Identification of Synthesis Approaches for IP/IC Piracy of Reversible Circuits

SAMAH MOHAMED SAEED, City College of New York, City University of New York
NITHIN MAHENDRAN, University of Washington, Tacoma
ALWIN ZULEHNER and ROBERT WILLE, Johannes Kepler University Linz
RAMESH KARRI, New York University

Reversible circuits employ a computational paradigm that is beneficial for several applications including the design of encoding and decoding devices, low-power design, and emerging applications in *quantum computation*. However, similar to conventional logic, reversible circuits are expected to be subject to *Intellectual Property/Integrated Circuit piracy*. To counteract such attacks, an understanding of how to identify the target function from a reversible circuit is a crucial first step. In contrast to conventional logic, the target function is (implicitly or explicitly) embedded into the reversible circuit. Numerous synthesis approaches have been proposed for this embedding task. To recover the target function embedded in a reversible circuit, one needs to know what synthesis approach has been used to embed the circuit.

We propose a machine learning-based scheme to determine the used reversible synthesis approach based on the telltale signs it leaves in the synthesized reversible circuit. We study the impact of optimizing the synthesis approaches on the telltale signs that they leave. Our analysis shows that the synthesis approaches can be determined in the vast majority of cases even if optimized versions of the synthesis approaches are used.

Additional Key Words and Phrases: Reversible logic, Security, IP/IC piracy, BDD, QMDD, ESOP, TBS, Machine learning-based scheme.

1 INTRODUCTION

The rapid increase of the fabrication cost and the design efforts have forced companies to become fab-less and utilize a third-party to fabricate their *Integrated Circuits* (IC). This business model introduces trust issues as it includes unveiling design details, i.e. the *Intellectual Property* (IP) of the company, to third party foundries, end users, and others whose trustworthiness often is suspect. In fact, there is the risk that the third-party will reverse engineer the IC, steal the IP, and make pirated copies. Many companies started to developing and using countermeasures to thwart such malicious activities.

This led to several defense mechanisms to prevent IP/IC piracy. Some of these techniques use logic locking (obfuscation) by inserting additional combinational or sequential logic into the design controlled by secret keys [29, 31, 46]. Others partition the design and fabricate each part in a different fab [27]. At the layout level, IC camouflaging is used to hide the implementation of the design from the end user [7, 26, 28]. While this impact of the global supply chain and the resulting security issues has been studied for conventional circuits, the security implications for circuits and devices that follow novel, alternate computational paradigms and rely on new technologies are not available.

Reversible circuits provide such an example. Reversible computations are conducted in a bijective fashion only (i.e. only one-to-one mappings between the inputs and the outputs are realized). This is an ideal computation model for several established as well as emerging applications. Examples include encoding and decoding devices [41, 43, 51], quantum computing [21, 36], and low-power design [4, 5, 17]. Other areas which benefit from reversible computations include adiabatic circuits [3, 30, 48], formal verification [2], and optical computing [9].

While many of these application areas are coming out of the research stage, significant improvements in the design and (physical) realization of reversible circuits have been made recently. A huge variety of realizations that relies on reversible computations exists. Promising directions have recently been featured in the IEEE Spectrum [12] and conferences [1, 10, 24]. Besides, several investigations on low-power design which exploit reversible computations are currently ongoing (e.g. [15, 45]). Hence, although they are ongoing research, there is a lot of potential.

Considering that conventional CMOS design approaches will reach certain limits soon, it is important to develop a full understanding of all aspects of those emerging technologies – including issues such as synthesis and technology mapping [1, 10, 24], as well as potential security threats. In fact, the respective reversible designs might be subject to attacks by untrustworthy third-parties. A detailed understanding about how to recover the target function of a reversible circuit is crucial to counteract any malicious activities such as IP piracy [23, 31] and the insertion of malicious circuitry into the design [16].

In this work[1], we aim for a better understanding of the security issues. In order to derive a reversible circuit, non-reversible target functions are (implicitly or explicitly) embedded into the circuit – requiring the addition of ancillary inputs (with a constant value) and garbage outputs. Hence, to derive the target function, an attacker first needs to know the value of the ancillary inputs. As a prelude, it is crucial to know what synthesis approach was used to create the circuit. Numerous synthesis solutions have been proposed for reversible circuits. They employ different schemes for embedding. Nevertheless, the most established ones leave telltale signs that one can use to identify the approach.

This paper proposes a machine learning-based scheme to determines the reversible synthesis approach, as a prelude to identifying the target function of a given reversible circuit. To this end, telltale signs of reversible synthesis approaches are extracted and discussed. Furthermore, the suitability of the telltale signs in reversible circuits, which are generated using the optimized versions of the synthesis approaches, is considered. Experimental evaluations show the effectiveness of the scheme in identifying the synthesis approach of a given reversible circuit as a prelude to IP piracy.

The paper is organized as follows. We review the background on reversible circuits and the main motivation of this work in detail in Section 2 and Section 3, respectively. We review the synthesis approaches and extract their telltale signs in Section 4. We then derive corresponding telltale features which help in identifying the synthesis approach in Section 5. Those features are employed in a machine learning-based framework as part of the experimental evaluations in Section 6.

2 REVERSIBLE CIRCUITS

A function $f : B^n \rightarrow B^m$ is *reversible*, if and only if $n = m$ and, each input combination maps to a unique output combination. *Reversible circuits* realizing such functions cannot only derive the output assignment from a given input assignment, but also vice versa; computations can be conducted in two directions. To this end, a different kind of logic gate is used: the *Toffoli gate*.

Let $X = \{x_1, \ldots, x_n\}$ be the variables of a reversible function which are realized in a reversible circuit by *circuit lines*. Then, a Toffoli gate $TOF(C, t)$ is composed of a set $C \subseteq \{x_j \mid x_j \in X\} \cup \{\overline{x}_j \mid x_j \in X\}$ of positive (x_j) and negative ($\overline{x}_j$) *control lines* and a *target line* $t \in X \setminus C$. The Toffoli gate inverts the value of the target line iff all positive control lines are assigned 1 and all negative control lines are assigned 0. The values of all remaining lines are passed through. If the number of control lines with positive and negative polarity exceeds two the resulting reversible circuit is called a

[1] A preliminary version of this paper has appeared at IEEE International Conference on Computer Design in [34].

Multiple-controlled Toffoli gate (MPMCT). All reversible functions can be realized by a cascade of Toffoli/MPMCT gates.

In order to realize a non-reversible function, *ancillary inputs* and *garbage outputs* are employed. The target non-reversible function is embedded into the reversible circuit and executed when all ancillary inputs are assigned a pre-determined set of constants and only non-garbage outputs are considered[2].

