Identifying Reversible Circuit Synthesis Approaches to Enable IP Piracy Attacks

Samah Mohamed Saeed, Nithin Mahendran
University of Washington
Tacoma, WA 98402
{samahs,nithin}@uw.edu

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

Ramesh Karri
New York University
Brooklyn, NY 11201
rkarri@nyu.edu

***Abstract*—Reversible circuits are vulnerable to intellectual property and integrated circuit piracy. To show these vulnerabilities, a detailed understanding on how to identify the function embedded in a reversible circuit is crucial. To obtain the embedded function, one needs to know the synthesis approach used to generate the reversible circuit in the first place. We present a machine learning based scheme to identify the synthesis approach using telltale signs in the design.**

***Index Terms*—Reversible logic, IP piracy, QMDD, ESOP, TBS, BDD, Security, Machine learning.**

I. Introduction

Reversible circuits compute bijective functions (i.e., they implement one-to-one mappings between the inputs and the outputs). Applications of reversible circuits include encoder/decoders [1], [2], quantum computing architectures [3], [4], and low-power designs [5]–[7]. Other areas that benefit from reversible computing include adiabatic circuits [8], [9], formal verification [10], and optical computing [11].

Progress in the design and (physical) realization of reversible circuits is underway. Concurrently, an understanding of security threats such as intellectual property (IP) piracy and malicious circuitry [12]–[14] is important and is the focus of this paper. This paper sheds light on recovering the function of a reversible circuit.

Non-reversible functions are implicitly or explicitly embedded into reversible circuits by using ancillary inputs and garbage outputs [15]. Hence, to derive the function, an attacker first needs to know the ancillary inputs. With the knowledge of the synthesis approach used to generate a reversible circuit, an attacker can identify most of the functions [16]. To this end, it is crucial to identify the synthesis approach which has been used to create the reversible circuit in the first place. This is covered in this work.

To this end, we first review the background on reversible circuits and motivate our work in Section II-A and Section II-B, respectively. Section III-A extracts the telltale signs of reversible circuit synthesis approaches. Section III-B derives features that can be used to identify the synthesis approach. These features are used by machine learning algorithms to discover the synthesis approach as discussed in Section III-C and demonstrated in the experimental evaluations in Section III-D. Finally, the paper is concluded in Section IV.

II. Reversible Circuits

A. Background

A function $f : B^n \to B^m$ is *reversible*, if and only if $n = m$ and each input combination maps to a unique output combination. Computations can be conducted in both directions using the *Toffoli gate*. Let $X = \{x_1, \ldots, x_n\}$ be the inputs to a reversible function. Then, the Toffoli gate $TOF(C, t)$ is 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 on the target line if and only if all positive control lines are assigned 1 and all negative control lines are assigned 0. The values on all remaining lines pass through unaltered. The Toffoli gate is universal, i.e., all reversible functions can be realized using them.

In order to realize an arbitrary function, an *embedding* step is conducted [15], which utilizes *ancillary inputs* and *garbage outputs*. An ancillary input is an input that is set to a fixed value (either 0 or 1). A garbage output is a don't care output for all possible input conditions. A function is implemented on a reversible circuit when ancillary inputs are assigned dedicated constant values and only the non-garbage outputs are considered.

Example 1. *Fig. 1 shows a reversible circuit implementation of a full adder. Black dots denote positive control lines, while* $\oplus$ *denotes a target line. The adder is implemented by assigning a constant 0 to the ancillary input* x_1*. The non-garbage output* y_2 *is the sum and* y_1 *is the carry-out.*

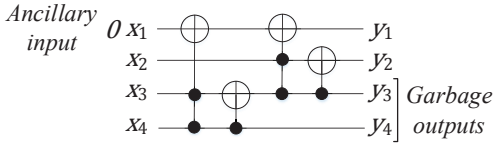

Fig. 1. Reversible circuit implementation of a full adder.

