IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 17, 2016, accepted May 23, 2016, date of publication June 1, 2016, date of current version June 24, 2016.

Digital Object Identifier 10.1109/ACCESS.2016.2575039

A New Characterization of Hardware Trojans

SAMER MOEIN[1], (Member, IEEE), THOMAS AARON GULLIVER[1], (Senior Member, IEEE), FAYEZ GEBALI[1], (Life Senior Member, IEEE), AND ABDULRAHMAN ALKANDARI[2]

[1]Department of Electrical and Computer Engineering, University of Victoria, Victoria, V8W 2Y2, Canada
[2]Department of Computer, Public Authority for Applied Education and Training, Kuwait City 34053, Kuwait

Corresponding author: S. Moein (samerm@uvic.ca)

ABSTRACT This paper examines hardware trojan threats to semiconductor chips, which is particularly important for chips intended for vital infrastructure and critical applications. The phases of the chip production life-cycle are considered in terms of the opportunities for trojan insertion. Trojans are examined based on eight attribute categories. A matrix identifying the relationships between these attributes is defined. This matrix is used to characterize hardware trojans from both the attacker and defender perspectives. Two case studies are given to illustrate the usefulness of the proposed approach.

INDEX TERMS Hardware trojans, chip life cycle, hardware trojan taxonomy, hardware trojan attributes.

I. INTRODUCTION

In September 2007, Israeli jets bombed a suspected nuclear installation in northeastern Syria. Among the mysteries surrounding this airstrike is the failure of the Syrian radar systems. It has been suggested that the commercial off-the-shelf microprocessors used in these systems may have been fabricated with a hardware backdoor which was used to temporarily disable them [1]. There are many documented cases of similar incidents which have caused significant government, industry, and consumer concerns. In response, the Defence Advanced Research Projects Agency (DARPA) initiated the Trusted Integrated Circuits (TRUST) program to develop techniques for trojan detection [1], [2]. This highlights the fact that hardware designers and researchers must be vigilant to the insertion of hardware trojans during all phases of the chip production life-cycle. The characterization of these trojans is the main goal of this paper.

Integrated Circuits (ICs) are becoming increasingly vulnerable to hardware trojan attacks due to design outsourcing, overseas fabrication, and increasing reliance on third-party Intellectual Property (IP). In addition, hardware attacks can originate from the use of unverified design automation tools. Fig. 1 shows the modern IC production life-cycle phases: ***specification***, ***design***, ***fabrication*** (including interfacing, masking, wafer fabrication and assembly, and packaging), ***testing*** and ***deployment*** [3]–[6]. Complete protection of a chip during its life-cycle is only possible if it is produced by a trusted source.

Several researchers have proposed taxonomies for hardware trojans based on their attributes [7]–[12].

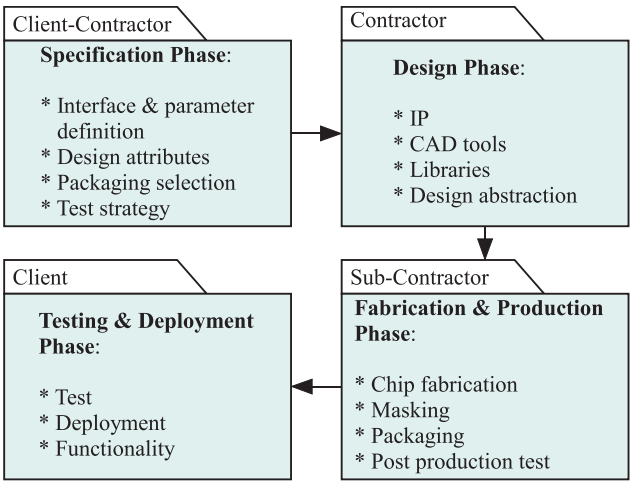


FIGURE 1. The Integrated Circuit (IC) design life-cycle phases.

In [7] and [8], hardware trojans were classified based on two categories: trigger and payload. These are in fact activation mechanisms for trojans. In [9] and [10], the classification was based on three categories: physical, activation, and action. Although this adds two categories to the previous taxonomy, the classification is not related to the chip life-cycle. In [11], a more detailed classification was developed based on five categories: insertion phase, abstraction level, activation mechanism, effect, and location. This classification considers the chip life-cycle and the targeted location, but not the physical characteristics of trojans. The taxonomy in [11] has been used to examine trojans and identify the

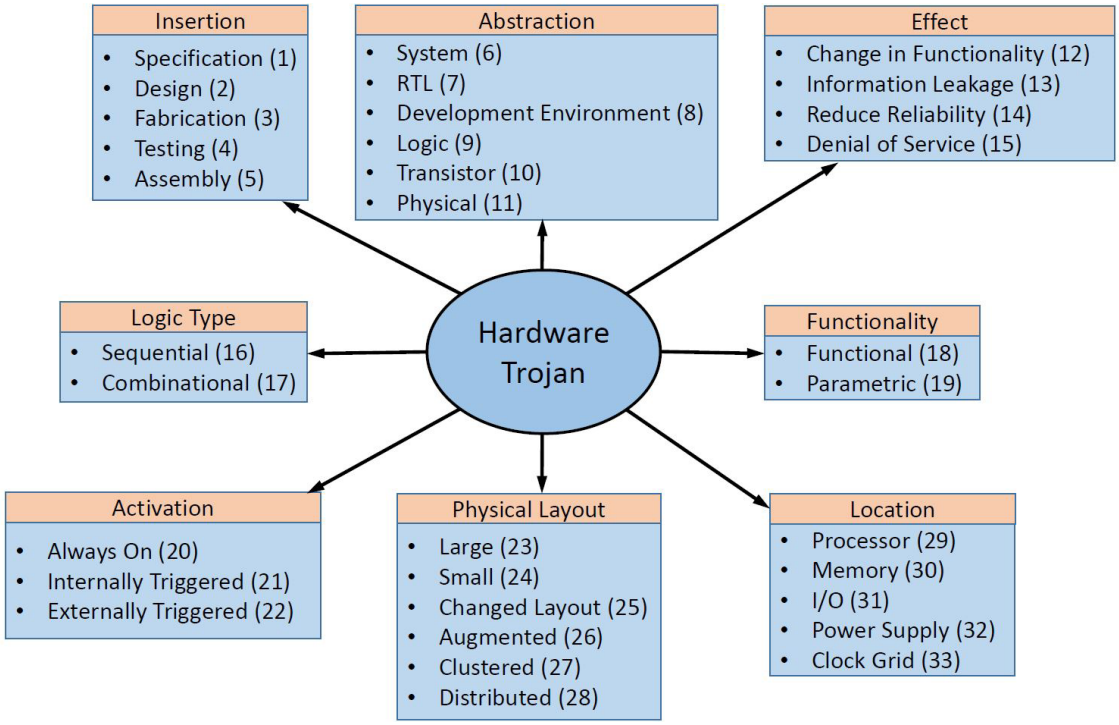


FIGURE 2. The hardware trojan taxonomy.

corresponding attributes. A comprehensive classification was introduced in [12] which is based on 33 attributes in eight categories as shown in Fig. 2. This includes the logic type of a trojan, which plays a major role in predicting the effect of a trojan and thus determining appropriate detection and mitigation techniques.

