From Reversible Logic to Quantum Circuits: Logic Design for an Emerging Technology

Robert Wille[1,2] Anupam Chattopadhyay[3] Rolf Drechsler[2,4]
[1]Institute for Integrated Circuits, Johannes Kepler University Linz, A-4040 Linz, Austria
[2]Cyber Physical Systems, DFKI GmbH, 28359 Bremen, Germany
[3]Nanyang Technological University, Singapore
[4]Institute of Computer Science, University of Bremen, 28359 Bremen, Germany
robert.wille@jku.at anupam@ntu.edu.sg drechsle@informatik.uni-bremen.de

***Abstract*—Quantum computing has been attracting increasing attention in recent years because of the rapid advancements that have been made in quantum algorithms and quantum system design. Quantum algorithms are implemented with the help of quantum circuits. These circuits are inherently reversible in nature and often contain a sizeable Boolean part that needs to be synthesized. The logic design of such quantum circuits constitutes a non-trivial task and, hence, have heavily been investigated by researchers in the recent past. This paper provides a brief overview of these research. We review the major steps to be conducted in the logic design of quantum circuits and provide a sketch for each single step. These descriptions are enriched with discussions as well as references to the respective related work.**

I. Introduction

Exploiting quantum mechanical phenomena for computational purposes established the field of quantum computation [1] which received significant attention in the recent past. In a quantum circuit, in contrast to conventional bits, qubits are used as storage, which do not only allow to represent the (Boolean) basis states 0 and 1, but also superpositions of both. By this, qubits can represent multiple states at the same time which enables massive parallelism. Additionally exploiting further quantum mechanical phenomena such as phase shifts or entanglement enables asymptotical speed-ups for many relevant problems (e.g. database search [2] or integer factorization [3], [4]), offers new methods for secure communication (e.g. quantum key distribution), and has several other appealing applications [1].

These promises also lead to new challenges for computer-aided design. While first netlists of the respective quantum circuits have been developed by hand, the design of more complex quantum functionality will require automatic methods particularly for synthesis. Here, the fact that quantum circuits are inherently reversible in nature and often contain a sizeable Boolean part can be exploited. Therefore, how to design an efficient quantum circuit for a respectively desired Boolean functionality emerged as a major research area.

This paper provides an overview on the resulting approaches for the logic design of quantum circuits. To this end, we first review the basics on reversible as well as quantum circuits and summarize the main steps to be conducted in Section II and Section III, respectively. After a brief review on reversible circuits synthesis in Section IV, we provide a more detailed sketch of the respective steps, namely (1) the mapping of the reversible circuit to a quantum circuit (Section V), (2) the consideration of nearest neighbor constraints (Section VI), as well as (3) the test of the resulting circuit (Section VII). Note that, due to the focus on the pure logic design, other important issues such as fault tolerance or error correcting [5], [6], [7] are not covered in this overview. At the end of this paper, some final conclusions are provided in Section VIII.

II. Background

This section briefly reviews the basics on reversible as well as quantum circuits which are used in order to realize quantum functionality.

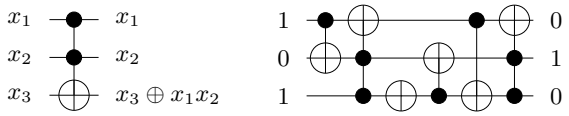
A. Reversible Circuits

Reversible circuits are digital circuits with the same number of input signals and output signals. Furthermore, reversible circuits realize bijections, i.e. each input assignment maps to a unique output assignment. Accordingly, computations can not only be performed from the inputs to the outputs but also in the other direction.

Reversible circuits are composed as cascades of reversible gates. The *Toffoli gate* [8] is widely used in the literature and also considered in this paper.

Definition 1. *Given a set of variables or signals* $X = \{x_1, \ldots, x_n\}$*, a* Toffoli gate $T(C, t)$ *is a tuple of a possibly empty set* $C \subset X$ *of* control lines *and a single* target line $t \in X \setminus C$*. The Toffoli gate inverts the value on the target line if all values on the control lines are set to* 1 *or if* $C = \emptyset$*. All remaining values are passed through unaltered.*

Example 1. *Fig. 1a shows a Toffoli gate drawn in standard notation, i.e. control lines are denoted by* $\bullet$ *while the target line is denoted by* $\oplus$*. A circuit composed of several Toffoli gates is depicted in Fig. 1b. This circuit maps e.g. the input* 101 *to the output* 010 *and vice versa.*



