硬件木马研究动态综述

赵剑锋, 史 岗

中国科学院信息工程研究所 第五实验室 北京 中国 100093

摘要 由于集成电路的设计制造技术越来越复杂, 使得芯片在设计及生产过程中充满潜在的威胁, 即有可能被加入硬件木马。硬件木马有可能改变系统功能, 泄漏重要信息, 毁坏系统或造成拒绝服务等危害。文章系统的介绍了硬件木马的产生背景、概念及性质、国内外研究现状, 对国内外研究现状进行了分析比较; 介绍了硬件木马的常见分类方式, 并结合实例加以说明; 常用的检测方法分类, 并结合案例, 说明各种检测方法的优缺点; 介绍了硬件木马的安全防范措施, 结合案例说明各种防范措施的特点; 最后总结全文, 指出现有研究存在的问题, 并展望了硬件木马的研究方向与重点。

关键词 硬件木马; 设计; 检测; 安全措施
中图法分类号 TP309.2 DOI 号 10.19363/j.cnki.cn10-1380/tn.2017.01.006

A Survey on the studies of Hardware Trojan

ZHAO Jianfeng, SHI Gang

Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

Abstract With the increasing complexity of IC design and fabrication, IC design and fabrication is endangered by potential threat like the inserted Hardware Trojan. Hardware Trojan may change the function of system, lead to the leakage of important information, damage the system or cause denial of service. This paper introduces the background, concepts and properties of Hardware Trojan. And then it analyzes the research status at home and abroad. Combined with cases of Hardware Trojan, this paper introduces its common classification, the general detection methods as well as their advantages and disadvantages. What's more, it applies typical cases to illustrate the features of each safety measure of Hardware Trojan. Finally, it puts forward the research direction and emphasis of Hardware Trojan after summarizing the existing problems in current researches.

Key words Hardware Trojan; design; detection; security approach

1 引言

如今, 信息安全[1-10]越来越受到人们的关注和重视。信息安全防御和保护技术更是多种多样。信息安全工具也在不断更新。攻击者为了达到某种恶意目的, 也在进一步研究新的方法和技术, 这些攻击方法和技术更是层层深入, 从软件层渗透到硬件层, 由此催生了硬件木马。然而, 硬件木马位于系统核心的、基础的层面, 很难被发现和检测出来, 它有可能给系统带来毁灭性的打击。

有关硬件的危害, 已有案例可循。1991 年,在海湾战争中, 美军通过激活设置在打印机芯片中的硬件木马, 导致伊拉克防空指挥系统的瘫痪[11]; 1994 年的“奔腾”处理器芯片内浮点运算器的缺陷, 导致个别情况下会出现计算结果的错误, 给 Intel 公司造成了 4.75 亿美元的损失[12]; 1999 年的奔腾 III 处理器芯片的序列号事件[13]; 2012 年剑桥大学的 Sergei Skorobogatov 和伦敦库欧-瓦迪斯实验室的 Christopher Woods 发表论文指出他们发现了 FPGA 芯片 ProASIC3 A3P250 上的硬件后门, 攻击者可以利用该后门监视和改变芯片上的数据[14]; 2012 年发生的 INTEL 指令集严重漏洞: 美国计算机应急预备小组发布一份安全报告, INTEL 处理器在 X86-64 扩展中执行 SYSRET 指令集的方式, 影响 INTEL 处理器上 INTEL64 扩展使用, 受影响的操作系统包括: 64 位 Windows 7、Windows Server 2008 R2、64 位

通讯作者: 赵剑锋, 博士研究生, 讲师, Email: zhaojianfeng@iie.ac.cn。
本课题得到国家“核高基”科技重大专项基金项目(No.2013ZX01029003-001); 国家“八六三”高技术研究发展计划基金项目(No.2012AA01A401)资助。
收稿日期: 2016-09-19; 修改日期: 2016-11-15; 定稿日期: 2016-12-22

FreeBSD 和 NetBSD、Xen 虚拟化软件、红帽企业 Linux、SUSE Linux Enterprise Server[15]。

由此看出, 虽然某些安全事件是硬件缺陷本身所致, 但是, 芯片中一旦加入硬件木马后, 就有可能改变系统功能, 造成信息泄漏或导致拒绝服务的发生, 对军事系统、金融设施甚至家用设备都可能造成危害。尤其是随着芯片设计与制造的全球化, 芯片的安全性变得越来越脆弱。芯片有可能在如下环节被植入硬件木马:

(1) 设计阶段。集成电路的设计离不开 EDA 工具, 即使是普通设计者, 只要掌握了 Verilog 或 VHDL 等硬件设计语言就可以进行芯片的设计, 这样一来导致从业人员成分复杂, 难以对其进行有效监督和管理。在这种情况下, 部分从业人员可能利用自己的专业知识, 在芯片设计过程中故意加入恶意的设计, 或者是从业人员本身水平较低, 无意中留下了有危害的漏洞。

(2) 进口和代工阶段。硬件设计耗资巨大, 再加上激烈的行业竞争, 使得大量硬件制造商依赖进口和代工方式来降低成本。这样, 一个硬件设备有可能需要不同的国家、地区、公司来共同完成, 然而, 难以保证每个参与方都是安全可靠的。

(3) 第三方的 IP 核。在芯片设计期间, 为了加快设计时间, 降低设计复杂度, 经常购买并使用第三方的 IP 核, 但是, 第三方 IP 核的安全性也是值得怀疑的。

一系列硬件安全事件的发生, 加上芯片生产过程复杂, 需要经过设计、综合、仿真、布局、布线、加工、测试、封装和组装等各个环节, 涉及各方面人员、EDA 工具及第三方的 IP 核, 难以保证其中任何一个环节都是安全的[16-25]。所有这些因素使得硬件安全受到极大的关注。

硬件木马已经成为目前新的研究课题。国内外的大学或学术机构在不同程度上对硬件木马展开了研究, 主要包括硬件木马的设计, 检测, 以及针对硬件木马采取的安全防范措施(研究文献情况如图 1 所示)。文献[19]主要对硬件木马的设计和危害作了介绍, 另外就硬件木马检测而言, 主要介绍了旁路信号分析方法。Julien Francq[26]等人主要从硬件木马的危害和检测两个方面来介绍。S. Bhunia [27] 等人是在 Chakraborty R. S 文章基础上总结的, 没有具体的案例分析。牛小鹏等人的综述[28]主要给出了硬件木马的概念和分类, 并给出几中具体的检测方法, 对当时的检测方法进行了总结。刘华锋等人的综述[29]主要介绍了硬件木马的概念及检测措施。任立儒等人的文章[30]主要介绍了硬件木马的危害。陈华锋等人的文章[31]主要讲到了硬件木马检测技术的发展趋势。文献[86]提出了一种芯片模型, 在这个模型基础上来描述硬件木马, 并且探讨了“pre-silicon”阶段和“post-silicon”阶段的硬件木马检测方法、防护措施。

以上所述是比较典型的介绍硬件木马相关研究情况的文献, 随着硬件木马研究的进一步深入, 出现了很多新情况, 新问题, 新技术, 新方法。因此, 本文在深入调研的基础上, 参考中外各主要数据库相关研究文章(如图 1 所示), 结合已有的研究成果, 对硬件木马的研究情况做了深入全面的论述。

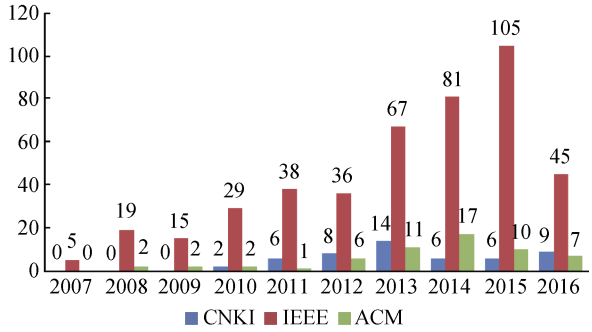

图 1 以“硬件木马”为标题的文章数量

与以往文献相比, 本文的贡献主要有: 列出了硬件木马危害的具体真实案例; 梳理了硬件木马概念的发展过程, 在总结以往成果的基础上给出了硬件木马的概念, 并结合案例做出详细解释; 增加了硬件木马与软件恶意代码以及硬件缺陷的比较; 给出了硬件木马设计实现的典型案例; 对硬件木马的检测技术进行了详细的划分, 并针对每一种检测技术的特点、适用场景及存在问题进行了归纳总结; 详细分析了硬件木马防护技术及其特点; 提出了具体明确的研究课题; 就硬件木马防护而言, 指出了一些具体的技术细节, 对国内硬件安全研究具有积极的推进意义。

文章主要包括以下部分: 硬件木马的概念、性质及研究现状; 硬件木马的分类方式; 硬件木马的典型案例; 硬件木马的检测方法; 防范硬件木马的安全措施; 在总结部分, 指出现在研究存在的问题, 并对未来的研究方向进行了较为具体细致的讨论。

2 硬件木马的概念及研究现状

2.1 概念

2007 年 IBM 沃森研究中心和伍斯特理工学院联合发表文章, 第一次正式提出了硬件木马的概念[32]。

硬件木马的概念大体经过了以下发展过程。

2007 年, D．Jia 等研究人员认为, 硬件木马是指芯片中含有被恶意篡改的电路, 这些电路一旦被激活, 就可以发起攻击, 使芯片损毁、难以正常运行或泄露用户私密信息[35]。

2008 年, X. Wang 和 M.Tehranipoor 等人在 IEEE 国际会议 HOST(Hardware-Oriented Security and Ttust)上, 把硬件木马定义为实现某种恶意目标的电路, 它能够在特定的触发条件下被激活, 以实现破坏性的功能, 或达到泄漏芯片内部信息的目的[99]。

2009 年, Yier Jin 等人认为, 硬件木马是能够对电子系统功能和可信性造成影响的恶意电路, 通过对芯片故意地或恶意地改变、添加、移除等方式来减弱电路的功能和可信性[37]。

2011 年, 天津大学任立儒等人认为: 硬件木马是一种被注入到电子电路系统的恶意电路模块, 通过改变原电路系统功能以达到监控或者打击敌手的目的[30]。同年, Mohammad Tehranipoor 认为硬件木马不仅指对微处理器、数字信号处理器等硬件进行修改, 也可以是对固件的修改, 如 FPGA 的位流[69]。Berk Sunar 与 Mohammad Tehranipoor 观点一样, 认为硬件木马不仅是对芯片硬件的修改, 可也以是对 IP 核或固件的修改[70]。

2012 年, 华中科技大学的郑朝霞等研究人员提出, 硬件木马是一种特殊的电路, 该电路能够实现特定的恶意目的[100]。同年, Mark R. Beaumont 在 2012DATE(Design, Automation and Test in Europe Conference)会议指出: 硬件木马是对原硬件设计进行了恶意修改, 在一定条件下被触发来达到改变系统功能的目的[69]。