In this paper, a new methodology is presented to study hardware trojan attributes and their relationships. The classification in Fig. 2 from [12] is used to illustrate the proposed approach, but it is flexible and can be used with any hardware trojan classification. Further, new attributes based on technology or chip manufacturing developments can easily be accommodated. This is important, as with any circuit a hardware trojan goes through several production phases as it becomes embedded into the target system.

The contributions of this paper are as follows:

1) The relationships between the hardware trojan attributes are defined algebraically based on a comprehensive taxonomy with four levels. These results can be used to identify unknown attributes and characterize trojans. Further, this information can be employed to improve chip security during manufacturing, and determine appropriate trojan detection techniques based on vulnerable attributes.
2) The trojan life-cycle of a chip is used to determine how and where a trojan can be inserted into a chip.
3) The proposed approach is flexible and can be used with any trojan classification or taxonomy.

The remainder of this paper is organized as follows. An algebraic approach to characterizing hardware trojans is given in Section II. Based on this approach, two case studies are presented in Section III. Finally, Section IV provides some concluding remarks.

II. ALGEBRAIC APPROACH TO HARDWARE TROJAN ATTRIBUTES

In this section, an algebraic approach to hardware trojan attributes is presented. Fig. 2 provides a comprehensive classification of these attributes. They belong to one of four trojan levels: chip life-cycle, abstraction, properties, and location, as shown in Fig. 3.

The chip life-cycle level in Fig. 3 is equivalent to the insertion category in Fig. 2. In this level, the attacker decides at which phase of the life-cycle the trojan will be inserted. According to this decision, the abstraction of the trojan insertion is determined. The abstraction category in Fig. 2 is the same as the abstraction level in Fig. 3. The trojan properties level contains the properties a trojan may have according to the chip life-cycle and abstraction levels. The possible location for a trojan is based on the chip life-cycle, abstraction, and properties levels. Each trojan has a particular combination of attributes which define its characteristics. These can be represented using an $n \times n$ matrix defined as

$$\mathbf{R} = \begin{bmatrix} \mathbf{R_1} & \mathbf{R_{12}} & 0 & 0 \\ 0 & \mathbf{R_2} & \mathbf{R_{23}} & 0 \\ 0 & 0 & \mathbf{R_3} & \mathbf{R_{34}} \\ 0 & 0 & 0 & \mathbf{R_4} \end{bmatrix}$$

where n is the number of attributes.

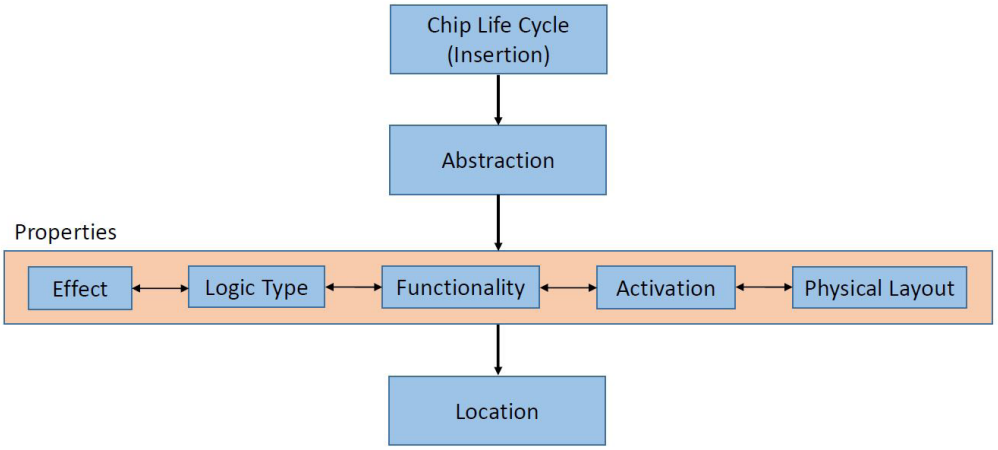


FIGURE 3. The four hardware trojan levels.

A. HARDWARE TROJAN MATRIX

The hardware trojan matrix $\mathbf{R}$ has four square submatrices on the diagonal: $\mathbf{R_1}$, $\mathbf{R_2}$, $\mathbf{R_3}$, and $\mathbf{R_4}$. These matrices represent the levels in Fig. 3. The transfer matrices $\mathbf{R_{12}}$, $\mathbf{R_{23}}$, and $\mathbf{R_{34}}$ represent the connections between the levels. The characteristics of this matrix are described below.

1) SINGLE ELEMENT

In matrix $\mathbf{R}$, $r(i,j) = 1$ indicates that attribute i can lead to attribute j

$$\forall i,j: \quad r(i,j) = 1 \Rightarrow \text{attribute}(i) \rightarrow \text{attribute}(j). \tag{1}$$

By definition $r(i,i) = 0$.