B. Threat Model and Motivation

We consider an attacker in the foundry with access to the gate-level implementation of the reversible circuit. We assume that the attacker does not have access to a functional chip [17], which can be applied to non-commercial applications such as military applications. We further assume that manufacturing test is conducted in either a separate test facility [18] or in the foundry. In the latter case, we consider random values assigned to the ancillary inputs during manufacturing test. Conducting

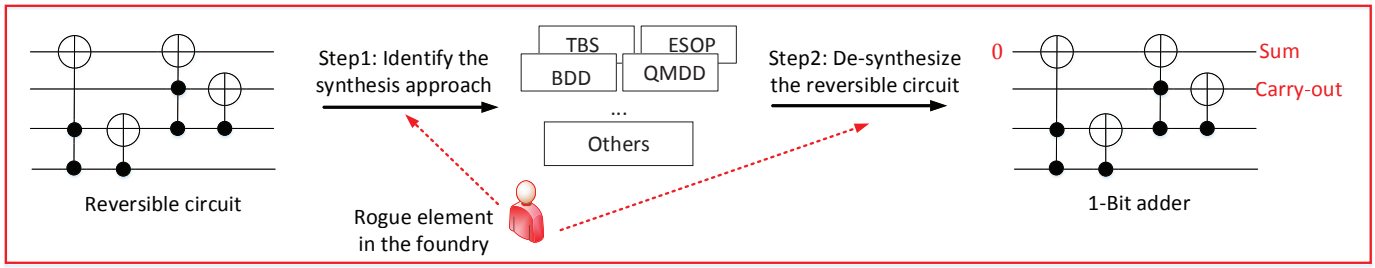

Fig. 2. The "reversing the reversible circuit" threat model (i.e., steps involved in recovering the function from a reversible circuit with no access to functional chip.). This paper focuses on the first step of the attack.

the test prior to activating an IC, improves the security and yields better test quality [19]. We consider two scenarios wherein an attacker can infer ancillary inputs and garbage outputs. If the chip is on the motherboard, an attacker can use the peripheral circuits to distinguish between the primary and the garbage outputs of the reversible circuits. Then, the attacker needs to recover the ancillary inputs only. On the other hand, if the design is sent to an untrusted foundry without peripheral circuits, the attacker has to identify the location of the ancillary inputs and garbage outputs in addition to the value of the ancillary inputs. In both scenarios, identifying the synthesis approach that generates a reversible circuit is important to recover the function.

Example 2. *Consider the circuit in Fig. 1. From an attacker's perspective, each input is a potential ancillary input and each output is a potential garbage output. Without any more information available during fabrication, the circuit appears to realize an arbitrary 4-bit reversible function.*

Even if an attacker locates the ancillary inputs and the garbage outputs, she/he can not recover the ancillary values to implement the function. The ancillary inputs hide the function in a reversible circuit. However, the attacker can circumvent this if he/she knows the synthesis approach used to generate the circuit; the ancillary inputs and garbage outputs depend on the embedding or the synthesis approach used to generate a reversible circuit. In Fig. 2, the attacker can recover the function in two steps. First, the attacker identifies the synthesis approach. Next, he/she obtains the ancillary inputs. This paper sheds light on the first step of the attack.

III. Identifying the Synthesis Approach

The telltale signs of reversible synthesis approaches can be used to identify them given a synthesized reversible circuit. This is sketched in the following sub-section by considering a selection of well-established synthesis approaches.

A. Telltale Signs of Synthesis Approaches

Transformation-based synthesis (TBS, [20]) starts with a truth table description and adds reversible gates to modify the given function until the identity-function is obtained. The resulting circuits have the following telltale TBS sign. A gate g_i at position i in a circuit is likely to have fewer (or as many) control lines than the preceding gates g_j ($j < i$), i.e., it is likely that $|C_j| \geq |C_i|$ if $i > j$. This is because, in TBS, the rows of the truth table are consecutively transformed to identity and, hence, gates are added such that no truth table row above the considered row is altered. As TBS proceeds, more rows are fixed. Hence, more control lines are added to the gates. Since in TBS the gates are implemented output to input, gates with more control lines appear towards the input-side of the circuit.

Example 3. *TBS generated the circuit shown in Fig. 3. The number of control lines increases from the output to the input.*

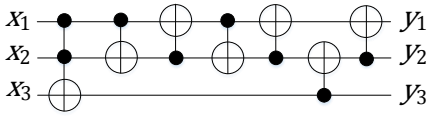

Fig. 3. Hamming code circuit generated by TBS.