(a) Toffoli gate (b) Toffoli circuit

Fig. 1. Toffoli gate and Toffoli circuit

B. Quantum Circuits

Quantum computation [1] is a promising application of reversible logic. The corresponding *quantum circuits* are very similar to reversible circuits but work on quantum bits (*qubits*) instead of bits. In contrast to Boolean logic, qubits do not only allow to represent the conventional Boolean states 0 and 1, but also the superposition of them. More precisely, a qubit is a linear combination of the conventional Boolean states in the two dimensional complex Hilbert space. The two orthonormal quantum states $|0\rangle = \begin{bmatrix}1\\0\end{bmatrix}$ and $|1\rangle = \begin{bmatrix}0\\1\end{bmatrix}$ are used to represent the Boolean values 0 and 1. Thus, any state of a qubit may be written as $|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle$, where α and β are complex numbers such that $|\alpha|^2 + |\beta|^2 = 1$. All quantum bits reside on the Bloch sphere, a unit 2-sphere.

Each operation on these qubits can be defined by a unitary matrix [1] which is represented by means of quantum gates. In the design of quantum circuits, often the following definition of quantum gates (forming the so-called *NCV library*) is applied[1]:

Definition 2. *A* quantum gate $\mathrm{q}(C, t)$ *over the inputs* $X = \{x_1, \ldots, x_n\}$ *consists of a single target line* $t \in X$ *and, in some cases, of a single control line* $c \in X$ *with* $t \neq c$*. That is,* C *is either empty or equals* $\{c\}$*. When further assuming that the inputs to the circuit as well as the inputs to the control lines of the gates are restricted to the conventional Boolean values* 0 *and* 1[2]*, the following four gates define the commonly used quantum gate library.*

- NOT gate*: The qubit on the target line* t *is inverted.*
- Controlled NOT gate *(CNOT): The target qubit* t *is inverted if the control qubit* c *is 1.*
- Controlled V gate*: The* V*-operation is performed on the target qubit* t *if the control qubit* c *is 1. The* V*-operation is also known as the square root of NOT, since two consecutive* V*-operations are equivalent to an inversion.*
- Controlled $V^\dagger$ gate*: The* $V^\dagger$ *gate performs the inverse operation of the* V *gate, i.e.* $V^\dagger = V^{-1}$.

Due to the assumption for the circuit inputs and control lines, the set of possible values $\{|0\rangle, |1\rangle, |v_0\rangle, |v_1\rangle\}$ is closed under the above mentioned operations. That is, we are dealing with a 4-valued logic, where $|v_0\rangle = \frac{1+i}{2}\begin{bmatrix}1\\-i\end{bmatrix}$ and $|v_1\rangle = \frac{1+i}{2}\begin{bmatrix}-i\\1\end{bmatrix}$.

Example 2. *Fig. 2 shows a quantum circuit composed of several of the gates introduced above realizing a reversible function. Control lines are again denoted by* $\bullet$ *while target lines are either denoted by* $\oplus$ *in case of NOT and CNOT or by* V *and* $V^\dagger$ *in case of* V *gates and* $V^\dagger$ *gates, respectively. This circuit maps e.g. the input* 101 *to the output* 111 *and vice versa, whereby the intermediate value* $|v_0\rangle$ *occurs.*

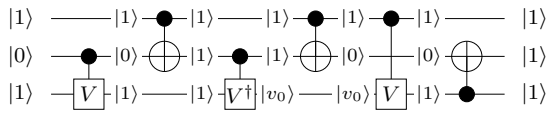


Fig. 2. Quantum circuit

All elementary gates are assumed to have unit cost [9].

III. Logic Design for Quantum Circuits

Since the design of quantum circuits is a rather complex task, many (automatic) methods employ a synthesis flow that *does not directly* realize the given quantum functionality, but follow a *multi-step* approach. For this purpose, two main characteristics are exploited, namely

- many important quantum algorithms, like Grover's database search algorithm [2] and Shor's factorization algorithm [3], contain a considerable reversible (Boolean) component that needs to be synthesized, and
- all quantum operations are inherently reversible.

Consequently, the Boolean components of the quantum algorithms are first realized as a reversible circuit, rather than a quantum circuit. This significantly reduces synthesis complexity. Besides that, a huge variety of synthesis approaches for reversible circuits is already available (briefly summarized in Section IV).

