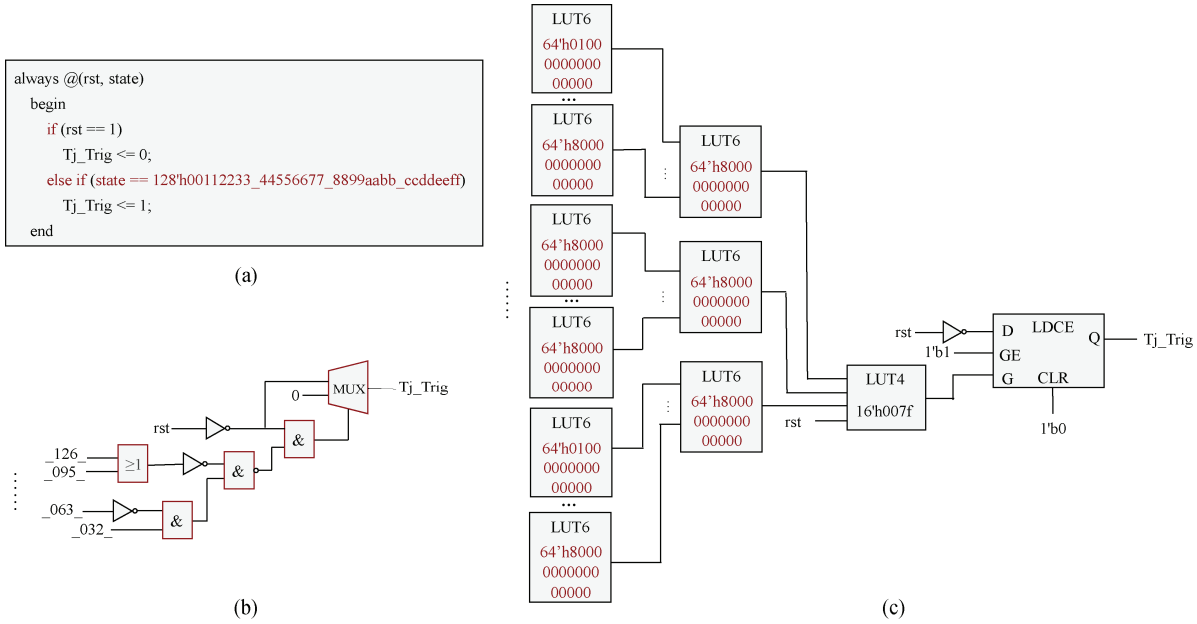

图 1 AES-T1000 的木马触发逻辑在不同抽象层次的特征
(a) RTL 级木马特征; (b) 门级木马特征; (c) LUT 级木马特征
Figure 1 Features of AES-T1000 Trojan trigger logic at different levels of abstraction
(a) Trojan features at RTL level; (b) Trojan features at gate level; (c) Trojan features at LUT level

在 RTL 可以通过对代码语句的语法结构和操作进行分析, 寻找具备硬件木马特征的 RTL 设计。在门级可以提取结构特征如逻辑门的扇入单元数、多路选择器的数量、反相器的数量等, 也可以利用 SCOAP 算法[11]对每个节点计算可测试性度量, 寻找控制和观测难度较大的节点, 但由于输出节点的可测性计算需要依赖输入节点的值, 因此分析效率会随着电路规模呈指数级增长, 而且门级网表只是描述了电路元件的连接关系, 无法直接体现实际的电路功能。在 LUT 级只需要寻找初始化向量中 0、1 分布不均匀的 LUT 单元, 与 RTL 和门级的检测方法相比, 将集成电路设计综合为 FPGA 网表后, 木马特征更为直观和显著, 更易于实现特征自动提取。

本文利用 LUT 级的木马结构和行为特征, 提出 FPGA 网表级的硬件木马检测方法, 利用了低翻转的木马触发控制逻辑对应的信号取值具有显著不均匀的概率分布, 即信号取逻辑 0 或逻辑 1 的概率远远大于它取另一个值的概率。在 FPGA 网表中, 信号通常为 LUT 输出, 信号特征即反映在 LUT 的初始化向量中。LUT 的初始化向量表征了电路输入和输出之间的逻辑关系, 为翻转率分析提供了直接依据。

本文借助信息理论中的信息熵来量化分析每个 LUT 初始化向量的 0、1 分布概率特征, 以识别具备低翻转概率特征的可疑木马单元。相较于直接采用概率作为度量标准, 信息熵能够统一地将初始化向量趋向全 0 或全 1 量化为低翻转特征。信息熵是信

息理论中的概念，它可以用于度量信源的不确定性，或者随机事件发生的不确定性。数学上，信息熵定义为式(1):

$$H(X)=-\sum p(x)\cdot \log_2 p(x) \tag{1}$$

其中，$H(X)$表示随机变量X的信息熵；$p(x)$表示X取值为 x 的概率。

考虑随机事件 X 只存在发生或不发生的情况，例如木马激活或不激活。当信息熵越高时，意味着事件的不确定性越大，事件发生的概率将接近 50%。当信息熵越低时，意味着事件的不确定性越小，该事件发生的概率将接近 100%(或 0)。因此，信息熵较低的单元符合木马单元激活概率极低的特征，而信息熵较大的单元为易激活单元。例如一个初始化向量为 64'h4000000000000000 的 LUT，可视为一个 6 位地址线的随机存取存储器(Random Access Memory, RAM)，输出 1 和 0 的概率分别为 1/64 和 63/64，代入信息熵计算公式，可得到该 LUT 的信息熵约为 0.116 比特，该 LUT 具有低信息熵的特征，符合木马单元的特点。

本文通过计算硬件设计各个模块对应 FPGA 网表中 LUT 初始化向量的信息熵，然后对每个模块中全部 LUT 对应信息熵进行统计分析，从 LUT 信息熵的统计分布特征即可快速区分哪些模块具有典型的木马特征。本文将在实验部分对此进行测试分析。

