基于温度特征分析的硬件木马检测方法

钟晶鑫* 王建业 阚保强
(空军工程大学防空反导学院 西安 710051)

摘 要：硬件木马是一种在特定条件下使集成电路失效或泄露机密信息等的恶意电路，给现代信息系统带来了严重的安全隐患。该文基于硬件木马在芯片工作之初造成的温度响应特征，提出一种利用芯片温度变化特性并进行比对的硬件木马检测方法。该方法采用环形振荡器作为片内温度特征测量传感器，提取温度变化特征信息，并采用曲线拟合评价指标来评估硬件木马对温度变化特征的影响，通过比对无木马芯片温度响应特征从而完成木马检测。通过对 10 个不同芯片的检测，结果表明该方法能够对面积消耗 32 个逻辑单元硬件木马的检测率达到 100%，对 16 个逻辑单元检测概率也能达到 90%；同时检测结果表明该方法完成硬件木马检测后，能够对硬件木马的植入位置进行粗定位。

关键词：硬件木马；温度变化；环形振荡器；检测定位
中图分类号： TN406 **文献标识码**： A **文章编号**：1009-5896(2018)03-0743-07
DOI: 10.11999/JEIT170443

Hardware Trojan Detection Through Temperature Characteristics Analysis

ZHONG Jingxin WANG Jianye KAN Baoqiang
(*Air and Missile Defenses College, Air Force Engineering University, Xi'an* 710051, *China*)

Abstract: Hardware Trojan is the malicious circuit modification which can disable the Integrated Circuit (IC) or leak confidential information covertly to the adversary, and brings potential safety hazard for ICs. In this paper, a new approach for hardware Trojan detection based on compare the temperature variation characteristics when IC starts working. Ring Oscillator (RO) is used as a detector to obtain the information about IC's temperature variation characteristics. In order to describe temperature variation characteristics accuracy, a parameter about the D-value of RO's oscillation cycle counts is presented, and parameters about the quality of the fitting curve are used to estimate the hardware Trojan's effect on IC's temperature characteristics. Results from ten chips show that the proposed approach is effective towards increasing successful detection ratio and can achieve better Trojan detection probability 100% on average over conventional patterns for Trojan which is 32 logic elements, and for Trojan which is 16 logic elements can also achieve Trojan detection probability 90%, besides the proposed approach locating the Trojan's insertion place roughly.

Key words: Hardware Trojan; Temperature variation; Ring Oscillator (RO); Detection and location

1 引言

硬件木马是指集成电路(Integrated Circuit, IC)在设计或制造过程中对原始电路进行的恶意修改或者植入的恶意电路，主要目的是在某些特定条件下导致集成电路失效、泄露机密信息或者缩短集成电路使用寿命等[1]。由于集成电路芯片生产工艺的精密性和复杂性，要求其制造过程必须在具备特殊工艺和环境条件的专业工厂内完成，单独建设一条芯片生产线需要复杂的技术和昂贵的费用[2]，大部分都需要第三方工厂代加工流片，因此无法保证生产过程不被植入硬件木马。我国目前80%的芯片都交由海外工厂代流片，因此防止芯片收到硬件木马的监测和攻击、保证集成电路芯片的可靠性和安全性，成为了保证国防信息安全和人民正常生活的重要课题之一。目前硬件木马的检测方法主要有4类：基于逆向工程、基于侧信道分析、逻辑测试和可测性设计。

基于逆向工程的版图比对技术是一种将芯片进行剖片、拍照，再与芯片设计版图比较区别的技术，该方法对芯片具有很强的破坏性[3]，并且检测所需的资金投入大、时间消耗长[4]。

基于侧信道分析的硬件木马检测方法是目前硬件木马检测领域最有效的手段之一，其原理是通过检测芯片的工作时候反应出的异常侧信道特征信息

收稿日期：2017-05-11；改回日期：2017-09-18；网络出版：2017-11-08
*通信作者：钟晶鑫 zhjx_08@163.com

(包括：功耗[5]、延时[6]、电压、电流[7]、热[8]、电磁[9,10]等)来检测硬件木马。由于硬件木马产生的异常信息较弱，因此很容易淹没在芯片工作时的各类噪声中，因此基于侧信道分析的硬件木马检测方法容易受噪声影响，很难提高检测精度。侧信道分析也可以作为一种攻击方法，分析密码芯片侧信道信息(功耗特征等)，获取密钥等重要信息[11]。

逻辑测试的方法是通过向芯片输入遍历测试矢量来激发潜藏的木马。随着集成电路规模增大，遍历所有测试矢量并不可行，因此该领域研究都集中在了如何找到有效测试矢量上[12−14]。但对于规模不断增大的集成电路和多变的木马激活方式，利用有效的测试向量仍然需要消耗大量的测试时间才有可能激活木马。