Example 2.1. Fig. 1 is a reversible circuit that realizes a full adder. Considering the input assignment $x_1x_2x_3x_4 = 1000$, the left most gate $g_1 = TOF(\{x_1\}, x_2)$ inverts the value of the target line x_2 since the positive control line is 1. The second gate from the left $g_2 = TOF(\{x_2\}, x_3)$ inverts the value of the target line. In contrast, the gate $g_3 = TOF(\{x_1, \overline{x}_2\}, x_4)$ keeps the value of the target line x_4 since the negative control line x_2 is assigned 1. The right most gate $g_4 = TOF(\{x2, \overline{x}_3\}, x_4)$ keeps the value of its target line. The desired adder function is executed by assigning constant 0 to the ancillary input x_4. The non-garbage output y_3 produces the sum and y_4 produces the carry-out.

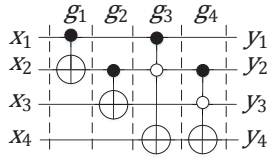


Fig. 1. Reversible circuit realizing a full adder.

3 MOTIVATION AND CONSIDERED PROBLEM

The (future) design flows for emerging technologies/applications relying on reversible logic likely will be subject to security threats similar to those encountered by conventional CMOS designs. This motivates the evaluation of security implication of reversible circuits for promising emerging technologies/applications.

In order to secure reversible circuits against third-parties, a detailed understanding on how a rogue actor would recover the target function of a reversible circuit is crucial. We consider an attacker in the foundry who has access to the gate-level implementation of the reversible circuit. Initially when the technology is rolled out, it is reasonable to assume that the distribution of the chips is strictly controlled. Hence, there is no access to the functional chips. In contrast to conventional CMOS circuits, reversible circuits usually do not explicitly realize the desired target function, but embed them into a reversible one. This process can introduce ancillary inputs and garbage outputs which may "hide" the actual target function as the functional constant assignment for all the ancillary inputs and the position of the ancillary inputs and the garbage outputs are ambiguous.

Example 3.1. Consider the circuit in Fig. 1. From the attacker perspective, each input and output is a potential ancillary input and garbage output, respectively. Without any information on the position of the ancillary inputs and their constant values as well as the position of the garbage outputs (this information is not needed to fabricate the circuit), the circuit realizes an arbitrary reversible function.

Even if an attacker in the foundry identifies the location of the ancillary inputs and the garbage outputs, she/he can still not recover the ancillary inputs values to activate the target function. Hence, also the values of the ancillary inputs naturally "hide" the target function in a reversible

[2]To this end, usually an embedding step is conducted as e.g. described in [18, 50].

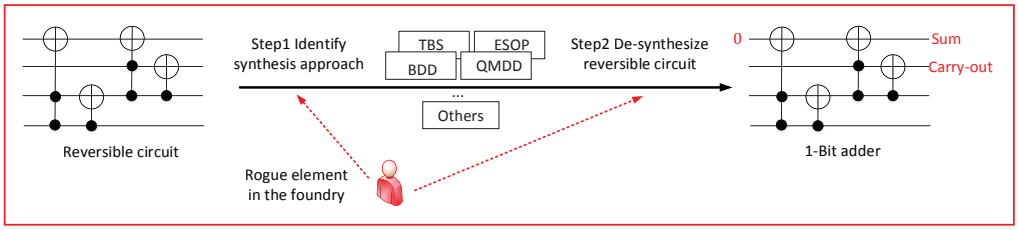


Fig. 2. Steps for recovering the target function of a reversible circuit.

circuit. However, the attacker can circumvent this obstacle if he/she knows what synthesis approach was used to generate the reversible circuit.

Once the synthesis approach is known, the attacker can reverse the synthesis process, which are implemented with the objective of minimizing the cost and reducing the run time of the tool. For example, reversible circuits generated using BDD-based synthesis approach are composed of finite set of predefined sub-circuits. Most of the sub-circuit types are associated with a unique ancillary input value. Thus, identifying the sub-circuits entitles the corresponding functional ancillary inputs value, and thus the target function. Hence, the question of how securely the target function is hidden in the embeddings boils down to the question of how hard it is to identify the synthesis approach.

This motivates the attacker to recover the target function in two steps as shown in Fig. 2. First, the attacker identifies the synthesis approach used to generate the reversible circuit. Next, he/she can obtain the ancillary inputs yielding the target function (de-synthesis step). The value and the location of the ancillary inputs and garbage outputs are determined using the synthesis approach that creates the reversible circuits. Once the synthesis approach is known, the attacker can reverse the synthesis process (as sketched above) and, by this, obtain the target function. This paper focuses on the first step of the attack.

4 TELLTALE SIGNS OF SYNTHESIS APPROACHES FOR REVERSIBLE CIRCUITS

This section reviews established and well-known synthesis approaches which realize reversible circuits for given functions. For each synthesis approach, its main ideas are sketched first. Afterwards, its main telltale signs are discussed. These signs provide the basis for our proposed telltale features and identification scheme in Section 5.

4.1 Transformation-Based Synthesis

Transformation-based synthesis (TBS) is one of the oldest and most known synthesis approaches [20]. It relies on a truth table description and rests on adding reversible gates to modify the given function until the identity-function is obtained. Each line of the truth table is traversed and gates are chosen in a way so that they do not alter already considered lines. The gates are added starting from the output side of the circuit to the input side, i.e. the output values are transformed until the identity is achieved. The following example illustrates the idea.

Example 4.1. Consider the function shown in Table 1. The first column denotes the numbers of the truth table lines, while the second and third column give the function specification, i.e. the respective input/output mappings. The algorithm starts at line 0 of the truth table. Since the input of this line is equal to the output (both are assigned to 0000 and, hence, already realize the identity), no gate has to be added. In contrast, to match the output with the input in line 1 of the truth table, the values for c and b must be inverted. Thus, two gates $g_1 = TOF(\{d\}, c)$ (1^{st} Step) and

Table 1. Transformation-Based Synthesis Steps.

line (i)	input abcd	output abcd	1^{st} abcd	2^{nd} abcd	3^{rd} abcd	4^{th} abcd	5^{th} abcd	6^{th} abcd
0	0000	0000	0000	0000	0000	0000	0000	0000
1	0001	0111	01**0**1	0**0**01	0001	0001	0001	0001
2	0010	0110	0110	0110	0**0**10	0010	0010	0010
3	0011	1001	10**1**1	1**1**11	1**0**11	**0**011	0011	0011
4	0100	0100	0100	0100	0100	0100	0100	0100
5	0101	1011	10**0**1	1**1**01	1101	1101	**0**101	0101
6	0110	1010	1010	1010	1**1**10	1110	1110	**0**110
7	0111	1101	11**1**1	1**0**11	1**1**11	**0**111	**1**111	**0**111
8	1000	1000	1000	1000	1000	1000	1000	1000
9	1001	1111	11**0**1	1**0**01	1001	1001	1001	1001
10	1010	1110	1110	1110	1**0**10	1010	1010	1010
11	1011	0001	00**1**1	0**1**11	0**0**11	**1**011	1011	1011
12	1100	1100	1100	1100	1100	1100	1100	1100
13	1101	0011	00**0**1	0**1**01	0101	0101	**1**101	1101
14	1110	0010	0010	0010	0**1**10	0110	0110	**1**110
15	1111	0101	01**1**1	0**0**11	0**1**11	**1**111	**0**111	**1**111