2015 年, 文献[86]主要提出了一种芯片模型来说明硬件木马的概念, 该模型分为触发电路、硬件木马电路、负载电路三部分。

目前, 硬件木马还没有统一的概念, 总结之前的研究成果可以认为, 硬件木马主要是指: 在芯片或电子系统中故意植入的特殊模块或者设计者无意留下的缺陷模块, 在特殊条件触发下, 该模块能够被攻击者利用以实现破坏性的功能。硬件木马可以独立完成攻击功能, 如泄露信息给攻击者、改变电路功能、甚至直接破坏电路, 也可能与软件协同破坏系统功能。

与软件恶意代码(如软件木马、蠕虫、后门等)相比较而言, 硬件木马与硬件设计生产紧密相连, 它产生的历史要早于软件恶意代码, 随着软件恶意代码和软件杀毒技术的矛盾式发展, 攻击者开始深入到计算机的底层, 意图在底层硬件上植入硬件木马来达到攻击目的。硬件木马与软件恶意代码共同点: 都是人为设计的, 为了实现特定的攻击目的。但是, 两者最大的区别在于它们处于计算机体系的不同层面, 软件恶意代码处于计算机的软件层面, 依赖于具体的操作系统; 而硬件木马则处于硬件层面, 与具体的操作系统无关; 软件恶意代码具有传染性, 在特定的情况下可以在不同的计算机之间传播, 而硬件木马不可能具有传染性; 软件恶意代码通过杀毒工具或重装操作系统, 可以得到彻底清除, 但是, 硬件木马的隐蔽性极强, 不可能通过软件查杀的方法来检测和清除, 即便被发现, 也只能更换新的、安全的硬件来清除。

另外, 硬件木马也不同于硬件缺陷, 硬件缺陷是人们的粗心大意或者生产工艺本身的限制造成的, 例如, 器件的磨损、老化等原因都可能造成硬件缺陷, 这是人们无法控制的, 不以人的意志为转移的; 而硬件木马则是攻击方故意预先注入的, 完全是人为行为。再者, 硬件木马与硬件缺陷的引入时机也不同。硬件缺陷可能出现在硬件设计、生产或使用的各个阶段; 而硬件木马一般只能在设计或生产过程中人为地加入。

通过与软件恶意代码及硬件缺陷对比可知, 硬件木马具有预先注入、与硬件紧密相关、需要特定条件激活等特征。

通过本小节以下论述, 可以对硬件木马的概念有个具体的认识。

一般而言, 芯片包含一系列功能模块, 每个模块执行一种特定任务。模块间的数据经系统总线传送, 总线仲裁器模块会对总线上的数据流进行控制。如下图 2 所示, 以手机为例, 数据可能会从内存(1)传送到执行计算的模块(2), 然后再传送到负责信息编码和解码的模块(3), 最后再传送到负责与芯片外部交换数据的模块(4)和(5)。

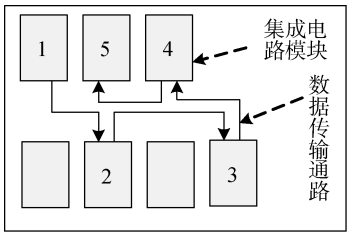

图 2　数据在芯片模块间的传递

在正常情况下, 芯片会一直工作。当其中的硬件木马电路被激活后, 芯片会按照预先设定的恶意目标发起攻击。触发条件可以有多种形式, 如特定数据和时间, 或者是外部发送的特定编码格式数据包中

的“唤醒呼叫”。一旦激活, 木马随即就会公开或秘密地展开攻击。

在公开攻击时, 硬件木马可以使正常运行的芯片罢工。比如在手机中, 被感染的模块拒绝让出被它占用的总线, 使其他模块之间无法通信。在这种情况下, 该芯片会完全停止工作。如下图 3 所示。

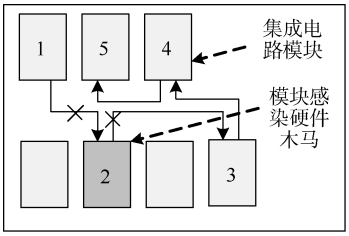

图 3　芯片模块受到攻击

在隐蔽攻击中, 硬件木马电路的行为不会留下任何痕迹。隐蔽攻击特别令人担忧, 因为没有明显迹象表明系统运行出现了问题。但是, 硬件木马电路会将机密数据发送到芯片外部的某个地址, 或配合其他被感染的系统发动攻击。如下图 4 所示。

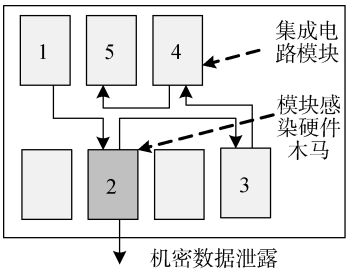

图 4　机密数据被芯片模块泄漏

从以上介绍可以看出, 硬件木马具有破坏性、寄生性、隐蔽性和潜伏性。

简单来讲, 集成电路, 也就是常说的芯片, 是一种刻蚀在半导体晶片(通常是硅)上的电路。现代芯片非常小, 最大也就几平方厘米, 但可容纳数十亿个晶体管。由于现代芯片拥有如此高的复杂性, 一些可能被木马利用的漏洞也存在其中。

总之, 在一般情况下, 芯片可以按照正常功能来工作; 在特定情况下, 芯片中的硬件木马被触发, 随后展开攻击。

2.2　研究现状

从硬件木马的概念提出以后, 硬件木马逐渐成为研究热点[33-41]。

2007 年, 美国开始举办第一届信息安全月活动, 而后两年的信息安全意识周竞赛(Cyber Security Awareness Week Challenge)题目就是如何设计、检测及防范硬件木马; King S.T 等人设计完成了称为 IMPs(Illinois Malicious Processors)恶意处理器[33]; 随后, 他又与 Hicks .M 等人一起研究了在芯片设计阶段, 通过修改源文件或者添加 IP 核这两种方式来引入硬件木马[40]; Jia Di 等人构建了硬件威胁模型[35]; Abhishek Das 等人提出了针对硬件木马的监控系统[36]; Agrawal D 等人对于 RSA 电路中的一些硬件木马的实例进行了分析介绍, 并提出了 IC 指纹检测法[32]; Yier Jin 等人在 RTL 层次上实现了 8 种硬件木马攻击技术[37]; Santos 等人在 8051 微处理器中加入了硬件木马[39]; Sturton C 等人研究了更为隐蔽的硬件木马的注入方法[23]; Hicks 等人提出了 UCI(Unused Circuit Identification)方法用于检测电路设计阶段注入的硬件木马[40]; Yier Jin 等人研究了无线加密芯片中的硬件木马, 以及它的工作方式和检测技术[41]。2008 年, IEEE 开始举办专门针对硬件木马和硬件安全的国际学术会议 IEEE international Workshop/Symposium on Hardware- Oriented Security and Trust。

国外的一些学校、公司等机构较早展开了对硬件木马的研究。例如, 康涅狄格州大学、凯斯西储大学、新墨西哥大学、加利福尼亚大学、耶鲁大学、德克萨斯大学、莱斯大学等高校都有相关研究人员对硬件木马问题进行了研究。另外, 德国波鸿 IT 安全研究中心、IBM 华盛顿研究中心、Intel 公司、美国 Phoenix Technologies Ltd 公司、加拿大海军研究院等都陆续开展了硬件木马的研究工作。

国内如军械工程学院、华中科技大学、华南理工大学、天津大学、国防科学技术大学、信息工程大学等高校相关研究人员对硬件木马的设计检测等问题进行了初步的研究。

总之, 国外对硬件木马的研究开展较早, 国内 2010 年才有中文文献(见图 1)。但是, 硬件木马的研究是刚刚起步, 没有统一的概念、理论或指导方法, 大多研究都是基于仿真和 FPGA(Field－Programmable Gate Array)实验, 到目前为止, 各主要研究单位还在进一步探索当中。

3　硬件木马分类

关于硬件木马的分类, 目前比较流行的分类标准有两种[42,43]。

第一种分类根据硬件木马的形成阶段、所处层次、激活方式、功能效果及所处位置进行了相应地划分, 如图 5 所示。

所谓硬件木马的形成阶段, 其实就是芯片的生产过程, 从提出概念到封装测试的整个过程, 在这

个生产过程的每个阶段都有可能被注入硬件木马。

就抽象层次而言, 主要是指, 对于一个物理电路, 可以在不同的层次上用硬件描述语言(如 Verilog、VHDL 等)来描述它, 如果只从行为和功能来描述, 则属于行为描述类型, 包含系统级、行为描述级、寄存器传输级三个层次, 这三个层次描述的较为抽象。如果从电路的结构来描述, 那么就属于结构类型, 门级属于结构类型描述。晶体管级和物理级则属于具体的、与生产芯片有关的电路器件的描述了。硬件木马也是一样的, 可以在不同的抽象层次来描述它。

就激活方式而言, 从硬件木马运行时间来说, 可以分为: 硬件木马一直运行和等待特定条件触发运行两种; 就触发而言, 可以分为内部触发(如在特定时间触发; 或特定物理条件, 如温度)和外部触发(如用户输入特定的字符串, 或其他部件给出特定的信号)两种。

就功能效果划分来说, 硬件木马主要实现这样几个功能: 一是改变芯片原来的功能, 二是改变芯片的属性, 三是造成芯片当中信息的泄漏, 四是造成芯片无法运行, 即所谓的拒绝服务。

就硬件木马所处的位置而言, 主要是根据计算机的组成部分来划分的, 如处理器, 内存、输入输出设备, 或时钟、电源等。

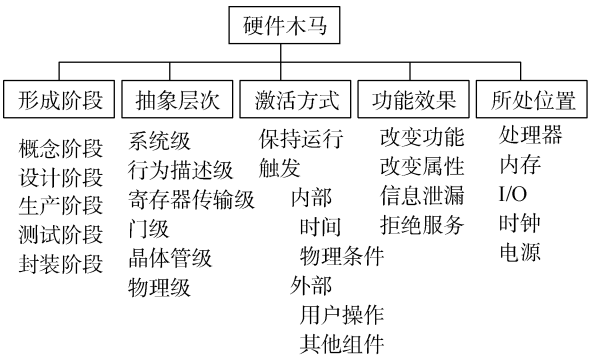

图 5 第一种硬件木马分类[42]

第二种分类把硬件木马分成触发部分与负载部分, 如图 6 所示, 图 7、图 8 和图 9 给出了具体的例子。

在第二种分类方式中, 就触发部分而言, 主要看硬件木马电路的特点, 即是数字的, 还是模拟的, 如果是数字电路, 那么是组合数字电路还是时序数字电路(时序又分为同步、异步等)。负载部分也是根据电路是数字的还是模拟的来划分。