可信性设计是贯穿硬件木马防御、检测和定位的一个热门发展方向，通过在芯片内添加额外的测试电路来辅助对抗硬件木马的入侵，该设计主要完成的功能有增加木马激活概率[15,16]、辅助各类硬件木马检测[17−19]以及防止木马植入[20,21]。可信性设计会增加芯片的额外面积消耗，同时会对芯片正常工作造成一定的影响。

硬件木马的植入会反映出额外的热信息，即使在非激活状态下硬件木马也会产生热量，从而影响周边电路的温度变化趋势。本文结合侧信道分析检测法和可信性设计检测法优缺点，在芯片内部构建高灵敏度的环形振荡器网络来测量芯片内部温度变化信息，降低了环境温度噪声影响，提高了检测精度，同时利用侧信道分析方法分析有木马和无木马芯片的温度趋势变化差异，并提出了一种新描述温度变化趋势的参数，利用曲线拟合评价指标来评估硬件木马对温度特征的影响，形成了完整的检测和评估体系。

2 基于温度特征分析的硬件木马检测理论

2.1 芯片温度变化信息提取

环形振荡器(Ring Oscillator, RO)是对芯片内部温度变化比较敏感一种集成电路结构，广泛应用于芯片内部温度检测技术。RO在集成电路芯片中可以采用两种结构来设计：非门结构和与非门结构。图1所示为两种结构下的N阶RO(N为大于1的奇数)，其中图1(b)结构的RO的其中一个输入端接入电源，从而使该结构RO对于芯片内部的变化更加灵敏[19]。因此本文采用与非门结构RO作为检测硬件木马的传感器。

当RO作为芯片内部传感器工作时，通常设置一个计数器在设定时间t_{f}内测定振荡器周期计数值C，可转换出此时RO的振荡频率为$f=C/t_{\mathrm{f}}$。由于$f\propto C$，且RO的振荡频率与芯片内部温度T之间几乎呈线性关系，可认为$f\propto 1/T$，因此$C\propto 1/T$。C的变化可以体现芯片内温度的变化，但C的值一般远远大于T，若将C转换为T来进行硬件木马检测，必将丢失许多特征信息，增加误差以及降低检测精度，所以在本文中用C代替温度T作为描述芯片内部温度变化的观测参数。

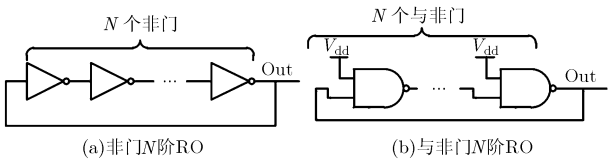

图1 两种RO结构

2.2 基于温度特征分析的硬件木马检测原理

从芯片通电开始工作，芯片内部的温度会受热扩散影响而逐渐均衡，从而会掩盖硬件木马产生的差异信息，因此硬件木马检测最有效的时间是到达热平衡前这段时间。图2是3阶与非门RO构成的RO在CycloneIII EP3C16F484 FPGA芯片上的振荡周期计数值C_j随工作时间的变化图(C_j是自芯片开始测试的第j个测试值)。实验在恒温箱中进行，实验环境温度$T_{\mathrm{temp}}=15$℃，振荡周期计数时间$t_{\mathrm{f}}=16.384$ μs，测试时间t_{test}为10 min。其中有木马芯片表示在RO附近植入了采用200 MHz时钟驱动的32位移位寄存器代替木马。从图2中可见，RO振荡周期计数C随时间增加而减少并且趋于稳定，且木马的植入使C_j在相同时刻的下降更快，C在t_{test}内的变化量从117增加到了138，同时测试过程中后5 min测试时间内C的变化量为总变化量的15%，为了提高检测效率，在后期的验证实验中取t_{test}=5 min。

在实际情况中由于工艺偏差的存在，同型号的不同芯片中RO的振荡周期计数值C_j差异也很大。图3是在图2相同测试条件下，10块无木马FPGA中RO计数值C_j的值域分布。从图3中可以看出，C_j并不能直接区分是由工艺偏差引起的还是由硬件木马植入引起的差异。为了有效检测木马，将测试时间t_{test}内的计数值C_j与所有计数最小值$C_{\min}$之差 D_j($D_j=C_j-C_{\min}$)作为检验硬件木马的参数，如图3所示，10块的无木马的FPGA芯片的$D_{\max}$(D_j的最大值)很接近110，差值D_j可以在有效去除芯片间计数差异的同时，完整地保留了计数值C_j的变化趋势。而图2中看出木马的植入使$D_{\max}$值增大到了138。因此本文以D_j值可以作为检测硬件木马的有效参数，通过对比D_j值的变化趋势来确认芯片内是否存在硬件木马。