$g_2 = TOF(\{d\}, b)$ (2^{nd} Step) are added as depicted in Fig. 3, which have no effect on the previous truth table line. In line 2 and line 3, a $g_3 = TOF(\{c\}, b)$ gate as well as a $g_4 = TOF(\{c, d\}, a)$ gate are needed to match the values of b and a, respectively (Step 3 and Step 4). For the latter, two control lines are needed to keep the already traversed truth table lines unaltered. Afterwards, only two more gates $g_5 = TOF(\{d, b\}, a)$ (5^{th} Step) and $g_6 = TOF(\{c, b\}, a)$ (6^{th} Step) are necessary to achieve the input-output identity. The resulting circuit is shown in Fig. 3.

Circuits obtained from transformation-based synthesis employ the following telltale TBS sign.

Telltale TBS sign. A gate g_i at position i, from the input side, in a circuit obtained from *transformation-based synthesis* is very likely to have fewer (or as many) control lines than the preceding gates g_j ($j < i$), i.e it is very likely that $|C_j| \geq |C_i|$ if $i > j$. This is because, in transformation-based synthesis, the truth table lines are consecutively transformed to the identity. Thus, gates have to be applied such that no truth table line above the currently considered one is altered. The further we proceed with the synthesis algorithm, the less truth table lines can be altered by the applied gates. Control lines are added to the gates. Since the gates are applied from right to left, the gates with more control lines are likely to appear at the beginning of the circuit.

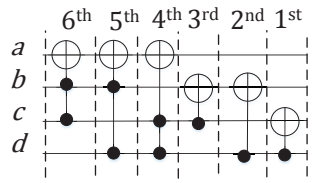


Fig. 3. Reversible circuit generated by transformation-based synthesis.

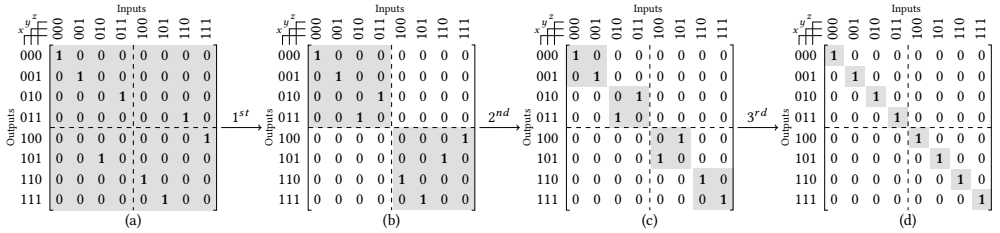


Fig. 4. QMDD-based synthesis steps.

4.2 QMDD-Based Synthesis

As for transformation-based synthesis, the main idea of QMDD-based synthesis (originally proposed in [38] and recently optimized in [49]) is to transform the function to the identity by applying reversible gates. However, instead of consecutively transforming the output patterns towards the input, the identity is established for one variable (i.e. the mapping from one bit of the input to one bit of the output) after the other. As each reversible function represents a permutation of input/output patterns, QMDD-based synthesis relies on a permutation matrix representation of the function to be synthesized. To increase the scalability of the approach, this matrix is represented by a *Quantum Multi-valued Decision Diagram* (QMDD [22][3]), which also exploits redundancies in the matrix, and thus, leads to circuits with significantly smaller costs.

In QMDD, the function to be synthesized is decomposed by its most significant variable – yielding four sub-matrices. Since the identity matrix requires the second and third quadrant to be composed of 0-entries only, Toffoli gates are applied which accordingly swap the respective matrix columns. Recursively applying this scheme to all other variables (sub-matrices) eventually yields the identity matrix and, hence, the desired circuit. The following example illustrates the idea.

Example 4.2. Consider the embedded function of a half adder as represented in Fig. 5a. Fig. 4(a) provides the corresponding permutation matrix representing the mapping from the inputs (columns) to the outputs (rows). First, this matrix shall be transformed so that the second and third quadrant are composed of 0-entries only. Thus, columns 010 and 110 must be swapped. This can be accomplished by adding a gate $TOF(\{x_2, \overline{x_3}\}, x_1)$ (x_1 is flipped when $x_2 = 1$ and $x_3 = 0$) yielding the matrix as shown in Fig. 4(b). Afterwards, the same scheme is recursively applied to the first and fourth quadrant leading to a swap of columns 100 and 110 as well as 101 and 111. Both swaps are accomplished by a single gate: $TOF(\{x_1\}, x_2)$ as in Fig. 4(c). Another recursive application of the scheme leads to a swap of columns 010 and 011 as well as 100 and 101 (accomplished by $TOF(\{\overline{x_1}, x_2\}, x_3)$ and $TOF(\{x_1, \overline{x_2}\}, x_3)$, respectively as in Fig. 4(d). The resulting reversible circuit that realizes a half adder is shown in Fig 5b.

Circuits obtained from QMDD-based synthesis employ the following telltale QMDD sign.

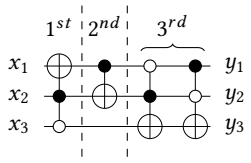
Telltale QMDD sign. In QMDD-based synthesis, one variable is transformed to the identity after the other. Therefore, the circuit can be divided into n regions where n is the number of variables. In each region, there exists a variable that occurs in each gate of the region as either a control or a target line. Furthermore, a circuit line is never used as a target line after the corresponding variable has been transformed to the identity.

Example 4.3. The reversible circuit in Fig. 5b has three regions separated by dashed lines. Each region corresponds to one variable in the identity metric.

[3]QMDDs are suited for compactly representing unitary matrices. Since a permutation matrix is a special case of a unitary matrix, this type of decision diagram is also suited for this purpose.

x_1	x_2	x_3	y_1	y_2	y_3
0	**0**	**0**	0	**0**	**0**
0	**0**	**1**	0	**0**	**1**
0	**1**	**0**	1	**0**	**1**
0	**1**	**1**	0	**1**	**0**
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	0	1	1
1	1	1	1	0	0

(a) Reversible function of a half adder.



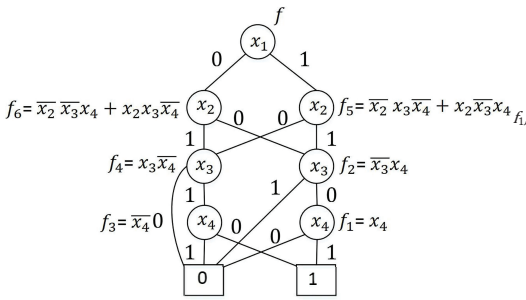
(b) Resulting reversible circuit.