4 LUT 级木马特征提取与匹配

本文结合信息熵研究基于图神经网络的 LUT 级硬件木马特征自动提取与匹配方法，并设计了一种基于扇入结构分析的子图划分方法。将硬件木马检测问题转换为机器学习中的二分类问题，用于检测第三方 IP 核中硬件木马所在位置。本文中硬件木马检测训练与测试流程如图 2(a)所示。

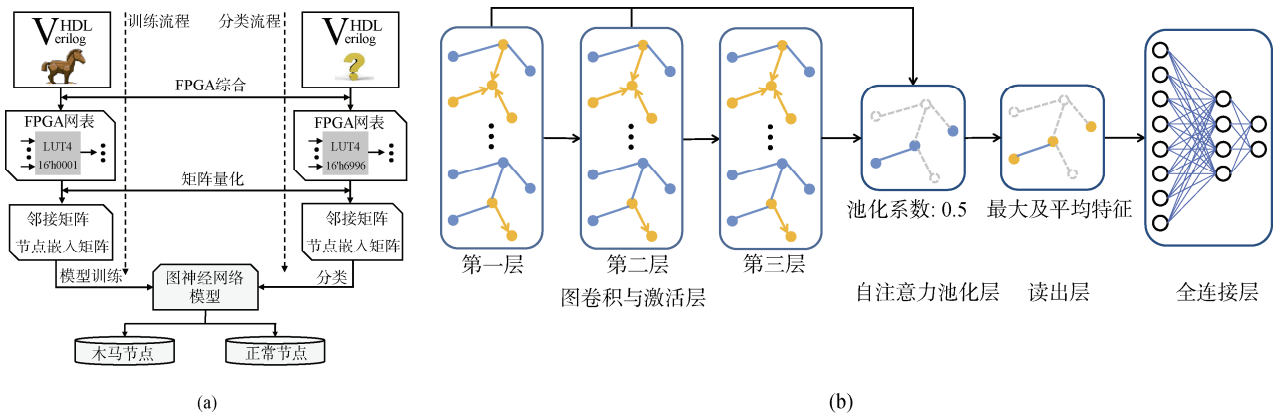

图 2 基于图神经网络的 LUT 级硬件木马特征自动提取与检测
(a) 硬件木马检测与训练流程图; (b) 图神经网络模型

Figure 2 Automatic extraction and detection of Hardware Trojan features at LUT level based on Graph Neural Network
(a) Process of Hardware Trojan detection and training; (b) Graph Neural Network model

如图 2(a)，将 Verilog 形式描述的硬件木马基准电路通过逻辑综合生成 FPGA 网表。基于 FPGA 网表进行矩阵量化，构建邻接矩阵及节点嵌入矩阵。

在模型训练阶段，本文采用了基于电路结构分析的电路节点打标签方法，首先对基于 Verilog 形式描述的数据集中的木马节点和正常电路节点分别打标签 1 和 0，本文采用深度优先算法，对于逻辑综合时 FPGA 网表中产生的节点，赋予与其父节点一样的标签。之后使用矩阵量化得到的邻接矩阵及节点嵌入矩阵用于训练图神经网络，图神经网络模型结构如图 2(b)所示，该神经网络模型主要包括 4 个功能层操作。第一个功能层操作为图卷积与激活层，该部分可视作由三层图卷积函数进行的节点嵌入操作，三层图卷积操作的隐藏节点数相同，并通过整流线性单元(Rectified Linear Unit, ReLU)函数进行非线性激活。其中，每个图卷积层接收上层传入的嵌入矩阵以及硬件设计有向图表示的邻接矩阵，经过卷积计算聚合邻居节点信息，并将结果传输到下一层。硬件设计有向图表示在经过三层图卷积计算并进行拼接后，可以得到有向图表示中每个节点的聚合特征嵌入向量。

模型第二部分为自注意力池化层，该层网络将图卷积与激活层输出的节点聚合特征嵌入向量作为输入，经过卷积操作聚合邻居节点信息，计算出每个节点的自注意力分数。对图中所有节点的自注意力分数进行排序，保留自注意力分数高的节点及其

连接结构, 并将新的邻接矩阵和嵌入矩阵输入到下层网络结构中。其中, 保留的节点比例通过池化率进行控制, 本文采用的池化率为 0.5。

模型结构的第三部分为读出层, 该层网络接收自注意力池化层更新的邻接矩阵和嵌入矩阵, 通过节点特征嵌入的最大化和平均值策略, 聚合图表示中所有节点的特征信息, 实现全局操作, 生成图表示的特征嵌入。

该模型结构的最后一部分可视作由 3 个全连接层和激活函数对图表示的特征嵌入进行的分类操作。其中每个全连接层分别接收上层结构传入的节点嵌入, 经过带有偏置项的线性变换计算出新的节点嵌入, 并传入下层网络结构。模型通过全连接层的计算, 输出一个二维节点嵌入, 该节点嵌入的两个维度分别表示无木马的预测概率和有木马的预测概率, 若无木马概率大于有木马概率, 待测硬件设计被判断为无木马硬件设计, 反之判断待测硬件设计中包含硬件木马。例如, 输出结果为[0.7, 0.3]时, 待测硬件设计被判断为无木马硬件设计; 为[0.13, 0.87]时, 判断待测硬件设计中包含硬件木马。

模型通过图卷积与激活层聚集图表示中的邻居节点信息, 经过自注意力池化层计算图中节点的自注意力分数并筛选出重要节点, 读出层聚合图表示中所有节点信息, 并通过全连接层实现分类。在模型训练时, 本文采用加权交叉熵损失函数, 通过为每个类别分配不同的权重, 使模型在训练时更关注少数类, 从而改善木马检测模型的检测效果。在图 2(a)中的分类流程中, 基于训练完成的图神经网络模型, 待测集成电路设计 FPGA 网表中的节点被分类为木马节点或正常节点, 实现了对硬件木马的检测。