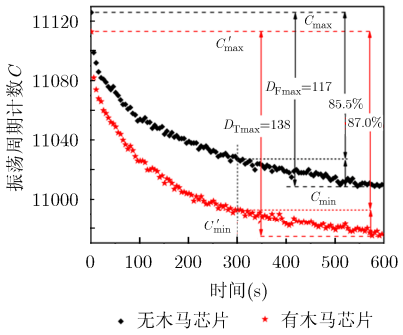

图 2　RO振荡周期计数随工作时间的变化关系

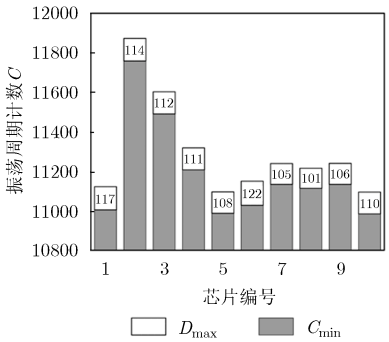

图 3　10块FPGA芯片中RO的振荡周期计数值

2.3 判定算法

为了量化和扩大硬件木马对芯片温度变化趋势的影响，本文引入了曲线拟合时所用的两个评价标准：残差平方和(SSE)与拟合系数(R_square)，该标准能够体现拟合曲线与目标曲线之间的相似度。SSE 越大，表明拟合数据与目标数据之间差异大，R_square 越小，表明拟合度越低。SSE 和 R_square 的定义为

$$\mathrm{SSE}=\sum_{j=1}^{n}w_j\left(y_j-\hat{y}_j\right)^2 \tag{1}$$

$$\mathrm{R\_square}=\frac{\sum_{j=1}^{n}w_j\left(\hat{y}_j-\bar{y}\right)^2}{\sum_{j=1}^{n}w_j\left(y_j-\bar{y}\right)^2},\ \bar{y}=\frac{1}{n}\sum_{j=1}^{n}y_j \tag{2}$$

其中，w_j 是权重参数，本文中取1；y_j 是目标数据，$\hat{y}_j$ 是拟合数据，$\bar{y}$ 是目标数据的平均值。在硬件木马检测时，首先从 M 个无木马的芯片中分别提取 $\boldsymbol{D}_i$ 值：

$$\left.\begin{aligned}\boldsymbol{D}_1&=\left[\begin{array}{cccc}D_{\mathrm{F}11}&D_{\mathrm{F}12}&\cdots&D_{\mathrm{F}1N}\end{array}\right]\\ \boldsymbol{D}_2&=\left[\begin{array}{cccc}D_{\mathrm{F}21}&D_{\mathrm{F}22}&\cdots&D_{\mathrm{F}2N}\end{array}\right]\\ &\vdots\quad\ \vdots\quad\ \vdots\quad\ \ddots\quad\ \vdots\\ \boldsymbol{D}_M&=\left[\begin{array}{cccc}D_{\mathrm{F}M1}&D_{\mathrm{F}M2}&\cdots&D_{\mathrm{F}MN}\end{array}\right]\end{aligned}\right\} \tag{3}$$

本文公式中 i 代表芯片编号，j 代表测试数据编号。其中 N 为 $\boldsymbol{D}_i$ 的维度，即与时间相关的数据长度。再计算出 $\boldsymbol{D}_i$ 在每一时刻的平均值，以此构成目标数据：

$$\begin{aligned}\overline{\boldsymbol{D}}_{\mathrm{F}}&=\left[\begin{array}{cccc}\overline{D}_{\mathrm{F}1}&\overline{D}_{\mathrm{F}2}&\cdots&\overline{D}_{\mathrm{F}N}\end{array}\right]\\&=\frac{1}{M}\left[\begin{array}{cccc}\sum_{i=1}^{M}D_{\mathrm{F}i1}&\sum_{i=1}^{M}D_{\mathrm{F}i2}&\cdots&\sum_{i=1}^{M}D_{\mathrm{F}iN}\end{array}\right]\end{aligned} \tag{4}$$

目标数据形成后，每一个待测芯片 i 的 SSE_i 和 $\mathrm{R\_square}_i$ 值可以从下式获得

$$\mathrm{SSE}_i=\sum_{j=1}^{N}\left(\overline{D}_{\mathrm{F}j}-d_{ij}\right)^2 \tag{5}$$

$$\mathrm{R\_square}_i=\frac{\sum_{i=1}^{N}\left(d_{ij}-\overline{D}\right)^2}{\sum_{i=1}^{N}\left(\overline{D}_{\mathrm{F}j}-\overline{D}\right)^2},\ \overline{D}=\frac{1}{N}\sum_{j=1}^{N}\overline{D}_{\mathrm{F}j} \tag{6}$$