Fig. 5. QMDD-based synthesis.

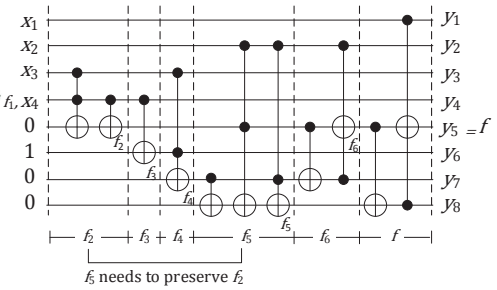
4.3 BDD-Based Synthesis

BDD-based synthesis introduced in [39] is based on *Binary Decision Diagrams* (BDDs [6]). A BDD is a directed graph $G = (V, E)$ where each terminal node represents the constant 0 or 1 and each non-terminal node represents a (sub-)function. Each non-terminal node $v \in V$ has two succeeding nodes low(v) and high(v). If v is representing the function f and labeled with the variable x_i, then the corresponding sub-functions represented by the succeeding nodes are the co-factors $f_{x_i=0}$ (low(v)) and $f_{x_i=1}$ (high(v)). Thus, a BDD naturally exposes the Shannon decomposition. Given a BDD representing a function f and its sub-functions derived by Shannon decomposition, a reversible circuit for f can be obtained by applying corresponding sub-circuits as shown by the following example.

Example 4.4. Fig. 6a shows a BDD representing the function $f = \overline{x}_1\overline{x}_2\overline{x}_3x_4 + \overline{x}_1x_2x_3\overline{x}_4 + x_1\overline{x}_2x_3\overline{x}_4 + x_1x_2\overline{x}_3x_4$ and the respective co-factors resulting from the application of the Shannon decomposition. The co-factor f_1 can easily be represented by the primary input x_4. Given the value of f_1, the co-factor f_2 can be realized by the first two gates depicted in Fig. 6b[4]. Respective sub-circuits can be added for all remaining co-factors until a circuit representing the overall function f is built. The remaining steps are shown in Fig. 6b.



(a) BDD.



(b) Resulting reversible circuit.

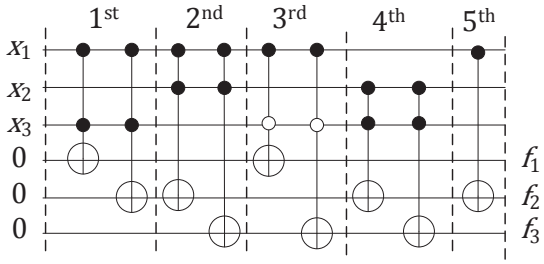
Fig. 6. BDD-based synthesis.

Circuits obtained from BDD-based synthesis employ the following telltale BDD signs.

[4]Note that an additional circuit line is added to preserve the values of x_4 and x_3 which are still needed by the co-factors f_3 and f_4, respectively.

	x_1	x_2	x_3	f_1	f_2	f_3
1st	1	-	1	1	1	0
2nd	1	1	-	0	1	1
3rd	1	-	0	1	0	1
4th	-	1	1	0	1	1
5th	1	-	-	0	1	0

(a) ESOP.



(b) Resulting reversible circuit.

Fig. 7. ESOP-based synthesis.

Telltale BDD sign 1. Circuits obtained from BDD-based synthesis consist of predefined sub-circuits for each type of the BDD nodes based on Shannon decomposition[5].

Telltale BDD sign 2. Each circuit line of a reversible circuit generated using BDD-based synthesis is used as control or target line for a small set of reversible gates, which indicates that the interference between circuit lines is low.

4.4 ESOP-Based Synthesis

ESOP-based synthesis introduced in [11] generates a reversible circuit from a Boolean function provided as *Exclusive Sum of Products* (ESOPs). ESOPs are two-level descriptions of Boolean functions that are represented as the exclusive disjunction (EXOR) of conjunctions of literals (called *products*). A *literal* is either a Boolean variable or its negation. An ESOP is the most general form of two-level AND-EXOR expressions.

Given a function $f : B^n \rightarrow B^m$, ESOP-based synthesis generates a circuit with $n + m$ lines, where the first n lines are considered as primary inputs, while the last m lines are initialized to constant 0 and considered as primary outputs. Under these design constraints, Toffoli gates are selected such that the desired function is realized. This selection exploits the fact that a single product $x_{i_1} \dots x_{i_k}$ of an ESOP description directly corresponds to a Toffoli gate with control lines $C = \{x_{i_1}, \dots, x_{i_k}\}$. In case of negative literals, NOT gates or negative control lines are applied accordingly. A circuit realizing a function given as ESOP can be derived as illustrated in the following example.

Example 4.5. Consider the function f to be synthesized as depicted in Fig. 7(a)[6]. The first product x_1x_3 affects f_1 and f_2. Hence, two Toffoli gates with target lines f_1 and f_2 and control lines $C = \{x_1, x_3\}$ are added. The third product $x_1\overline{x}_3$ includes a negative literal. Thus, the corresponding Toffoli gates have a negative control line on x_3. This procedure is continued until all products have been considered. The resulting circuit is shown in Fig. 7(b).

Circuits obtained from ESOP-based synthesis employ the following Telltale ESOP sign.

Telltale ESOP sign. The set of lines for circuits obtained from ESOP-based synthesis can be split into two distinct subsets. The lines in one subset are used as positive or negative control lines throughout the entire circuit, whereas the lines in the other subset are only used as target lines.

[5]Note that other decompositions such as positive/negative Davio can also be applied which result in different sub-circuits.
[6]The column on the left-hand side gives the products, where a "1" on the i^{th} position denotes a positive literal (i.e. x_i) and a "0" denotes a negative literal (i.e. $\overline{x}_i$), respectively. A "–" denotes that the respective variable is not included in the product. The right-hand side gives the primary output patterns.

5 RESULTING TELLTALE FEATURES AND IDENTIFICATION SCHEME

The synthesis approaches discussed above impose different telltale signs in the resulting reversible circuits, i.e. for the same target function, different reversible circuits are generated by different synthesis approaches. In this section, we provide telltale features that formalize these signs. We then propose a machine learning-based scheme that exploits these telltale features for the identification of the synthesis approach that generates a given reversible circuit. Finally, we discuss how these telltale signs can be extended so that they even work if optimizations of the considered synthesis approaches or even completely different synthesis approaches are employed.

5.1 Proposed Telltale Features

We map the telltale signs of the synthesis approaches to features that aid their identification.

5.1.1 Control Line Reduction. Control line reduction is based on the **telltale TBS sign**. This feature is used to check whether the number of control lines decreases when traversing the circuit from the inputs to the outputs. This feature outputs a ratio of reversible gates, which show a reduction in the number of control lines as follows: We count the number of reversible gates g_i where $|C_i| \geq |C_{i+1}|$ and divide it by the total number gates. The larger the value of the respective feature, the more likely the circuit has been generated using transformation-based synthesis.