图 7 中的硬件木马触发部件为一个或非门, 负载部分是一个异或门。当输入端 A 和 B 同时为 0 时, 通过与门之后 C 为 0, 通过触发逻辑或非门输出为 1, C 和或非门的输出端到异或门, 输出 C modified 为 1。所以在 A 和 B 输入同为 0 时, 硬件木马通过触发部件和负载部件工作, 改变了 C 的输出。A 和 B 在其他输入情况下, 输出结果不改变。

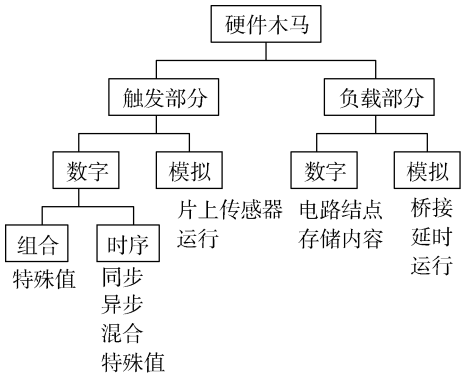

图 6 第二种硬件木马分类[43]

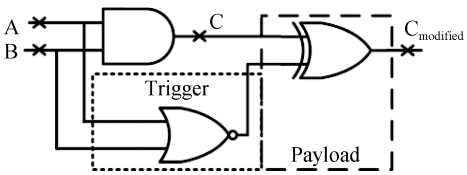

图 7 组合逻辑触发[43]

图 8 是通过一个同步计数器作为触发部件, 一个异或门作为负载。当计数值达到 2^{k-1} 时, 硬件木马触发, 从而改变了 ER 的值为 ER*。

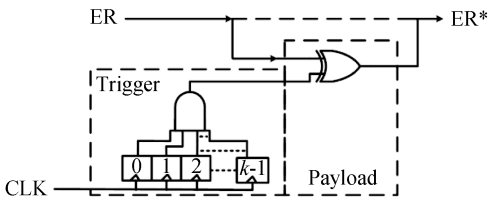

图 8 同步时序逻辑触发[43]

图 9 通过增加电容, 影响路径的延时, 达到改变输出值的目的。

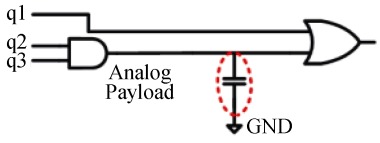

图 9 模拟负载[43]

4 硬件木马举例

下面通过一些具体的、典型的案例来了解硬件木马的设计产生过程, 以及硬件木马的具体功能。

文献 [33] 通过修改 Leon3(one open source

SPARC design)处理器的 VHDL(Very-High-Speed Integrated Circuit Hardware Description Language)源代码, 提出并设计了一款带有硬件木马的处理器 IMPs(Illinois Malicious Processors)。该处理器实现了两种攻击模式: 特殊内存访问模式和影子模式。

特殊内存访问机制使硬件支持恶意程序突破操作系统的内存访问控制策略, 即绕过 MMU(Memory Management Unit)保护, 不通过内存访问检查机制, 使得在用户模式下可以访问特权存储区域, 比如操作系统运行的区域。具体过程如下:

(1) 恶意软件在数据总线发送一个事先定义好的数据序列;

(2) 该数据序列触发内存访问电路;

(3) MMU 检查到此数据序列后, 对内存访问不作保护检查;

(4) 授予恶意软件访问所有内存区域的特权, 达到其攻击目的。

影子模式允许攻击者隐蔽地执行固件代码。如图 10 所示。影子模式中的 shadow i-memory 和 shadow d-memory 中的指令和数据, 可以执行全部特权操作。在正常模式下, 处理器可以通过高速缓存进行取指令等操作。但在影子模式运行时, 处理器限制了内存总线上的活动。

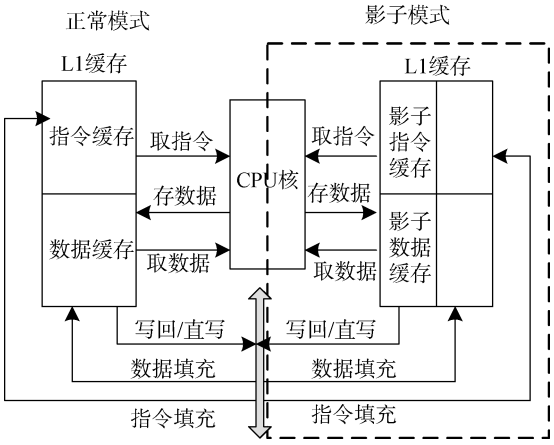

图 10 影子模式激活时硬件的对比

图 11 所示的是, 攻击者通过网络发送 UDP 数据包, 含有硬件木马的处理器在接收到特定的字节序列时, 把固件代码拷贝到保留的 shadow i-memory 和 shadow d-memory 中, 然后, 输入预先定义好的密码“letmein”, 就能以 root 权限登录系统。

通过以上内容可以看出, 文献[33]主要在设计阶段加入了硬件木马, 它属于寄存器传输级, 激活方式涵盖了内部和外部两种, 它的功能主要是提升权限和泄漏信息。硬件木马所处的位置为处理器内部。

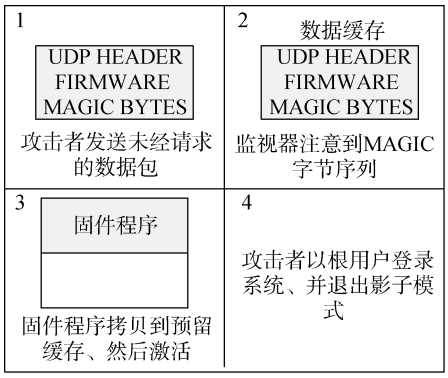

图 11 通过影子模式实现 root 登录

文献[33]在 FPGA 上进行了实验验证。

文献[37]在 RTL(Register Transfer Level)级别上设计出了一款嵌入硬件木马的加密机, 代号叫做 Alpha, 如图 12 所示, 并在 Spartan-3 FPGA 上实现, 文章分析了各个硬件木马的触发与负载部分, 并给出了加入硬件木马前后的 FPGA 的资源开销变化情况。

文中设计的加密机使用 AES-128(Advanced Encryption Standard)加密算法。若要传输一条消息, 操作员要按照以下步骤操作:

(1) 使用“Key Select”开关选择密钥。

(2) 按下“INI System”按钮。

(3) 利用与“KB IN”端口连接的键盘输入明文。VGA 监视器与“VGA OUT”端口连接来显示明文。“Input Status”LEDs 表明输入缓存已经使用了多少。

(4) 按下“Start Encryption”按钮启动加密, 加密引擎从输入缓存中读入明文, 把相应的密文写入到输出缓存中。

(5) 按下“Transmit”按钮, 把输出缓存中的内容发到 RS232 端口。

图 12 中标注有①符号的地方, 表明在源文件中加入了硬件木马。硬件木马 Tro1 在文件 alphatop.v 中, Tro2 在文件 kb2ascii.v 中, Tro3、Tro4、Tro6 在文件 alphatop.v 中, Tro5 在文件 async_transmitter.v 中, Tro7 在文件 async_receiver.v 中, Tro8 在文件 kbtop.v 和 kb2ascii.v 中。以 Tro1 为例, 当输入“New Haven”时, 触发硬件木马, 信息从 RS-232 TxD 泄露出去; 对于 Tro3, 当输入“Moscow”时, 触发木马, 输出结果用“boston”替换了“Moscow”, 其他各个硬件木马作用各异, 不再赘述。

文献[38]在制造阶段修改处理器, 以达到提升权限的功能。该文的特点主要通过较少的硬件开销, 达到植入硬件木马的目的, 如通过增加缓冲器, 或改变总线的根数, 或者是在距离相近的总线之间搭建一个桥等方式来实现。文中展现了在高性能处理器

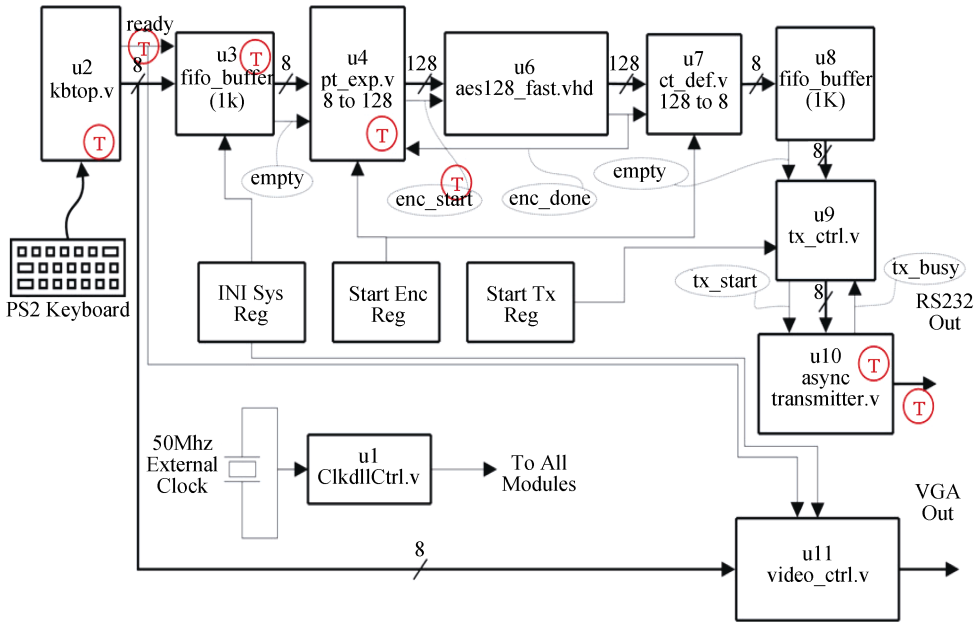

图 12　Alpha 体系结构[37]

和一般的嵌入式处理器上的硬件木马的特性。

文献[39]在 8051 IP 核(intellectual property core)中加入了硬件木马。共有三个木马: 第一个是后门木马(Back door Trojan), 主要用来把部分或全部内存内容通过串口复制出来; 第二是计数器炸弹木马(Bomb Counter Trojan), 通过一个计数器来控制其被编程的次数, 一旦超过预先设定的数值, 则将模块的功能锁住或使其无法使用; 第三个是能量沉陷木马(Power Sink Trojan), 它通过获悉 RC5 加密算法每半轮的进位标志, 从而得到密钥。可以看出, 文献[39]也主要是在设计阶段加入了硬件木马, 以达到信息泄漏和拒绝服务的功能。

文献[44]通过向 AES(Advanced Encryption Standard)模块加入硬件木马逻辑电路, 如图 13, 主要是两条硬件木马路径, 一条与控制器连接, 另外一条把加密模块与解密模块连接到一起。

在图 13 中, 加入硬件木马之后, 一旦满足攻击者预先设定好的条件, 就会触发硬件木马, 那么半编码的数据就会沿着预先设定好的木马路径, 从加密模块发送到解密模块, 结果是, 明文直接从输出接口输出。另外, 当从 AES 核输入密钥时, 也会通过木马路径传到控制器, 密钥也会直接输出。文献[44]也是通过修改 HDL 源文件来实现硬件木马, 通过外部触发达到泄漏信息的目的。