Once a reversible circuit realizing the desired functionality is available, it is mapped to a functionally equivalent quantum circuit. To this end, the following steps are usually conducted:

1) *Mapping Reversible Circuit to Quantum Circuit*
   Each reversible gate is mapped to an functional equivalent cascade of quantum gates. To this end, various mapping methods and subsequent optimizations can be applied which e.g. depend on the respectively desired quantum gate library. Section V provides an overview of this step.
2) *Consideration of Physical Constraints*
   The resulting quantum gates may not satisfy dedicated physical constraints – so-called *Nearest Neighbor Constraints* represent one prominent example. Hence, dedicated procedures have to be undertaken to satisfy such constraints. Section VI provides an overview of this step.
3) *Testing of Quantum Circuits*
   The logical behavior of the reversible circuits can be tested with efficient test-set generation. To that end, researchers established fault models for quantum circuits and proposed various testing techniques. Section VII provides an overview of this step.

A brief overview of this is presented in the following.

[1]Later in Section V-C, also references to other gate libraries are provided.

[2]This restriction is common when considering the design of Boolean components of quantum circuits as motivated in Section I.

IV. Synthesis of Reversible Circuits

Although not the focus of this paper, we briefly review the synthesis of reversible circuits which provide the basis for all following steps. A key differentiator between conventional and reversible logic synthesis methods arise due to the restriction of fanout in reversible circuits. Increasing fanout leads to more ancilla lines and often prevents the application in quantum computation (due to the no-cloning theorem [1]), which is costly to implement in quantum technologies. To that end, different synthesis methods were adopted, which can broadly be classified in terms of the chosen Boolean function representation (more detailed overviews are provided in [10], [11]).

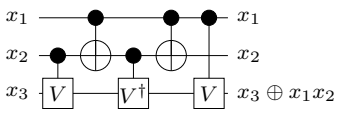
- Early logic synthesis methods were based on truth tables [12], [13], which guaranteed ancilla-free reversible logic synthesis. These were further optimized to include diverse gate libraries [14].

- Afterwards, more elaborated heuristics have been proposed which also rely on truth-table like function descriptions but employ more advanced heuristics such as ant colony optimization [15], simmulated anneling [16], group theory based techniques [17], or output permutation [18]. However, these methods mainly suffer from their poor scalability.

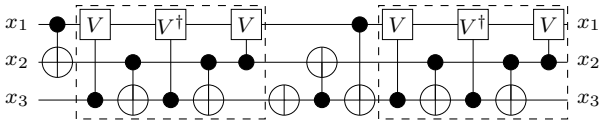
- *Exclusive Sum-of-Products* (ESOPs) allow for a more compact representation of a Boolean function. Hence, ESOP formulations have been used to synthesize ancilla-free [19] or bounded-ancilla [20] circuits for larger functions.

- Due to the inherent scalability of decision diagrams, also multiple methods based on variants of decision diagrams were developed to address the scalability problem [21], [22], [23]. The default, node-wise mapping of a decision diagram on to reversible circuits results in high ancilla, which were addressed by an alternative mapping that utilizes reversible functions descriptions such as QMDDs [24] or so called *characteristic function* to achieve ancilla-free reversible circuit [25].

A set of other function representations such as *Quantum Multiple-valued Decision Diagram* (QMDD) [26] or *Rotation-based Decision Diagrams* (RbDD) [27] have been explored, too. Furthermore, note that most of these methods additionally rely on an embedding step to be conducted prior to synthesis (see e.g. [28]). This usually also motivates a trade-off between ancilla and gate count/quantum costs (see e.g. [29]). Finally, it is also important to note that, the above variants propose heuristic methods due to the complexity of reversible logic synthesis [30]. Multiple exact synthesis methods have been proposed, too [31], [32], which, however have limited capacity to scale. A recent proposition of hybrid synthesis flow showed that scalability can be achieved by following conventional logic synthesis flows, whereas ancilla can be restricted by building internal data-structures with a functional view [33].



(a) For the single Toffoli gate shown in Fig. 1a



(b) For the reversible circuit shown in Fig. 1b

Fig. 3. Mapping reversible circuits to quantum circuits

V. Mapping Reversible Circuits to Quantum Circuits

As motivated in Section III, an established synthesis approach is to first realize the desired (Boolean) functionality as a reversible circuit and, afterwards, map it to an equivalent quantum circuit. To this end, the first set of approaches considered the realization of quantum circuits based on the NCV gate library as reviewed in Section II. However, beyond that, also further – more application-specific – libraries emerged in the recent past. This motivated the development of further mapping schemes for these new libraries. This section reviews the main idea of the mapping scheme for this library and exemplarily illustrates its optimization potential. Afterwards, we provide a brief overview on the further libraries and their mappings.