矩阵量化过程如图 3 所示。首先对逻辑综合后的 FPGA 网表进行结构化分析, 构建出完整硬件设计对应的有向图表示, 然后对每个节点进行扇入分析构建表征该节点的子图。由于硬件设计信号的输出结果往往受输入端方向信号及操作类型的影响, 因此信号扇入方向的硬件设计结构在一定程度上能够决定该信号的输出情况, 即信号输入端的扇入结构可以作为该信号的特征, 经统计分析, 4 级扇入结构足以反映硬件木马电路结构。

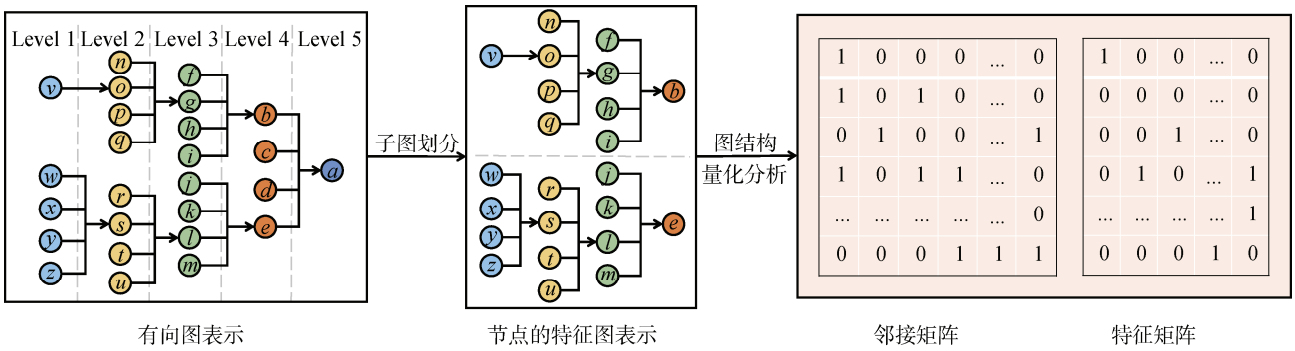

图 3 矩阵量化过程图

Figure 3 Process of matrix quantization

对有向图表示进行子图划分。例如 LUT 单元节点 b 和 e, 选择其扇入结构中 Level 1~4 的 LUT 单元节点, 根据完整的有向图表示保存节点之间的连接关系, 可获得 LUT 单元节点 b 和 e 的特征图表示。邻接矩阵及嵌入矩阵的构建过程如图 4 所示。

由硬件木马 LUT 级特征分析可知, 在 FPGA 网表中, LUT 单元的初始化向量反映了其翻转行为, 因此在构建节点嵌入矩阵时本文提取了 LUT 单元的初始化向量作为特征。对于非 LUT 单元的其他 FPGA 单元节点, 例如输入信号、输出信号、反相器和触发器, 分别规定其特征标识为 0、1、2 和 3。具体的 LUT 单元对应的特征标识计算方法如式(2):

$$feat = hex2dec(init) + 4 \tag{2}$$

其中, *feat* 为该查找表单元的特征标识, *init* 为该查找表单元的初始化向量值, *hex2dec*(*init*) 表示将十六进制数转换为十进制数。

为了能够区分具有相同特征标识的同类节点, 本文还为所有 FPGA 网表中的单元节点分配了一个数值标识作为该节点唯一标号, 它是基于 FPGA 网表构建电路的有向连接图, 应用深度优先搜索算法遍历有向连接图中的所有节点得到的。

如图 4(a)所示, 根据式(2), 初始化向量值分别为 16'h0040、16'h0040、16'h3000、16'h8000、16'h4000 的 5 个 LUT 单元对应的节点特征标识分别为 68、68、12292、32772、16388, 分配的节点唯一标号为 0、1、2、3、4。为提高模型对重要 LUT 单元节点及其连接

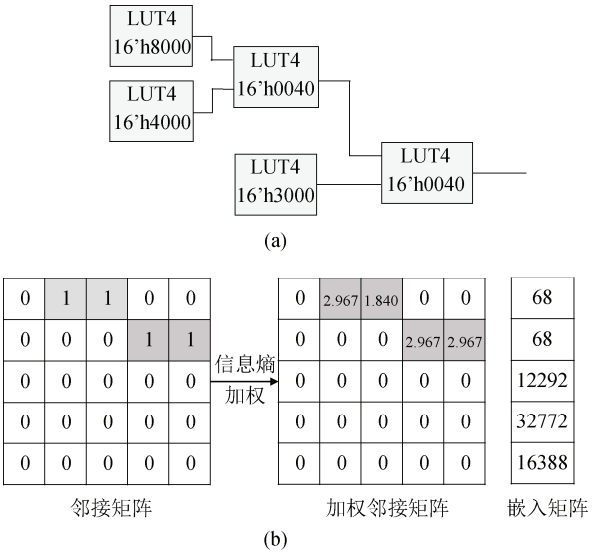

图 4 FPGA 网表及对应的图表示矩阵
(a)FPGA 网表结构图; (b)图表示矩阵

Figure 4 FPGA netlist and corresponding matrix representation of graph
(a) Structure of FPGA netlist; (b) Matrix representation of graph

关系的关注度, 本文基于 LUT 单元的信息熵值, 为表示 LUT 单元连接关系的邻接矩阵添加了权重, 将 LUT 单元所连输出边的权重设为其信息熵的倒数。例如初始化向量为 16'h3000 的 LUT 单元, 其信息熵值为 0.5435, 其输出边的权重为 1.840。最终构建出的加权邻接矩阵及嵌入矩阵实现了对数据集的图表征, 如图 4(b)所示。

5 LUT 级木马相关属性提取与验证

本文进一步提出 LUT 级木马相关属性提取与验证方法, 图 5 显示了该方法的流程。

本文以综合后的 FPGA 网表为分析对象, 使用 ModelSim 对网表进行有限随机仿真, 导出仿真轨迹, 通过分析轨迹来识别低翻转和低覆盖的 LUT。其中, 低翻转 LUT 是指 LUT 的输出为低翻转信号, 在有限次的随机仿真中很少或从不改变其逻辑状态; 低覆盖是指 LUT 的某些地址行输入组合在有限仿真中未出现。这些低翻转和低覆盖的 LUT 极有可能是与木马相关的设计。