2) ROW

In matrix $\mathbf{R}$, $r(i,j) = 1$ in row i indicates that attribute i can lead to these attributes in column j

$$\forall i: \quad r(i,j) = 1 \Rightarrow \text{attribute}(i) \rightarrow \text{attribute}(j). \tag{2}$$

3) COLUMN

In matrix $\mathbf{R}$, $r(i,j) = 1$ in column j indicates the attributes that can lead to attribute j

$$\forall j: \quad r(i,j) = 1 \Rightarrow \text{attribute}(i) \rightarrow \text{attribute}(j). \tag{3}$$

4) FAN IN

The fan in is the number of connections to an attribute from other attributes. This indicates how many other attributes can lead to this attribute. The fan in can be expressed as

$$F_{in}(j) = \sum_{i=1}^{n} r(i,j). \tag{4}$$

5) FAN OUT

The fan out is the number of connections from an attribute to other attributes. This indicates how many attributes this attribute can lead to. The fan out can be expressed as

$$F_{out}(i) = \sum_{j=i}^{n} r(i,j). \tag{5}$$

6) ROOT ATTRIBUTE

An attribute with $F_{in} = 0$ is called a root attribute. It is an attribute that is not the result of any other attribute so that

$$F_{in}(j) = \sum_{i=1}^{n} r(i,j) = 0. \tag{6}$$

7) TERMINAL ATTRIBUTE

A terminal attribute is an attribute with $F_{out} = 0$. It is the location of a trojan inserted in a system. It does not lead to any attributes so that

$$F_{out}(i) = \sum_{j=i}^{n} r(i,j) = 0. \tag{7}$$

8) INTERMEDIATE ATTRIBUTE

An intermediate attribute has $F_{in} \neq 0$ and $F_{out} \neq 0$. Thus, it is an attribute that can be a consequence of other attributes, and also lead to other attributes.

B. SUBMATRIX R_1

The submatrix $\mathbf{R_1}$ given by

$$\mathbf{R_1} = \left[\begin{array}{c|ccccc} A & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 1 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 0 \\ 4 & 0 & 0 & 0 & 0 & 1 \\ 5 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

represents the trojan chip life-cycle level shown in Fig. 3. It also indicates the relationships between attributes in

the insertion phase in Fig. 2. These start with specification (attribute 1), and end with assembly (attribute 5). For example, $\mathbf{R_1}(1, 2) \equiv \mathbf{R_1}(\text{Specification, Design}) = 1$ indicates that if there is an incomplete or incorrect specification, it can lead to an improper or defective design.

C. SUBMATRIX R_2

The submatrix $\mathbf{R_2}$ given by

$$\mathbf{R_2} = \left[\begin{array}{c|cccccc} A & 6 & 7 & 8 & 9 & 10 & 11 \\ \hline 6 & 0 & 1 & 0 & 0 & 0 & 0 \\ 7 & 0 & 0 & 1 & 0 & 0 & 0 \\ 8 & 0 & 0 & 0 & 1 & 0 & 0 \\ 9 & 0 & 0 & 0 & 0 & 1 & 0 \\ 10 & 0 & 0 & 0 & 0 & 0 & 1 \\ 11 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

represents the trojan design level in Fig. 3. It also indicates the relationships between the attributes in the abstraction level in Fig. 2. The abstraction level has six sublevels beginning with system (attribute 6) and ending with physical (attribute 11). A modification in a sublevel can propagate to the following sublevels. For example, $\mathbf{R_2}(6, 7) \equiv \mathbf{R_2}(\text{System, RTL}) = 1$ indicates that if there is an alteration to the interconnections, hardware modules, or communication protocols, it can lead to the RTL signals, registers, or boolean functions being compromised.

D. SUBMATRIX R_{12}

The submatrix $\mathbf{R_{12}}$ given by

$$\mathbf{R_{12}} = \left[\begin{array}{c|cccccc} A & 6 & 7 & 8 & 9 & 10 & 11 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 1 \\ 4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 1 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

describes the connections between $\mathbf{R_1}$ and $\mathbf{R_2}$, and thus the relationships between the insertion attributes and the abstraction attributes. For example, $\mathbf{R_{12}}(2, 7) \equiv \mathbf{R_{12}}(\text{Design, RTL}) = 1$ indicates that if there is an attempt to insert a trojan in the design level, a modification of the RTL can be used to accommodate the alteration.

It is clear that development environment (attribute 8) and transistor (attribute 10) are not directly connected to the insertion phase, but both are indirectly connected through paths in $\mathbf{R}$. The system level (attribute 6) has the highest fan in as it is connected to specification (attribute 1), testing (attribute 4), and assembly (attribute 5) from the insertion category.

E. SUBMATRIX R_3

Submatrix $\mathbf{R_3}$ given by

$$\mathbf{R_3} = \left[\begin{array}{c|cccccccccccccccc} A & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 & 26 & 27 & 28 \\ \hline 12 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 13 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 14 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 15 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 16 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 17 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 18 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 19 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 20 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 21 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 22 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 23 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 24 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 25 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 26 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 27 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 28 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{array}\right]$$

represents the properties of a trojan given in Fig. 3 and contains attributes 12 to 28. It indicates the relationships between the effect, logic type, functionality, activation, and physical layout attributes. For example, $\mathbf{R_3}(14, 19) \equiv \mathbf{R_3}(\text{Reduced Reliability, Parametric}) = 1$ indicates that if a trojan reduces chip reliability (i.e. quicker battery drain), it can lead to changes in the power consumption, and thermal and delay profiles.

F. SUBMATRIX R_{23}

Submatrix $\mathbf{R_{23}}$ given by

$$\mathbf{R_{23}} = \left[\begin{array}{c|cccccccccccccccc} A & 12 & 13 & 14 & 15 & 16 & 17 & 18 & 19 & 20 & 21 & 22 & 23 & 24 & 25 & 26 & 27 & 28 \\ \hline 6 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 7 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 8 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 9 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 10 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 11 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

is the transfer matrix that describes the connections between $\mathbf{R_2}$ and $\mathbf{R_3}$. Thus it indicates the connections between the abstraction attributes and the trojan property attributes.

1) SINGLE ELEMENT EXAMPLE

$\mathbf{R_{23}}(7, 12) \equiv \mathbf{R_{23}}(\text{RTL, Change in Functionality}) = 1$ indicates that if a trojan has been inserted in the RTL abstraction level, which changes the function description, it can also change the function behaviour.

2) ROW EXAMPLE

For row $= 7$,
$\mathbf{R_{23}}(7, j) \equiv \mathbf{R_{23}}(\text{RTL}, j) = 1$:

$\mathbf{R_{23}}(7, 12) \equiv \mathbf{R_{23}}$(RTL, Change in Functionality),
$\mathbf{R_{23}}(7, 15) \equiv \mathbf{R_{23}}$(RTL, Denial of Service),
$\mathbf{R_{23}}(7, 16) \equiv \mathbf{R_{23}}$(RTL, Sequential),
$\mathbf{R_{23}}(7, 17) \equiv \mathbf{R_{23}}$(RTL, Combinational),
$\mathbf{R_{23}}(7, 18) \equiv \mathbf{R_{23}}$(RTL, Functional),
$\mathbf{R_{23}}(7, 20) \equiv \mathbf{R_{23}}$(RTL, Always On),
$\mathbf{R_{23}}(7, 21) \equiv \mathbf{R_{23}}$(RTL, Internally Triggered),
$\mathbf{R_{23}}(7, 22) \equiv \mathbf{R_{23}}$(RTL, Externally Triggered),
$\mathbf{R_{23}}(7, 23) \equiv \mathbf{R_{23}}$(RTL, Large), and
$\mathbf{R_{23}}(7, 24) \equiv \mathbf{R_{23}}$(RTL, Small) $= 1$,
gives the trojan property attributes that may occur as a result of a trojan being inserted in the RTL abstraction level.

