IC/IP Piracy Assessment of Reversible Logic

Samah Mohamed Saeed
City University of New York
New York, USA
ssaeed@ccny.cuny.edu

Xiaotong Cui
Chongqing University
Chongqing, China
xiaotong.sd@gmail.com

Alwin Zulehner, Robert Wille
Johannes Kepler University
Linz, Austria
{azulehner,robert.wille}@jku.at

Rolf Drechsler
University of Bremen/DFKI
Bremen, Germany
drechsler@uni-bremen.de

Kaijie Wu, Ramesh Karri
New York University
New York, USA
kaijie@gmail.com,rkarri@nyu.edu

ABSTRACT

Reversible logic is a building block for adiabatic and quantum computing in addition to other applications. Since common functions are non-reversible, one needs to embed them into proper-size reversible functions by adding ancillary inputs and garbage outputs. We explore the Intellectual Property (IP) piracy of reversible circuits. The number of embeddings of regular functions in a reversible function and the percent of leaked ancillary inputs measure the difficulty of recovering the embedded function. To illustrate the key concepts, we study reversible logic circuits designed using reversible logic synthesis tools based on Binary Decision Diagrams and Quantum Multi-valued Decision Diagrams.

KEYWORDS

Reversible logic, IC/IP piracy, Security, Number of embeddings, BDD, QMDD, Ancillary inputs, Garbage outputs.

1 INTRODUCTION

The globalization of the design flow of Integrated Circuits (ICs) introduces security vulnerabilities. Intellectual Property (IP) piracy reveals design details to a fab, and other malicious parties at the different stages in the supply chain [1, 2]. Knowledge of the design can allow one to identify sensitive parts of the design, to make malicious modifications, and to co-opt the IP.

Reversible circuits implement bijective $n \times n$ functions that map each possible input to a unique output. Reversible circuits can be used for quantum computing [3], adiabatic computing [4–7], encoder/decoder design [8–10], and optical computing [11, 12]. In the near future, reversible computing is expected to be an alternative design methodology for low-power circuits [13]. A recent summary argues that the future of computing depends on "making it reversible" [13]. Efforts are underway to realize low-power reversible circuits [14, 15].

Reversible circuits differ from conventional circuits and hence have different constraints vis-a-vis IC/IP piracy attacks and defenses. Reversible circuits embed the originally intended function (which might not be reversible) into a reversible one. This introduces so-called ancillary inputs and garbage outputs, which already might obfuscate the function from an attacker's perspective. In this paper, we provide IC/IP piracy assessment of reversible logic. We assume that the design-manufacture-test flow for reversible circuits is similar to that for conventional circuits – exposing them to similar risks.

1.1 Related Work

Researchers have considered the IC/IP piracy problem and developed Design-for-Trust techniques to thwart IC/IP piracy in CMOS-based logic circuits. Logic obfuscation hides the implementation and the functionality of the design by including additional gates, which are controlled by a secret key [16–18]. IC camouflaging is a layout-level protection from a malicious end user preventing her from extracting the gate-level implementation of the design [19–21]. In camouflaging, layouts of all standard logic gates are designed to look alike. These techniques target conventional CMOS.

A recent study provides the first step in launching IP piracy attacks on reversible circuits [22]. Synthesis approaches employed to generate reversible circuits have telltale clues in the circuits, which can expose the synthesis approach. Hardware Trojans are another security vulnerability in the IC supply chain. Hardware Trojans in reversible circuits have been considered in [23], where the difficulty of detecting Trojans inserted into reversible circuits is assessed. However, thus far, no assessment of the difficulty of recovering the functionality of a reversible circuit has been conducted yet. This is considered in the following.

The paper is organized as follows. Section 2 provides the background on reversible logic. The motivation, threat model, and our analysis of the number of embeddings are described in Section 3. IP piracy of reversible circuits is analyzed in Section 4. Results in Section 5 demonstrate the difficulty of recovering functions embedded into reversible circuits and report the number of possible embeddings faced by an attacker. We conclude with remarks on the feasibility of this problem and the threat model in Section 6.

2 BACKGROUND

We provide an overview of reversible logic and synthesis approaches to embed a target function into a reversible circuit.

2.1 Reversible Logic

A reversible circuit implements a bijection wherein a computation is performed in both directions (i.e. inputs → outputs and vice-versa).

Reversible circuits are implemented as cascades of reversible gates. Each reversible gate over the inputs $X = \{x_1, \ldots, x_n\}$ consists of a (possibly empty) set $C_i \subseteq \{x_j \mid x_j \in X\} \cup \{\overline{x}_j \mid x_j \in X\}$ of positive (x_j) and negative ($\overline{x}_j$) *control lines* and a set $T \subset X \setminus C$ of *target lines*. The most commonly used reversible gate is the *Toffoli* gate $TOF(C, x_t)$ [24]. $TOF(C, x_t)$ consists of a target line $x_t \in X \setminus C$

whose value is inverted if all values on the positive (negative) control lines are set to 1 (0) or if $C = \emptyset$. All other lines are unaltered.

To realize non-reversible functions, *ancillary inputs* and *garbage outputs* are used. An ancillary input of a reversible circuit is an input that is set to a fixed value (0 or 1). A garbage output of a reversible circuit is a don't care for all inputs.

Example 1. *Figure 1 shows a full adder design using reversible gates. The top ancillary input of the circuit is set to 0. The bottom two outputs are don't care garbage outputs. For the input $x_1x_2x_3x_4 = 0100$, the leftmost gate $g_1 = TOF(\{x_3, x_4\}, x_1)$ passes the value on the target line x_1 unaltered since the two positive control lines are 0.*

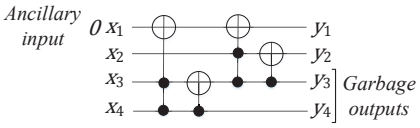

Figure 1: Reversible logic implementation of a full adder.

2.2 Reversible Logic Synthesis

Several approaches to reversible logic synthesis have been proposed [25–28]. These approaches either implicitly [25, 26] or explicitly [27, 28] *embed* a function into a proper-size reversible function.