文献[54]专门探讨了通过 JTAG 接口来进行攻击的可能性。如通过嗅探或者修改 JTAG 接口的数据输入引脚 TDI 和输出引脚 TDO 信号; 控制 JTAG 接口的模式选择引脚 TMS, 或修改时钟引脚 TCK; 另外,

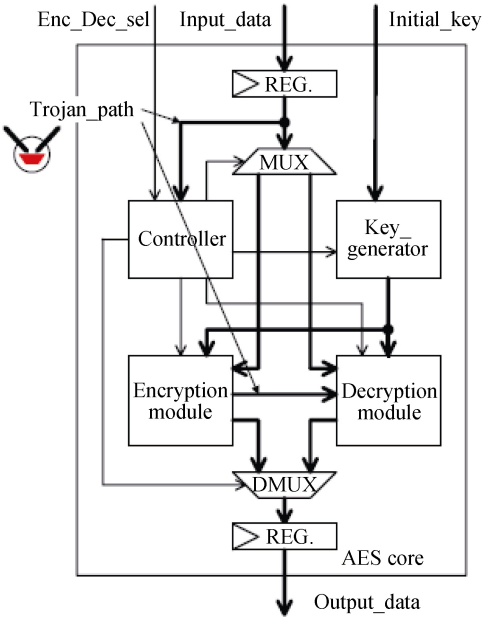

图 13　加入了硬件木马的 AES 核[44]

也可以通过 JTAG 接口, 窃取用户的密钥。

通过以上实例, 可以看出, 硬件木马的设计制作方式种类很多, 其存在的位置也是不一而足。硬件木马可以用来窃取信息, 可以破坏芯片功能, 也可以达到其他一些攻击目的。

5　硬件木马检测方法

硬件木马检测方法主要分为破坏式和非破坏式两大类。

对于破坏式方法具体实现而言, 主要有物理检测技术。

物理检测[95]属于破坏式检测方法, 严格地讲, 这种破坏式检测方法对于一些逻辑比较简单的芯片来说, 是一种非常有效的硬件木马检测手段。但是, 随着芯片集成度越来越高, 这种芯片逆向方法则显得力不从心, 并且会付出巨大的人力、物力、财力代价, 结果可能不尽人意。

物理检测通常将待鉴别器件开封后, 对电路进行逐层扫描, 然后根据扫描图像重建原始设计, 最后通过版图比较找到电路中的硬件木马。这其实是一种破坏式检测方法, 也就是芯片逆向工程分析方法, 即通过解剖、照相、制版、提图、仿真、验证、测试等工序完成检测。如文献[59]和[61]就是利用逆向工程来检测硬件木马。

非破坏式检测方法分为侵入式(如内建自测技术)与非侵入式(如功能测试技术和旁路分析技术)两种。侵入式分为保护类与辅助类, 即通过向芯片加入保护或辅助单元来完成硬件木马的检测。如文献[58]通过加入数据选择器来检测硬件木马。

非侵入式分为实时运行检测和测试检测。测试又分为逻辑向量测试与旁路信号检测。旁路信号检测是目前使用较多的检测方法。如文献[62]就是采用非侵入的方式, 通过监测外围输入输出信号来判断硬件木马。

对于非破坏式方法的具体实现而言, 主要包含功能测试技术、内建自测试技术和旁路分析技术。

功能测试方法是一种基于 ATPG(自动测试图形生成)的硬件木马检测技术[96]。它的基本原理是: 在芯片的输入端口施加激励, 在芯片的输出端口监测并观察, 如果输出的逻辑值与预计的输出不相符, 则可以断定发现了一个缺陷或木马。缺陷是在生产制造过程中, 由于设计、原材料、生产工艺、操作等不合理造成的, 具有随机性和不确定性, 如薄膜厚度、薄膜应力、折射率、掺杂浓度等, 这些因素造成的后果也具有随机性。而硬件木马是人为设计的, 具有明确的目的, 它主要为了实现逻辑上的一些功能, 如窃取信息等, 因此在满足触发条件情况下, 应该是稳定的和确定的。

内建自测技术: BIST(Built-In Self-Test)是一个芯片的额外功能模块[97]。芯片中除了包含规范中定义的那些功能元件外, 还可以设计一些额外的电路结构来监测芯片内部的信号, 或者监测芯片缺陷。可信的芯片通过 BIST 电路产生一个签名(校验和或指纹), 而有缺陷的芯片或被植入木马的芯片产生的却是另外一个签名。这种利用 BIST 来检测硬件木马的方法也被称为硬件可信性设计。

旁路分析技术[82]: 任何一个器件在工作时总是会发出各种各样的旁路信号。旁路信号主要包括: 热信号、电磁辐射信号、功耗信号、以及电路延时的信息等。硬件木马会对 IC 的一些物理参数, 如电源瞬态电流, 功耗或路径延时产生影响, 通过观察这些影响就有可能检测出 IC 中是否有木马存在。如文献[65]通过电流的变化来检测硬件木马。

表 1　主要检测技术特点比较

主要检测方法特点				
技术名称	借助工具	特点	适用场景	存在问题
物理检测[95]	1. 光学扫描显微镜。 2. 电子扫描显微镜。 3. 电压对比成像器。 4. 电荷诱发电压调整器。 5. 皮秒成像电路分析仪等。	1. 逆向工程剖析。	结构较简单的芯片。	1. 检测耗时, 投入巨大; 2. 难以获取对比标准, 会造成电路损坏, 对日益精细的芯片内部电路没有较好的措施; 3. 设备精度达不到要求。
功能测试[96]	1. 自动测试平台	1. 通过对管脚输入测试向量来进行穷举测试。 2. 稳定性高, 受噪声影响小。	特定引脚触发硬件木马的芯片	1. 测试向量数量庞大; 2. 测试时间过长; 3. 应用范围有限。
内建自测[97]	不需要	1. 在芯片生产过程中要加入测试模块或扫描链, 对关键点进行测试。	在特定管脚能产生“签名”信息的芯片。	1. 不能全面监测芯片电路。只能对关注点进行监测。
旁路分析[82]	1. 高精度示波器。 2. 高精度温度检测器。 3. 高精度功耗分析仪器。 4. 高精度频谱分析仪器等。	1. 检测精度高, 条件限制少, 不要求木马被触发, 只要电路工作, 就可以采集信号。 2. 需要与标准芯片比较相应的信号值, 如电流、温度等。	目前应用比较广泛; 利用功耗和时序分析的场景较多。	1. 对设备要求精度高, 纳米技术和测量噪声会影响旁路信号, 影响判决; 2. 需要标准芯片电路, 在实际情况下难以获得标准芯片电路。

在众多研究文献中, 目前应用比较广泛的是旁路分析方法, 这种方法具有典型性和代表性, 现介绍几个案例如下。

文献[45]提出了基于瞬时功耗的硬件木马检测方法, 通过奇异值分解算法处理功耗值, 在 FPGA (Field－Programmable Gate Array)上验证了该方法的可行性。其基本原理是: 当电路处于工作状态时, 会消耗电能量, 芯片中的硬件木马也是如此。因此, 有硬件木马的芯片与没有硬件木马的芯片,在相同的输入测试向量的激励下, 它们功耗瞬态变化有细微的差别。可以推测, 硬件木马规模越大, 其消耗的能量也越多, 那么表现出来的差别就越大。所以, 通过比较待测芯片与无木马芯片之间的功耗瞬态变化, 就能判断出待测芯片中是否存在硬件木马, 但是, 该方法的使用前提是要有无硬件木马的芯片。文献[45]检测结果如图 14。

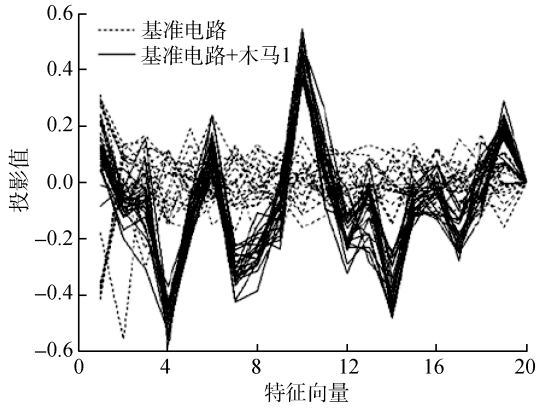

图 14　基于瞬时功耗的硬件木马检测[45]

文献[46]也用了功耗分析的方法, 采用了投影寻踪算法来处理, 通过基于绝对信息散度的投影寻踪技术, 将芯片运行过程中产生的高维旁路信号投影到低维子空间, 在信息损失尽量小的前提下发现原始数据中的分布特征, 从而实现芯片旁路信号特征提取与识别。如图 15 所示。

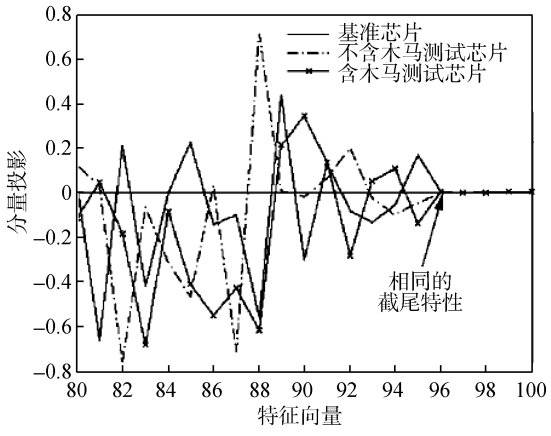

图 15　采用 K-L 变换分析的硬件木马检测结果[46]

文献[45]、[46]虽然提供了一种通过旁路信号检测电路的方法, 但是这种方法在实际中是不可行的, 因为它需要一款标准的不存在硬件木马的芯片。在实际生产过程中, 不可能从数以千万计的芯片中找出这样的芯片, 所以这种研究方法只能在能停留在实验中, 没有实际的应用价值。

文献[47]提出了基于路径延时的硬件木马检测方法。该文献中, 通过向 SEA(Scalable Encryption Algorithm crypto)中加入硬件木马(在门级和布局级), 它基于木马路径与正常路径延时不同的事实, 提出路径延时检测方法, 该方法开销少、有一定效果。下图是加入硬件木马与未加入硬件木马的路径时延的对比。如下图 16 所示。

Data path	Enc_data [4]	Enc_data [5]	Enc_data [6]	Enc_data [7]
W/O Trojan	5063ps	5063ps	5063ps	5063ps
W/T Trojan	5069ps	5143ps	5096ps	5068ps
Increased delay	*6 ps*	*80 ps*	*33 ps*	*5 ps*

图 16　基于路径延时的硬件木马检测[47]