5.1.2 Control-Only and Target-Only Lines. Control-only and target-only lines is a discrete feature that checks whether the **telltale ESOP sign** is satisfied. For a given reversible circuit, the proposed feature is equal to 1 if the circuit is divided into control-only and target-only lines; otherwise, the feature is equal to 0.

5.1.3 BDD Sub-Circuits-Only. In order to check whether **telltale BDD sign 1** holds in a reversible circuit, we propose BDD sub-circuits-only feature. The proposed discrete feature evaluates to 1 if a reversible circuit consists of only BDD sub-circuits and 0 otherwise.

5.1.4 Cone-Structural Analysis. Cone-structural analysis exploits **telltale BDD sign 2** to identify reversible circuits generated using BDD-based synthesis. Our proposed analysis considers the logic cone driven by each pair of circuit lines to compute the number of gates reachable from each pair of the circuit lines. The more common reversible gates driven by the pair of circuit lines, the more circuit lines interfere with each others and, thus, the less likely that this circuit has been generated by BDD-based synthesis.

The sum of the reversible gates that each pair of circuit lines converge at quantifies the interference among circuit lines. This value, upon normalization provides the Convergent Ratio (CR), which denotes the interference between circuit lines in terms of the number of reversible gates, i.e.

$$CR = Normalize\{\sum_{i,j} |gate(i) \cap gate(j)|\}, \tag{1}$$

where i and j refer to different inputs (lines) of the reversible circuit and $gate(i)$ refers to the set of reversible gates driven by input (line) i. Normalization indicates a division of the summation of the number of reversible gates that each pair of circuit lines converge at by the number of all input pairs times the total number of gates.

A smaller convergent ratio indicates a low interference between the circuit lines, which suggests that BDD-based synthesis has been used to generate the reversible circuit. This cone-structural analysis can distinguish reversible circuits generated using BDD-based synthesis from other synthesis approaches such as QMDD- and transformation-based synthesis, in which circuit lines heavily interact with each other.

5.1.5 QMDD Identity Transformation. Another discrete feature that assists the identification of the synthesis approach of a given reversible circuit is the QMDD Identity Transformation feature that determines whether a reversible circuit is generated by QMDD-based synthesis based on the **telltale QMDD sign**. We propose a two-phase algorithm to compute the QMDD identity transformation feature in a top-down approach as shown in Algorithm 1. The reversible circuit is scanned first to determine the order of the variables in the QMDD identity feature. The reversible circuit is scanned again to check whether the circuit employs the desired structure. We first consider the top most variable x_i in the identity matrix. We search for the last gate (starting from the beginning of the circuit) in which x_i occurs as a target line $LG(x_i)$. All gates before $LG(x_i)$ must have x_i as a control or target line. If any of the gates violates this rule, we consider that the circuit is not generated using QMDD-based synthesis, and thus our feature evaluates to 0. Otherwise it is set to 1. The same process is applied to the remaining variables in the identity matrix.

ALGORITHM 1: QMDD Identity Transformation Feature.

Input: Reversible circuit
Output: 1 (QMDD-based synthesis) or 0 (Others)

LG(x_i) is the last occurred reversible gate with target line x_i.;
for *each variable (circuit line)* x_i **do**
  report LG(x_i)
end
Reorder circuit variables (lines) starting from x_i with first LG(x_i) to x_j with last LG(x_j) in the reversible circuit.;
for *each variable* x_i **do**
  for *each reversible gate* g_j *between* $LG(x_{i-1})$ *and* $LG(x_i)$ **do**
    if $x_i \notin C_j$ *(control line) and* $x_i \notin T_j$ *(target line)* **then**
      return 0
    end
  end
end
return 1

5.2 Machine Learning-Based Scheme

We exploit machine learning models to identify the synthesis approach that realizes a given reversible circuit. We use the **control line reduction, control-only and target-only lines, BDD sub-circuits-only, cone-structural analysis,** and **QMDD identity transformation** features which are employed on different machine learning models, namely the decision tree [25], the random forest [14], the support vector machine [8], and the logistic regression [13] models. Three of the telltale features are discrete features, while the other two are continuous features. As for training data, we utilized supervised learning using reversible circuits from which it was known how they have been synthesized. First, we extract the five telltale features from the reversible circuits with known synthesis approach (training data) to build the machine learning model. Next, we extract the five telltale features from unknown reversible circuits (test data) and apply them to the proposed machine learning model to predict the corresponding synthesis approach.

5.3 Telltale Features for Optimizations or Other Synthesis Approaches

Reversible circuit synthesis approaches as considered above are frequently optimized. This of course may affect their telltale signs and thus features. In order to evaluate whether and, if yes,

how these optimized synthesis approaches affect the proposed machine learning-based scheme, optimization of these synthesis approaches as well as corresponding telltale signs are additionally considered in this work. More precisely:

- For ESOP-based synthesis, we allow shared target functions as proposed in [35]. Thus, target lines can also be used as control lines later in the circuits. Consequently, the telltale ESOP sign of the circuit has to be adjusted.
- For BDD-based synthesis, we allow complement edges in the BDD to get a more compact representation (as proposed in [40]). Thus, more pre-defined sub-circuits have to be considered. Consequently, the telltale BDD sign 1 should be extended to include the additional pre-defined sub-circuits.
- For TBS, we consider a bidirectional synthesis approach as optimization (as e.g. proposed in [20]). Gates can be added to the circuit at the beginning or at the end of the circuit depending on which gate is cheaper, and thus, affecting the telltale TBS sign. As a result, the control line reduction ratio of circuits generated using TBS may be reduced.
- For QMDD-based synthesis, we consider the optimization scheme discussed in [49] which exploits more redundancies in the nodes and paths of the QMDDs. While the resulting reversible circuits are significantly smaller, they still maintain the same telltale sign as the ones generated by QMDD-based synthesis.

We refined the proposed telltale features of reversible circuits to enable the identification of the synthesis approach that generates a reversible circuit, even if the optimization outlined above are applied. The control-only and target-only lines feature outputs a ratio of circuit lines that are either control only or target only lines, which is expected to be high. The BDD sub-circuits-only feature is modified to include new types of BDD sub-circuits. The other features will remain intact.

Besides that, in a similar fashion telltale features can be revised, extended, or newly obtained for completely other synthesis approaches as well. For example, the solution proposed in [37] generates reversible circuits where the number of control lines from the inputs to the outputs and from the outputs to the inputs are expected to exhibit partial increments – both within groups/regions of the reversible circuits. Thus, two telltale features can be added to identify this synthesis approach, which are the ratio of reversible gates where $|C_{i+1}| \geq |C_i|$ and the ratio of the reversible gates where $|C_i| \geq |C_{i+1}|$. Based on these values, a likely identification is possible. Similarly, telltale features for other synthesis approaches can be derived[7].