Example 2. *Consider the full adder in Table 1(a). The full adder has the carry in c_{in} and summands x and y as the inputs and the carry out c_{out} and the* sum *as the outputs. The full adder is not reversible since (1) the number of inputs is not equal to the number of outputs and (2) there is no unique input-output mapping. Adding an additional output to the function does not make it reversible. The first four rows of the truth table can be embedded with respect to reversibility as shown in the rightmost column of Table 1(a). However, since $c_{out} = 0$ and sum $= 1$ appear twice (marked in bold), a unique embedding for the fifth row of the truth table is not possible. The same holds for the italicized rows. One possible embedding of the full adder is shown in Table 1(b). Here, the full adder is obtained if the ancillary input is set to 0. g_1 and g_2 are the garbage outputs.*

Table 1: Embedding a full adder into 4x4 reversible logic.

(a) Full Adder

c_{in}	x	y	c_{out}	sum	
0	0	0	0	0	0
0	0	1	**0**	**1**	**0**
0	1	0	**0**	**1**	**1**
0	1	1	*1*	*0*	*0*
1	0	0	**0**	**1**	**?**
1	0	1	*1*	*0*	*1*
1	1	0	*1*	*0*	*?*
1	1	1	1	1	1

(b) Embedding

0	c_{in}	x	y	c_{out}	sum	g_1	g_2
0	**0**	**0**	**0**	**0**	**0**	0	0
0	**0**	**0**	**1**	**0**	**1**	1	1
0	**0**	**1**	**0**	**0**	**1**	1	0
0	**0**	**1**	**1**	**1**	**0**	0	1
0	**1**	**0**	**0**	**0**	**1**	0	0
0	**1**	**0**	**1**	**1**	**0**	1	1
0	**1**	**1**	**0**	**1**	**0**	1	0
0	**1**	**1**	**1**	**1**	**1**	0	1
1	0	0	0	1	0	0	0

In the following, we briefly review the reversible logic synthesis approaches considered for IC/IP piracy assessment.

2.2.1 BDD-based Reversible Logic Synthesis. BDD-based synthesis introduced in [26] embeds the function implicitly. The function is specified as a *Binary Decision Diagram (BDD)* [29]. A BDD is a directed acyclic graph $G = (V, E)$ where a Shannon decomposition

$$f = \overline{x}_i \cdot f_{x_i=0} + x_i \cdot f_{x_i=1}$$

is carried out in each node $v \in V$. The function $f_{x_i=0}$ ($f_{x_i=1}$) is the negative (positive) co-factor of f obtained by assigning x_i to 0 (1). In this paper, the node representing $f_{x_i=0}$ ($f_{x_i=1}$) is denoted by low(v) (high(v)), while x_i is the select variable.

Example 3. *Figure 2 shows a BDD representing $f = \overline{x}_1\overline{x}_2\overline{x}_3x_4 + \overline{x}_1x_2x_3\overline{x}_4 + x_1\overline{x}_2x_3\overline{x}_4 + x_1x_2\overline{x}_3x_4$ and the respective co-factors using Shannon decomposition.*

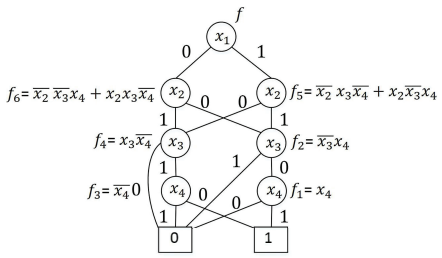

Figure 2: BDD of $f = \overline{x}_1\overline{x}_2\overline{x}_3x_4 + \overline{x}_1x_2x_3\overline{x}_4 + x_1\overline{x}_2x_3\overline{x}_4 + x_1x_2\overline{x}_3x_4$

Given a BDD $G = (V, E)$ of a function, a reversible circuit can be derived. All nodes $v \in V$ of G are traversed depth-first and substituted with a cascade of reversible gates. The cascade of gates depends on the successors of the node v. Figure 3 shows the mapping between different nodes in the BDD and the reversible sub-circuits. This process requires an ancillary circuit line in order to realize the non-reversible decomposition employed in a node. To obtain a reversible circuit realizing f, the entire BDD is traversed.

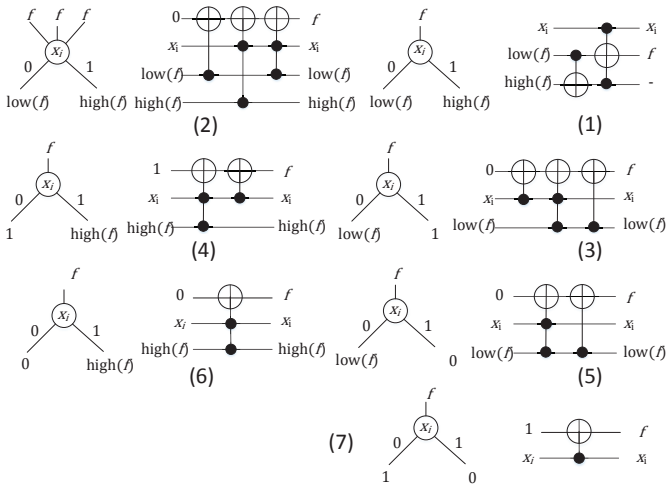

Figure 3: Reversible cascades representing Shannon decomposition of BDD. High(f) (low(f)) indicates the value of function f when the input x_i is set to 1 (0).

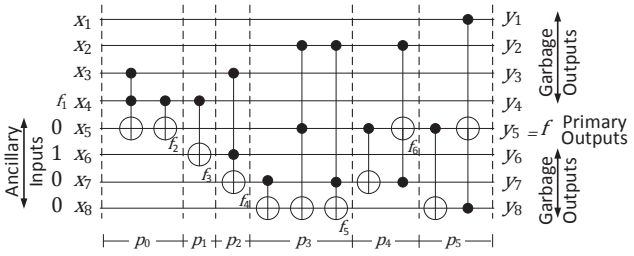

Figure 4: A reversible circuit derived from the BDD of function f, where p_i is the i_{th} partition of the reversible circuit.

Example 4. *Consider the BDD from Figure 2. The co-factor f_1 can be represented by the primary input x_4. The co-factor f_2 can be realized by the first two gates in Figure 4*[1]*. This way sub-circuits of all the co-factors are composed to realize the function f. The remaining steps are shown in Figure 4.*

2.2.2 QMDD-based Reversible Logic Synthesis. QMDD-based synthesis [28] embeds the function explicitly. The function is specified as a *Quantum Multi-valued Decision Diagram* (QMDD) which is a compact representation of a reversible function/permutation matrix [30]. In QMDDs, the function is decomposed by its most significant variable x_i yielding four sub-matrices, each of them represents one of the four possible mappings of x_i. The sub-matrices $M_{0->0}$, $M_{1->0}$, $M_{0->1}$, and $M_{1->1}$, map the variable x_i from 0 to 0, 1 to 0, 0 to 1, and 1 to 1, respectively. Toffoli gates can be applied to swap the columns of the permutation matrix M and, by this, to form the identity matrix. Those gates can be applied in reverse to eventually form a circuit realizing the desired function (see [28] for details). Since this approach works if the given function is reversible, the original function has to be explicitly embedded first[2]. That is, ancillary inputs and garbage outputs which make the function reversible are determined during embedding. The function embedded into a reversible circuit remains hidden for different values and locations of ancillary inputs and garbage outputs.