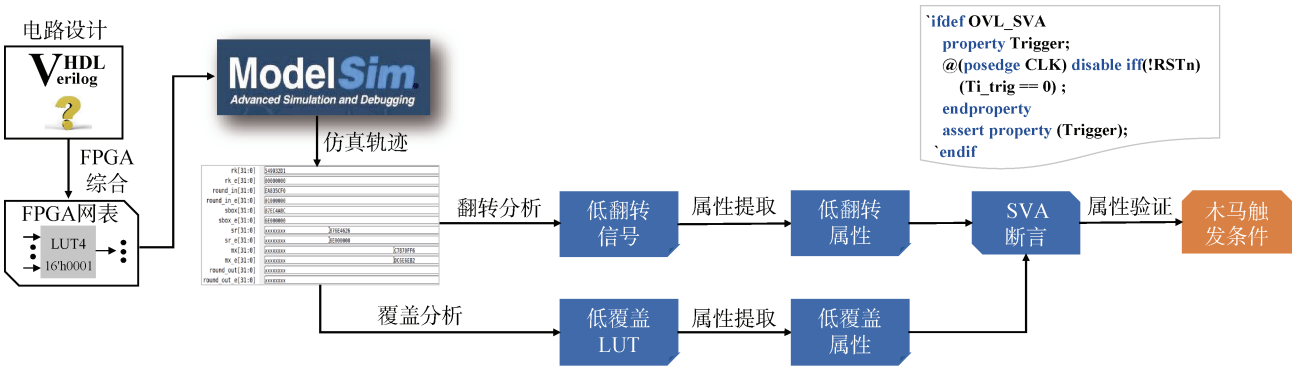

图 5 LUT 级木马相关属性提取与验证流程图

Figure 5 Process of extracting and validating Trojan-related properties at LUT level

为了进一步区分该LUT是难以触发的木马电路, 还是低可测性的正常逻辑, 可从仿真轨迹中提取翻转和覆盖率属性, 通过对属性进行验证来搜索能够使信号发生翻转或是覆盖地址线组合的输入条件(即验证反例), 并通过反例重放来确认设计是否存在恶意行为。以下分别对低翻转和低覆盖率属性的提取方法进行介绍。

例如对 Trust-Hub 中 AES-T1000 综合后的 FPGA 网表进行仿真, 随着仿真时间的增加, 可以筛选出图 6(a)所示的一个低翻转 LUT。

若仿真轨迹显示该 LUT 的输出信号始终为 1, 对应的翻转属性为: I[3:0]为任意取值组合时, 输出均为 1, 即输出信号恒定。可表示为伯克利逻辑交换格式(Berkeley Logic Interchange Format, BLIF)文件, 该文件前 3 行分别描述了该 LUT 的名称、输入和输出, 下面几行列出了不同输入对应的输出值, 即将 LUT 以真值表的形式展现出来。对该 BLIF 文件进行逻辑化简, 即可得到翻转属性断言 assert (_194_ == 1)。

对论文[33]中提出的 AES SDC 木马综合后的 FPGA 网表进行仿真, 随着仿真时间的增加, 可以筛选出图 6(b)所示的一个低覆盖 LUT。根据仿真轨迹可以得到地址线组合 I[3:0]的覆盖情况, 4 位的输入对应一个 16 位的覆盖向量, 仿真过程中被覆盖过的输入组合对应的覆盖向量位为 1, 覆盖向量最终收敛到 16'h0fff, 表示无法覆盖到高四位对应的输入组合, 即地址线 I2 和 I3 均为 1 的情况, 对应的覆盖属性为: 在 I0 和 I1 为任意取值组合且 I2 和 I3 同时为 1 时, 覆盖状态均为 0, 对应 BLIF 文件所描述的真值表代表

了 I2 和 I3 同时为 1 时输出 0 的属性对应逻辑, 对该 BLIF 文件进行逻辑化简, 即可得到覆盖属性断言 **assert** (dc1&dc2 == 0)。

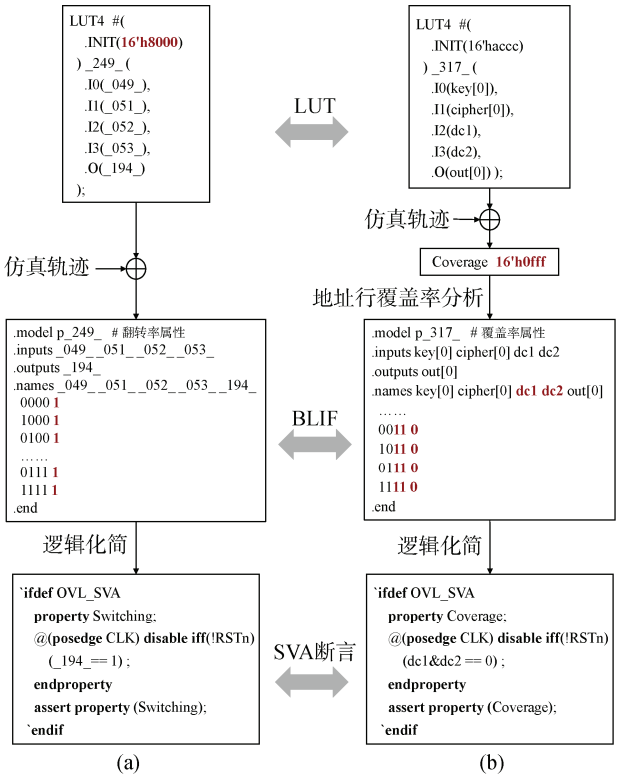

图 6 LUT 属性自动提取与断言生成

(a)低翻转 LUT 断言生成; (b) 低覆盖 LUT 的断言生成

Figure 6 Automatic extraction and assertion generation of LUT properties

(a) Assertion generation of low-switching LUT properties; (b) Assertion generation of low-coverage LUT properties