In *Quantum Multi-valued Decision Diagram (QMDD)-based* synthesis [21], the function is transformed to the identity using reversible gates. The identity is established one variable at a time (i.e., the mapping from one bit of the input to one bit of the output). Circuits obtained using QMDD-based synthesis have the following telltale QMDD sign. Divide the circuit into n regions where n is the number of variables. In each region, there is 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 variable has been transformed to identity.

Example 4. *QMDD-based synthesis generated the circuit shown in Fig. 4. Dashed lines show the three regions. Each region corresponds to a variable.*

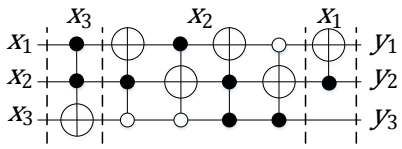

Fig. 4. Hamming code circuit generated by QMDD-based synthesis. White dots denote negative control lines.

BDD-based synthesis [22] uses *Binary Decision Diagrams* (BDDs), which represent a function and its sub-functions derived by Shannon decomposition. Circuits obtained using BDD-based synthesis have the following telltale BDD signs. Telltale BDD sign 1: The reversible circuits use pre-defined sub-circuits for each type of the BDD node. Telltale BDD sign 2: Each line in a reversible circuit is used as a control or a target line for a small set of reversible gates; the interference between circuit lines is low.

Example 5. *BDD-based synthesis generated the circuit shown in Fig. 5. The circuit is obtained by applying corresponding sub-circuits $f_{1->7}$. Circuit lines $x_{5->7}$ do not control each other, resulting in low interference.*

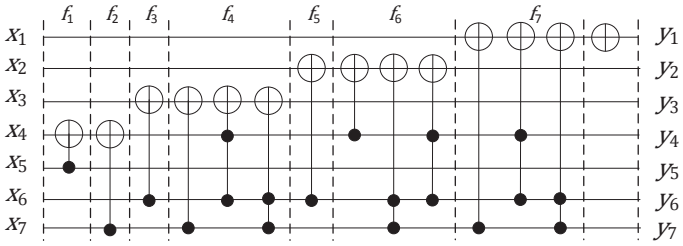

Fig. 5. Hamming code circuit generated by BDD-based synthesis.

Exclusive Sum of Products (ESOP)-based synthesis [23] generates a reversible circuit from an exclusive sum of products specification. Circuits obtained by ESOP have the following telltale ESOP sign. The lines in the circuits obtained using ESOP synthesis can be split into two distinct subsets. One subset is composed of (positive or negative) control lines only, while the other subset is solely composed of target lines.

Example 6. *ESOP-based synthesis generated the circuit shown in Fig. 6. Circuit lines $x_{4->6}$ are only control lines, while the others are target lines only.*

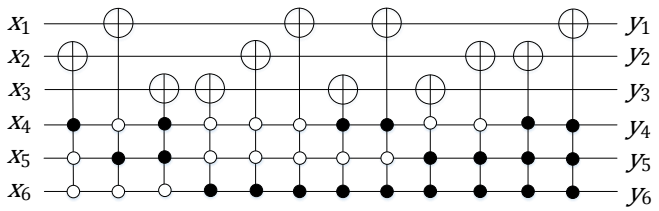

Fig. 6. Hamming code circuit generated by ESOP-based synthesis.

B. Telltale Features

We use the telltale signs in the synthesized circuits of the synthesis approaches to derive features to identify the synthesis approach.

1) Control Line Reduction: The telltale TBS sign motivates a feature that checks whether the number of control lines decrease when traversing the circuit from the inputs to the outputs. This feature outputs a ratio of the reversible gates, which show a reduction in the number of control lines as follows: 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 output of this feature, the more likely the circuit was generated using TBS.

2) Control-Only and Target-Only Lines: Control-only and target-only lines is a feature to check whether the circuit has the ESOP telltale sign. This feature is equal to 1 if the circuit can be divided into control-only and target-only lines; otherwise, the feature is equal to 0.

3) BDD Sub-Circuits-Only: This feature is 1 if a reversible circuit has BDD sub-circuits only and 0 otherwise.

4) Cone-Structural Analysis: This analysis is motivated by BDD telltale sign 2 and considers the logic cone driven by each pair of circuit lines. The feature computes the number of gates reachable from each pair of the circuit lines. The larger the number of reversible gates driven by the pair of circuit lines, the more the circuit lines interfere with each other. Thus, it is less likely that this circuit has been generated using BDD-based synthesis. For each pair of reversible circuit inputs, logic can be traced to compute the number of reversible gates that both circuit lines converge at. The sum of all the values collected in the analysis quantifies interference among circuit lines. Our analysis returns Convergent Ratio (CR), denoting interference between circuit lines in terms of the number of reversible gates, i.e.