其中，d_{ij} 是第 i 个测试芯片的第 j 个时刻的测试值($j\in[1,N]$)，$\overline{D}$ 是无木马芯片在测试时间内的均值。SSE_i 和 $\mathrm{R\_square}_i$ 获取后，可以直观地看出测试数据与目标数据之间的拟合度，当 SSE_i 和 $\mathrm{R\_square}_i$ 超过一定的阈值之后，便可认为该芯片被植入了硬件木马。后文中以值域范围较大的 SSE_i 的大小作为检测硬件木马的主要判决依据，$\mathrm{R\_square}_i$ 作为变化趋势的量化体现，辅助分析硬件木马植入带来的影响，而判决阈值则由无木马芯片决定，在考虑一定虚警率和漏警率情况下，通过抽取一定的无木马芯片计算出 SSE_i 对于目标数据之间的容差来确定阈值。

3　实验及算法设置

3.1 FPGA 设置

本文中选取3阶RO构成环形振荡网络(Ring Oscillator Network, RON)进行检测。如图4所示，16个RO均匀分散布局在EP3C16F484 FPGA之中，使能端口(EN)可控制RO只工作在测试环节，不影响芯片正常工作。计数器每秒对RO振荡周期在固定时间($t_{\mathrm{f}}=16.384\ \mu\mathrm{s}$)内的个数 C 进行4次计数。有限状态机(FSM)作为控制的核心，控制RO的选择、周期计数、测试时间($t_{\mathrm{test}}=5\ \mathrm{min}$)和数据传输。

布局时将设计好的DES加密模块布局到FPGA中，以模拟芯片真实工作情况。如图5所示，A和B是选定用来放置硬件木马位置。测试用3种不同面积的移位寄存器代替硬件木马发热，T1, T2, T3分别代表64位、32位和16位移位寄存器，移位寄存器的驱动时钟为200 MHz。实验过程中，分别将T1, T2, T3放置于A或B位置，即每次芯片中只出现1个木马。表1是FPGA中各个模块所消耗的芯片资源比例。

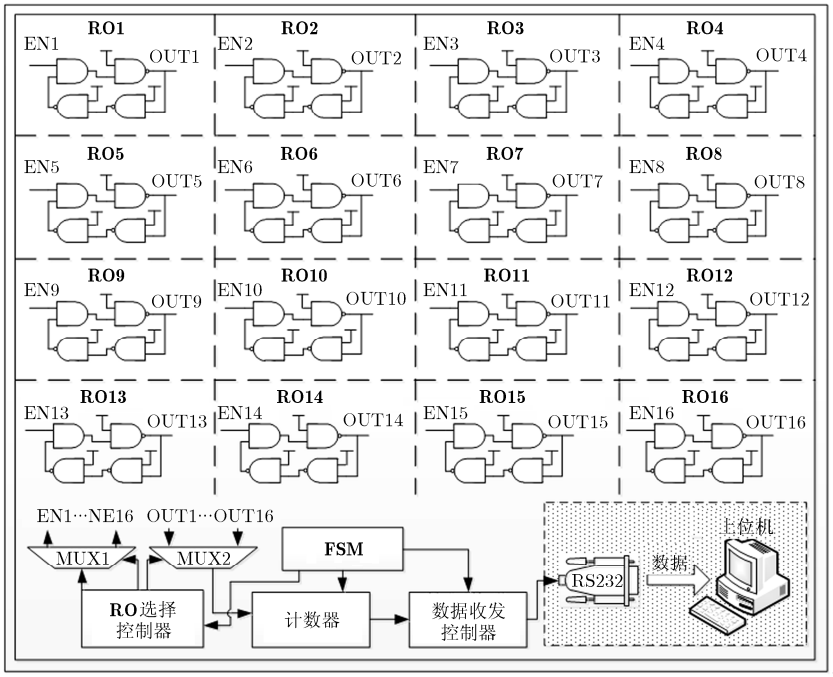

图 4 FPGA中RON结构

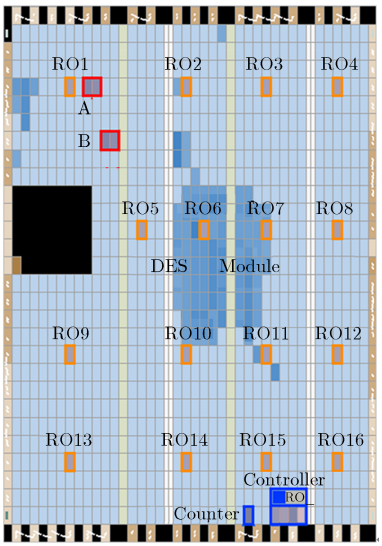

图 5 EP3C16F484 FPGA芯片布局

表 1 各模块逻辑资源消耗比例

模块	逻辑单元	总单元数	百分比(%)
RON	233	15408	1.51
DES	1088	15408	7.06
T1	64	15408	0.42
T2	32	15408	0.21
T3	16	15408	0.10