3) COLUMN EXAMPLE

For column $= 14$,
$\mathbf{R_{23}}(i, 14) \equiv \mathbf{R_{23}}$(i, Reduce Reliability) $= 1$:
$\mathbf{R_{23}}(10, 14) \equiv \mathbf{R_{23}}$(Transistor, Reduce Reliability), and
$\mathbf{R_{23}}(11, 14) \equiv \mathbf{R_{23}}$(Physical, Reduce Reliability) $= 1$,
indicates that if a trojan reduces the reliability of a chip, it may be because of a modification in the transistor and/or physical abstraction levels.

4) FAN IN AND FAN OUT EXAMPLES

Table 1 shows the fan in and fan out for each attribute in $\mathbf{R_{23}}$. The maximum fan out is 14 for development environment (attribute 8). Thus if a CAD tool is exploited by an attacker, there can be many chip vulnerabilities. The maximum fan in is 6 for change in functionality, functional, and always on (attributes 12, 18, and 20, respectively). This means that many trojans will have these attributes.

G. SUBMATRIX R_{34}

Submatrix $\mathbf{R_{34}}$ given by

$$\mathbf{R_{34}} = \left[\begin{array}{c|ccccc} A & 29 & 30 & 31 & 32 & 33 \\ \hline 12 & 1 & 1 & 1 & 1 & 1 \\ 13 & 1 & 1 & 1 & 1 & 1 \\ 14 & 1 & 1 & 1 & 1 & 1 \\ 15 & 1 & 0 & 1 & 1 & 1 \\ 16 & 1 & 1 & 1 & 1 & 1 \\ 17 & 1 & 1 & 1 & 1 & 1 \\ 18 & 1 & 1 & 1 & 1 & 1 \\ 19 & 0 & 0 & 1 & 1 & 1 \\ 20 & 1 & 1 & 1 & 1 & 1 \\ 21 & 1 & 1 & 1 & 1 & 1 \\ 22 & 1 & 1 & 1 & 1 & 1 \\ 23 & 1 & 1 & 0 & 0 & 0 \\ 24 & 1 & 1 & 1 & 1 & 1 \\ 25 & 1 & 1 & 1 & 1 & 1 \\ 26 & 1 & 1 & 1 & 1 & 1 \\ 27 & 1 & 1 & 1 & 1 & 1 \\ 28 & 1 & 1 & 1 & 1 & 1 \end{array}\right]$$

describes the connections between $\mathbf{R_3}$ and $\mathbf{R_4}$. Thus it indicates the connections between the trojan properties and the hardware locations. For example, $\mathbf{R_{34}}(15, 29) \equiv \mathbf{R_{34}}$(Denial of Service, Processor) $= 1$ indicates that if a denial of service trojan has been inserted, it may be located in the processor to prevent the chip from executing properly. It is clear from this matrix that all locations are almost equally vulnerable to trojans.

TABLE 1. Matrix $\mathbf{R_{23}}$ attributes fan in and fan out.

Attribute	F_{in}	F_{out}	Attribute	F_{in}	F_{out}
6	-	6	18	6	-
7	-	10	19	3	-
8	-	14	20	6	-
9	-	8	21	3	-
10	-	9	22	3	-
11	-	12	23	3	-
12	6	-	24	4	-
13	2	-	25	2	-
14	2	-	26	3	-
15	4	-	27	3	-
16	3	-	28	2	-
17	4	-	-	-	-

H. MATRIX R

Matrix $\mathbf{R}$ represents the complete set of attributes in Fig. 2, and the hardware trojan levels in Fig. 3. The characteristics of a trojan can be inferred from the paths in $\mathbf{R}$. Note that an empty submatrix indicates that all entries are zero. For example, $\mathbf{R_4} = 0$ since the location attributes are terminal attributes and are not interconnected.

1) SINGLE ELEMENT EXAMPLE

$\mathbf{R}(17, 29) \equiv \mathbf{R}$(Combinational, Processor) $= 1$ indicates that if there is a trojan inserted in a processor, it may be a combinational circuit.

2) ROW EXAMPLE

For row $= 6$,
$\mathbf{R}(6, j) \equiv \mathbf{R}$(System, j) $= 1$:
$\mathbf{R}(6, 7) \equiv \mathbf{R}$(System, RTL),
$\mathbf{R}(6, 12) \equiv \mathbf{R}$(System, Change in Functionality),
$\mathbf{R}(6, 13) \equiv \mathbf{R}$(System, Information Leakage),
$\mathbf{R}(6, 15) \equiv \mathbf{R}$(System, Denial of Service),
$\mathbf{R}(6, 18) \equiv \mathbf{R}$(System, Functional),
$\mathbf{R}(6, 19) \equiv \mathbf{R}$(System, Parametric), and
$\mathbf{R}(6, 20) \equiv \mathbf{R}$(System, Always On) $= 1$,
which implies that if a trojan is introduced at the system abstraction level, it may lead to a combination of RTL, change in functionality, information leakage, denial of service, functional, parametric, and always on.

3) COLUMN EXAMPLE

For column $= 6$,
$\mathbf{R}(i, 6) \equiv \mathbf{R}$(i, System) $= 1$:
$\mathbf{R}(1, 6) \equiv \mathbf{R}$(Specification, System),
$\mathbf{R}(4, 6) \equiv \mathbf{R}$(Testing, System), and
$\mathbf{R}(5, 6) \equiv \mathbf{R}$(Assembly, System) $= 1$,
which implies that if a trojan is inserted at the system abstraction level, it may be because of an incomplete or modified specification, control of the testing to evade trojan detection, and/or modification in the assembly phase.