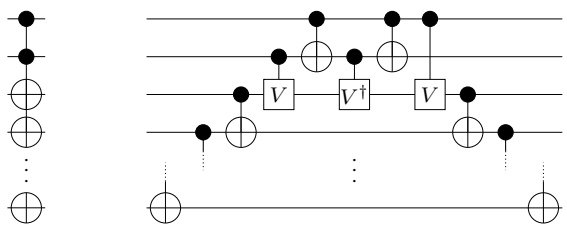
A. NCV Library

The main idea of mapping reversible circuits to quantum circuits can be found in the seminal work by Barenco et al. [9] which is illustrated by the following example.

Example 3. *Consider a Toffoli gate with two control lines as shown in Fig. 1a. A functionally equivalent realization in terms of quantum gates is depicted in Fig. 3a. This cascade can be applied to fully map the reversible circuit shown in Fig. 1b into an equivalent quantum circuit. For this purpose, all corresponding Toffoli gates are respectively substituted with a corresponding quantum gate cascade. The 1$^{\text{st}}$, 3$^{\text{rd}}$, 4$^{\text{th}}$, and 5$^{\text{th}}$ gate remain unchanged as they already represent quantum gates. The resulting fully equivalent quantum circuit is shown in Fig. 3b.*

Similar mappings exist for Toffoli gates with more than two control lines. But with increasing number of control lines, the resulting quantum circuits become more expensive, i.e. require more quantum gates. Furthermore, also the number of the *ancillary lines*, i.e. the number of circuit lines which neither are a control line nor a target line, affect the size of the resulting quantum circuit. Details on that are described in e.g. [9], [34], [35] – the resulting costs are frequently applied as a cost metric in order to evaluate reversible circuits.

While the basic idea of the mapping scheme illustrated above is almost identical in all corresponding quantum circuit synthesis methods, several adjustments and alternatives have been proposed in the recent past which e.g. optimize the ap-



(a) Toffoli Gate (b) Quantum gate cascade

Fig. 4. Improved mapping of Toffoli gate with multiple targets

proach or address further gate libraries. The remainder of this section reviews some representatives of these developments.

B. Possible Optimization: Exploiting Multiple-Target Lines

Mapping reversible circuits to quantum circuits has mainly been considered in a local manner, i.e. single Toffoli gates only have been mapped to corresponding quantum gate cascades as illustrated by Example 3. In contrast, considering more than one Toffoli gate allows for significant reductions. This has been investigated e.g. in [36] where mappings of pairs of Toffoli gates have been proposed. Besides that, it has been observed in [37] that synthesis methods for reversible circuits often lead to cascades of Toffoli gates which differ only in the position of their respective target lines, but have an equal set of control lines. This motivated the definition and exploitation of *Toffoli gates with multiple target lines*.

Definition 3. *A* Toffoli gate with multiple target lines $T(C, T)$ *over the inputs* $X = \{x_1, \ldots, x_n\}$ *consists of a (possibly empty) set of* control lines $C \subset X$ *and a non-empty set of* target lines $T \subseteq X \setminus C$. *The Toffoli gate inverts the value on all target lines if all values on the control lines are set to* 1 *or if* $C = \emptyset$. *All remaining values are passed through unaltered.*

Toffoli gates with multiple target lines enable an easier and more efficient mapping to quantum gate cascades. More precisely, a Toffoli gate with multiple target lines $T(C, T)$ and $T = \{t_1, t_2, \ldots, t_k\}$ can be mapped to a quantum gate cascade as follows:

1) Consider a Toffoli gate $T(C, t)$ with a single target line $t \in T$.
2) Map $T(C, t)$ to a functionally equivalent quantum gate cascade using any of the existing methods introduced in the past.
3) For each remaining target line $t' \in T \setminus \{t\}$, add a Toffoli gate $T(\{t\}, t')$ before and after the quantum gate cascade generated in Step 2.

Example 4. *Consider the Toffoli gate with multiple target lines as shown in Fig. 4a. Applying the proposed scheme, a functionally equivalent quantum gate cascade as shown in Fig. 4b results.*

Using this improved mapping, the size of the resulting quantum gate cascades can further be reduced to $3 + 2|T|$ for a Toffoli gate with two control lines and $|T|$ target lines – compared to the standard mapping scheme a substantial improvement particularly for gates with a large number of control lines.