文献[47]虽然提供了另外一种方法, 其本质与文献[45]、[46]是一样, 都是通过把感染硬件木马的芯片的某一特征参数与标准的不存在硬件木马的芯片做比较, 在比较时, 设定某一个合理的阈值, 以此来判断待测芯片中是否含有硬件木马。

6　安全措施

文献[48]提出了硬件木马的安全防范措施。如图 17 所示和表 2 所示。

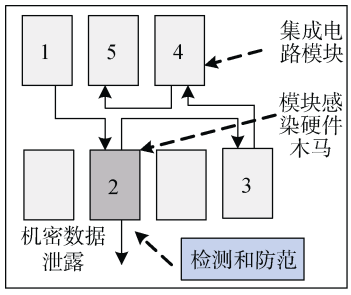

图 17　硬件木马防范示意图

文献[49]总结了内存受到的威胁, 比如欺骗攻击(spoofing attack), 重组、拼接攻击(splicing attack), 重放、重演攻击(replay attack), 相应地介绍了目前内存保护方面所采用的技术, 如斯坦福大学提出的 XOM(eXecute Only Memory)架构, 它用来保护数字

表 2 安全措施[48]

安全措施	阻止内容	工作原理
内存守卫	对流氓模块访问存储禁区的所有尝试予以阻止, 有助于防止监听或数据盗窃。	内存守卫可以确保模块只能访问授权的存储区域, 并且标记未经授权的访问企图。
安全系统总线	阻止可能导致芯片运行变慢甚至完全停止的恶意占用系统总路线的行为。	总线通过不同模块来分析总线访问统计模式, 并标记异常行为。
输入/输出监控	当芯片企图将数据复制到芯片外时予以阻止。	电路对芯片上的数据流入与流出进行分析, 并与预期模式进行比较, 标记其间的差别。
模块现场测试	对试图感染的健康模块的木马攻击进行防御。	芯片上的检测器会定期检测模块, 确保芯片正常运行。
额外的可编程逻辑电路	允许电路隔离受感染的模块并替代它工作。	配置额外逻辑电路来替代被隔离的功能块, 但这会降低芯片运行速度。

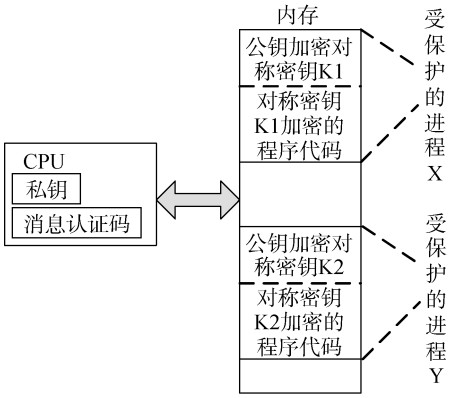

图 18 XOM 安全模型示意图

版权问题。XOM 在处理器内部有加密引擎, 程序运行在相互独立的存储隔间内(Compartment), 不同的数据属于不同的隔间, 系统运行时不允许程序访问其他隔间的数据[50]。图 18 是 XOM 架构示意图。XOM 上的软件由软件厂商进行加密, 它只能运行在特定的处理器上。为了增加安全性并且提高性能, 根据对称加密算法(处理速度相对快)和非对称加密算法(处理速度相对慢)在性能上的不同, XOM 使用了密钥共享协议。首先, XOM 处理器芯片中存储一对非对称密钥对(Kxom, Kp), 其中, Kxom 为私钥, Kp 为对外公开的公钥。应用程序发布者使用自己的对称密钥 Ks 对程序加密生成相应的密文, 然后通过用户处理器公钥 Kp 来加密密钥 Ks, 最后将程序密文和加密后的 Ks 一起发给用户。系统运行此应用软件时, 先使用自己的私钥 Kxom 来解密得到软件密钥 Ks, 然后通过 Ks 来解密程序并运行。通过这种方法, 软件发布者可以通过使用相应的处理器公钥 Kp 来加密 Ks, 从而使得相应的程序只能运行在特定的处理器上, 实现了对软件版权的保护。

另外, XOM 还通过为流出处理器的数据附加 MAC 值来实施完整性保护。通过将地址合并计算在 MAC 中, XOM 可以阻止攻击者替换不同地址处的存储器块来破坏完整性。不过, XOM 无法对抗重放攻击。

另外, 麻省理工学院提出的 AEGIS(Architecture for Tamper-Evident and Tamper-Resistant Processing) 架构, 它对内存的数据提供了机密性和完整性保护[50]。它假定处理器和操作系统的部分内核是安全的(Security Kernel)。它提供了两种工作模式: TE (Tamper-Evident) 和 PTR(Private and authenticated Tamper-Resistant)。TE 为系统数据提供了完整性保护, 确保软件运行过程中能够探测到对数据的篡改行为。PTR 提供了对数据的完整性和机密性保护。AEGIS 架构如图 19 所示。

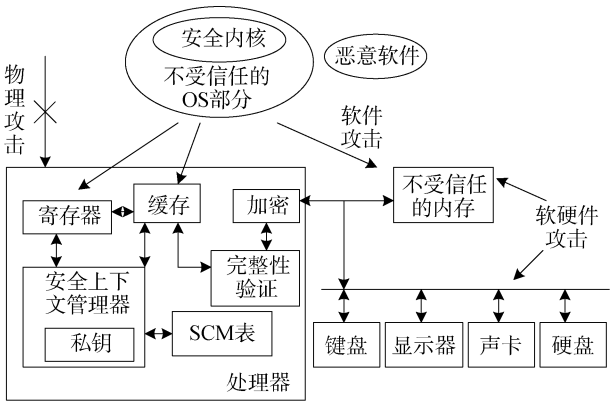

图 19 AEGIS 安全模型示意图

在 Security Kernel 中, AEGIS 采用安全上下文管理器 SCM(Secure Context Manager)来维护每一个进程的相关安全信息。每条记录如下表 3 所示。

表 3 AEGIS 的进程安全条目[50]

SPID	H(Prog)	Regs	Hmem	0/1	Kstatic,Kdynamic

其中, SPID 为安全进程 ID, 值为 0 时表示不受安全保护的普通进程。H(Prog)表示相应的哈希值。Regs 代表相应的寄存器及其值。Hmem 用于完整性校验。

0/1 表示该进程的工作保护模式状态(TE 或 PTR)。Kstatic 为对称密钥, 用于加解密应用程序, 每个应用程序都有一个唯一的 Kstatic, 在程序运行过程中保持不变。程序运行过程中产生的数据使用 Kdynamic 进行加密, 不同会话产生的 Kdynamic 是不同的, 在会话结束时该密钥也就失效了。

AEGIS 架构不仅可以用于软件版权保护, 也可以用于认证执行和数字版权管理上, 但就机密性和完整性而言, AEGIS 采用了直接块加密和 Hash 树校验方法, 系统延迟开销较大, 效率较低, 实用性不是很强。

文献[51]提出一种安全系统总线来阻止非法的访问。图 20 是在总线上加入了硬件木马。图 21 是提出的一种安全总线架构。其原理是: 当总线仲裁器(arbiter)收到主机的通信请求后, 在其释放总线控制权前, 向请求者(requester)和响应者(responder)同时发送一个由随机数发生器(random number generator)产生的随机数。发送者利用随机数来同步数据。合法的数据随机掺杂假数据发送到总线上。只有授权的接收者才能利用随机数同步数据得到合法源数据。

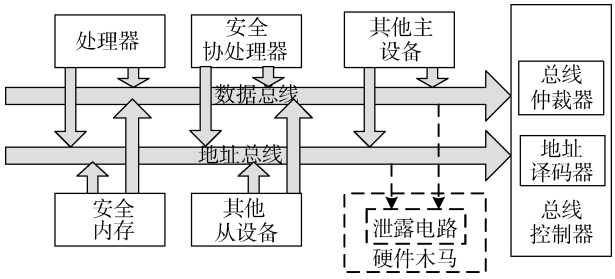

图 20 在芯片中加入硬件木马

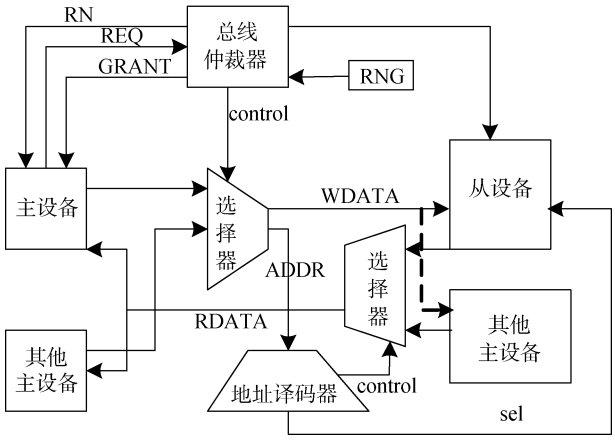

图 21 提出的安全总线架构[51]

文献[52]提出: 在芯片中加入监测和防御模块, 包括传感器、中央控制逻辑和信号控制单元等, 如图 22 所示。其中信号探测(signal probe networks, SPN)模块是可编程的, 实现了对正常电路信号进行选择采样监测; 安全监视(security monitors, SM)模块也是可编程的, 它负责分析 SPN 送来的信号; 安全控制和处理模块实现对 SPN 和 SM 模块的重配置编程, 而配置数据则以加密的方式存放在安全 FLASH 中; 加密/解密模块对存放在安全 FLASH 中的数据进行加密和解密; 一旦监测模块发现异常, 信号控制模块能提供相应的应对措施, 以防止危险的发生。

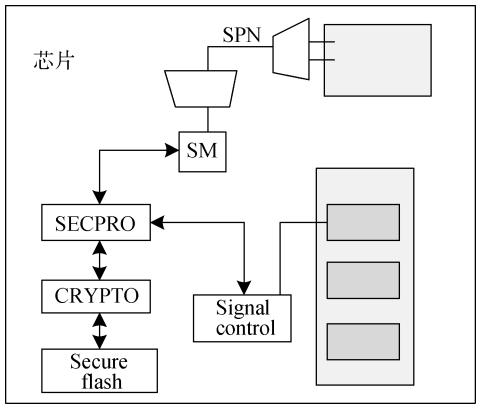

图 22 包含信号探测与监视模块的芯片架构

在文献[54] 中, 攻击者利用 JTAG[55,56]接口设计了硬件木马。图 23 只列出了危害之一, 其他几种攻击也同样是利用了 JTAG 调试接口的性质。相应地, 文献提出了对 JTAG 的保护, 分无保护、认证、加密、完整性四个层次。如表 4 所示。