6 EXPERIMENTAL EVALUATION

In this section, we evaluate the effectiveness of the proposed machine learning-based scheme. To this end, we considered 645 different reversible circuits for functions provided in RevLib [42]. The reversible circuits are generated using different synthesis approaches and, afterwards, used in the decision tree, the random forest, the support vector machine, and the logistic regression machine learning models for the identification of the synthesis approach of a given reversible circuit. More precisely, our support vector machine is based on a radial basis function kernel. The models are trained and tested with reversible circuits generated using four synthesis approaches, namely ESOP, BDD, QMDD, and TBS using 10-fold cross validation. 90% of the reversible circuits serve as training data, while the remaining 10% serve as test data.

Table 2 shows the resulting identification accuracy of the original (non-optimized) synthesis approaches for the individual machine learning models. For each model, we list the classification results in a table where the rows denote the actual synthesis approach used to generate the circuit

[7]Please note that our evaluations later in Section 6 focuses on the synthesis approaches considered above.

Table 2. Identifying the Original Synthesis Approach.

(a) Random Forest.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	98.4	0.0	0.8	0.8
ESOP	0.0	91.4	0.5	8.1
QMDD	0.0	1.7	86.8	11.5
TBS	3.4	2.4	6.4	87.8

Overall accuracy: 90.5%

(b) Decision Tree.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	96.9	0.0	0.5	2.6
ESOP	0.0	91.4	1.3	7.3
QMDD	0.5	1.3	86.8	11.4
TBS	1.9	7.1	9.2	81.8

Overall accuracy: 88.7%

(c) Logistic Regression.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	86.6	0.0	0.5	12.9
ESOP	0.5	81.1	4.6	13.8
QMDD	0.0	0.0	85.8	14.2
TBS	0.0	0.0	1.3	98.7

Overall accuracy: 87.0%

(d) Support Vector Machine.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	90.4	0.0	0.5	9.1
ESOP	0.0	80.7	5.7	14.6
QMDD	0.0	0.0	87.0	13.0
TBS	2.5	0.0	1.7	95.8

Overall accuracy: 88.0%

and the columns list the predicted synthesis approaches using our machine learning models. The diagonal of each table indicates the percentage of correctly identified reversible circuits generated using the corresponding synthesis approach. For example, the random forest model classifies 98.4% of all circuits that where generated using BDD-based synthesis correctly, while 0.8% of those were classified as circuits generated using QMDD-based synthesis or ESOP-based synthesis, respectively (as can be seen in the first row of Table 2a). Furthermore, we list the overall accuracy for each model underneath the respective table.

As can be seen in Table 2, the random forest machine learning model reaches the highest accuracy by identifying the correct synthesis approach in 90.5% of all cases. However, the other models reach a similar accuracy ranging from 87.0% to 88.7%. These results clearly confirm that the proposed methods can indeed with reasonable accuracy identify the respectively used synthesis scheme for the vast majority of the cases.

However, for all models, we face problems distinguishing QMDD-based synthesis and TBS for a small set of reversible circuits, since these two approaches are rather similar. The difference is that the TBS transforms the logic function to the identity pattern by pattern, while QMDD-based synthesis conducts this task variable by variable. Consequently, it is not surprising that similar circuits result in certain cases.

In a second series of experiments, we apply the proposed machine learning-based scheme for the optimized reversible circuit synthesis approaches using the modified telltale features as explained in Section 5.3. Table 3 provides the resulting identification accuracy for this case.

As can be seen in Table 3, the random forest model again reaches the highest accuracy (i.e. 91.9%). This accuracy is even higher than in the first series of experiments. Furthermore, the accuracy of the logistic regression model as well as the support vector machine significantly decrease to 85.6%.

For the machine learning models that reach an average accuracy above 90% (i.e. random forest and decision tree), we can also observe that the problem of distinguishing QMDD-based synthesis and TBS decreases when taking the optimized synthesis approaches into account. In fact, the number of incorrectly classified circuits generated by QMDD-based synthesis drop from 11.5% and 11.4% to 5.5% and 5.0%, respectively by using the refined telltale signs. However, the overall

Table 3. Identifying the Optimized Synthesis Approach.

(a) Random Forest.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	96.6	0.0	1.1	2.3
ESOP	0.0	87.7	3.3	9.0
QMDD	0.0	5.5	89.3	5.2
TBS	2.5	1.2	3.4	93.9

Overall accuracy: 91.9%

(b) Decision Tree.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	99.0	0.0	0.5	0.5
ESOP	0.0	88.1	4.9	7.0
QMDD	1.3	5.0	89.0	4.7
TBS	2.7	3.2	6.3	87.8

Overall accuracy: 90.5%

(c) Logistic Regression.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	86.0	0.0	1.9	12.1
ESOP	0.0	89.3	2.6	8.1
QMDD	0.0	10.9	77.6	11.5
TBS	0.0	0.0	1.6	98.4

Overall accuracy: 85.6%

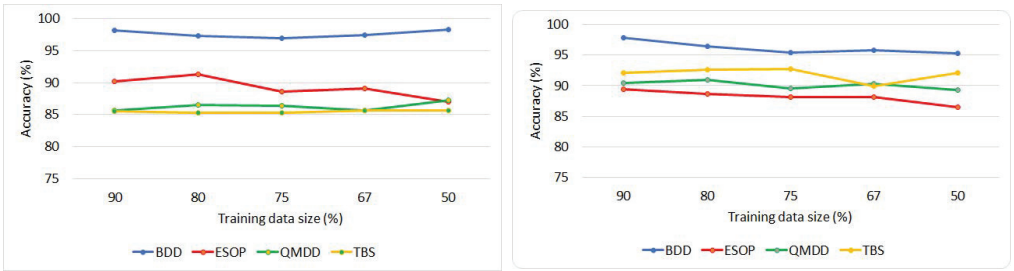
(d) Support Vector Machine.

Synth. with \ Classified as [%]	BDD	ESOP	QMDD	TBS
BDD	87.3	0.0	0.6	12.1
ESOP	0.0	89.2	1.4	9.4
QMDD	0.0	13.0	76.9	10.1
TBS	0.0	0.0	4.1	95.9

Overall accuracy: 85.6%

accuracy of these models only improves slightly, since it is more complicated to distinguish circuits generated by ESOP-based synthesis and QMDD-based synthesis under optimization.

Finally, we evaluate the effect of the size of training data on the accuracy of the machine learning based identification scheme for both optimized and original (non-optimized) synthesis approaches. We reduce the percentage of the training data to 80%, 75%, 67%, and 50% of the total number reversible circuits and run the random forest model, which yields the highest accuracy. The obtained results for original and optimized synthesis approaches are provided in Fig. 8a and Fig. 8b, respectively. The graphs show that the reduction in the size of the training data can slightly affect the identification accuracy, which indicates that the proposed metrics are highly representative for corresponding reversible circuit properties under different synthesis approaches.