通过对断言进行形式化验证, 以检查该属性是否会被满足, 当属性在证明期间不能被满足时, Yosys 将报告一个反例, 即为木马的触发条件。

6 实验结果

6.1 木马特征量化分析结果

采用 Trust-Hub AES-T1000 测试向量, 利用第 3 节给出的信息熵分析方法对其设计模块中 LUT 的信息熵进行了计算, 并采用柱状图来展示信息熵分布情况。从图 7 中可以看出, AES-T1000 中 S 模块和 xS 模块存在一些非 1 的信息熵, 这是因为 S 盒和逆 S 盒变换本质做的是基于字节的查表替换, 会损失一部分随机性。而 lfsr_counter 模块、Trojan_Trigger 模块、TSC 模块都与木马功能相关, 其中 Trojan_Trigger 模块存在大量信息熵为 0.1 比特左右的极低值, 这表明该模块存在着许多翻转概率极低的信号, 与硬件木马触发器相关。aes_128 模块所有 LUT 单元的信息熵均为 1 比特, 表明信号翻转概率均匀, 为 AES 正常加密功能模块。通过信息熵的计算, 可以快速准确地定位可疑木马模块以及具备低翻转率特征的硬件木马相关 LUT 单元。

6.2 木马特征提取与匹配结果

本文使用了 Trust-Hub.org 中的 14 个木马测试向量进行 GNN 模型训练与测试, 基于 Yosys 的 synth_intel 命令将集成电路设计综合为 FPGA 网表, 利用第 4 节的方法进行特征提取和模型训练。本文使用了留一法对 GNN 模型进行评估, 硬件木马检测结果如表 1 所示。留一法是一种交叉验证方法, 常用于评估机器学习模型的性能。其核心思想是每次从数据集中留出一个样本作验证, 其余样本用于训练模型, 重复进行直到所有样本都被验证一次。留一法可以确保训练集与测试集的交集为空, 因此训练出的模型能够检测训练集中包含相似特征的木马设计样本, 但是, 模型不一定能够检测训练集中不包含相似特征的木马样本。由表 1 可见本文提出的 LUT 级硬件木马检测方法可以达到良好的检测效果, TNR、TPR、Accuracy、AUC 的平均值分别能够达到 0.990、0.969、0.971、0.979。以上结果表明本文提出的方法能够实现 FPGA 网表中硬件木马节点的精准检测。

表 2 中进一步将本文方法同其他基于机器学习的木马检测方法进行定量和定性比较。其中定量比较包括 Accuracy 和 TPR 指标的比较, 定性比较包括各类方法适用的设计抽象层次、特征提取过程是否自动化以及是否需要黄金参考模型。表 2 中的结果说明本文提出的硬件木马检测方法相较于文献[11, 14, 17, 21]中的方法能够实现木马特征提取的自动化, 且木马检测的实现无需依赖于黄金参考模型。相较于文献[23-24], 本文方法具有更高的木马检出率。

6.3 木马相关属性提取与验证结果

以 Trust-Hub.org 中的 BasicRSA-T200 为例说明木马相关属性提取与验证方法的有效性。在 BasicRSA-T200 提供的 3 个模块上运行测试, 图 8 给出了不同仿真时间与模块下, 输出低翻转信号的 LUT 个数。RSACypher 模块在仿真时间超过 5000 ns 后, 低翻转信号数量收敛为 1 个, 即 eqOp 信号; modmult 和 modmult_0 模块在经过 200 ns 的仿真后, 低翻转信号数量快速下降到 25 个。经对比检查,