$\mathbf{R} =$

A	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
1	0	1	0	0	0	1	0	0	0	0	0																						
2	0	0	1	0	0	0	1	0	0	0	0																						
3	0	0	0	1	0	0	0	0	0	0	1																						
4	0	0	0	0	1	1	0	0	1	0	0																						
5	0	0	0	0	0	1	0	0	0	0	0																						
6						0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0					
7						0	0	1	0	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	0	0	0	0					
8						0	0	0	1	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1					
9						0	0	0	0	1	0	1	0	0	1	1	1	1	0	1	1	1	0	0	0	0	0	0					
10						0	0	0	0	0	1	1	0	1	0	0	1	1	1	1	0	0	0	1	0	1	1	0					
11						0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	1	1	1	1	1	1					
12												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13												0	0	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	1	1	1	1	1
14												0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1
15												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1
16												1	0	0	1	0	0	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1
17												1	1	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18												1	0	0	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19												0	1	1	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	0	1	1	1
20												1	1	1	1	0	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
21												1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1
22												1	1	0	1	1	1	1	0	0	0	0	0	1	0	1	1	0	1	1	1	1	1
23												1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	1	1	0	0	0
24												1	1	1	1	0	1	1	1	1	1	1	0	0	0	1	1	0	1	1	1	1	1
25												1	0	0	1	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	1	1
26												1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1
27												1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	1	1	1	1	1
28												1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	1	1	1	1	1
29																																	
30																																	
31																																	
32																																	
33																																	

4) ROOT ATTRIBUTE EXAMPLE

From (6), specification is a root attribute of any trojan as it is at the start of the chip life-cycle. Thus it does not depend on any other attributes.

5) TERMINAL ATTRIBUTE EXAMPLE

From (7), the location attributes are terminal attributes since they do not lead to other attributes.

6) FAN IN AND FAN OUT EXAMPLES

Table 2 shows the fan in and fan out for the attributes in **R**. The maximum fan in of 19 occurs for functional, always on, and augmented (attributes 18, 20, and 26, respectively). Thus, these three attributes occur most frequently due to other attributes. Adding an inverter gate to a chip to insert a trojan is a simple example which combines all of these attributes. The maximum fan out of 21 occurs only for augmented (attribute 26). Thus, this attribute most frequently leads to other attributes. Table 2 also gives the sums of the fan in and fan out for the attributes in **R**. These values reflect the connectivity of an attribute with other attributes. A critical attribute is an attribute with the highest sum, since it has the greatest number of connections with other attributes. For the analysis presented here, augmented (attribute 26) is the critical attribute. Thus inserting a trojan without modifying the chip layout makes detecting it a very difficult task. Note that **R** is flexible and can be updated to accommodate different taxonomies or changes in the relationships between attributes.

I. PATHS

A path in **R** is a sequence of attributes that lead to other attributes (via rows) or are the result of other attributes (via columns). The paths visually show the attribute relationships

TABLE 2. The fan in and fan out for the attributes in R.

Attribute	F_{in}	F_{out}	$F_{in}+F_{out}$	Attribute	F_{in}	F_{out}	$F_{in}+F_{out}$	Attribute	F_{in}	F_{out}	$F_{in}+F_{out}$
1	0	3	3	12	18	17	35	23	14	13	27
2	1	2	3	13	10	13	23	24	16	17	33
3	1	2	3	14	8	11	19	25	11	14	25
4	1	3	4	15	16	16	32	26	19	21	40
5	1	1	2	16	13	15	28	27	17	19	36
6	3	7	10	17	17	18	35	28	14	17	31
7	2	11	13	18	19	18	37	29	16	0	16
8	1	15	16	19	9	9	18	30	15	0	15
9	2	9	11	20	19	18	37	31	16	0	16
10	1	10	11	21	13	15	28	32	16	0	16
11	2	12	14	22	12	14	26	33	16	0	16

and the connections between the levels in Fig. 3 that characterize a trojan. They will be used in the next section to obtain directed graph representations of two hardware trojans.

III. CASE STUDIES

In this section, two case studies of hardware trojan characterization are presented based on the methodology in the previous section.

A. COMBINATIONAL TROJAN

Assume that an attacker works as a design engineer in a chip manufacturing facility. He uses the trojan shown in Fig. 4 to change the functionality of a chip when a particular event occurs. This is a combinational logic triggered trojan where $A = B = 0$ causes the output C to have an incorrect value (C modified). This circuit reflects an attacker inserting a trojan based on a rare event which is unlikely to be detected during the testing phase. From the figure, the trojan characteristics are change in functionality, combinational, functional, internally triggered, small, clustered, and augmented (attributes 12, 17, 18, 21, 24, 26, and 27 in $\mathbf{R_3}$). Examining the corresponding columns in $\mathbf{R_{23}}$, these attributes are all directly connected to the development environment in the abstraction category (attribute 8).

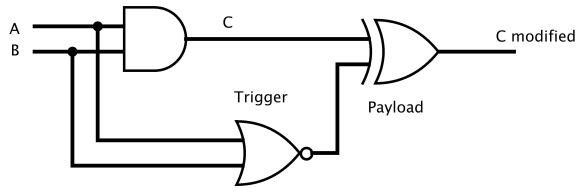


FIGURE 4. A combinational logic triggered trojan.

From $\mathbf{R_{12}}$, there is no direct connection between attribute 8 and an attribute in the insertion category (attributes 1 to 5). However, $\mathbf{R_2}$ provides a connection between the development environment (attribute 8) and RTL (attribute 7). Then attribute 7 is connected to design (attribute 2) and system (attribute 6), and attribute 6 is connected to specification, testing, and assembly (attributes 1, 4, and 5). In addition, $\mathbf{R_1}$ provides connections between the attributes in the insertion category. Matrix **R1** shows all candidate paths that characterize this trojan. In **R1**, the light blue colour shows the row connections, the light red colour shows the column connections, and the direct attribute connections are shown in green.

Assume that attributes 12, 17, 18, 21, 24, 26, and 27 have been identified for the trojan. These attributes all lie in $\mathbf{R_3}$. Moving forward (increasing attribute numbers), the expected trojan locations can be identified, and moving backward (decreasing attribute numbers), the suspect attributes in the insertion and abstraction phases can be identified. It is clear that the attributes in rows 12, 17, 18, 21, 24, 26, and 27 can lead to all trojan locations (columns 29, 30, 31, 32, and 33). In **R1**, attributes 12, 17, 18, 21, 24, 26, and 27 all have value 1 in row 8 only. Thus, a trojan with these seven attributes can be inserted using attribute 8 (development environment) in the abstraction phase. $\mathbf{R_{12}}$ can then be used to reach the life-cycle phase. **R1** provides the complete paths from the insertion phase to the location phase for this trojan. These paths are represented by the directed graph in Fig. 5, which provides a complete characterization of the hardware trojan in Fig. 4. In this figure, S indicates a possible trojan insertion point.