(a) (b)

Fig. 8. Accuracy under different sizes of training data for (a) original and (b) optimized synthesis approaches.

Overall, our experimental evaluation confirms that the proposed approach is capable of identifying with reasonable accuracy the synthesis approaches of reversible circuits with dedicated telltale features that can be used by different machine learning algorithms. Moreover, the experiments have shown that a refinement of these telltale signs to also support optimized versions of the synthesis approaches is possible and results in an even higher overall classification accuracy.

7 DISCUSSION

To show the feasibility of the assumed attack model, and how it impacts the current design flow, we provide further discussions of it in the form of questions and answers.

How realistic is the assumed threat model with respect to the different applications of reversible circuits?

The fabrication process of reversible circuits may significantly differ depending on the target technology. For example, reversible circuits used for encoding and decoding devices, adiabatic computations, or low-power designs have similar fabrication processes (and, hence, threats) as CMOS chips, while reversible circuits used in quantum computation are not necessarily applicable to this threat. In this work, we consider reversible circuits in general as the respective security issues can occur in many of the possible technologies such as encoding and decoding devices, adiabatic computation, or low-power designs. Our future work will target revealing the IP of quantum circuits in which gates are not physically realized but employed through dedicated manipulations of qubits.

Does the attacker have access to functional inputs and outputs of the reversible circuit?

We assume that the attacker has no access to a functional reversible chip. This is a valid assumption for reversible circuits utilized for non-commercial applications such as military applications [19]. We further assume that manufacturing test is conducted in either a separate test facility [44] or in the foundry itself. In the later one, we consider random values assigned to the ancillary inputs during manufacturing test. It has been shown that conducting the test prior to activating an IC, which is obfuscated using additional inputs (key) improves the security and yields better test quality [47]. Thus, in our threat model, the attacker has no access to functional inputs and outputs of the reversible circuit.

Can the adversary distinguish between the garbage and the primary outputs through the routing information?

We consider two scenarios. If the chip is on the motherboard, an attacker in the foundry can utilize the peripheral circuits to distinguish between the primary and the garbage outputs of the reversible circuits. Thus, the attacker needs to recover the ancillary inputs value only. On the other hand, if the chip design is sent alone to a foundry without peripheral circuits to mitigate attacks by an untrusted foundry, the attacker has to identify the location of the ancillary inputs and garbage outputs in addition to the value of the ancillary inputs to determine the target function. In both scenarios, identifying the synthesis approach that generates a reversible circuit is very important to recover the target function.

8 CONCLUSION

In this paper, we show how the attacker can determine the synthesis approach that generates a reversible circuit as a first step towards recovering the target function. We extract the telltale signs of well-established synthesis approaches and then utilize their corresponding telltale features in the proposed machine learning-based scheme to reveal the respectively applied synthesis approach. Experimental results show that, using the proposed machine learning-based scheme, an attacker can determine the synthesis approach of a given reversible circuit in the vast majority of cases. With this information on hand, an attacker can reverse the synthesis approach to obtain the target function of the reversible circuit. The method proposed here can easily be extended for further synthesis approaches to be supported. To this end, the telltale signs of those synthesis approaches should be determined and described in terms of new features.

The contributions of this paper provide the basis for much future work. Once the synthesis approach is known, the next step is to explore the locations and the value of the ancillary inputs as

well as the location of the garbage outputs (see the second step in Fig. 2). For approaches such as BDD-based synthesis or ESOP-based synthesis, this often can already be derived once the synthesis approach is known [32, 33]. For other synthesis approaches, details on how the embedding has been conducted is also required [33]. Besides that, this work also motivates the need for obfuscation countermeasures since, as shown by the results, the respective information can easily be derived for the majority of the considered circuits.

ACKNOWLEDGEMENTS

The 3rd and 4th authors are supported by the EU COST Action IC1405. The 5th author is partly funded by NYU/NYU-AD CCS.

REFERENCES

[1] 2019. Special issue: Design of Reversible Computing Systems. In *IEEE Transactions on Emerging Topics in Computing*.
[2] L. Amaru, P. E. Gaillardon, R. Wille, and G. De Micheli. 2016. Exploiting inherent characteristics of reversible circuits for faster combinational equivalence checking. In *Proceedings of Design, Automation Test in Europe Conference Exhibition*. 175–180.
[3] W. C. Athas and L. J. Svensson. 1994. Reversible logic issues in adiabatic CMOS. In *Proceedings of Workshop on Physics and Computation*. 111–118.
[4] C. H. Bennett. 1973. Logical reversibility of computation. *IBM J.Res.Dev* 17, 6 (Nov 1973), 525–532.
[5] A. Berut, A. Arakelyan, A. Petrosyan, S. Ciliberto, R. Dillenschneider, and E. Lutz. 2012. Experimental verification of Landauer's principle linking information and thermodynamics. *Nature* 483 (2012), 187–189.
[6] R. E. Bryant. 1986. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computer* 35, 8 (Aug. 1986), 677–691.
[7] L.W. Chow, J.P. Baukus, B.J. Wang, and R.P. Cocchi. 2012. Camouflaging a standard cell based integrated circuit. Google Patents.
[8] N. Cristianini and J. Shawe-Taylor. 2000. *An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods*. Cambridge University Press.
[9] R. Cuykendall and D. R. Andersen. 1987. Reversible optical computing circuits. *OPTICS LETTERS* 12, 7 (1987), 542–544.
[10] S. J. Devitt and I. Lanese (Eds.). 2016. *Reversible Computation - 8th International Conference, RC 2016, Bologna, Italy, July 7-8, 2016, Proceedings*. Lecture Notes in Computer Science, Vol. 9720. Springer.
[11] K. Fazel, M.A. Thornton, and J.E. Rice. 2007. ESOP-based Toffoli gate cascade generation. In *Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*. 206 –209.
[12] M. P. Frank. 2017. The Future of Computing Depends on Making It Reversible. *IEEE Spectrum* (2017).
[13] D. A. Freedman. 2009. *Statistical Models: Theory and Practice*. Cambridge University Press.
[14] T. K. Ho. 1995. Proceedings of International Conference on Document Analysis and Recognition. In *ICDAR*, Vol. 1. 278–282 vol.1.
[15] T. Hogg, M. S. Moses, and D. G. Allis. 2017. Mechanical Computing Systems Using Only Links and Rotary Joints. *Molecular Systems Design & Engineering* 2, 3 (2017), 235–252.
[16] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor. 2010. Trustworthy hardware: identifying and classifying hardware trojans. *Computer* 43, 10 (Oct 2010), 39–46.
[17] R. Landauer. 1961. Irreversibility and heat generation in the computing process. *IBM J.Res.Dev* 5, 3 (1961), 183–191.
[18] D. Maslov and G. W. Dueck. 2004. Reversible cascades with minimal garbage. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 23, 11 (Nov 2004), 1497–1509.
[19] M. El Massad, J. Zhang, S. Garg, and M. V. Tripunitara. 2017. Logic locking for secure outsourced chip fabrication: a new attack and provably secure defense mechanism. *CoRR* abs/1703.10187 (2017).
[20] D. M. Miller, D. Maslov, and G. W. Dueck. 2003. A transformation based algorithm for reversible logic synthesis. In *Proceedings of the ACM/ESDA/IEEE Design Automation Conference*. 318–323.
[21] M. Nielsen and I. Chuang. 2000. *Quantum Computation and Quantum Information*. Cambridge Univ. Press.
[22] P. Niemann, R. Wille, D. M. Miller, M. A. Thornton, and R. Drechsler. 2016. QMDDs: efficient quantum function representation and manipulation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35, 1 (2016), 86–99.
[23] M. Pecht and S. Tiku. 2006. Bogus: electronic manufacturing and consumers confront a rising tide of counterfeit electronics. *IEEE Spectrum* 43, 5 (2006), 37–46.