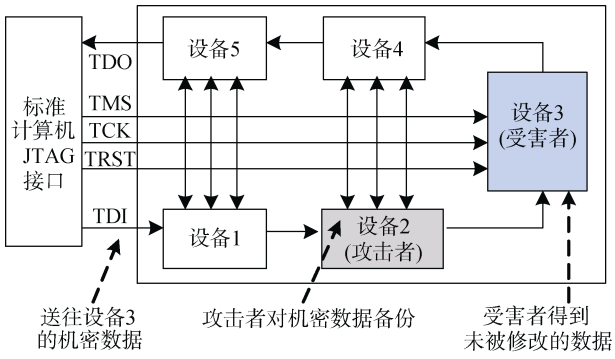

图 23 攻击者利用 JTAG 嗅探数据

表 4 JTAG 保护层次

保护等级	认证	机密性	完整性
0	no	no	no
1	yes	no	no
2	yes	yes	no
3	yes	yes	yes

文献[57]提出了一种检测 CPU 操作码的保护单元, 即 PPU(processor protection unit)。如图 24 所示。PPU 检查操作码的合法性、操作码执行时钟周期的

合法性、有限状态机的合法性及处理器内部信号的合法性。通过检验操作码的取值, 可以防止攻击者在处理器中植入恶意的指令。有限状态机的验证可以防止攻击者执行非法行为的企图。

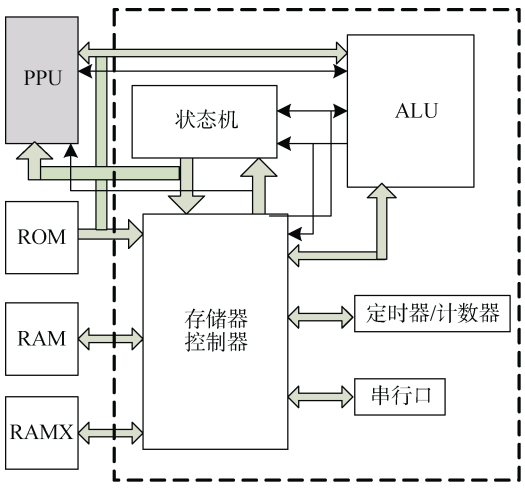

图 24 PPU 保护

文献[60]提出了一种芯片模块之间相互认证来传递数据的方法, 即一个模块想往另一个模块写数据或读数据, 需要得到该模块的认证, 只有通过认证, 才可以进行数据的读写操作。这样可以防止非法的硬件木马模块窃取合法的数据。

另外, 一些比较成熟的算法在硬件木马检测当中也有所应用, 如文献[63]中用到的机器学习算法, 文献[64]中用到的非线性分析算法等。文献[66]提出了一种新的算法来解决环形振荡器中硬件木马检测的问题。

7 总结与展望

半导体产业发展的一个趋势就是集成电路设计与制造过程的分离, 这造成了芯片的制造常常处于不受设计方控制的状态。目前我国的高端、高档集成电路主要依赖进口,对于那些应用于国防系统、政府机构、金融、交通等安全敏感领域的 IC 来说, 供应过程的不可控, 使得在使用这些芯片时面临极大的安全隐患: 攻击者可以在设计或制造过程中往芯片中植入硬件木马, 这些硬件木马可能在将来某个时候被攻击者触发, 也可能在某些情况下自行触发。硬件木马一旦被触发, 就可以将密钥等加密信息隐蔽地泄露给攻击者, 另外, 还可以执行破坏行为, 导致整个系统功能瘫痪。

本文系统概括分析了硬件木马的产生、分类、检测及安全措施的研究情况, 虽然硬件木马的研究开展时间较短, 研究深度也受到客观条件的限制, 但是目前相关研究人员已经在硬件木马的设计与检测中取得相当数量的成果, 其中部分成果具有重要的理论和工程参考价值 [67−83]。

从大量文献中, 不难发现目前存在的问题:

(1) 虽然目前硬件木马的设计主要出现在 VHDL 或 Verilog 源文件中, 但是这为研究硬件木马提供了一个思路和实验平台, 并且可以借鉴和参考相关研究成果。

(2) 在检测当中, 比如功耗检测或时延检测, 都需要有标准芯片做比较, 这也是一个假设比较强的条件。在实际检测中, 面对成千上万片芯片, 很难区分哪些是没有硬件木马的, 哪些是有硬件木马的; 另外, 关于逆向工程技术, 当芯片逻辑电路比较少的时候, 还能起到作用, 但是, 如今随着集成电路技术的发展, 生产工艺提升, 以及生产成本的下降, 导致芯片逻辑门数很大, 采用逆向工程进行解剖是不可行的, 而且成本也是让人难以接受的。正是由于上述困难的存在, 才给从事硬件木马研究的相关人员提出了比较新颖的研究课题, 同时也要求研究人员要找新的思路、新的方法和新的理论解决以上的问题。

(3) 关于对硬件木马的保护问题, 虽然目前只是提出了对芯片某些区域(比如内存、总线、处理器等)的针对性保护方案[84], 但是这些个案为最终提出整套研究方案打下了基础, 在此基础上有望提出芯片保护的完整性方案[85,86]。

因此, 硬件木马是一个较为崭新的研究领域, 近几年在国内外学者、研究人员的关注下, 逐渐成为研究的热点, 有许多方面值得深入研究下去。具体来说, 有如下几个比较重要的研究方向:

(1) 如果能够获得 VHDL 或 Verilog 源文件, 如何从源文件中分析出潜在的硬件木马威胁。这方面可以借助于软件病毒、软件木马分析的思路, 以及一些成熟的算法。例如通过模型检验, 即通过对有限状态空间进行穷举遍历来判断给定性质是否满足的一种形式化验证方法, 模型检验是一种自动化形式验证方法, 可以利用很多成熟的工具, 而且便于实用化; 具体步骤如下: 利用检验器对系统进行建模, 生成状态迁移系统; 对待验证的性质进行编码, 生成时态逻辑公式; 把状态迁移系统和时态逻辑公式送往模型检验器进行自动验证, 根据输出结果判断系统是否满足待检验的性质。就硬件木马防护来说, 在硬件源文件设计过程中, 可以通过添加专门的逻辑电路或者是通过更加细致的设计来避免硬件木马的注入。

(2) 对于购买或使用的第三方的 IP 核, 如何在确保其功能正确的同时, 也确保它的安全性, 或者说如何开展对第三方 IP 核安全性的评估[87] [89] [91] [92], 这都是非常重要并且有重大理论价值和工程价值的, 如 SoC 芯片的安全设计[88]。这方面的工作目前方兴未艾, 在具体的实现方法、算法上有很大的研究空间。由于硬件木马的设计要求非常高, 既不能在正常情况下影响芯片功能, 还要求在特定条件下准确触发, 因此硬件木马注入者必须对电路结构非常清楚, 利用这个特点, 可以通过电路欺骗技术, 在芯片电路中加入伪电路模块, 也可以通过增加电路状态迁移图的复杂度, 来欺骗迷惑攻击者, 使攻击者难以对电路结构有清楚认识, 以此来增加硬件木马插入难度。

(3) 对于国外引进的处理器, 如何找出潜在的威胁[90,93]; 另外, 对于自主研发的处理器, 如何提出处理器的安全防护机制, 或提出一种全新的安全处理器架构。再者, 如何确保处理器指令集的安全, 或者处理器指令集中没有预留的后门指令。为此, 可以采用电路模糊技术, 即用较难逆向分析的设计代替原始设计, 具体来说, 可以把一个功能独立完整的模块打散, 分布到多个组件中, 在布线过程中进行扰乱设计, 比如采用多金属层混合布局布线。

(4) 通信接口在计算机系统中是必不可少的, 如 UART、网络适配器、蓝牙以及一些调试接口。这些接口的安全问题为研究者提供了较好的课题。关于接口的安全防范, 可以采用可信性设计技术, 具体而言, 针对关键信号, 可以添加额外的监测电路, 提升潜在的硬件木马的激活概率, 增加对敏感信号的实时监控, 及时发现异常并发出报警。

(5) 如何提出一套完整的硬件木马检测机制、检测平台[89-90], 这也是一个很好的研究课题。如文献[94]提出的一种通过电磁分析来检测硬件木马的平台。该平台通过 RS232 通信端口给 FPGA 提供检测输入, 当 FPGA 运行时, 触发示波器记录磁场探头接收电磁信号, 并实时向计算机传输, 探头位置可以更改, 从而实现对 FPGA 各个区域电磁信号的采集。

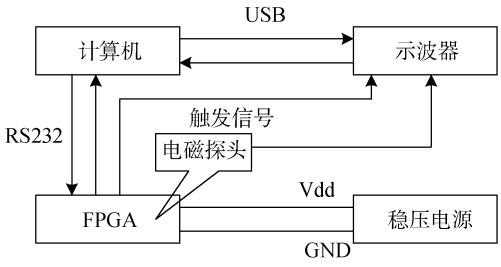

图 25 文献[94]实验平台原理图

总之, 硬件木马研究中虽然存在着一些问题, 但是这些问题为科研人员提供了有重要理论价值和工程价值的课题, 也揭示了硬件木马未来研究方向。

致 谢 本文的工作是在对硬件木马的调研基础上完成的。通过对 ACM、CNKI、IEEE 数据库有关硬件木马的文献资料研究分析, 总结出了其中的研究规律和热点。本文是在信息工程研究所第五研究室多位老师指导下完成的, 在此表示诚挚感谢。

参考文献

[1] McAfee Corp. New Paradigm Shift: Comprehensive Security beyond the Operating System. *White paper*, pp.1-10, 2012.

[2] B.Kauer, OSLO: Improving the security of Trusted Computing, *16th USENIX Security Symposium (Security'07)*, pp.229-237. 2007.

[3] A.Sanjaya, Hardware Assisted Security: Anticipating Digital Threat and Challenges. *FIRST TC 2012 IDF (TC'12)*, pp.1-10, 2012.

[4] J.Winter. Trusted Computing Building Blocks for Embedded Linux-based ARM Trust Zone Platforms, *Proceedings of the 3rd ACM workshop on Scalable trusted computing (STC'08)*, pp. 21-30. 2008.

[5] "Qubes".http://qubes-os.org.Sept.2014.

[6] "EMET Technology",.https://windowssecrets.com/top-story/protecting-pcs-from-the-next-zero- day-threat/. Sept.2013.

[7] IBM, Microsoft, etc.Workshop on Advancing Computer Architecture Research (ACAR-II): Laying a New Foundation for IT, *Computer Architecture for 2025 and beyond*, pp.1-20, 2010.

[8] Ruby B. Lee. Hardware-Enhanced Security. Keynote, *ACM CCS,* 2012.

[9] "Security issues", http://www.cctime.com/html/2014-12-10/201412101135225626.htm, Dec, 2014.

[10] "Global information Security issues", http://sec.chinabyte.com/162/13439662.shtml.May, 2015.

[11] P. Jiang, D.J. Li. "Information countermeasure". *Tsinghua University press, Chinese People's Public Security University press*, pp.8-11. 2007 (蒋平,李冬静. "信息对抗".*清华大学出版社*,*中国人民公安大学出版社* 2007:8-11.)