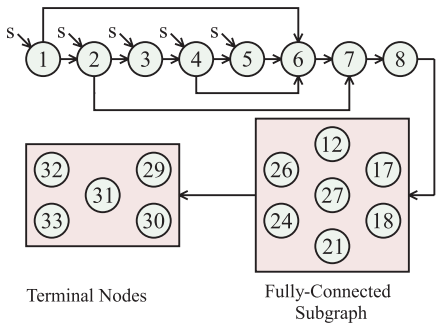


FIGURE 5. A directed graph characterization of the hardware trojan in Fig. 4.

An attacker will choose a particular point in the chip life-cycle for insertion. Assuming the design phase is selected, the graph shown in Fig. 6 represents the inserted trojan. Conversely, a defender has no information about the attacker decisions, so it is necessary to consider all paths in Fig. 5. However, if the specification and assembly phases are trusted, the graph in Fig. 5 will reduce to the graph in Fig. 7.

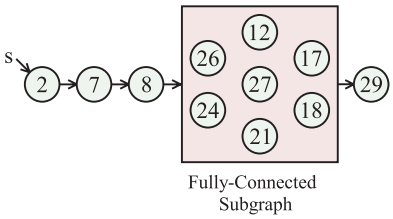


FIGURE 6. The trojan graph assuming it is inserted in the design phase.

Then if a defender is able to verify that attribute 6 is not a trojan characteristic, the trojan in Fig. 6 is identified exactly. The defender has now discovered the suspect attributes (attributes 2, 7, and 8). A hardware trojan detection technique is required to check chips based on these attributes. Techniques such as those in [13]–[15] can be used to detect a trojan with a combination of these attributes. If the trojan is detected, the defender should designate this design facility as untrusted.

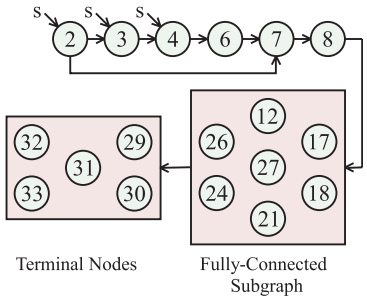


FIGURE 7. The defender graph assuming the trojan is inserted during the chip life cycle.

B. BACKDOOR TROJAN

Consider a chip manufacturer that designs a circuit to compute a function $F(X)$ for a system to authenticate user-password pairs X and $F(X)$. The design specifies

R1 =

A	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33
1	0	1	0	0	0	1	0	0	0	0	0																						
2	0	0	1	0	0	0	1	0	0	0	0																						
3	0	0	0	1	0	0	0	0	0	0	1																						
4	0	0	0	0	1	1	0	0	1	0	0																						
5	0	0	0	0	0	1	0	0	0	0	0																						
6						0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0					
7						0	0	1	0	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	0	0	0	0					
8						0	0	0	1	0	0	1	0	0	1	1	1	1	0	1	1	1	1	1	1	1	1	1					
9						0	0	0	0	1	0	1	0	0	1	1	1	1	0	1	1	1	0	0	0	0	0	0					
10						0	0	0	0	0	1	1	0	1	0	0	1	1	1	1	0	0	0	1	0	1	1	0					
11						0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	0	1	1	1	1	1	1					
12												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
13												0	0	0	0	0	1	0	1	1	0	1	0	1	0	1	1	1	1	1	1	1	1
14												0	0	0	0	0	0	0	1	1	0	0	0	1	0	1	1	1	1	1	1	1	1
15												0	0	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1	1	0	1	1	1
16												1	0	0	1	0	0	1	0	0	1	1	1	0	1	1	1	1	1	1	1	1	1
17												1	1	0	1	0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
18												1	0	0	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1
19												0	1	1	0	0	0	0	0	1	0	0	0	1	0	1	0	1	0	0	1	1	1
20												1	1	1	1	0	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1
21												1	0	0	1	1	1	1	0	0	0	0	1	1	0	1	1	1	1	1	1	1	1
22												1	1	0	1	1	1	1	0	0	0	0	0	1	0	1	1	0	1	1	1	1	1
23												1	0	0	1	1	1	1	0	1	1	0	0	0	1	1	1	1	1	1	0	0	0
24												1	1	1	1	0	1	1	1	1	1	1	0	0	0	1	1	0	1	1	1	1	1
25												1	0	0	1	1	1	1	0	1	0	0	1	0	0	1	1	0	1	1	1	1	1
26												1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1
27												1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	0	0	1	1	1	1	1
28												1	1	1	1	1	1	1	1	1	1	0	1	0	0	1	0	0	1	1	1	1	1
29																																	
30																																	
31																																	
32																																	
33																																	

ten users I_0 to I_9 with $F(X) = X^2$. Since there are ten users, the designer uses four bits, X_1 to X_4, to encode them. As $F(X) = X^2$, the largest function output is 81, so seven bits are required, Z_1 to Z_7. The resulting circuit is shown in Fig. 8a. Testing using these ten input values shows that the outputs are correct and so the circuit functions properly.

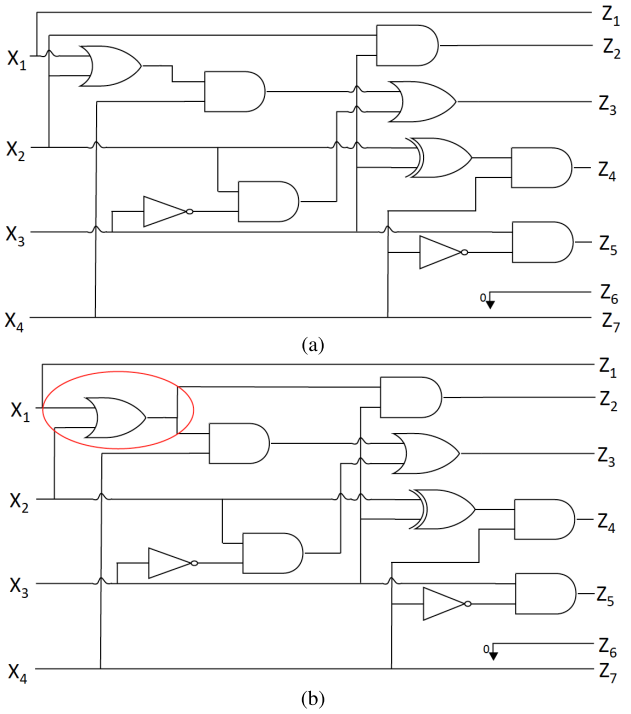


FIGURE 8. The circuits with and without the trojan. (a) Trojan free circuit. (b) Circuit with a backdoor trojan.