Different embedding approaches can be used as a pre-processing step. While a random embedding can generate the reversible function, it is inefficient and does not scale. Scalable algorithmic embeddings reduce the run time and we consider the embedding method from [32]. This method guarantees a minimum number of ancillary inputs and garbage outputs. The resulting permutation matrix is divided into sub-matrices to determine the input-output combinations mapping. A sub-matrix may consist of don't care assignments (x), potential functional input-output assignments (*), and no assignment (0). The embedding transposes the matrix to the identity matrix to assign (*) and (x) entries with actual values; the functional input assignments are moved to the diagonal of the permutation matrix using swap operations of columns with hamming distance of 1. Each swap is implemented using a Toffoli gate.

Example 5. *To transform the permutation matrix of Figure 5(b) to the identity matrix, the input combination of the seventh column from the left is swapped with the eighth column. The updated input combination of the eighth column is swapped with the one in the fourth column resulting in the identity matrix. All the (*) and (x) entries in the main diagonal are assigned to 1. The resulting reversible embedding and circuit are shown in Figure 5(c) and (d), respectively.*

[1] An additional circuit line is added to preserve the values of x_4 and x_3 which are needed by the co-factors f_3 and f_4.

[2] A QMDD synthesis approach embeds and synthesizes in one step [31].

3 MOTIVATION AND THREAT MODEL

Reversible circuits generated by state-of-the-art synthesis tools are challenging for an adversary aiming to recover the functionality of the circuit even if he/she has access to gate-level implementations. We illustrate these challenges, discuss the threat model and count the number of embeddings in the reversible circuit.

3.1 IP Piracy: Challenges and Threat Model

An adversary with access to the gate-level implementation of a reversible function can trivially identify the function. However, when a non-reversible function is embedded in a reversible function, the adversary who is unaware of the location of the inputs and outputs of the target function can not easily identify the location and the value of the ancillary inputs and the location of the garbage outputs. Both are essential to extract the function.

Example 6. *Consider the full adder embedding in Figure 1. If the attacker is unaware of the location and values of the ancillary inputs, he/she cannot determine the target function. If c_{in} is the ancillary input, setting c_{in} to 0, results in sum $= x \oplus y$, while setting c_{in} to 1, results in sum $= \overline{x \oplus y}$. Distinguishing between the primary care outputs and the garbage don't care outputs is another challenge. In Figure 1, any of the four outputs can be a garbage output.*

We consider an attacker in the foundry with access to the gate-level implementation of a reversible circuit. The I/Os of the reversible circuits are connected to the I/O pins of the chip. Even, if the location of the ancillary inputs is known to the attacker, the unknown values of the ancillary inputs and the location of the garbage outputs may obfuscate the function. In an untrusted foundry threat model, the attacker does not have access to a functional chip [33, 34]

3.2 Number of Embeddings as a Security Metric

The number of embeddings quantifies the difficulty of IC/IP piracy. Our analysis applies to the base-case where an attacker does not know the location of the ancillary inputs and garbage outputs and the value of the ancillary inputs.

Consider an $n \times n$ reversible function $f(x_1, \cdots, x_n) = (y_1, \cdots, y_n)$. Each x_i is either a primary or an ancillary input, while each y_i, is either a primary or a garbage output. Each output y_i, is computed as $y_i = f_i(x_{i_1}, x_{i_2}, \cdots, x_{i_{m_i}})$, where m_i $(1 \leq m_i \leq n)$ is the number of inputs that drive y_i. Let k_i $(0 \leq k_i \leq m_i)$ be the number of inputs that drive y_i but not y_p, where p is in the interval 1 to $i-1$ $(1 \leq p < i)$. The number of functions embedded in y_i is obtained by considering any subset of the k_i inputs as ancillary inputs:

$$e(k_i) = \sum_{j=0}^{k_i} C(k_i, j) \times 2^j \tag{1}$$

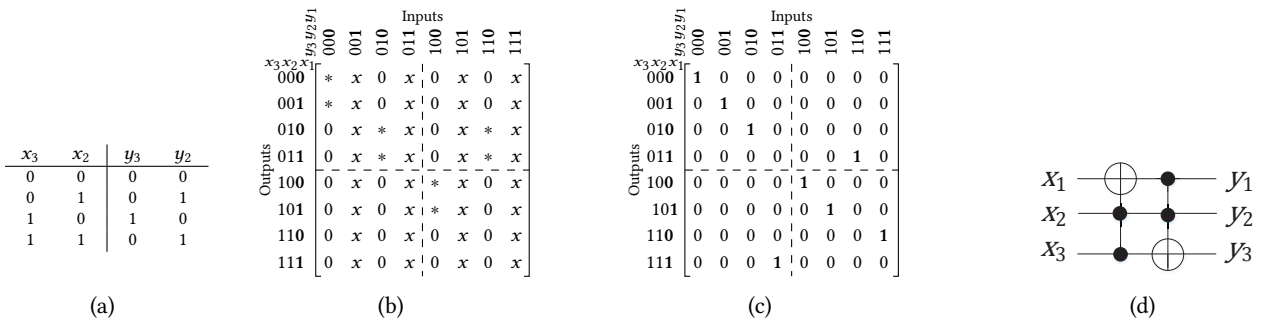

Figure 5: Embedding into a reversible function: (a) The truth table of the target function f. (b) The permutation matrix of f after adding an ancillary input x_1 and a garbage output y_1. The target function is activated when $x_1 = 0$. (c) The reversible function after swapping columns to yield the identity matrix. (d) The resulting reversible circuit.

The binomial coefficient $C(k_i, j)$ is the number of ways one can select j un-ordered ancillary inputs from k_i. The number of embedded functions with n primary outputs is

$$E(n, K) = \prod_{i=1}^{n} e(k_i) \tag{2}$$

where $K = \{k_1, \ldots, k_n\}$.