C. Consideration of Further Gate Libraries

The NCV gate library as reviewed in Def. 2 was, for a long time, the state-of-the-art gate model which has been assumed in logic design of quantum circuits (motivated by the original work from [9] and respective follow-ups such as [34], [35]). However, in the recent past and motivated by new physical accomplishments as well as theoretical discussions, further gates libraries received attention. Accordingly, also dedicated mapping schemes have been considered. This includes, but is not limited to, the following gate libraries.

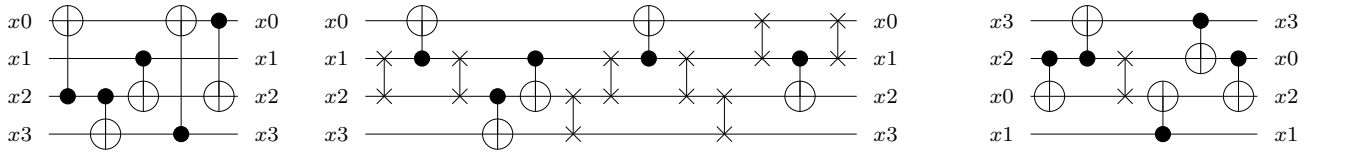
The *Clifford+T gate library*, which is composed of gates for the so-called *Clifford group* operation together with so-called T and $T^\dagger$ gates (see e.g. [38] for details). The motivation for this gate library is the increasing need to consider fault tolerance in quantum circuits for what Clifford+T gates have been found particularly suited for. From a design perspective, a major objective is to reduce the number of T/$T^\dagger$ gates since their fault tolerant implementations lead to costs which are by a factor of 100 greater than for the other gates. Accordingly adjusted mappings and dedicated circuit constructions have been discussed e.g. in [38], [39], [40], [41].

Gate libraries based on *Physical Machine Descriptions* (PMDs, [42]), which have been motivated by a broad survey of quantum systems conducted in the ARDA quantum computing roadmap [43]. Here, different technologies (namely *Quantum Dots* (QD), *Superconducting Qubits* (SC), *Ion Traps* (IT), and many more) have been considered resulting in PMDs which are different in terms of the supported (primitive) operations and, hence, in terms of the supported gates. Consequently, mapping schemes for the resulting gate libraries have also been considered. An overview and comparison has recently been conducted e.g. in [44].

The *NCV-$|v_1\rangle$ library* [45], [46], in which *qudits* instead of qubits are assumed, i.e. a basic building block which does not rely on a two level quantum system but a (multiple-valued) d-level quantum system is assumed. This allows for gates whose control lines may be sensitive to non-Boolean values (while e.g. the NCV gates considered above always require Boolean control lines). This in turn enables to realize smaller mappings for Toffoli gates and an easier satisfaction of certain physical constraints (as discussed in [46]). While this is interesting from a design perspective, physical realizations and concepts on how to realize this library still is subject to future work.

VI. Consideration of Nearest Neighbor Constraints

In the recent years, research has led to several physical realizations for quantum computers [47] which uncovered serious intrinsic limitations [48], e.g. with respect to the interaction distance, decoherence times, or scaling. Consequently, further constraints have been introduced, which needs to be considered during quantum circuit synthesis. One of the most common



(a) Given circuit (b) Nearest neighbor compliant circuit (c) Optimal nearest neighbor compliant circuit

Fig. 5. Establishing nearest neighbor compliance

ones is the so called *linear nearest neighbor constraint*. This limits the interaction distance between gate qubits and applies a constraint so that computations are only performed between adjacent, i.e. nearest neighbor, qubits [49].

Usually, synthesis of quantum circuits as reviewed above often does not consider nearest neighbor constraints. Accordingly, researchers started considering synthesis of nearest neighbor compliant quantum circuits. Complementary directions are thereby investigated.

One direction addressed nearest neighbor constraints directly during synthesis. One of the approaches presented in [50] provides an example of this design paradigm. Here, an exact synthesis method is proposed which directly realizes a nearest neighbor compliant quantum circuit with minimal quantum cost. Although not very scalable, this approach can be used to generate macros (i.e. minimal sub-circuits) which, in turn, allow for improvements when mapping a reversible circuit onto a quantum circuit. Another approach that embeds the nearest neighbor constraint within a heuristic reversible logic synthesis technique is presented recently in [51]. Efficient nearest neighbor-compliant quantum realization of prominent quantum circuits are proposed in [52].