With four bits, there are 16 input combinations, but in this case the six inputs 10 to 15 are not used so they are don't care conditions. With this incomplete specification, the designer asserts that the circuit performs the function $F(X) = X^2$ correctly. If an attacker makes the small modification indicated in Fig. 8b, the output for the ten valid inputs is the same as with the original circuit. However, it also provides two additional correct outputs for inputs 10 and 11. Therefore, the modified circuit contains a backdoor trojan.

Table 3 shows the outputs from both circuits for all possible inputs. With an input of 10, circuit (a) produces 68, which is not the square of 10, while an input of 11 produces 89. Thus, circuit (a) provides outputs for the don't care conditions which are not correct values for $F(X)$. However, circuit (b) outputs the correct values of $F(X)$ for inputs 10 and 11, so the user-password pairs (10, 100) and (11, 121) are valid, which confirms that circuit (b) contains a backdoor trojan. It can be used to allow users who do not have permission to access the system. Note that both circuits have the same type and number of gates. The only difference is a wire connection, so the layouts are almost identical.

Assuming the trojan in Fig. 8b is used to gain control of the system to create a denial of service attack, the characteristics are denial of service, combinational logic, functional, and externally triggered (attributes 15, 17, 18, and 22 in $\mathbf{R_3}$). Examining the corresponding columns in $\mathbf{R_{23}}$, this combination of attributes can occur if the trojan is inserted at the RTL, development environment, or logic type in the abstraction phase (attributes 7, 8, or 9). As the attack is enabled by an incomplete specification (attribute 1), the attacker needs to control testing (attribute 4) so that the trojan is not discovered.

There are two steps an attacker must take, insert the trojan during the design phase (attribute 2), and control the testing phase (attribute 4) to evade detection during testing. From $\mathbf{R_{12}}$, attribute 7 is connected to design (attribute 2) and system (attribute 6), and attribute 6 is connected to specification, testing, and assembly (attributes 1, 4, and 5). Logic (attribute 9) is directly connected to testing (attribute 4). This trojan is inserted due to an incomplete specification, and $\mathbf{R_1}$ indicates a connection between design (attribute 2) and specification (attribute 1). These paths are represented by the directed graph in Fig. 9, which is a complete characterization of the hardware trojan in Fig. 8b. In this figure, S indicates a possible trojan insertion point.

An attacker will choose a particular point in the chip lifecycle for insertion. Assuming the design phase or testing phase (attribute 2 or attribute 4) is chosen, the graph shown

TABLE 3. The outputs for the circuits in Fig. 8.

		Inputs					Circuit (a)								Circuit (b)							
		X_1	X_2	X_3	X_4	X	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	$F(X)$	Z_1	Z_2	Z_3	Z_4	Z_5	Z_6	Z_7	$F(X)$
Inputs	I_0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	I_1	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1
	I_2	0	0	1	0	2	0	0	0	0	1	0	0	4	0	0	0	0	1	0	0	4
	I_3	0	0	1	1	3	0	0	0	1	0	0	1	9	0	0	0	1	0	0	1	9
	I_4	0	1	0	0	4	0	0	1	0	0	0	0	16	0	0	1	0	0	0	0	16
	I_5	0	1	0	1	5	0	0	1	1	0	0	1	25	0	0	1	1	0	0	1	25
	I_6	0	1	1	0	6	0	1	0	0	1	0	0	36	0	1	0	0	1	0	0	36
	I_7	0	1	1	1	7	0	1	1	0	0	0	1	49	0	1	1	0	0	0	1	49
	I_8	1	0	0	0	8	1	0	0	0	0	0	0	64	1	0	0	0	0	0	0	64
	I_9	1	0	0	1	9	1	0	1	0	0	0	1	81	1	0	1	0	0	0	1	81
Undefined	I_{10}	1	0	1	0	10	1	0	0	0	1	0	0	68	1	1	0	0	1	0	0	100
	I_{11}	1	0	1	1	11	1	0	1	1	0	0	1	89	1	1	1	1	0	0	1	121
	I_{12}	1	1	0	0	12	1	1	1	0	0	0	0	112	1	0	1	0	0	0	0	80
	I_{13}	1	1	0	1	13	1	1	1	1	0	0	1	121	1	0	1	1	0	0	1	89
	I_{14}	1	1	1	0	14	1	1	0	0	1	0	0	100	1	1	0	0	1	0	0	100
	I_{15}	1	1	1	1	15	1	1	1	0	0	0	1	113	1	1	1	0	0	0	1	113

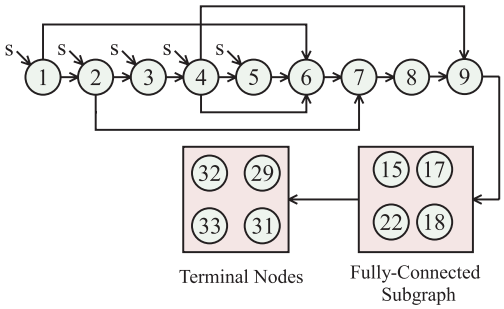


FIGURE 9. A directed graph characterization of the hardware trojan in Fig. 8b.

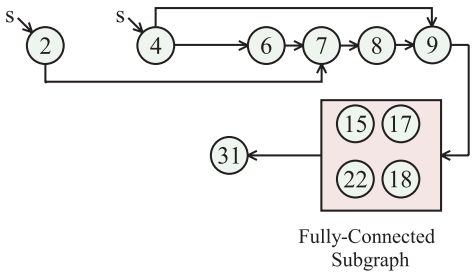


FIGURE 10. The trojan graph assuming it is inserted during the design phase or testing phase.

in Fig. 10 represents the trojan. Conversely, a defender has no information about the attacker decisions, so it is necessary to check all possible paths in Fig. 9. These provide a defender with the suspect attributes (attributes 2, 4, 6, 7, and 9), that can be used by an attacker to insert the trojan. A hardware trojan detection technique is required to check chips based on these attributes. The technique in [16] can be used to detect a trojan with any combination of these attributes.