3.2 测试环境设置

测试中为了降低外部噪声和减小环境温度对测试结果的影响，测试过程在恒温箱内进行。恒温箱恒定在15℃，在每次测试开始前，先将芯片放置于恒温箱内，确保每个芯片每次测试的起始温度都为15℃。

3.3 算法设置

3.3.1 数据预处理 由于三阶 RO 测试数据受噪声影响大，数据上下浮动较强，在测试过程中，每秒测试上传 4 次 16 个 RO 的振荡周期计数值，将每秒内的 4 次测试均值作为当前测试数据来降低噪声影响。在测试时间 5 min 内将会最终提供 $N=300$ 个数据用于硬件木马检测。

3.3.2 检测算法 算法设置包括两个方面内容：目标数据建立和测试数据验证，如表 2 所示。本文算法应用时测试芯片数量 $M=10$，数据长度 $N=300$。

4 实验结果与分析

4.1 硬件木马检测精度分析

图 6 是 T1, T2 和 T3 分别植入图 5 中 FPGA 布局图 A 位置时，从 RO1 中获取的 $D_{\max}$ 值分布，由于植入位置离 RO1 较近，首先考虑分析 RO1 中数据以验证方法有效性。从图 6 中可见，无木马芯片的 $D_{\max}$ 值除 6 号芯片外都在 100 以下，若取 $D_{\max}$ 值的判决阈值为 100，$D_{\max}$ 值已经能将几乎所有 T1 和 T2 植入的芯片区分开(除 T2 植入的 4 号芯片)，但 T3 植入的芯片中仍有高达 50%的芯片与无木马芯片混杂难以区分。

表 3 是对图 7 中检测芯片取 RO1 的数据利用文

表 2 检测算法设置

步骤 1　目标数据建立

(1)获取 M 个无木马芯片的 $\boldsymbol{D}_{\mathrm{F}}$ 值，且对每个 RO

(2) for $(j=1, j<N, j++)\{$　　//选择第 j 个数据

(3)　for $(i=1, j<M, i++)\{$　　//选择第 i 个芯片

(4)　　$D_{\mathrm{sum\_}j}=D_{\mathrm{sum\_}j}+D_{\mathrm{F}ij};\}$　　//第 j 个数据总和

(5) $\overline{D}_{\mathrm{F}j}=D_{\mathrm{sum\_}j}/M;\}$　　//第 j 个数据均值

(6) $\overline{\boldsymbol{D}}_{\mathrm{F}}=\left[\overline{D}_{\mathrm{F1}}\ \overline{D}_{\mathrm{F2}}\ \cdots\ \overline{D}_{\mathrm{F}N}\right];$　　//构成目标数据

(7)再获取 M 个无木马芯片的 $\boldsymbol{D}_{\mathrm{F}}^{'}$ 值，且对每个 RO

(8) for $(i=1, i<M, i++)\{$　//计算无木马芯片 SSE 等值

(9)　计算 $\mathrm{SSE}_{\mathrm{F}i}$ 和 $\mathrm{R\_square}_{\mathrm{F}i}$;$\}$

(10)依据 $\mathrm{SSE}_{\mathrm{F}i}$，选取 SSE 的判决门限 $\mathrm{SSE}_{\mathrm{T}}$。

步骤 2　测试数据验证

(1)获取 M 个待测芯片的 $\boldsymbol{D}$ 值，且对每个 RO

(2) for $(i=1, j<M, i++)\{$　//计算待测芯片 SSE 等值

(3)　计算 SSE_i 和 $\mathrm{R\_square}_i$;

(4) if $(\mathrm{SSE}_i>\mathrm{SSE}_{\mathrm{T}})\ T_{\mathrm{flag\_}i}=1;$　　//判定植入木马