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

where i and j refer to different circuit lines and $gate(i)$ refers to the set of reversible gates driven by line i. Normalization is effected via 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* total number of gates).

5) QMDD Feature: This feature determines if a reversible circuit is generated by QMDD synthesis. The feature is computed by checking the circuit for the telltale QMDD sign.

C. Machine Learning Scheme

We use machine learning to reveal the synthesis approach. We use control line reduction, control-only and target-only lines, BDD sub-circuits-only, cone-structural analysis, and QMDD telltale features for four machine learning algorithms, namely the decision tree [24], the random forest [25], the support vector machine (SVM) [26], and the logistic regression (LR) [27] models. In the training phase, we apply supervised learning with reversible circuits from which it is known how they have been synthesized. These circuits are traced to compute each of our proposed feature.

D. Experimental Evaluation

We consider 443 reversible circuits generated using different synthesis approaches. We use the decision tree, random forest, SVM, and LR machine learning algorithms. The models are trained using reversible circuits generated by four synthesis approaches – ESOP, BDD, QMDD, and TBS. We use the telltale features from Section III-B for classifying the test data. The results of the machine learning were verified by taking the average of 1000 runs of the 10× cross validation (10% test data and 90% training data).

Table I shows the accuracy of our machine learning identification scheme. The first column denotes the used machine learning algorithm, and the other columns denote the percentage of correctly identified reversible circuits generated by different synthesis approaches. The results confirm that the machine learning scheme can correctly identify the used synthesis approach most of the time. The random forest machine learning algorithm performs best. Circuits realized using BDD- and QMDD-synthesis can be identified more easily than circuits realized using TBS and ESOP-synthesis. This is because some ESOP-based reversible circuits satisfy the QMDD telltale sign and some of the TBS-based reversible circuits consist of BDD sub-circuits.

TABLE I
IDENTIFYING THE SYNTHESIS APPROACH.

Machine Learning Algorithm	BDD(%)	ESOP(%)	QMDD(%)	TBS(%)
Random Forest	96.1	87.6	93.8	74
Desicion tree	91.6	86.3	90.6	73.5
LR	96.7	76	96.8	39.3
SVM	98.8	74.5	94.5	33.5

Besides that, we also evaluated the effect of the training data size on the accuracy of the identification. We decreased the percentage of training data from 80% to 75% to 67% to 50% and ran the random forest model. The results are summarized in Fig. 7. The reduction in the size of the training data slightly affects the identification accuracy. This indicates that our proposed features capture the essential telltale signs of different synthesis approaches.

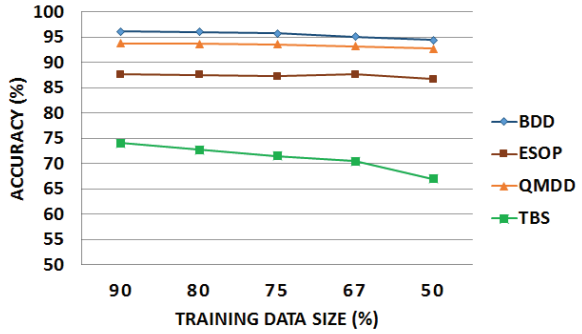

Fig. 7. Accuracy as a function of percentage of training data.

IV. DISCUSSION AND CONCLUSIONS

We analyzed how the attacker can identify the synthesis approach that generates a reversible circuit. This is a first step towards the identification of the function embedded into the reversible circuit. We first extract the telltale signs of well-established synthesis approaches and then map them to features that are utilized in several machine learning algorithms to determine the applied synthesis approach.

Our work provides a better understanding of IP piracy and possible attacks on reversible circuits. However, the application, and thus, the fabrication process of reversible circuits may differ depending on the target technology. For example, reversible circuits used for encoding and decoding devices and low-power designs have CMOS fabrication processes. In contrast, they significantly differ for quantum computation (where reversible circuits do not describe real hardware, but sequences of operations). Future work will focus on approaches which are more dedicated to a technology and/or an application.