[12] E.M.Clarke, M.S.Khaira and X.D.Zhao, Word level model checking-avoiding the Pentium FDIV error. *Proceedings of 33rd Design Automation Conference (DAC'96)*, pp.645-648, 1996.

[13] http://www.gmw.cn/3_wlzk/yg/0002/19990520_1.html/,May,1999.

[14] S.Sergei, W.Christopher, Breakthrough Silicon Discovers Backdoor in Military Chip. *Proceedings of 14thInternational*

Workshop on Cryptographic Hardware and Embedded Systems (CHES'12), pp.23-40.2012.

[15] "Vulnerablity notes database", http://www.kb.cert.org/vuls/id/649219/, Sept, 2012.

[16] Y.Sun, Q.B Li, H.B Gao, and P Zhang, Towards Hardware Trojan: Problem Analysis and Trojan Simulation. *Proceedings of 2nd International Conference on Information Engineering and Computer Science(IECS'10)*, Wuhan, China,pp.1-4, 2010.

[17] R.Karri, J.Rajendran, K.Rosenfeld K, and M.Tehranipoor, "Trustworthy Hardware", *Identifying and Classifying Hardware Trojans*. vol.43, no.10, pp.39-46, Oct, 2010.

[18] S.Narasimhan, W.Yueh, and W.Xinmu, "Improving IC Security against Trojan Attacks through Integration of Security Monitors". *IEEE Design & Test of Computers*, vol.29, no.5, pp. 37-46. 2012.

[19] M. Tehranipoor, K. Farinaz, "A Survey of Hardware Trojan Taxonomy and Detection". *IEEE Design & Test of Computers*, vol.27, no.1, pp.10-25, 2010.

[20] H. Salmani, M. Teharanipoor and J. Plusquelllic, "A Novel Technique for Improving Hardware Trojan Detection and Reducing Trojan Activation Time". *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol.20, no.1, pp. 112-125, 2012.

[21] M.Tehranipoor, H.Salmani and X.H Zhang, "Trustworthy Hardware: Trojan Detection and Design-for-Trust Challenges". *Computer*, vol.44, no.7,pp.66-74,2011.

[22] E.Love, Y.Jin and Y.Makris, Enhancing security via provably trustworthy hardware Intellectual property. *Proceedings of 2011 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'11)*, pp.12-17, 2011.

[23] T.R.Reece, D.B.Limbrick and W.H.Robinson, Design Comparison to Identify Malicious Hardware in External Intellectual Property. *Proceedings of 2011 IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications (TSPCC'11)*, pp. 639-646,2011.

[24] B.Sunar, Rise of the hardware Trojans. Proceedings of 2011 IEEE *17th International On-Line Testing Symposium (OLTS'11)*, pp.138,2011.

[25] X.H Zhang, M.Tehranipoor, Case study: Detecting hardware Trojans in third-party digital IP cores. *Proceedings of 2011 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'11)*, pp.67-70,2011.

[26] J.Francq, F.Frick, "Overview of Hardware Trojans Detection and Prevention Methods", *Circuit Theory and Design (ECCTD'15)*,2015.

[27] S. Bhunia, M. Hsiao, M. Banga and S. Narasimhan, "Hardware Trojan Attacks: Threat Analysis and Countermeasures", *Proceedings of the IEEE*, vol. 102, no. 8, pp. 1229-1247, 2014.

[28] X.P. Niu, Q.B. Li, W.Wang and D. Zhang. "Research on the technology of hardware Trojan". *Journal of Information Engineering University*, vol.13, no.6, pp.742, 2012. (牛小鹏, 李清宝, 王炜, 张丹. 硬件木马技术研究综述[J] .*信息工程大学学报*, 2012, 13(6): 742.)

[29] H.F. Liu, H.W.Luo and L.W. Wang. "Overview of hardware Trojans." *Microelectronics*, vol.41,no.5,pp.109-113,2011.(刘华锋, 罗宏伟, 王力纬.硬件木马综述[J] .*微电子学*, 2011, 41(5):109-113.)

[30] L.R. Ren, "Under the new circumstances of the special weapons -- hardware Trojans". *China Institute of Electronic Science Journal of settings*, vol.6,no.2,pp.140 – 142,2011.(任立儒.新情况下的特殊武器——硬件木马[J] .*中国电子科学研究院学报*.2011, 6(2): 140－142.)

[31] H.F.Chen. "Technology and development trend of hardware Trojans detection circuit". *Journal of Zhejiang University (SCIENCE EDITION)*, vol.41,no.1,pp.49-51,2014.(陈华锋 等. 硬件木马电路检测技术及发展趋势[J] .*浙江大学学报(理工版)*, 2014, 41(1): 49－51.)

[32] D.Agrawal, S.Baktir and D. Karakoyunlu, Trojan Detection using IC Fingerprinting. *Proceedings of IEEE Symposium on Security and Privacy (SSP'07)*, pp.296-310, 2007.

[33] S.T.King, J.Tucek and A.Cozzie, Designing and implementing malicious hardware. *Proceedings of First USENIX Workshop on Large-Scale Exploits and Emergent Threats*(WLEET'08), 2008.

[34] H.Matthew, F.Murph and T.K.Samuel, Overcoming an Untrusted Computing Base: Detecting and Removing Malicious Hardware Automatically. *Proceedings of 2010 IEEE Symposium on Security and Privacy (SSP'10)*, pp.159-172, 2010.

[35] D.Jia and S.Smith, A hardware threat modeling concept for trustable integrated circuits. *Proceedings of IEEE Region 5 Technical Conference*, pp.354-357, 2007.

[36] D.Abhishek, M.Gokhan, Z.Joseph and A.Choudhary, Detecting/ Preventing Information Leakage on the Memory Bus due to Malicious Hardware. *Proceedings of 2010 Design, Automation & Test in Europe Conference & Exhibition(ATE'10)*, pp.861-866, 2010.

[37] Y.Jin, N.Kupp, Y.Makris, Experiences in Hardware Trojan Design and Implementation. *Proceedings of IEEE International Workshop on Hardware-Oriented Security and Trust*, pp.50-57, 2009.

[38] N.G.Tsoutsos and M.Maniatakos, "Fabrication Attacks: Zero-Overhead Malicious Modifications Enabling Modern Microprocessor Privilege Escalation Emerging Topics in Computing", *IEEE Transactions* vol. 2, no.1pp. 81 – 93,2014.

[39] J.Santos, M.Carlos and F. Yunsi, Designing and implementing a Malicious 8051 processor. *Proceedings of 2012 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, pp.63-66, 2012.

[40] C.Sturton, M.Hicks, D.A.Wagner, S.T.King, Defeating UCI: Building Stealthy and malicious Hardware. *Proceedings of 2011 IEEE Symposium on Security and Privacy (SSP'11)*, pp.64-77, 2011.

[41] Yier J., Makris Y. "Hardware Trojans in Wireless Cryptographic ICs ". *IEEE Design & Test of Computers*, vol.27, no.1, pp.26-35, 2010.

[42] J.Rajendran, E.Gavas and J.Jimenez, Towards a comprehensive and systematic classification of hardware Trojan. *Proceedings of 2010 IEEE international Symposium on Circuits and Systems (ISCAS'2010)*.pp.1871-1874, 2010.

[43] R.S.Chakraborty, N.Seetharam and B.Swarup, Hardware Trojan: Threats and emerging solutions. *Proceedings of the 14th IEEE International High Level Design Validation and Test Workshop*, pp.166-171, 2009.

[44] T.Kumaki, M.Yoshikawa and T.Fujino, Cipher-destroying and secret-key-emitting hardware Trojan against AES core Circuits and Systems (MWSCAS), *2013 IEEE 56th International Midwest Symposium,* pp. 408 – 411,2013.

[45] L.W Wang, H.W. Luo and R.H. Yao, "A hardware Trojan detection method based on bypass analysis". *Journal of South China University of Technology (NATURAL SCIENCE EDITION)*, vol.40, no.6,pp.9,2012.(王力纬, 罗宏伟, 姚若河. 基于旁路分析的硬件木马检测方法[J] .*华南理工大学学报(自然科学版)*, 2012, 40(6): 9.)

[46] P. Zhang, X.C.Wang and Q. Zhou, "Based on projection pursuit analysis of hardware Trojans detection". *Journal of communication (NATURAL SCIENCE EDITION)*, vol.34,no.4, pp. 122-125, 2013.(张鹏, 王新成, 周庆. .基于投影寻踪分析的芯片硬件木马检测[J] .*通信学报(自然科学版)*, 2013, 34(4): 122-125.)

[47] P.Kumar and R.Srinivasan, Detection of hardware Trojan in SEA using path delay. *Electrical, Electronics and Computer Science (SCEECS'14),* pp. 1 – 6, 2014.

[48] V.John, "hardware hacker". *Global Science*, pp.18-23,2010.

[49] H.Michael and T.Stephen, 2014. Memory encryption: A survey of existing techniques. *ACM Comput*. pp. 26, March, 2014.

[50] D. Lie, C. Thekkath, M. Mitchell, P. Lincoln, D. Boneh, J. Mitchell and M. Horowitz. Architectural Support for Copy and Tamper Resistant Software. *In Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating System (ASPLOSIX'2000)*, pp. 169-177, 2000.

[51] G.Suh, D.Clarke, B.Gassend, M.dijk and S.Devadas. AEGIS: Architecture for Tamper-Evident and Tamper-Resistant Processing. *In Proc. International Conference of Supercomputing (ICS'03)*, pp.160-171, 2003.

[52] L. Kim and J. D. Villasenor, "A System-On-Chip Bus Architecture for Thwarting Integrated Circuit Trojan Horses", *IEEE Transactions on VLSI*, vol. 19, no. 10, pp. 1921-1926, 2011.

[53] M.Abramovici and P.Bradley, Integrated circuit security-new threats and solutions. *Proceedings of the 5th Annual Cyber Security and Information Intelligence Research Workshop*. 2009.

[54] C.X. Wang, P.H. Jiang and M.Y.Yu, "Chip level Trojan detection technology research". *Semiconductor technology*, vol.37,no.5, pp.341-345, 2012. (王晨旭, 姜佩贺, 喻明艳. 芯片级木马检测技术研究综述[J] .*半导体技术*, 2012, 37(5): 341-345.)

[55] K. Rosenfeld and R. Karri, "Attacks and Defenses for JTAG," *Des. Test Comput.IEEE*, vol. 27, no. 1, pp. 36–47, Feb. 2010.

[56] Test Access Port and Boundary-Scan Architecture. IEEE Std 1149.1-2001,

[57] In-System Configuration of Programmable Devices. IEEE Std 1532-2002,

[58] D.Dubeuf and R. Karry, "Run-time detection of hardware Trojans: The processor protection unit," *IEEE Eur. Test Symp*, pp. 156–161, May 2013.