(5) else $T_{\mathrm{flag\_}i}=0;$ end if;$\}$　　//判定未植入木马

中算法求解的SSE和R_square值分布表。为了更好地选择判决阈值，求解无木马芯片的SSE和R_square值所用的数据，是在构成目标数据之外单独测量的一组无木马芯片的数据。从表 3 数据中可以看出，无木马芯片的SSE值大多低于1000，且无木马芯片直接的趋势拟合度都接近 1。而对于 T1, T2 和 T3 植入的芯片，SSE和R_square相对无木马芯片的差异很大，拟合度大部分低于 0.95，且木马面积越大，SSE越大，拟合度R_square越小。在选取判决门限时，需要考虑一定的容差，降低误警概率，同时也要降低漏警概率。如表 3 中选取SSE判决门限为 1000, SSE 大于 1000 的芯片被认为植入木马，那么对 10 号芯片将造成误判，对 T3 植入的 2 号芯片漏判；若选取SSE判决门限为 1500，扩大容差后对无木马芯片判定成功，对 T3 植入的 2 号芯片漏判；若选择SSE判决门限为 900，那么会造成 8 号芯片和 10 号芯片误判，但对 T1, T2 和 T3 植入的所有芯片都能完全检测，相对于 10 个芯片来说，就是以增加 10%误警概率来降低 10%漏警概率达到硬件木马 100%检测。检测的目的就是为了排除隐患，所以判决门限在误警概率容忍范围内要尽可能地小。

4.2 硬件木马检测定位

硬件木马的植入位置对检测结果也有很大的影响。表 4 是选取 T1 和 T3 植入位置 B 与植入位置 A 时 RO1 的检测结果对比。从表 4 中可以看出，T1 和 T3 植入位置 B 得到的SSE值大部分都比植入位置 A 的SSE值小，也就是说硬件木马植入位置的离对应检测 RO 的距离越远，RO 检测的效果越低。RON 的分散布局正是为了提高硬件木马的检测能力，通过分析多个 RO 的检测结果，判定该芯片是否存在硬件木马。

芯片内的温度在 30 s 左右芯片内温度就扩散均匀并逐渐向平稳靠近[8]，硬件木马产生的额外热量将被扩散到整个芯片，会造成了所有 RO 的测试SSE值都偏高。图 7 是 1 号芯片中 16 个 RO 分别测试的SSE值对比。从图 7 中可以看出，每个 RO 所测得的SSE值都相对较大，很难直接判定硬件木马植入的大致区域。在测试时间 $t_{\mathrm{test}}=5$ min 内的测试数据可以看作对硬件木马影响的积累，在芯片达到热平衡前积累越高越有利于判定硬件木马的存在，因此需要对 30 s 内的数据进行单独分析。

图 8 是取前 30 s 的测试数据形成的测试结果。从图 8 中的 T1 植入位置 A 中 RO1 时SSE值明显偏高，可以明显判断出硬件木马植入在 RO1 附近位置；对于 T1 植入位置 B 中，RO5 相对较高，则可大致判断硬件木马离RO5较近。同时RO15和RO16离测试控制器和计数器太近，受到的工作噪声影响较大，测试结果的SSE值偏高。