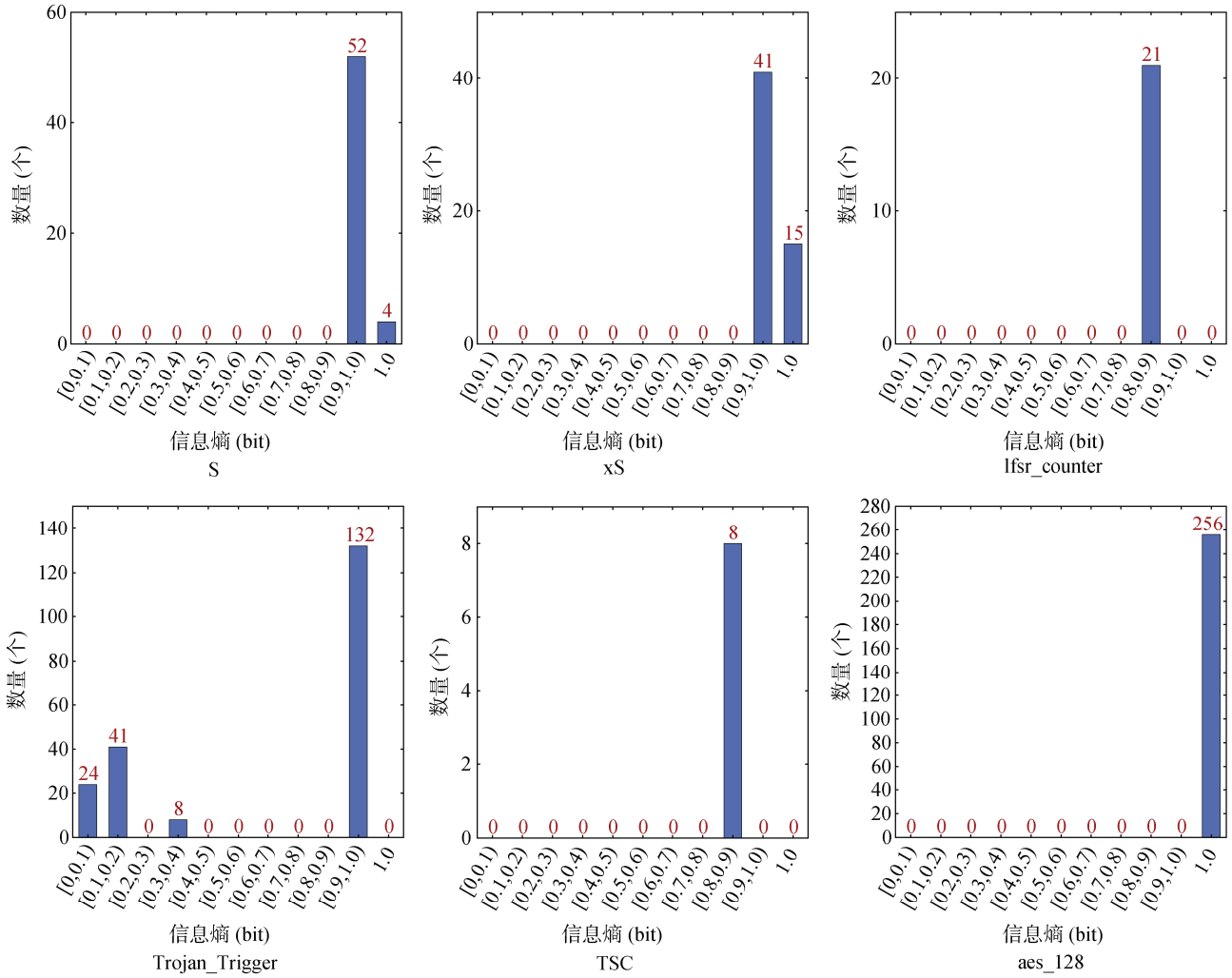

图 7　AES-T1000 各模块信息熵分布直方图

Figure 7　Information entropy distribution histogram of each module of AES-T1000

表 1　硬件木马检测结果

Table 1　Results of Hardware Trojan detection

测试向量	TNR	TPR	Accuracy	AUC
AES-T1600	0.956	0.976	0.976	0.966
AES-T1700	0.977	0.997	0.997	0.987
AES-T1800	1	0.996	0.996	0.998
AES-T1900	1	0.995	0.995	0.997
AES-T2000	1	0.996	0.996	0.998
AES-T2100	1	0.994	0.994	0.987
RS232-T700	1	0.921	0.931	0.960
RS232-T900	1	0.952	0.957	0.976
RS232-T901	1	0.971	0.975	0.985
RS232-T600	0.933	0.941	0.940	0.937
c6288-T002	1	0.979	0.979	0.989
c6288-T003	1	0.941	0.941	0.970
c6288-T004	1	0.938	0.939	0.969
c6288-T009	1	0.974	0.974	0.987
平均值	**0.990**	**0.969**	**0.971**	**0.979**

表 2　基于机器学习的硬件木马检测方法比较

Table 2　Comparison of hardware Trojan detection methods based on machine learning

检测方法	抽象层次	Accuracy	TPR	特征提取自动化	无需黄金参考模型
聚类算法[11]	门级网表	NA	100%	×	√
决策树[14]	RTL 代码	99.1%	100%	×	×
支持向量机[17]	门级网表	66.0%	81.0%	×	×
概率神经网络[21]	RTL 代码	NA	100%	×	√
图卷积网络[23]	RTL 代码	99.6%	88.4%	√	√
信息熵+聚类[24]	门级网表	NA	79.0%	√	√
本文方法	**LUT 网表**	**96.9%**	**97.1%**	**√**	**√**

RSCypher 模块中的低翻转信号为硬件木马的触发信号，而 modmult 和 modmult_0 模块中的低翻转信号为未连接信号以及正常运行时不会被使用的溢出控制逻辑。从实验结果可以得到，本文提出的木马相关属性提取与验证方法能精确捕获木马触发信号。

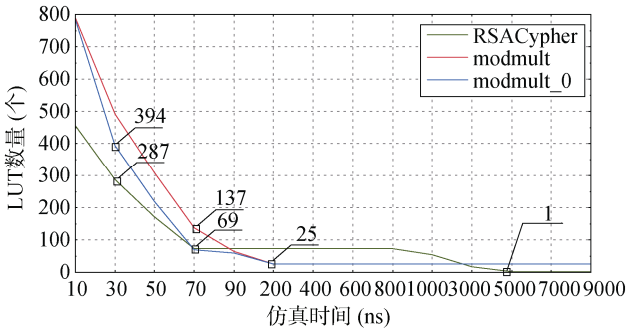

图 8　BasicRSA-T200 各模块低翻转 LUT 数收敛情况

Figure 8　Low-switching LUT number for each module of BasicRSA-T200

通过化简 eqOp 信号所在 LUT 的真值表, 提取的属性用断言语言描述为: assert (eqOp == 0)。使用 Yosys 的 SAT 求解器对该属性进行形式化验证, 证明结果如图 9 所示:

使用 Yosys 工具证明低翻转的 eqOp 信号是否始终为 0, 结果显示属性验证失败, SAT 求解器给出的反例表明在 indata＝32'h1fa0301 时会发生违反属性的情况, 此时信号 eqOp 为逻辑 1。属性验证求解结果与 BasicRSA-T200 测试向量中木马设计的触发条件完全一致。

7　结论

本文提出了一种基于 LUT 特征分析的硬件木马检测方法, 利用 LUT 初始化向量所包含的结构和行为特征, 可以实现木马特征量化分析、特征提取与匹配、木马相关属性的自动提取与形式化验证。以 Trust-Hub.org 中的木马基准为测试的实验结果表明本方法在检测准确度、定位精度、实现难易度等方面均具有显著优势, 能够为硬件木马的检测提供有效支撑。在后续研究中, 我们将探索如何提升模型的泛化能力, 使人工智能模型能够检测更多类型的木马。