Another direction employs a post-synthesis process: If a given circuit does not satisfy the nearest neighbor constraints, additional SWAP gates are added.

Definition 4. *A* SWAP gate *is a quantum gate $S(q_i, q_j)$ including two signal x_i, x_j and maps $(x_0, \ldots, x_i, x_j, \ldots, x_{n-1})$ to $(x_0, \ldots, x_j, x_i, \ldots, x_{n-1})$. That is, a SWAP gate realizes the exchange of two quantum values (in figures drawn using two connected $\times$ symbols).*

These SWAP gates allow for making all control lines and target lines adjacent and, by this, help to satisfy the nearest neighbor constraint. More precisely, a cascade of adjacent SWAP gates can be inserted in front of each gate g with non-adjacent circuit lines in order to shift the control line of g towards the target line, or vice versa, until they are adjacent. The following example illustrates the idea.

Example 5. *Consider the circuit depicted in Fig. 5a. As can be seen, gates g_1, g_4, and g_5 are non-adjacent. Thus, in order to make this circuit nearest neighbor compliant, SWAP gates in front and after all these gates are inserted as shown in Fig. 5b.*

While such a naive approach is able to transform any given circuit into a nearest neighbor compliant version in linear time, the insertion of SWAP gates obviously increases the quantum cost of the resulting circuit. In fact, for each non-adjacent gate $2 \cdot (|t - c| - 1)$ SWAP gates are additionally inserted to the circuit, where t and c denote the position of the target and control line, respectively. Moreover, the fashion in which SWAP gates are inserted has a significant effect: Reordering the circuit lines or considering SWAP gate insertion not only locally for each single gate but for the whole cascade may reduce the costs significantly. As an example, the nearest neighbor compliant circuit from Fig. 5b can actually be reduced to the circuit shown in Fig. 5c – reducing the number of SWAP gates from 8 to 1.

Motivated by this, researchers started to intensely investigate how to reduce the number of SWAP gate insertions in order to make a given quantum circuit nearest neighbor compliant. Approaches addressing this design problem have recently be presented e.g. in [53], [54], [55], [56]. Besides that, recent work focuses on employing this restriction to 2D quantum circuits [57], [58]. Finally, this physical constraints can also specifically be considered to a given quantum technology, such as the one outlined in [59].

VII. Testing of Quantum Circuits

The fragility of quantum states naturally requires high error resilience, achieved by error correcting codes, such as the ones proposed in [5], [6], [7]. The efficiency of practical realizability of such error correction codes also lead to heightened activity in experimental studies, characterization of the cost model for the gate libraries and novel physical constraints. In this part of the paper, however, we restrict ourselves to the logical correctness of the reversible circuit. This task is fundamentally different from conventional digital circuit testing [60], partly due to the fault models that arise from the implementation aspects of quantum circuits.

A. Fault Models

In [61], different fault models for a reversible circuit are proposed. These include, for example, *single/multiple missing gate fault*, *single/multiple missing control fault* and *repeated gate fault*. Such fault models were derived from the implementation principles of quantum circuits using trapped ion technology [1]. It was shown in [61] that testing for certain classes of fault models is sufficient to detect faults in other classes. Further, automatic test pattern generation for reversible circuits based on these faults are proposed [61], [62]. Bounds for the test set size for certain classes of faults are provided in [63].

B. Approaches for Test Pattern Generation

Beyond the standard test-set generation approach proposed in [63], [61], [62], the diagnosis of faults is studied in [64]. It is observed that by inserting certain additional logic to

the original reversible circuit, the test effort can be reduced and also the fault diagnosis can be improved. Efforts in that direction is made by the online testing approach outlined in [65]. Quantum circuits are probabilistic in nature, where the outcome is accompanied with a probability distribution. Therefore, testing of such circuits under this stochastic behavior essentially means, testing that the observed non-deterministic behavior is within acceptable limit. This is emphasized in the binary tomographic testing proposed in [66].

VIII. Conclusions and Future Work

Quantum circuits are inherently reversible. Because of that, mapping a reversible circuit realizing the desired functionality into an equivalent quantum circuit received significant interest. In this overview paper, we reviewed the main idea and motivation behind this concept and discussed state-of-the-art research.

Moving forward, the following research goals are envisioned, specifically in the context of automated quantum circuit implementation.