5　结束语

硬件木马的植入会带来额外的热量从而引起芯片内温度变化，环形振荡器是对温度变化极为敏感，

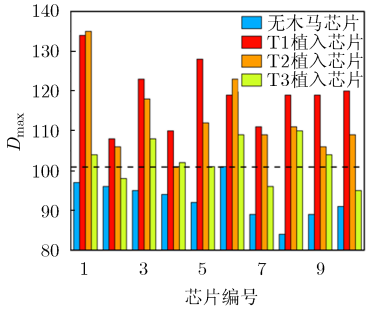

图 6 有无木马芯片的 $D_{\max}$ 值对比

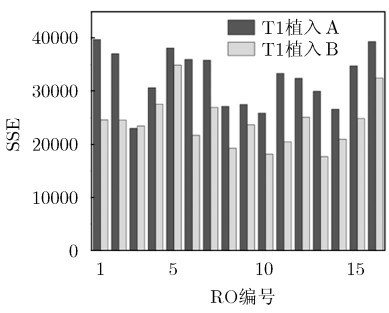

图 7　1 号芯片测试 5 min SSE 值分布

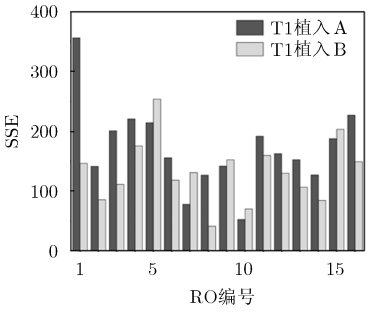

图 8　1 号芯片测试 30 s SSE 值分布

表 3 RO1 检测结果分布表

芯片编号	无木马芯片		T1 植入芯片		T2 植入芯片		T3 植入芯片	
	SSE	R_s	SSE	R_s	SSE	R_s	SSE	R_s
1	573.4	0.994	39763.6	0.790	18349.4	0.878	1540.0	0.985
2	609.1	0.994	6737.3	0.948	9013.0	0.929	998.4	0.990
3	466.5	0.995	31081.5	0.830	25462.5	0.849	11738.1	0.916
4	530.2	0.995	7804.3	0.937	11213.8	0.911	9236.8	0.930
5	393.1	0.996	57873.2	0.727	15723.5	0.897	3123.6	0.973
6	587.7	0.995	36100.6	0.791	32452.2	0.818	8593.9	0.937
7	507.3	0.995	19785.7	0.886	10252.5	0.931	2277.5	0.980
8	990.9	0.988	39187.4	0.785	21816.5	0.853	10326.0	0.932
9	357.0	0.996	31687.1	0.823	8060.0	0.943	4489.6	0.963
10	1130.0	0.989	76542.0	0.666	67762.8	0.702	7160.9	0.945

表 4 不同植入位置下 RO1 检测结果分布表

芯片编号	T1 植入 A		T1 植入 B		T3 植入 A		T3 植入 B	
	SSE	R_s	SSE	R_s	SSE	R_s	SSE	R_s
1	39763.6	0.790	21293.0	0.857	1540.0	0.985	1517.1	0.987
2	6737.3	0.948	4240.0	0.966	998.4	0.990	874.3	0.991
3	31081.5	0.830	11784.4	0.917	11738.1	0.916	4960.1	0.959
4	7804.3	0.937	16496.0	0.895	9236.8	0.930	5660.0	0.954
5	57873.2	0.727	14597.0	0.900	3123.6	0.973	566.1	0.994
6	36100.6	0.791	30133.8	0.827	8593.9	0.937	7430.0	0.945
7	19785.7	0.886	4859.0	0.963	2277.5	0.980	422.5	0.996
8	39187.4	0.785	24486.8	0.837	10326.0	0.932	6516.5	0.944
9	31687.1	0.823	14557.7	0.890	4489.6	0.963	1880.4	0.985
10	76542.0	0.666	13236.7	0.903	7160.9	0.945	1009.5	0.991

本文通过构建环形振荡器网络监测芯片内部温度特征，分析硬件木马的植入会对温度变化趋势带来的影响，设计新算法并利用曲线拟合评价参数SSE和R_square对是否存在硬件木马进行判决。实验结果表明，基于温度变化趋势的硬件木马检测方法能够有效地完成硬件木马的检测，通过对 10 个不同芯片的测试数据分析得出，当硬件木马单元为 32 个逻辑单元时，对硬件木马能达到 100%检测，对面积消耗在 16 个逻辑单元硬件木马能达到 90%的检测概率，同时 RON 结构能够扩大芯片内部监测范围，在一定程度上达到硬件木马的粗定位。本文重点在于验证文中检测方法的可行性，对于硬件木马的多样性和硬件木马的定位研究有限，在下一步工作将加强该检测方法对于硬件木马检测的多样性和硬件木马定位检测定位。

参 考 文 献

[1] 刘长龙. 基于侧信道分析的硬件木马检测技术研究[D]. [博士论文], 天津大学, 2013: 1–8.
LIU C L. Research of hardware Trojans detection technology based on side channel analysis[D]. [Ph.D. dissertation], Tianjin University, 2013: 1–8.

[2] YANG K and HICKS M. Analog malicious hardware[C]. IEEE Symposium on Security and Privacy Conference, San Jose, USA, 2016: 18–37. doi: 10.1109/SP.2016.10.

[3] SUBRAMANYAN P, TSISKARIDZE N, and LI Wenchao. Reverse engineering digital circuits using structural and functional analyses[J]. *IEEE Transactions on Emerging Topics in Computing*, 2014, 2(1): 63–80. doi: 10.1109/TETC.2013.2294918.

[4] BAO Chongxi, FORTE D , and SRIVASTAVE A. On Reverse engineering-based hardware Trojan detection[J]. *IEEE*

Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2016, 35(1): 49–57. doi: 10.1109/TCAD.2015.2488495.

[5] AGRAWAL D and BAKTIR S. Trojan detection using IC fingerprinting[C]. IEEE Symposium on Security and Privacy Conference, Berkeley, USA, 2007: 296–310. doi: 10.1109/SP.2007.36.

[6] JIN Y and MAKRIS Y. Hardware Trojan detection using path delay fingerprint[C]. IEEE International Workshop on Hardware-Oriented Security and Trust Conference, Anaheim, USA, 2008: 51–57. doi: 10.1109/HST.2008.4559049.