Each output of a reversible circuit is either a primary or a garbage output. Thus, the number of functions embedded into an $n \times n$ reversible circuit is:

$$EMB(n, K) = (2^n - 1) \times E(n, K) \tag{3}$$

where $2^n - 1$ is all possible combinations of output bits of a reversible circuit. We subtract one to exclude the case where all outputs of a reversible circuit are garbage.

Example 7. *In Figure 1, $k_1 = 4, k_2 = 0, k_3 = 0$ and $k_4 = 0$. Thus, $e_1 = 81$, $e_2 = 1, e_3 = 1$, and $e_4 = 1$. The number of embedding is $81 \times (2^4 - 1) = 1215$.*

At a first glance, it seems difficult to derive the target function. The next section evaluates the difficulty (or ease) of IC/IP piracy of reversible circuits.

4 IP PIRACY OF REVERSIBLE CIRCUITS

4.1 The Attack Principle

The attacker can recover the target function from a reversible circuit in two steps as illustrated in Figure 6. First, the attacker identifies the synthesis approach used to generate the reversible circuit [22]. Next, the attacker reconstructs the target function from the reversible circuit using the properties of the synthesis approach, referred to as de-synthesis.

Reversible logic synthesis approaches leave telltale signs in the circuits, which can be used to reveal the synthesis approach [22]. We consider reversible circuits obtained using BDD- [26] and QMDD-based synthesis [32]. De-synthesis applies to other reversible logic synthesis approaches as well.

4.2 Desynthesis of BDD-based Reversible Logic

BDD-based synthesis yields reversible logic with two telltale signs.

(1) The primary inputs are connected to garbage outputs.
(2) The target line of a reversible gate that doesn't control any successor gate is connected to a primary output.

Example 8. *Consider the circuit created by a BDD-based synthesis approach in Figure 4. The positions of the ancillary inputs and garbage outputs are specified. The primary output of the reversible circuit is connected to the target line of right most gate.*

The attacker uses these telltale signs to recover the function embedded in the reversible circuit. First, the attacker distinguishes primary outputs from the garbage outputs and primary inputs from the ancillary inputs using the two telltale signs. Next, the attacker discovers the value of the ancillary inputs by:

(1) Partitioning the reversible circuit into sub-circuits.
(2) Identifying the ancillary input values of the sub-circuits.

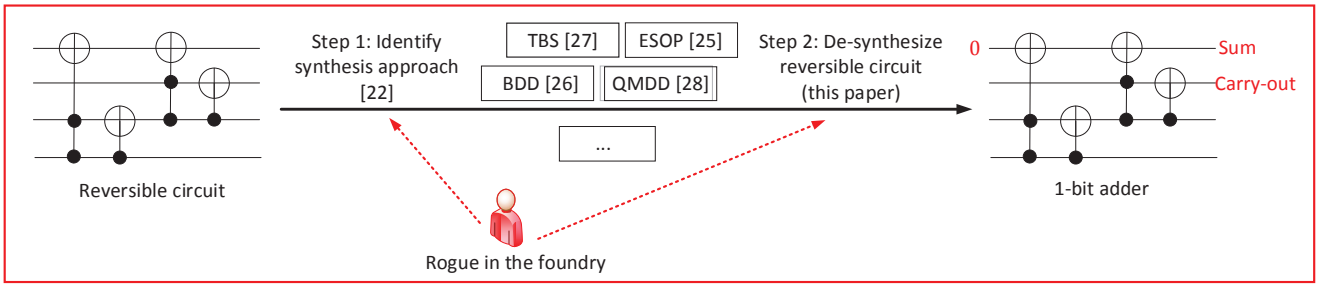

Figure 6: The "reversing the reversible circuit" threat model (i.e., steps involved in recovering the function from a reversible circuit without access to the functional chip) [22]. This paper focuses on the step 2 of the attack.

The attacker partitions the reversible circuit into sub-circuits, where each sub-circuit consists of the maximum number of adjacent reversible gates that match at least one sub-circuit in Figure 3[3]. Many sub-circuits have a unique structure, which enables identifying the ancillary input value. Others behave as *universal gates*, which can be reconfigured to support different sub-functions depending on the associated ancillary input value. In Figure 3, cases 4, 5 show an example of a sub-circuit that can represent any two co-factors depending on the ancillary input value. On the other hand, the sub-circuits in cases 1, 2, 3, 6, 7 represent unique co-factors. When the location of the ancillary inputs is known, the attacker applies the two steps to recover the value of the ancillary inputs.

Example 9. *Consider the reversible circuit in Figure 4. BDD-based synthesis using Shannon decomposition was used to generate it. The attacker identifies the primary output y_5 using the second telltale sign. Also, she/he identifies the location of garbage outputs y_1, y_2, y_3, and y_4 as they are directly connected to the inputs (using the first telltale sign). From the attacker's perspective y_6, y_7, and y_8 are potential primary outputs. The attacker determines the location of the ancillary inputs, x_5, x_6, x_7, and x_8 using the first telltale sign. Next, the attacker partitions the reversible gates into sub-circuits. As shown in Figure 4, the number of partitions is six. Consider p_i as the i^{th} partition. p_1, p_2, and p_3 are mapped to cases 7, 6, and 2 in Figure 3, respectively. Thus, p_1, p_2, and p_3 can be uniquely identified. As a result, the values of ancillary inputs x_6, x_7, and x_8 are 1, 0, and 0, respectively. However, p_0 can be mapped to either case 4 or case 5. These two cases result in a different co-factor, which results in a different ancillary input value. The number of embeddings depends on the number of unknown ancillary inputs and potential primary outputs. As a result, there are two possible functions for y_5. For the remaining outputs y_6, y_7, and y_8, the number of possible embeddings is one given that the number of possible embeddings for y_5 is two. Overall, the number of possible embeddings for the reversible circuit in Figure 4 is $2^3 \times 2$. This number can be further reduced to 2 if the attacker ignores the potential primary outputs. Thus, the attacker can recover most of the target function of the reversible circuit.*

4.3 Desynthesis of QMDD-based Reversible Logic

QMDD-based synthesis in [32] transposes the permutation matrix of the target function to the identity matrix by swapping columns of the functional input assignments with other columns using Toffoli gates. Following is the telltale sign of the embedding associated with this QMDD-based synthesis:

(1) The maximum number of Toffoli gates is activated by functional input assignments.