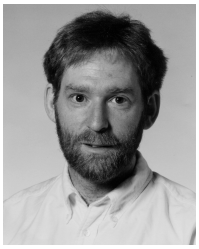
IV. CONCLUSION

Hardware trojans are a growing concern in modern Integrated Circuit (IC) development and production. This is particularly true for chips intended for vital infrastructure and critical applications. This paper considered the attributes of hardware trojans for all phases of the chip life-cycle. Previous results in the literature assume that some phases of this life-cycle can be trusted, while others may be untrusted. The attributes of hardware trojans were characterized, and a hardware trojan matrix was developed to show their connections. This matrix provides many insights into the characteristics of trojans, and thus is a valuable tool for their design and detection.

REFERENCES

[1] S. Adee, "The hunt for the kill switch," *IEEE Spectr.*, vol. 45, no. 5, pp. 34–39, May 2008.
[2] Defence Science Board Task Force on High Performance Microchip Supply. (2005). High Performance Microchip Supply. [Online]. Available: http://www.acq.osd.mil/dsb/reports/ADA435563.pdf
[3] B. Sharkey. (2007). *TRUST in Integrated Circuit Program—Briefing to Industry*. [Online]. Available: http://cryptocomb.org/DARPA - Trust in Integrated Circuits Program.pdf
[4] T. Zhou and T. B. Tarim, "An efficient and well-controlled IC system development flow: Design approved specification and design guided test plan," in *Proc. IEEE Int. Symp Circuits Syst.*, Kobe, Japan, May 2005, pp. 2775–2778.
[5] S. Padmanabhan. (2013). *Discover a Better Way to go From C-Level to Synthesis for SoC Designs*. [Online]. Available: http://electronicdesign.com/technologies/discover-better-way-go-c-level-synthesis-soc-designs
[6] M. Tehranipoor and F. Koushanfar, "A survey of hardware Trojan taxonomy and detection," *IEEE Des. Test Comput.*, vol. 27, no. 1, pp. 10–25, Jan./Feb. 2010.
[7] S. Bhunia, M. S. Hsiao, M. Banga, and S. Narasimhan, "Hardware Trojan attacks: Threat analysis and countermeasures," *Proc. IEEE*, vol. 102, no. 8, pp. 1229–1247, Aug. 2014.
[8] F. Wolff, C. Papachristou, S. Bhunia, and R. S. Chakraborty, "Towards Trojan-free trusted ICs: Problem analysis and detection scheme," in *Proc. Conf. Design, Autom. Test Eur.*, Munich, Germany, Mar. 2008, pp. 1362–1365.
[9] R. M. Rad, X. Wang, M. Tehranipoor, and J. Plusquellic, "Power supply signal calibration techniques for improving detection resolution to hardware Trojans," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, Nov. 2008, pp. 632–639.
[10] X. Wang, M. Tehranipoor, and J. Plusquellic, "Detecting malicious inclusions in secure hardware: Challenges and solutions," in *Proc. IEEE Int. Workshop Hardw.-Oriented Secur. Trust*, Anaheim, CA, USA, Jun. 2008, pp. 15–19.
[11] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware Trojans," *IEEE Comput.*, vol. 43, no. 10, pp. 39–46, Oct. 2010.
[12] S. Moein, S. Khan, T. A. Gulliver, F. Gebali, and M. W. El-Kharashi, "An attribute based classification of hardware Trojans," in *Proc. Int. Conf. Comput. Eng. Syst.*, Cairo, Egypt, Dec. 2015, pp. 351–356.
[13] Y. Liu, K. Huang, and Y. Makris, "Hardware Trojan detection through golden chip-free statistical side-channel fingerprinting," in *Proc. ACM/EDAC/IEEE Design Autom. Conf.*, San Francisco, CA, USA, Jun. 2014, pp. 1–6.
[14] Y. Liu, Y. Jin, and Y. Makris, "Hardware Trojans in wireless cryptographic ICs: Silicon demonstration & detection method evaluation," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, Nov. 2013, pp. 399–404.
[15] X. Mingfu, H. Aiqun, and L. Guyue, "Detecting hardware Trojan through heuristic partition and activity driven test pattern generation," in *Proc. Commun. Secur. Conf.*, Beijing, China, May 2014, pp. 1–6.
[16] D. Forte, C. Bao, and A. Srivastava, "Temperature tracking: An innovative run-time approach for hardware Trojan detection," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design*, San Jose, CA, USA, Nov. 2013, pp. 532–539.

SAMER MOEIN received the B.Sc. degree and the M.Sc. degree from Kuwait University, Kuwait, in 2004 and 2011, respectively, and the Ph.D. degree from the University of Victoria, Victoria, BC, Canada, in 2015, all in computer engineering. He is currently a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, University of Victoria. His research interests include computer security, cryptography, and cryptoprocessors.



T. AARON GULLIVER received the Ph.D. degree in electrical engineering from the University of Victoria, Victoria, BC, Canada, in 1989. From 1989 to 1991, he was a Defence Scientist with the Defence Research Establishment Ottawa, Ottawa, ON, Canada. He has held academic appointments with Carleton University, Ottawa, and the University of Canterbury, Christchurch, New Zealand. He joined the University of Victoria in 1999, where he is a Professor with the Department of Electrical and Computer Engineering. In 2002, he became a fellow of the Engineering Institute of Canada. In 2012, he was elected as a fellow of the Canadian Academy of Engineering. From 2000 to 2003, he was Secretary and a member of the Board of Governors of the IEEE Information Theory Society. He is currently an Area Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. His research interests include information theory and communication theory, algebraic coding theory, multicarrier systems, smart grid, and security.

FAYEZ GEBALI received the B.Sc. (Hons.) degree in electrical engineering from Cairo University, the B.Sc. (Hons.) degree in mathematics from Ain Shams University, and the Ph.D. degree in electrical engineering from the University of British Columbia. He is a Professor with the Department of Electrical and Computer Engineering, University of Victoria, where he is currently Department Chair. His research interests include parallel algorithms, networks-on-chip, 3-D integrated circuits, digital communications, and computer arithmetic. He held an NSERC Postgraduate Scholarship from the University of British Columbia.

ABDULRAHMAN ALKANDARI received the B.Sc. and M.Sc. degrees in computer engineering from Kuwait University, Kuwait, in 2004 and 2011, and the Ph.D. degree in computer science from the International Islamic University Malaysia in 2014. He is an Assistant Professor with the Department of Computer Science, Public Authority for Applied Education and Training (Basic Education College). His research interests include intelligent systems, traffic engineering, algorithms, smart phone applications, IoT, smart cities, and wireless sensor networks.