[7] AARESTAD J, ACHARYYA D, and RAD R. Detecting Trojans through leakage current analysis using multiple supply pad I_{DDQ}s[J]. *IEEE Transactions on Information Forensics and Security*, 2010, 5(4): 893–904. doi: 10.1109/TIFS.2010.2061228.

[8] NOWROZ A N, HU Kangqiao, and KOUSHANFAR F. Novel techniques for high-sensitivity hardware Trojan detection using thermal and power maps[J]. *IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems*, 2014, 33(12): 1792–1805. doi: 10.1109/TCAD.2014.2354293.

[9] SOLL O and KORAK T. EM-based detection of hardware Trojans on FPGAs[C]. IEEE International Symposium on Hardware-Oriented Security and Trust Conference, California, USA, 2014: 84–87. doi: 10.1109/HST.2014.6855574.

[10] NGO X T, NAJM Z, and BHASIN S. Method taking into account process dispersion to detect hardware Trojan horse by side-channel analysis[J]. *Journal of Cryptographic Engineering*, 2016, 6(3): 239–247. doi: 10.1007/s13389-016-0129-2.

[11] 汪鹏君, 张跃军, 张学龙, 等. 防御差分功耗分析攻击技术研究[J]. 电子与信息学报, 2012, 34(11): 2774–2784. doi: 10.3724/SP.J.1146.2012.00555.

WANG Pengjun, ZHANG Yuejun, ZHANG Xuelong, *et al*. Research of differential power analysis countermeasures[J]. *Journal of Electronics & Information Technology*, 2012, 34(11): 2774–2784. doi: 10.3724/SP.J.1146.2012.00555.

[12] SREEDHAR A, KUNDU S, and KOREN I. On reliability Trojan injection and detection[J]. *Journal on Low Power Electronics*, 2012, 8(5): 674–683. doi: 10.1166/jolpe.2012.1225.34.

[13] 薛明富, 胡爱群, 王箭. 基于探索式分区和测试向量生成的硬件木马检测方法[J]. 电子学报, 2016, 44(5): 1132–1138. doi: 10.3969/j.issn.0372-2112.2016.05.017.

XUE Mingfu, HU Aiqun, and WANG Jian. A novel hardware Trojan detection technique using heuristic partition and test pattern generation[J]. *Acta Electronica Sinica*, 2016, 44(5): 1132–1138. doi: 10.3969/j.issn.0372-2112.2016.05.017.

[14] KULKARNI A, PINO Y, and MOHSENIN T. SVM-based real-time hardware Trojan detection for many-core platform[C]. IEEE International Symposium on Quality Electronic Design Conference, California, USA, 2016: 362–367. doi: 10.1109/ISQED.2016.7479228.

[15] CHAKRABORTY R S and PAUL S. On-demand transparency for improving hardware Trojan detectability[C]. IEEE International Workshop on Hardware-Oriented Security and Trust Conference, Anaheim, USA, 2008: 48–50. doi: 10.1109/HST.2008.4559048.

[16] ZHOU Bin, ZHANG Wei, THAMBIPILLAI S, *et al*. Cost-efficient acceleration of hardware Trojan detection through fan-out cone analysis and weighted random pattern technique[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2016, 35(5): 792–805. doi: 10.1109/TCAD.2015.2460551.

[17] LI Jie and LACH J. At-speed delay characterization for IC authentication and Trojan horse detection[C]. IEEE International Workshop on Hardware-Oriented Security and Trust Conference, Anaheim, USA, 2008: 8–14. doi: 10.1109/HST.2008.4559038.

[18] JIN Y and KUPP N. DFTT: Design for Trojan test[C]. IEEE International Conference on Electronics & Circuits & Systems, Athens, Greece, 2010: 1168–1171. doi: 10.1109/ICECS.2010.5724725.

[19] ZHANG Xuihui and TEHRANIPOOR M. RON: An on-chip ring oscillator network for hardware Trojan detection[C]. Design Automation & Test in Europe Conference & Exhibition, Grenoble, France, 2011: 1–6. doi: 10.1109/DATE.2011.5763260.

[20] XIAO Kan and TEHRANIPOOR M. BISA: Built-in self-authentication for preventing hardware Trojan insertion[C]. IEEE International Symposium on Hardware-Oriented Security and Trust Conference, Anaheim, USA, 2013: 45–50. doi: 10.1109/HST.2013.6581564.

[21] WU Tony F, GANESAN K, HU Yunqing, *et al*. TPAD: Hardware Trojan prevention and detection for trusted integrated circuits[J]. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2016, 35(4): 521–534. doi: 10.1109/TCAD.2015.2474373.

钟晶鑫：　男，1990 年生，博士生，研究方向为芯片安全.

王建业：　男，1962 年生，教授，研究方向为集成电路设计、芯片安全.