- The varied efforts towards developing a complete framework starting from a quantum algorithm towards a physically mapped circuit representations need to come together. Early efforts towards an integrated logic synthesis flow has been undertaken in [67] (mainly for the reversible logic part). More specific flows towards the most promising quantum technologies are considered in [68], [59]. It is likely that such flows will generate more close interactions between the researchers working on quantum algorithms, logic synthesis, verification, and experimental quantum technologies. One immediate goal in this line of research is to adopt a well-defined cost model for the underlying gate libraries. Further, similar to conventional logic synthesis, representation formalisms between different abstraction levels need to be established.
- Programming languages for Quantum computers differ significantly from the traditional software programming languages [69]. To provide seamless implementation flow for Quantum circuits, such programming languages need to be integrated to the synthesis flows outlined in this paper.

Acknowledgment

The authors would like to thank all researchers and collaborators which, in the past years, worked on the development of the approaches reviewed in this paper. This work has partially been supported by the European Union through the COST Action IC1405.

References

[1] M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*. Cambridge Univ. Press, 2000.

[2] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Theory of computing*, 1996, pp. 212–219.

[3] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Foundations of Computer Science*, pp. 124–134, 1994.

[4] L. M. K. Vandersypen, M. Steffen, G. Breyta, C. S. Yannoni, M. H. Sherwood, and I. L. Chuang, "Experimental realization of Shor's quantum factoring algorithm using nuclear magnetic resonance," *Nature*, vol. 414, p. 883, 2001.

[5] A. R. Calderbank and P. W. Shor, "Good quantum error-correcting codes exist," *Phys. Rev. A*, vol. 54, pp. 1098–1105, 1996.

[6] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, "Surface codes: Towards practical large-scale quantum computation," *Phys. Rev. A*, vol. 86, 2012.

[7] D. Bacon, S. T. Flammia, A. W. Harrow, and J. Shi, "Sparse quantum codes from quantum circuits," in *Symposium on Theory of Computing*, 2015, pp. 327–334.

[8] T. Toffoli, "Reversible computing," in *Automata, Languages and Programming*, W. de Bakker and J. van Leeuwen, Eds. Springer, 1980, p. 632, technical Memo MIT/LCS/TM-151, MIT Lab. for Comput. Sci.

[9] A. Barenco, C. H. Bennett, R. Cleve, D. DiVinchenzo, N. Margolus, P. Shor, T. Sleator, J. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *The American Physical Society*, vol. 52, pp. 3457–3467, 1995.

[10] R. Drechsler and R. Wille, "From truth tables to programming languages: Progress in the design of reversible circuits," in *Int'l Symp. on Multi-Valued Logic*, 2011, pp. 78–85.

[11] M. Saeedi and I. L. Markov, "Synthesis and optimization of reversible circuits - a survey," *ACM Computing Surveys*, 2011.

[12] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Design Automation Conf.*, 2003, pp. 318–323.

[13] D. Maslov, G. W. Dueck, and D. M. Miller, "Toffoli network synthesis with templates," *IEEE Trans. on CAD*, vol. 24, no. 6, pp. 807–817, 2005.

[14] M. Soeken and A. Chattopadhyay, "Fredkin-enabled transformation-based reversible logic synthesis," in *Int'l Symp. on Multi-Valued Logic*, 2015, pp. 60–65.

[15] M. Li, Y. Zheng, M. S. Hsiao, and C. Huang, "Reversible logic synthesis through ant colony optimization," in *Design, Automation and Test in Europe*, 2010, pp. 307–310.

[16] N. Abdessaied, M. Soeken, G. W. Dueck, and R. Drechsler, "Reversible circuit rewriting with simulated annealing," in *VLSI of System-on-Chip*, 2015, pp. 286–291.

[17] G. Yang, X. Song, W. N. N. Hung, F. Xie, and M. A. Perkowski, "Group theory based synthesis of binary reversible circuits," in *Int'l Conf. on Theory and Applications of Models of Computation*, 2006, pp. 365–374.

[18] R. Wille, D. Große, G. Dueck, and R. Drechsler, "Reversible logic synthesis with output permutation," in *VLSI Design*, 2009, pp. 189–194.

[19] P. Gupta, A. Agrawal, and N. K. Jha, "An algorithm for synthesis of reversible logic circuits," *IEEE Trans. on CAD*, vol. 25, no. 11, pp. 2317–2330, 2006.