[24] I. Phillips and H. Rahaman (Eds.). 2017. *Reversible Computation - 9th International Conference, RC 2017, Kolkata, India, July 6-7, 2017, Proceedings*. Lecture Notes in Computer Science, Vol. 10301. Springer.

[25] J. R. Quinlan. 1986. Induction of decision trees. *Mach. Learn.* 1, 1 (March 1986), 81–106.

[26] Jeyavijayan Rajendran, Michael Sam, Ozgur Sinanoglu, and Ramesh Karri. 2013. Security analysis of integrated circuit camouflaging. In *Proceedings of the ACM SIGSAC Conference on Computer & Communications Security*. 709–720.

[27] J. Rajendran, O. Sinanoglu, and R. Karri. 2013. Is split manufacturing secure?. In *Proceedings of Design, Automation Test in Europe Conference Exhibition*. 1259–1264.

[28] J. Rajendran, O. Sinanoglu, and R. Karri. 2013. VLSI testing based security metric for IC camouflaging. In *Proceedings of IEEE International Test Conference*. 1–4.

[29] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri. 2015. Fault analysis-based logic encryption. *IEEE Trans. Comput.* 64, 2 (2015), 410–424.

[30] A. Rauchenecker, T. Ostermann, and R. Wille. 2017. Exploiting reversible logic design for implementing adiabatic circuit. In *Proceedings of International Conference "Mixed Design of Integrated Circuits and Systems*.

[31] J. A. Roy, F. Koushanfar, and I. L. Markov. 2010. Ending piracy of integrated circuits. *Computer* 43, 10 (2010), 30–38.

[32] S. M. Saeed, X. Cui, R. Wille, A. Zulehner, K. Wu, R. Drechsler, and R. Karri. 2017. Towards reverse engineering reversible logic. *CoRR* abs/1704.08397 (2017).

[33] S. M. Saeed, X. Cui, R. Wille, A. Zulehner, K. Wu, R. Drechsler, and R. Karri. 2018. IC/IP Piracy Assessment of Reversible Logic. In *Proceedings of International Conference on Computer-Aided Design*.

[34] S. M. Saeed, N. Mahendran, A. Zulehner, R. Wille, and R. Karri. 2017. Identifying reversible circuit synthesis approaches to enable IP piracy attacks. In *Proceedings of IEEE International Conference on Computer Design*. 537–540.

[35] Y. Sanaee and G. W. Dueck. 2009. Generating Toffoli networks from ESOP expressions. In *Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*. IEEE, 715–719.

[36] P. W. Shor. 1997. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* 26, 5 (Oct. 1997), 1484–1509.

[37] M. Soeken, G. W. Dueck, M. M. Rahman, and D. M. Miller. 2016. An extension of transformation-based reversible and quantum circuit synthesis. In *Proceedings of IEEE International Symposium on Circuits and Systems*. 2290–2293.

[38] M. Soeken, R. Wille, C. Hilken, N. Przigoda, and R. Drechsler. 2012. Synthesis of reversible circuits with minimal lines for large functions. In *Proceedings of Asia and South Pacific Design Automation Conference*. 85–92.

[39] R. Wille and R. Drechsler. 2009. BDD-based Synthesis of Reversible Logic for Large Functions. In *Proceedings of the ACM/ESDA/IEEE Design Automation Conference*. 270–275.

[40] R. Wille and R. Drechsler. 2010. Effect of BDD Optimization on Synthesis of Reversible and Quantum Logic. *Electronic Notes in Theoretical Computer Science* 253, 1 (March 2010), 57–70.

[41] R. Wille, R. Drechsler, C. Osewold, and A. García Ortiz. 2012. Automatic design of low-power encoders using reversible circuit synthesis. In *DATE*. 1036–1041.

[42] R. Wille, D. Grosse, L. Teuber, G. W. Dueck, and R. Drechsler. 2008. RevLib: An Online Resource for Reversible Functions and Reversible Circuits. In *Proceedings of International Symposium on Multiple Valued Logic*. 220–225.

[43] R. Wille, O. Keszocze, S. Hillmich, M. Walter, and A. García Ortiz. 2016. Synthesis of approximate coders for on-chip interconnects using reversible logic. In *DATE*. 1140–1143.

[44] B. Wire. 2014. Research and Markets: Outsourced Semiconductor Assembly and Test Market (OSAT) Trends. http://www.businesswire.com/news/home/20140324005628/en/Research-Markets-Outsourced-Semiconductor-Assembly-Test-Market

[45] W. Wustmann and K. Osborn. 2017. Reversible Fluxon Logic: Topological particles allow gates beyond the standard adiabatic limit. In *arXiv:1711.04339*.

[46] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri. 2016. On improving the security of logic locking. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 35, 9 (2016), 1411–1424.

[47] M. Yasin, S. M. Saeed, J. Rajendran, and O. Sinanoglu. 2016. Activation of logic encrypted chips: Pre-test or post-test?. In *Proceedings of Design, Automation and Test in Europe Conference Exhibition*. 139–144.

[48] A. Zulehner, M. Frank, and R. Wille. 2019. Design Automation for Adiabatic Circuits. In *Proceedings of Asia and South Pacific Design Automation Conference*.

[49] A. Zulehner and R. Wille. 2017. Improving synthesis of reversible circuits: exploiting redundancies in paths and nodes of QMDDs. In *International Conference on Reversible Computation*. Springer, 232–247.

[50] A. Zulehner and R. Wille. 2017. Make It Reversible: Efficient Embedding of Non-reversible Functions. In *Proceedings of Design, Automation and Test in Europe Conference Exhibition*.

[51] A. Zulehner and R. Wille. 2017. Taking One-to-one Mappings for Granted: Advanced Logic Design of Encoder Circuits. In *Proceedings of Design, Automation and Test in Europe Conference Exhibition*.