The attack exploits the telltale signs of the QMDD embedding to reveal the target function embedded in a reversible circuit. The attack is formulated as an optimization problem, in which the objective is to maximize the number of activated gates of the reversible circuit. The rationale for the attack to return the functional ancillary inputs value is that reversible gates are inserted with the objective of converting a non-reversible function into a reversible one. These gates swap functional input assignment columns with functional/non-functional columns to achieve the identity matrix. Thus, the ancillary inputs value that activates the maximum number of reversible gates is the one that realizes the target function.

The attacker traverses the reversible circuit, while keeping track of the swap operations for different input assignments. The attacker identifies the input patterns that activate the maximum number of Toffoli gates using an automatic test pattern generation algorithm (ATPG) [35]. The ATPG is used to generate test patterns that detect missing target line faults, and thus, activate reversible gates. The test patterns are ranked based on the number of detected faults. The top test patterns are used to identify either the ancillary inputs value for known location of ancillary inputs or the location of primary inputs for unknown location of ancillary inputs.

We consider two attack scenarios. If the location of the ancillary inputs is known, the attacker selects ancillary inputs value of the test patterns that detect the maximum number of missing target line faults. These patterns correspond to functional input assignment columns with the maximum number of swap operations to construct an identify matrix. This attack is outlined in algorithm 1. When the location of the ancillary inputs is unknown, the attacker attempts to identify the primary inputs. According to the embedding procedure, functional input assignments activate the maximum number of Toffoli gates. Thus, the value of the ancillary inputs of the top test patterns are expected to be fixed, while some of the primary inputs can be fixed and others are different. The attacker can identify the inputs with different values in the top test patterns as primary inputs. The percentage of identified primary inputs depends on the test patterns with maximum number of detected faults.

Example 10. *Consider the reversible circuit in Figure 5, where the ancillary input is x_1 and the other two inputs are the primary inputs. The test pattern that activates the maximum number of missing target line faults (2 faults in this example) is $x_1x_2x_3 = 011$. If the attacker knows the location of the ancillary input, he/she can use this test pattern to reveal the value of the ancillary input ($x_1 = 0$). However, if the location of the ancillary input is unknown, the attacker observes the difference in the top test patterns. In this example, there is only one test pattern that detects the maximum number of faults. Thus, the attacker can not distinguish between ancillary and primary inputs.*

Algorithm 1: Attack circuits generated by QMDD synthesis.

Input: Reversible circuit
Output: Ancillary inputs value

Generate test patterns to detect missing target line faults.
for *each test pattern* **do**
  Count the number of detected missing target line faults.
end
Sort the test patterns based on the number of detected faults.
return Ancillary input values of test patterns with maximum number of detected faults.

Example 11. *To illustrate the attack on QMDD-based reversible circuits for unknown location of ancillary inputs, consider reversible circuit in Figure 7. The ancillary input is x_4. An ATPG generates two test patterns, $x_1x_2x_3x_4 = 0110$ and 1000, that detect the maximum number of missing target lines. x_1, x_2, x_3 have different values in these patterns indicating that they are primary inputs.*

[3]The partitioning maintains the BDD structure.

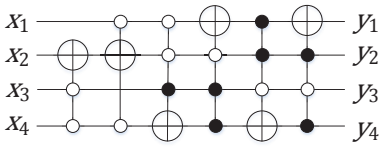

Figure 7: Reversible circuit activates function for $x_4 = 0$.

IP/IC piracy attack is expected to mis-predict most of the ancillary inputs values when a reversible circuit generated using one synthesis approach is de-synthesized using another synthesis approach. For example, de-synthesizing BDD-based reversible circuits using QMDD-based synthesis can fail to identify ancillary inputs values. In BDD-based reversible circuits, most of the control lines are primary inputs, whose value determines whether a gate is activated. Input patterns that activate the largest number of reversible gates may not contain the functional ancillary inputs value.

5 EXPERIMENTAL RESULTS

To evaluate the difficulty of extracting the implemented function from a reversible circuit, we conducted various experiments. In each experiment, we report the number of embeddings. First, we report the number of embeddings of reversible circuits for the base-case where the synthesis approach is not known. Second, we launch the attack on BDD- and QMDD-based reversible circuits. We use RevLib benchmarks [36].

5.1 Number of Embeddings as a Security Metric

In Tables 2 and 3, we count the number of embeddings of BDD- and QMDD-based reversible circuits, respectively. In each table, columns 1-5 summarize the benchmark, the number of inputs/outputs, garbage outputs, and ancillary inputs, and the gates number of the reversible circuit, respectively. Columns 6 and 7 provide the results of the base-case prior to the attack. %G in column 6 reports the percentage of identified garbage outputs due to the direct connection between some of the inputs and the outputs of the reversible circuit. Column 7 reports the number of embeddings.

Table 2 shows that on average 42% of the garbage outputs in BDD-based reversible circuits can be identified. However, the number of garbage outputs of BDD-based reversible circuits is large because every node of the BDD can produce a garbage output. Moreover, the location and the value of the ancillary inputs are unknown. Thus, the number of embeddings using equation 3 is large. Table 3 shows that on average 6.8% of the garbage outputs in QMDD-based reversible circuits can be recovered – resulting in a large number of embeddings. We conclude that without the knowledge of the synthesis approach, identifying the target function is infeasible.

5.2 IP Piracy Attacks

5.2.1 BDD-based Reversible Circuits. In the second set of experiments, we assume that the attacker knows the synthesis approach used to generate the reversible circuit [22]. First, we launch our attack on BDD-based reversible circuits. Columns 8 and 9 in Table 2 provide the percentage of recovered ancillary inputs and the number of possible embeddings after attacking the BDD-based reversible circuits used in the previous experiment, given that the attacker knows the synthesis approach.

Table 2: Results of the attack on reversible circuits created using BDD-based synthesis.