[59] B. Zhou, W. Zhang and S. Thambipilai, "Cost-efficient Acceleration of Hardware Torjan Detection Through Fan-out Cone Analysis and Weighted Random Pattern Technique". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. vol.35, no. 5. pp.792-805. 2016

[60] X.Wei, Y.Diao and Y.L.Wu, To Detect, locate and mask hardware Trojans in digital circuits by reverse engineering and functional ECO. *2016 21st Asia and South Pacific Design Automation Conference(ASP-DAC'16)*, pp.623-630. 2016.

[61] C. Liu, C.Patrick and C.G.Yang, A mutual auditing framework to protect IoT against hardware Trojans. *2016 21st Asia and South Pacific Design Automation Conference(ASP-DAC'16)*, pp.69-74. 2016.

[62] C.X.Bao, D.Forte and A.Srivastava, "On Reverse Engineering-Based Hardware Trojan Detection", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*.vol.35. no.1. pp.49-57. 2016.

[63] I.Voyiatzis, C. Sgouropoulou and C. Estathiou, Detecting untestable hardware Trojan with non-intrusive concurrent on line testing. *2015 10th international Conference on Design & Technology of Integrated Systems in Nanoscale Era(DTIS'15)*. pp.1-2. 2015.

[64] T.Iwase, Y.Nozaki, M.Yoshikawa and T.Kumaki, Detection

technique for hardware Trojans using machine learning in frequency domain. *2015 IEEE 4th Global Conference on Consumer Electronics(GCCE'15)*, pp.185-186. 2015.

[65] E.Zhou, S.Q. Li, Z.X. Zhao and L. Ni, Nonlinear analysis for hardware Trojan detection. *2015 IEEE international Conference on Signal Processing, Communications and Computing (ICSPCC'15)*, pp.1-4. 2015.

[66] B. Hou, C.H. He, L.W. Wang, Y.F. En and S.F. Xie, Hardware Trojan detection via current measurement: A method immune to process variation effects. *2014 international Conference on Reliability, Maintainability and Safety (ICRMS'14)*, pp.1039-1042. 2014.

[67] K.Qu, L.J Wu and X.G Zhang, A Novel Detection Algorithm for Ring Oscillator Network Based Hardware Torjan Detection with Tactful FPGA Implementation. *2015 11th international Conference on Computational intelligence and Security (CIS'15)*, pp.299-302. 2015.

[68] S.T.King, J.Tucek and A.Cozzie, "Designing and implementing malicious hardware". http://www.usenix. org/event/leet08/ tech/full paper/king/ king.pdf, 2008.

[69] R.B.Mark, D.H.Bradley, N.Tristan and P.Safer, Security architecture using fragmented execution and replication for protection against trojaned hardware. *Proceedings of Design, Automation and Test in Europe Conference*, pp.1000-1005, 2012.

[70] M.Tehranipoor and C.Wang, "Introduction to Hardware Security and Trust". *New York: Springer-Verlag*, 2011.

[71] S.Berk, Rise of the hardware Trojans. *Proceedings of IEEE International On-Line Testing Symposium (OLTS'11)*, pp.138, 2011.

[72] F.Wolff and C.A.Papachristou, Towards Trojan Free Trusted ICs: Problem Analysis and Detection Scheme. *Proceedings of Design, Automation and Test in Europe Conference(DAT'08)*, pp.1362-1365,2008.

[73] H.B.Gao, The Concept and Taxonomy of Hardware Trojan Horse. *Proceedings of 2010 3rd International Conference on Computer and Electrical Engineering(ICEEE'10)*, pp.12-32, 2010.

[74] H.Salmani and M.Tehranipoor, "Layout-Aware Switching Activity Localization to Enhance Hardware Trojan Detection". *IEEE Transactions on Information Forensics and Security*, vol.7, no.1, pp.76-87, 2012.

[75] S.Wei, K.Li, K.Farinaze and M. Potkonjak, Hardware Trojan horse benchmark via optimal creation and placement of malicious circuitry. *Proceedings of 2012 49th ACM/EDAC/ IEEE Design Automation Conference (ADAC'12)*, pp.90-95, 2012.

[76] R.Chakraborty and S.Bhunia, "Security Against Hardware Trojan Attacks Using Key-based Design Obfuscation". *Journal of Electronic Testing and Test Application*, vol.27, no.6, pp.767-785, 2011.

[77] S.Narasimhan, X.M Wang and D.D.Du, TeSR: A robust Temporal Self-Referencing approach for Hardware Trojan detection. *Proceedings of 2011 IEEE International Symposium on Hardware-Oriented Security and Trust (HOST'11)*, pp.71-74, 2011.

[78] W.C.Li, Z.Wasson and S.Seshia, Reverse engineering circuits using behavioral pattern mining. *Proceedings of 2012 IEEE International Symposium on Hardwware-Oriented Security and Trust(HOST'12)* ,pp.83-88,2012.

[79] G.Shrestha and M.S.Hsiao, Ensuring trust of third-party hardware design with constrained sequential equivalence checking. *Proceedings of 2012 IEEE Conference on Technologies for Homeland Security*, pp.7-12,2012.

[80] X.T.Ngo, Z.Najm, S.Guilley, S.Bhasinand J.L.Danger, Method Taking into Account Process Dispersion to Detect Hardware Trojan Horse by Side-Channel. *Proceedings of Security Proofs for Embedded Systems− PROOFS*, 2014.

[81] S. Dupuis, G. D. Natale, M. L. Flottes and B. Rouzeyre, Identification of Hardware Trojans Triggering Signals. *Workshop on Trustworthy Manufacturing and Utilization of Secure Devices − TRUDEVICE*, 2013.

[82] S.Dupuis, P.S.Ba, G.D.Natale, M.L. lottes and B.Rouzeyre, A Novel Hardware Logic Encryption Technique for thwarting Illegal Overproduction and Hardware Trojans. *Proceedings of IEEE International On-Line Testing Symposium − IOLTS*, pp.49–54, 2014.

[83] H.Salmani, M.Tehranipoor and J.Plusquellic, A Novel Technique for Improving Hardware Trojan Detection and Reducing Trojan Activation Time. *Proceedings of IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol.20, pp. 112–125, 2012.

[84] X.T.Ngo, S.Guilley, S.Bhasin, J.L.Danger and Z.Najm, Encodingthe State of Integrated Circuits: A Proactive and Reactive Protection against Hardware Trojans Horses. *Proceedings of ACM Workshop on Embedded Systems Security − WESS*, 2014.

[85] D.Vivek, B.Kerry and Y.Takefumi, Designing secure systems: Manufacturing, circuits and architectures. *International Solid-State Circuits Conference(ISSCC'16)*, pp.492-494, 2016.

[86] M.Tehranipoor, New Directions in Hardware Security. *2016 29th International Conference on VLSI Design and 2016 15th international Conference on Embedded Systems(VLSID'16)*, pp.50-52. 2016

[87] H. Li, Q. Liu. J.L. Zhang and Y.Q. Lyu. A Survey of Hardware Trojan Detecion, Diagnosis and Prevention. *2015 14th Inter-*

national Conference on Computer-Aided Design and Computer Graphics(CAD/Graphics'15), pp.173-180. 2015.

[88] J.Rajendran, O.Sinanoglu and R.Karri, "Building Trustworthy Systems Using Untrusted Components: A High-Level Synthesis Approach". *IEEE transactions on Very Large Scale Integration(VLSI) Systems*.vol.pp. no.99. pp.1-14. 2016.

[89] G. Bloom, B. Narahari and R. Simha, FPGA SoC architecture and runtime to prevent hardware Trojans from Leaking secres. *2015 IEEE international Symposium on Hardware Oriented Security and Turst(HOST'15)*, pp.48-51. 2015.

[90] Y.Zheng, S.Yang and S.Bhunia, "SeMIA: Self-Similarity-Based IC Integrity Analysis". *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. vol.35. no.1. pp.37-48.2016.

[91] M.Goll and S.Gueron, "Randomness Tests in Hostile Environments". *IEEE Transactions on Dependable and Secure Computing*. vol.pp. no.99. pp.1. 2016.

[92] J. Rajendran, A.M. Dhandayuthapany, V. Vedula and R. Karri. Formal Security Verification of Third Party Intellectual Property Cores for Information Leakage. *2016 19th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID'16)*, pp.547-552. 2016.

[93] Y.Xie, C.X.Bao and C.Serafy, "Security and Vulnerability Implications of 3D ICs". *IEEE Transactions on Multi-Scale Computing Systems*. vol.pp. no.99. pp.1. 2016.

[94] T.Meade, S.J.Zhang and Y.Jin, Netlist reverse engineering for high-level functionality reconstruction. *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC'16)*. pp.655-660. 2016.

[95] P.Zhang, X.C.Wang and Q.Zhou, "A Hardware Trojans Detection Method Based on Electromagnetic Emission Analysis" *Microelectronics and computer*, vol.30, no.12, pp.15-17, 2013. (张鹏, 王新成, 周庆.一种基于电磁分析的硬件木马旁路检测方法[J]. *微电子学与计算机*,2013,30(12):15-17.)

[96] T.D. Moore, J.L. Jarvi. Failure analysis and stress simulation in small multichip BGAs [J]. *IEEE* Transactions on Advanced Packaging, vol.24, no.5, pp.216-223, 2001.

[97] R.S. Chakraborty, F. Wolff. MERO: A Statistical Approach for Hardware Trojan Detection. *In: Proceedings of Workshop on Cryptographic Hardware and Embedded Systems(CHES'09)*, pp. 396-410,2009.

[98] L. Min, A. Davoodi, and M. Tehranipoor. A sensor-assisted self-authentication framework for hardware Trojan detection. *In: Proceedings of Design, Automation & Test in Europe Conference & Exhibition(DATE'12)*,pp. 1331-1336, 2012.

[99] X. Wang, M. Tehranipoor and J. Plusquellic. Detecting Malicious Inclusions in Secure Hardware: Challenges and Solutions. *IEEE International Workshop on Hardware-Oriented Security and Trust(HOST'08)*, pp.15-19,2008.

[100] Z.X. Zheng, L. Han, Y. Li and X.C. Zou. "The Implementation and Analysis of a Trojan Circuit in IC Chips". *Microelectronics and computer*, vol.29, no.10, pp.78-80, 2012.(郑朝霞, 韩玲, 李阳,邹雪城.一种木马电路的实现与特征分析[J]. *微电子学与计算机*,2012,29(10):78-80.)

赵剑锋 于 2006 年在北京理工大学通信与信息系统专业获得硕士学位。现在中国科学院信息工程研究所计算机系统结构专业攻读博士学位。研究领域为计算机系统安全。研究兴趣包括：架构安全。Email: zhaojianfeng@iie.ac.cn

史岗 于 2004 年在中国科学院计算技术研究所获得博士学位。现任中国科学院信息工程研究所第五研究室高级工程师。研究领域为计算机系统安全。研究兴趣包括：嵌入式系统、信息安全。Email: shigang@iie.ac.cn