Furthermore, post-synthesis optimization alters the circuit structure, although it does not completely re-write the circuits. Hence, future work includes investigations on how optimization methods affect the telltale signs, and thus, the identification scheme.

ACKNOWLEDGEMENTS

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

REFERENCES

[1] A. Zulehner and R. Wille, "Taking one-to-one mappings for granted: Advanced logic design of encoder circuits," in *DATE*, 2017.
[2] R. Wille, R. Drechsler, C. Osewold, and A. G. Ortiz, "Automatic design of low-power encoders using reversible circuit synthesis," in *DATE*, 2012, pp. 1036–1041.
[3] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000.
[4] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM J. Comput.*, vol. 26, no. 5, pp. 1484–1509, Oct. 1997.
[5] R. Landauer, "Irreversibility and heat generation in the computing process," *IBM J.Res.Dev*, vol. 5, no. 3, pp. 183–191, 1961.
[6] C. H. Bennett, "Logical reversibility of computation," *IBM J.Res.Dev*, vol. 17, no. 6, pp. 525–532, 1973.
[7] A. Berut, A. Arakelyan, A. Petrosyan, S. Ciliberto, R. Dillenschneider, and E. Lutz, "Experimental verification of Landauer's principle linking information and thermodynamics," *Nature*, vol. 483, pp. 187–189, 2012.
[8] W. C. Athas and L. J. Svensson, "Reversible logic issues in adiabatic cmos," in *PhysComp*, 1994, pp. 111–118.
[9] A. Rauchenecker, T. Ostermann, and R. Wille, "Exploiting reversible logic design for implementing adiabatic circuit," in *MIXDES*, 2017.
[10] L. G. Amarù, P. Gaillardon, R. Wille, and G. D. Micheli, "Exploiting inherent characteristics of reversible circuits for faster combinational equivalence checking," in *DATE*, 2016, pp. 175–180.
[11] R. Cuykendall and D. R. Andersen, "Reversible optical computing circuits," *Opt. Lett.*, vol. 12, no. 7, pp. 542–544, Jul 1987.
[12] 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.
[13] J. A. Roy, F. Koushanfar, and I. L. Markov, "EPIC: Ending piracy of integrated circuits," in *DATE*, 2008, pp. 1069–1074.
[14] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor, "Trustworthy hardware: Identifying and classifying hardware trojans," *Computer*, vol. 43, no. 10, pp. 39–46, Oct 2010.
[15] A. Zulehner and R. Wille, "Make it reversible: Efficient embedding of non-reversible functions," in *DATE*, 2017.
[16] S. M. Saeed, X. Cui, R. Wille, A. Zulehner, K. Wu, R. Drechsler, and R. Karri, "Towards reverse engineering reversible logic," *CoRR*, vol. abs/1704.08397, 2017.
[17] 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. [Online]. Available: http://arxiv.org/abs/1703.10187
[18] B. Wire, "Research and markets: Outsourced semiconductor assembly and test market (osat) trends," 2014. [Online]. Available: http://www.businesswire.com/news/home/20140324005628/en/Research-Markets-Outsourced-Semiconductor-Assembly-Test-Market
[19] M. Yasin, S. M. Saeed, J. Rajendran, and O. Sinanoglu, "Activation of logic encrypted chips: Pre-test or post-test?" in *DATE*, 2016, pp. 139–144.
[20] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *DAC*, 2003, pp. 318–323.
[21] M. Soeken, R. Wille, C. Hilken, N. Przigoda, and R. Drechsler, "Synthesis of reversible circuits with minimal lines for large functions," in *ASP-DAC*, 2012, pp. 85–92.
[22] R. Wille and R. Drechsler, "BDD-based synthesis of reversible logic for large functions," in *DAC*, 2009, pp. 270–275.
[23] K. Fazel, M. Thornton, and J. Rice, "ESOP-based Toffoli gate cascade generation," in *Proceedings of IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, 2007, pp. 206 –209.
[24] J. R. Quinlan, "Induction of decision trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.
[25] T. K. Ho, "Random decision forests," in *ICDAR*, vol. 1, 1995, pp. 278–282 vol.1.
[26] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods*. Cambridge University Press, 2000.
[27] D. A. Freedman, *Statistical Models: Theory and Practice*. Cambridge University Press, 2009.