Bench-mark	# I/O	# G	# A	# Gates	Basic %G	Basic #Emb	+ Synth. info. %L_A	+ Synth. info. #Emb.	+ Synth. info. Red. #Emb
4mod5	7	6	3	8	66.7	1.50E+04	66.7	8.00E+00	2.00E+00
4mod7	12	10	8	30	40	1.40E+08	100	3.20E+01	1.00E+00
5xp1	30	20	23	90	30	3.50E+21	91.3	6.60E+04	4.00E+00
add6	54	47	42	159	25.5	2.60E+38	100	3.40E+10	1.00E+00
adr4	16	11	8	34	72.7	1.10E+10	100	8.00E+00	1.00E+00
aj-e11	16	12	12	42	33.3	1.80E+11	83.3	1.00E+03	4.00E+00
alu1	28	20	16	47	60	1.50E+18	93.8	5.10E+02	2.00E+00
alu2	105	99	95	452	10.1	2.70E+76	89.5	6.30E+29	1.00E+03
alu3	66	58	56	200	17.2	2.20E+48	76.8	2.30E+18	8.20E+03
alu4	541	533	527	2186	2.6	9.27E+376	93.4	5.90E+166	3.40E+10
apex2	498	495	459	1746	7.9	1.13E+317	96.3	2.40E+142	1.30E+05
apex4	547	528	538	2551	1.7	1.24E+382	91.4	1.20E+170	7.00E+13
apla	103	91	93	326	11	1.40E+77	82.8	1.60E+29	6.60E+04
sao2	74	70	64	211	14.3	3.70E+54	73.4	1.50E+23	1.30E+05
clip	66	61	57	228	14.8	4.50E+48	78.9	1.80E+19	4.10E+03
cm150	37	36	16	62	58.3	3.00E+22	0	2.20E+09	6.60E+04
cm151	49	40	30	94	47.5	2.60E+32	70	1.10E+09	5.10E+02
cm42	22	12	18	45	33.3	8.20E+15	50	1.30E+05	5.10E+02
cm82	13	10	8	30	50	4.10E+08	100	3.20E+01	1.00E+00
cm85	36	33	25	87	33.3	5.00E+24	56	8.60E+09	2.00E+03
cmb	43	40	27	50	40	4.40E+28	100	1.70E+07	1.00E+00
co14	27	26	13	63	53.8	6.30E+16	100	4.10E+03	1.00E+00
con1	16	14	9	32	50	2.20E+10	77.8	5.10E+02	4.00E+00
cordic	52	50	29	101	46	3.50E+33	79.3	8.60E+09	6.40E+01
cu	38	28	24	80	50	2.30E+25	66.7	4.20E+06	2.60E+02
dc1	20	13	16	56	30.8	2.30E+14	87.5	2.00E+03	4.00E+00
decod	35	19	30	82	26.3	5.40E+25	50	5.40E+08	3.30E+04
dist	79	74	71	307	10.8	1.20E+59	100	7.40E+19	1.00E+00
dk17	58	47	48	154	21.3	1.30E+42	77.1	2.80E+14	2.00E+03
ex5p	206	143	198	647	5.6	7.80E+157	65.7	1.30E+61	3.00E+20
ex2	105	99	95	25	10.1	2.70E+76	89.5	6.30E+29	1.00E+03
f2	16	12	12	41	33.3	1.80E+11	58.3	8.20E+03	3.20E+01
f51m	385	377	371	1648	3.7	1.40E+252	91.9	2.00E+118	1.10E+09
hwb6	46	40	40	159	15	9.70E+33	92.5	1.40E+11	8.00E+00
hwb7	73	67	66	281	10.4	5.00E+54	92.4	1.80E+19	3.20E+01
hwb8	112	105	104	449	7.6	1.30E+80	95.2	2.50E+30	3.20E+01
max46	54	53	45	190	17	2.10E+39	77.8	1.80E+16	1.00E+03
%misex3	428	414	414	1473	3.4	8.8E+312	68.6	3.50E+159	1.40E+39
mlp4	103	95	95	362	8.4	5.50E+77	94.7	5.00E+27	3.20E+01
sqn	40	37	33	134	18.9	1.00E+29	69.7	1.10E+12	1.00E+03
sqrt8	30	26	22	76	30.8	8.60E+20	72.7	1.70E+07	6.40E+01
xor5	6	5	1	8	100	7.30E+02	100	1.00E+00	1.00E+00
z4ml	14	10	7	30	70	6.10E+08	100	8.00E+00	1.00E+00
tial	578	570	564	2253	2.5	9.75E+379	94.9	1.30E+176	5.40E+08

On average 81.6% of the ancillary inputs can be recovered. Thus, the attacker can identify most of the target function even if the location of the ancillary inputs is unknown. The large number of possible embeddings in column 9 is due to the large number of potential primary outputs. Although the attacker can recover the location of primary outputs that satisfy the second property of BDD-based reversible circuits in Section 4.2, he/she can not determine whether the outputs of sub-circuits, which are connected to the reversible circuit outputs and also used to control other target lines, are garbage outputs. Thus, the number of all primary output combinations increases significantly. **Our experiment summarized in Table 2 shows that the attacker can identify all the primary outputs of circuits created by BDD-based synthesis**. Thus, the number of embeddings of the BDD-based reversible circuits is reduced significantly, as illustrated in column 10 of Table 2, by ignoring the potential primary outputs[4].

[4]If some of the discarded outputs are primary outputs, the reduced number of embeddings can still assess the difficulty of partially recovering the function.

5.2.2 QMDD-based Reversible Circuits. Assuming that the location of the ancillary inputs is known to the attacker, we successfully identified the values of all the ancillary inputs for **94%** (i.e. 125-out-of-131) of the reversible circuits, while we partially identified the ancillary inputs value for the rest. These results use a naive test pattern generation algorithm [35] to generate test patterns that detect the maximum number of missing target line faults. These test patterns share the functional ancillary inputs value, which activates the target function. An advanced test pattern generation algorithm using SAT-solver [35] can yield a higher percentage of reversible circuits, in which the value of all the ancillary inputs is recovered. The number of embeddings depends only on the number of possible primary outputs, which is computed as $2^{\#\ possible\ primary\ outputs}$, excluding the leaked garbage outputs. Due to the space limitation we don't show the number of embeddings. However, it can be computed using the data in Table 3.

We also consider the scenario when the location of the ancillary inputs is unknown to the attacker. Table 3 shows the results on a set of reversible circuits generated using QMDD-based synthesis when the location of ancillary inputs is unknown. We consider the top test vectors, which detect the maximum number of missing target line faults. The eighth and the ninth columns indicate the percentage of identified locations of primary inputs and the number of embeddings, respectively. Our attack can identify on average %13.9 of the primary inputs. The number of embeddings is large, and thus, the attack can't recover the target function. Although, for many circuits the attacker can't identify the primary inputs, the number of embeddings using the attack is less than the corresponding one for the base-case. This is because each ancillary input under the proposed attack can have one possible value, which is given in the test patterns with maximum number of faults.