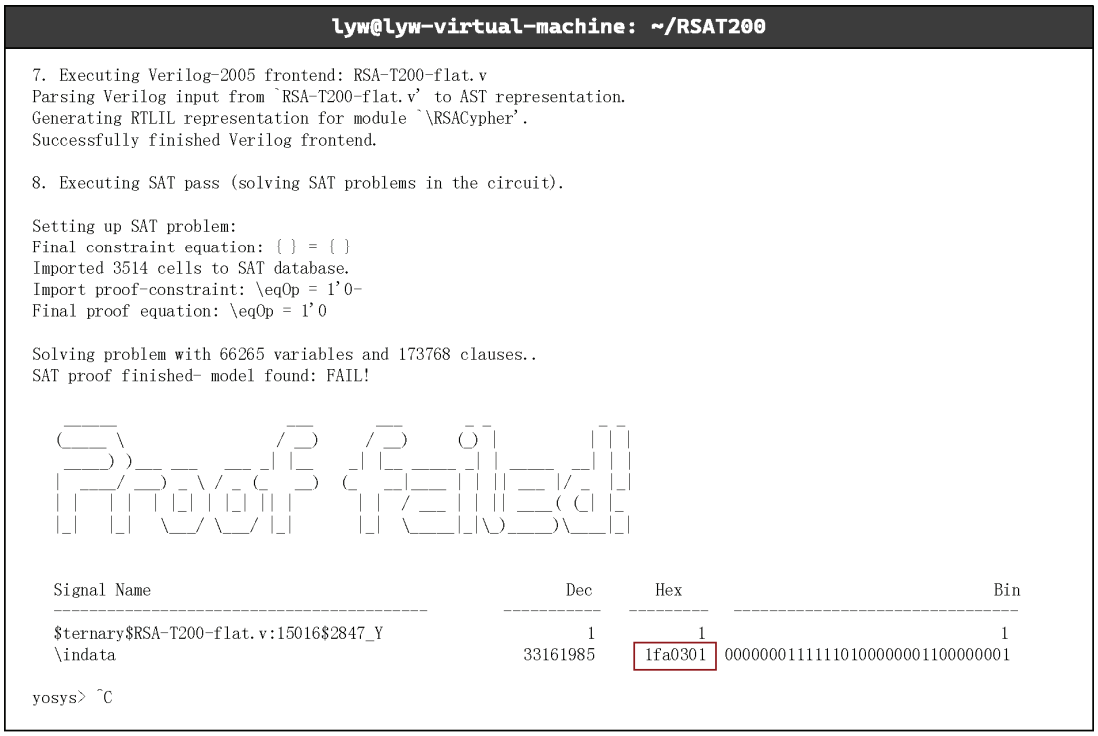

图 9　属性形式化验证结果

Figure 9　Formal verification results of properties

参考文献

[1] Huang Z, Wang Q, Yang P F. Hardware trojan: Research progress and new trends on key problems[J]. *Chinese Journal of Computers*, 2019, 42(5): 993-1017.
(黄钊, 王泉, 杨鹏飞. 硬件木马: 关键问题研究进展及新动向[J]. *计算机学报*, 2019, 42(5): 993-1017.)

[2] Baehr J, Hepp A, Brunner M, et al. Open Source Hardware Design and Hardware Reverse Engineering: A Security Analysis[C]. *2022 25th Euromicro Conference on Digital System Design*, 2023: 504-512.

[3] Mukherjee R, Rajendran S R, Chakraborty R S. A comprehensive survey of physical and logic testing techniques for hardware trojan detection and prevention[J]. *Journal of Cryptographic Engineering*, 2022, 12(4): 495-522.

[4] Pan Z X, Mishra P. Automated Test Generation for Hardware Trojan Detection Using Reinforcement Learning[C]. *2021 26th Asia*

and South Pacific Design Automation Conference, 2021: 408-413.

[5] Naveenkumar R, Sivamangai N M, Napolean A, et al. A Survey on Recent Detection Methods of the Hardware Trojans[C]. *2021 3rd International Conference on Signal Processing and Communication*, 2021: 139-143.

[6] Zhang J, Yuan F, Wei L X, et al. VeriTrust: Verification for Hardware Trust[C]. *The 50th Annual Design Automation Conference*, 2013: 1-8.

[7] Hu W, Ardeshiricham A, Kastner R. Hardware information flow tracking[J]. *ACM Computing Surveys*, 2022, 54(4): 1-39.

[8] Zhang Q Z, Zhao Y Q, Gao Y, et al. Survey on model checking based hardware trojan detection technology[J]. *Chinese Journal of Network and Information Security*, 2021, 7(2): 57-63.
(张启智, 赵毅强, 高雅, 等. 基于模型检测的硬件木马检测技术研究[J]. *网络与信息安全学报*, 2021, 7(2): 57-63.)

[9] Sabri M, Shabani A, Alizadeh B. SAT-based integrated hardware trojan detection and localization approach through path-delay analysis[J]. *IEEE Transactions on Circuits and Systems II*: *Express Briefs*, 2021, 68(8): 2850-2854.

[10] Pearce H, Surabhi V R, Krishnamurthy P, et al. Detecting hardware trojans in PCBS using side channel loopbacks[J]. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, 2022, 30(7): 926-937.

[11] Salmani H. COTD: Reference-free hardware trojan detection and recovery based on controllability and observability in gate-level netlist[J]. *IEEE Transactions on Information Forensics and Security*, 2017, 12(2): 338-350.

[12] Hasegawa K, Oya M, Yanagisawa M, et al. Hardware Trojans Classification for Gate-Level Netlists Based on Machine Learning[C]. *2016 IEEE 22nd International Symposium on On-Line Testing and Robust System Design*, 2016: 203-206.

[13] Piccolboni L, Menon A, Pravadelli G. Efficient control-flow subgraph matching for detecting hardware trojans in RTL models[J]. *ACM Transactions on Embedded Computing Systems*, 2017, 16(5s): 1-19.

[14] Huang H, Shen H H, Li S, et al. A Hardware Trojan Trigger Localization Method in RTL Based on Control Flow Features[C]. *2022 IEEE 31st Asian Test Symposium*, 2022: 138-143.

[15] Cheng X, Li L, Cheng W. A detection method of hardware trojans based on RTL[J]. *Microelectronics & Computer*, 2017, 34(3): 56-60.
(成祥, 李磊, 程伟. 基于 RTL 级硬件木马的检测方法[J]. *微电子学与计算机*, 2017, 34(3): 56-60.)