A summary of the attacks result is provided in Table 4, where L_A. indicates the percentage of leaked ancillary inputs. We conclude that if the attacker can identify the location of the ancillary inputs, she can recover most of the ancillary inputs values of a reversible circuit, and thus, most of the target function.

6 CONCLUSION AND DISCUSSION

We assessed the difficulty of recovering the target function from an embedding realized in terms of a reversible circuit. Identifying the target function becomes challenging since, in contrast to conventional circuits, reversible circuits do not directly realize the target function, but a corresponding embedding. While such an embedding may obfuscate the target function, it was unknown thus far how secure the resulting circuit really is with respect to piracy. We evaluated the effort and provided IP piracy attack strategies based on circuits generated by BDD-based and QMDD-based synthesis.

Next we discuss the realism feasibility of the considered problem and the threat model in a Q&A format:

Question: While theoretical interest might exist, isn't the considered problem a niche area of research since most technologies still rely on non-reversible CMOS circuits?

Table 3: Results of the attack on reversible circuits created using QMDD-based synthesis.

Bench-mark	# I/O	# G	# A	# Gates	Basic %G	Basic #Emb	+ Synth. info. %PI_L	+ Synth. info. #Emb.
4mod5	5	4	1	6	75	3.60E+05	0	1.90E+02
4mod7	5	2	1	24	50	2.10E+07	0	1.20E+04
5xp1	10	0	3	439	0	1.80E+29	0	3.70E+19
add6_92	13	6	1	8151	0	2.10E+47	0	2.00E+31
adr4	9	4	1	560	0	1.50E+24	100	1.80E+16
alu1	18	10	6	4718	0	1.00E+87	0	7.80E+56
alu2	14	8	4	2524	0	2.00E+54	0	6.60E+35
alu3	14	6	4	2119	0	2.00E+54	0	6.60E+35
alu4	19	11	5	61599	0	2.40E+96	50	8.20E+62
apex4	26	7	17	4715	0	2.00E+175	44.4	3.10E+113
clip	11	6	2	1121	0	6.30E+34	0	1.50E+23
cm150	22	21	1	1067	9.5	4.10E+121	61.9	3.70E+47
cm42	13	3	9	73	0	2.10E+47	0	2.00E+31
cm82	6	3	1	51	0	6.00E+11	0	1.30E+08
cm85	13	10	2	418	10	2.40E+46	0	2.50E+30
cmb	20	16	4	3126	6.3	2.30E+103	0	1.10E+68
con1	8	6	1	69	0	3.70E+19	14.3	1.70E+13
cu	25	14	11	13124	14.3	6.80E+154	7.1	2.10E+93
dc1	10	3	6	55	0	1.80E+29	0	3.70E+19
dk17	19	8	9	5695	0	2.40E+96	0	8.20E+62
example2	14	8	4	2524	0	2.00E+54	0	6.60E+35
f51m	19	11	5	40146	0	2.40E+96	0	8.20E+62
misex3	28	14	14	131668	0	1.40E+202	7.1	1.10E+127
mlp4	13	5	5	851	0	2.10E+47	0	2.00E+31
sqrt8	9	5	1	124	40	4.90E+23	25	1.40E+11
table3	28	14	14	127998	0	1.40E+202	78.6	4.40E+130
squar5	9	1	4	77	0	1.50E+24	0	1.80E+16
sym9	10	9	1	338	0	1.80E+29	0	3.70E+19
x2	16	9	6	1014	0	5.10E+69	30	1.40E+45

Table 4: Summary of IP piracy attacks, where ↓, ↓↓, ↓↓, and – indicate decrement, excessive decrement, excessive increment, and no effect, respectively.

Synthesis	Unknown Location of Ancill.		known Location of Ancill.	
	% L_A.	# Emb.	% L_A.	# Emb.
BDD	↑↑	↓↓	↑↑	↓↓
QMDD	–	↓	↑↑	↓↓

Response: As reviewed in Section 1, reversible logic is the basis for many applications and technologies. Significant progress is being made in some of these directions. Additionally, considering that conventional CMOS will reach its limit in the near future, it is important to be prepared on the implications (also with respect to security issues) which will come with applications/technologies relying on different paradigms such as reversible logic. In this regard, we would like to refer again to the discussion in [13] in which the potential technologies based on reversible logic are emphasized. Hence, a security assessment of reversible logic is timely.

Question: Is the study presenting a hypothetical reverse engineering strategy? Why would an IC pirate want to extract the non-reversible circuit when she/he already knows the reversible circuit and the embedded non-reversible circuit?

Response: This is a realistic scenario. The adversary is an untrusted foundry who has the layout and does not know the functionality. She/he is trying to reverse engineer the function. Further, the distribution of the chips is strictly controlled (e.g. chips used by the Govt and DoD). This is the state-of-the-art after IBM, a trusted foundry, was acquired by Global Foundries. This paper focuses on the IC/IP theft of reversible logic by an untrusted foundry. SAT attacks do not apply as there is no oracle (i.e. a functional chip) available to the attacker.

Question: The problem assumes (and shows) that the synthesis process leaves telltale signs which the attacker can leverage. But advanced synthesis algorithms may not suffer from this limitation?

Response: Even when we used optimized QMDD synthesis approach, our attack identifies the ancillary inputs value for known locations of ancillary inputs. This indicates that advanced synthesis approaches leave telltale signs. Re-designing synthesis tools so that they do not leave telltale signs is an important problem.

Question: Isn't it reasonable to assume that the reverse-engineer has some input/output pairs that she/he knows? Otherwise, it is questionable why the attacker wants to steal IP in the first place.

Response: We consider an attacker in the foundry. We assume that the chip is not publicly available. The malicious foundry does not have access to functional I/O. In this untrusted foundry threat model, the reversible circuits can be configured in many ways depending on the garbage outputs and the ancillary inputs.

Question: Is revealing a high percentage of ancillary inputs sufficient for a successful attack?

Response: Revealing a large number (and percentage) of ancillary inputs alone is not enough for a successful attack. The attacker should identify the location of the garbage outputs. Both the unknown ancillary inputs and garbage outputs determine the number of embeddings. However, the high percentage of identified ancillary inputs indicates that while the attack is not fully successful, the attacker can recover most of the target function. Our ideal secure synthesis approach should result in a large number of unknown ancillary inputs values and a large number of embeddings.

ACKNOWLEDGEMENTS

The 3rd and 4th author are supported by the EU COST Action IC1405. The 7th author is partly funded by NYU/NYU-AD CCS.

REFERENCES

[1] M. Pecht and S. Tiku, "Bogus: electronic manufacturing and consumers confront a rising tide of counterfeit electronics," *IEEE Spectrum*, vol. 43, no. 5, pp. 37–46, 2006.

[2] J. A. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending piracy of integrated circuits," in *Proceedings of Design Automation and Test in Europe*, 2008, pp. 1069–1074.

[3] *Quantum Computation and Quantum Information*. New York, NY, USA: Cambridge University Press, 2000.

[4] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM J.Res.Dev*, vol. 5, no. 3, pp. 183–191, 1961.

[5] C. H. Bennett, "Logical reversibility of computation," *IBM J.Res.Dev*, vol. 17, no. 6, pp. 525–532, 1973.

[6] W. C. Athas and L. J. Svensson, "Reversible logic issues in adiabatic CMOS," in *Proceedings of Workshop on Physics and Computation*, 1994, pp. 111–118.

[7] A. Rauchenecker, T. Ostermann, and R. Wille, "Exploiting reversible logic design for implementing adiabatic circuits," in *Proceedings of Mixed Design of Integrated Circuits and Systems*, 2017.

[8] R. Wille, R. Drechsler, C. Osewold, and A. G. Ortiz, "Automatic design of low-power encoders using reversible circuit synthesis," in *Proceedings of Design Automation and Test in Europe*, 2012, pp. 1036–1041.

[9] R. Wille, O. Keszocze, S. Hillmich, M. Walter, and A. G. Ortiz, "Synthesis of approximate coders for on-chip interconnects using reversible logic," in *Proceedings of Design, Automation Test in Europe*, 2016, pp. 1140–1143.

[10] A. Zulehner and R. Wille, "Taking one-to-one mappings for granted: Advanced logic design of encoder circuits," in *Proceedings of Design, Automation and Test in Europe*, 2017, pp. 818–823.

[11] R. Cuykendall and D. R. Andersen, "Reversible optical computing circuits," *Opt. Lett.*, vol. 12, no. 7, pp. 542–544, Jul 1987.

[12] S. Kotiyal, H. Thapliyal, and N. Ranganathan, "Mach-Zehnder interferometer based design of all optical reversible binary adder," in *Proceedings of Design, Automation and Test in Europe Conference Exhibition*, 2012, pp. 721–726.

[13] M. P. Frank, "The future of computing depends on making it reversible," *IEEE Spectrum*, 2017.

[14] W. Wustmann and K. Osborn, "Reversible fluxon logic: Topological particles allow gates beyond the standard adiabatic limit," in *arXiv:1711.04339*, 11 2017.

[15] M. S. M. Hogg, Tad and D. G. Allis, "Mechanical computing systems using only links and rotary joints," *Molecular Systems Design & Engineering*, vol. 2, no. 3, pp. 235–252, 2017.

[16] J. A. Roy, F. Koushanfar, and I. L. Markov, "Ending piracy of integrated circuits," *Computer*, vol. 43, no. 10, pp. 30–38, Oct 2010.

[17] J. Rajendran, H. Zhang, C. Zhang, G. S. Rose, Y. Pino, O. Sinanoglu, and R. Karri, "Fault analysis-based logic encryption," *IEEE Transactions on Computers*, vol. 64, no. 2, pp. 410–424, 2015.

[18] M. Yasin, J. J. Rajendran, O. Sinanoglu, and R. Karri, "On improving the security of logic locking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1411–1424, 2016.

[19] L. Chow, J. Baukus, B. Wang, and R. Cocchi, "Camouflaging a standard cell based integrated circuit," 2012.

[20] J. Rajendran, M. Sam, O. Sinanoglu, and R. Karri, "Security analysis of integrated circuit camouflaging," in *Proceedings of the ACM SIGSAC Conference on Computer & Communications Security*, 2013, pp. 709–720.

[21] J. Rajendran, O. Sinanoglu, and R. Karri, "VLSI testing based security metric for ic camouflaging," in *Proceedings of IEEE International Test Conference*, 2013, pp. 1–4.

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

[23] X. Cui, S. M. Saeed, A. Zulehner, R. Wille, K. Wu, R. Drechsler, and R. Karri, "On the difficulty of inserting trojans in reversible computing architectures," *IEEE Transactions on Emerging Topics in Computing*, pp. 1–1, 2018.

[24] T. Toffoli, "Reversible computing," in *Automata, Languages and Programming*, W. de Bakker and J. van Leeuwen, Eds. Springer, 1980, p. 632.

[25] K. Fazel, M. Thornton, and J. Rice, "ESOP-based Toffoli gate cascade generation," in *Proceedings of IEEE PacRim*, 2007, pp. 206 –209.

[26] R. Wille and R. Drechsler, "BDD-based synthesis of reversible logic for large functions," in *Proceedings of ACM/IEEE Design Automation Conference*, July 2009, pp. 270–275.

[27] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Proceedings of ACM/IEEE Design Automation Conference*, 2003, pp. 318–323.

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

[29] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Transactions on Computers*, vol. 35, no. 8, pp. 677–691, 1986.

[30] P. Niemann, R. Wille, D. M. Miller, M. A. Thornton, and R. Drechsler, "QMDDs: Efficient quantum function representation and manipulation," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 1, pp. 86–99, 2016.

[31] A. Zulehner and R. Wille, "One-pass design of reversible circuits: Combining embedding and synthesis for reversible logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 5, pp. 996–1008, 2018.

[32] A. Zulehner and R. Wille, "Make it reversible: Efficient embedding of non-reversible functions," in *Proceedings of Design, Automation and Test in Europe Conference Exhibition*, 2017, pp. 458–463.

[33] M. E. Massad, J. Zhang, S. Garg, and M. V. Tripunitara, "Logic locking for secure outsourced chip fabrication: A new attack and provably secure defense mechanism," *CoRR*, vol. abs/1703.10187, 2017.

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

[35] R. Wille, H. Zhang, and R. Drechsler, "ATPG for reversible circuits using simulation, Boolean satisfiability, and pseudo Boolean optimization," in *Proceedings of IEEE Computer Society Annual Symposium on VLSI*. IEEE, 2011, pp. 120–125.

[36] R. Wille, D. Grosse, L. Teuber, G. W. Dueck, and R. Drechsler, "Revlib: An online resource for reversible functions and reversible circuits," in *Proceedings of International Symposium on Multiple Valued Logic*, 2008, pp. 220–225.