[16] Yang J Z, Zhang Y, Hua Y F, et al. Hardware Trojans Detection through RTL Features Extraction and Machine Learning[C]. *2021 Asian Hardware Oriented Security and Trust Symposium*, 2022: 1-4.

[17] Hasegawa K, Yanagisawa M, Togawa N. A hardware-trojan classification method using machine learning at gate-level netlists based on trojan features[J]. *IEICE Transactions on Fundamentals of Electronics*, *Communications and Computer Sciences*, 2017, E100.A(7): 1427-1438.

[18] Yan Y J, Zhao C H, Liu Y J. Hardware trojan detection based on multiple structural features[J]. *Journal of Electronics & Information Technology*, 2021, 43(8): 2128-2139.
(严迎建, 赵聪慧, 刘燕江. 基于多维结构特征的硬件木马检测技术[J]. *电子与信息学报*, 2021, 43(8): 2128-2139.)

[19] Zhao P Y, Liu Q. Density-Based Clustering Method for Hardware Trojan Detection Based on Gate-Level Structural Features[C]. *2019 Asian Hardware Oriented Security and Trust Symposium*, 2020: 1-4.

[20] Han T, Wang Y Z, Liu P. Hardware Trojans Detection at Register Transfer Level Based on Machine Learning[C]. *2019 IEEE International Symposium on Circuits and Systems*, 2019: 1-5.

[21] Demrozi F, Zucchelli R, Pravadelli G. Exploiting Sub-Graph Isomorphism and Probabilistic Neural Networks for the Detection of Hardware Trojans at RTL[C]. *2017 IEEE International High Level Design Validation and Test Workshop*, 2017: 67-73.

[22] Yasaei R, Chen L K, Yu S Y, et al. Hardware Trojan Detection Using Graph Neural Networks[EB/OL]. 2022: arXiv: 2204.11431. https://arxiv.org/abs/2204.11431.

[23] Yasaei R, Faezi S, Al Faruque M A. Golden reference-free hardware trojan localization using graph convolutional network[J]. *IEEE Transactions on Very Large Scale Integration* (*VLSI*) *Systems*, 2022, 30(10): 1401-1411.

[24] Lu R J, Shen H H, Feng Z H, et al. HTDet: A clustering method using information entropy for hardware trojan detection[J]. *Tsinghua Science and Technology*, 2021, 26(1): 48-61.

[25] Love E, Jin Y E, Makris Y. Proof-carrying hardware intellectual property: A pathway to trusted module acquisition[J]. *IEEE Transactions on Information Forensics and Security*, 2012, 7(1): 25-40.

[26] Rajendran J, Vedula V, Karri R. Detecting Malicious Modifications of Data in Third-Party Intellectual Property Cores[C]. *2015 52nd ACM/EDAC/IEEE Design Automation Conference*, 2015: 1-6.

[27] Hu W, Althoff A, Ardeshiricham A, et al. Towards Property Driven Hardware Security[C]. *2016 17th International Workshop on Microprocessor and SOC Test and Verification*, 2017: 51-56.

[28] Liu Z X, Arias O, Fu W M, et al. Inter-IP Malicious Modification Detection through Static Information Flow Tracking[C]. *2022 Design*, *Automation & Test in Europe Conference & Exhibition*, 2022: 600-603.

[29] Wu J M, Fowze F, Forte D. EXERT: EXhaustive IntEgRiTy Analysis for Information Flow Security[C]. *2022 Asian Hardware Oriented Security and Trust Symposium*, 2023: 1-6.

[30] Zhao J F, Shi G. Research on RTL level hardware trojan[J]. *Journal of Cyber Security*, 2023, 8(4): 139-152.
(赵剑锋, 史岗. RTL 级硬件木马问题研究[J]. *信息安全学报*, 2023, 8(4): 139-152.)

[31] Wang C G, Cai Y C, Zhou Q. Automatic Security Property Generation for Detecting Information-Leaking Hardware Trojans[C]. *2017 IEEE International Conference on Computer Design*, 2017: 321-328.

[32] Li D J, Zhang Q Z, Zhao D Y, et al. Hardware trojan detection using effective property-checking method[j]. *electronics*, 2022, 11(17): 2649.

[33] Hu W, Zhang L, Ardeshiricham A, et al. Why You should Care about Don't Cares: Exploiting Internal Don't Care Conditions for Hardware Trojans[C]. *2017 IEEE/ACM International Conference on Computer-Aided Design*, 2017: 707-713.

李一玮 于2022年在西北工业大学信息安全专业获得学士学位, 2025 年在西北工业大学网络空间安全专业获得硕士学位。研究领域为集成电路硬件安全。研究兴趣包括硬件木马检测、芯片设计安全验证。Email: lywei@mail.nwpu.edu. cn

鲁逸晴 于 2025 年在西北工业大学网络空间安全专业获得学士学位。研究领域为网络空间安全。研究兴趣包括芯片设计安全验证、密码学。Email: llyiyi@mail.nwpu.edu.cn

胡伟 于 2012 年在西北工业大学控制科学与工程专业获得博士学位。现任西北工业大学长聘教授。研究领域为集成电路硬件安全、网络空间安全。研究兴趣包括芯片设计安全验证、密码侧信道安全、硬件木马检测、处理器安全、密码学。Email: weihu@nwpu.edu.cn

宿豪 于 2024 年在西北工业大学信息安全专业获得硕士学位。现在西北工业大学网络空间安全专业攻读博士学位。研究领域为集成电路硬件安全。研究兴趣包括芯片设计安全验证、硬件木马检测。Email: suh@mail.nwpu.edu.cn

武玲娟 于 2013 年在北京大学微电子学与固体电子学专业获得博士学位。现任华中农业大学副教授。研究领域为集成电路硬件安全。研究兴趣包括芯片设计安全验证、处理器安全、硬件木马检测。Email: wulj@mail.hzau.edu.cn