[20] R. Drechsler, A. Finder, and R. Wille, "Improving ESOP-based synthesis of reversible logic using evolutionary algorithms," in *Applications of Evolutionary Computation*, 2011, pp. 151–161.

[21] R. Wille and R. Drechsler, "BDD-based synthesis of reversible logic for large functions," in *Design Automation Conf.*, 2009, pp. 270–275.

[22] M. Krishna and A. Chattopadhyay, "Efficient reversible logic synthesis via isomorphic subgraph matching," in *Int'l Symp. on Multi-Valued Logic*, 2014, pp. 103–108.

[23] A. Chattopadhyay, A. Littarru, L. Amarú, P. E. Gaillardon, and G. D. Micheli, "Reversible logic synthesis via biconditional binary decision diagrams," in *Int'l Symp. on Multi-Valued Logic*, 2015, pp. 2–7.

[24] M. Soeken, R. Wille, C. Hilken, N. Przigoda, and R. Drechsler, "Synthesis of reversible circuits with minimal lines for large functions," in *ASP Design Automation Conf.*, 2012, pp. 85–92.

[25] M. Soeken, L. Tague, G. W. Dueck, and R. Drechsler, "Ancilla-free

synthesis of large reversible functions using binary decision diagrams," *Journal of Symbolic Computation*, vol. 73, pp. 1 – 26, 2016.

[26] P. Niemann, R. Wille, D. M. Miller, M. A. Thornton, and R. Drechsler, "QMDDs: Efficient quantum function representation and manipulation," *IEEE Trans. on CAD*, vol. 35, no. 1, pp. 86–99, 2016.

[27] A. Abdollahi and M. Pedram, "Analysis and synthesis of quantum circuits by using quantum decision diagrams," in *Design, Automation and Test in Europe*, 2006, pp. 1–6.

[28] M. Soeken, R. Wille, O. Keszocze, D. M. Miller, and R. Drechsler, "Embedding of large Boolean functions for reversible logic," *CoRR*, 2014. [Online]. Available: http://arxiv.org/abs/1408.3586

[29] R. Wille, M. Soeken, D. M. Miller, and R. Drechsler, "Trading off circuit lines and gate costs in the synthesis of reversible logic," *INTEGRATION, the VLSI Jour.*, vol. 47, no. 2, pp. 284–294, 2014.

[30] A. Chattopadhyay, C. Chandak, and K. Chakraborty, "Complexity analysis of reversible logic synthesis," *CoRR*, vol. abs/1402.0491, 2014.

[31] D. Grosse, R. Wille, G. W. Dueck, and R. Drechsler, "Exact multiple-control toffoli network synthesis with sat techniques," *IEEE Trans. on CAD*, vol. 28, no. 5, pp. 703–715, 2009.

[32] O. Golubitsky and D. Maslov, "A study of optimal 4-bit reversible toffoli circuits and their synthesis," *IEEE Trans. on Comp.*, vol. 61, no. 9, pp. 1341–1353, 2012.

[33] M. Soeken and A. Chattopadhyay, "Unlocking efficiency and scalability of reversible logic synthesis using conventional logic synthesis," in *Proceedings of the 53rd Annual Design Automation Conference (DAC)*, 2016, pp. 149:1–149:6.

[34] D. Maslov, C. Young, G. W. Dueck, and D. M. Miller, "Quantum circuit simplification using templates," in *Design, Automation and Test in Europe*, 2005, pp. 1208–1213.

[35] D. M. Miller, R. Wille, and Z. Sasanian, "Elementary quantum gate realizations for multiple-control Toffolli gates," in *Int'l Symp. on Multi-Valued Logic*, 2011, pp. 288–293.

[36] N. O. Scott and G. W. Dueck, "Pairwise decomposition of Toffoli gates in a quantum circuit," in *Great Lakes Symp. VLSI*, 2008, pp. 231–236.

[37] R. Wille, M. Soeken, C. Otterstedt, and R. Drechsler, "Improving the mapping of reversible circuits to quantum circuits using multiple target lines," in *ASP Design Automation Conf.*, 2013, pp. 85–92.

[38] M. Amy, D. Maslov, M. Mosca, and M. Roetteler, "A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 32, no. 6, pp. 818–830, 2013.

[39] D. M. Miller, M. Soeken, and R. Drechsler, "Mapping NCV circuits to optimized Clifford+T circuits," in *Conf. on Reversible Computation*, 2014, pp. 163–175.

[40] M. Amy, D. Maslov, and M. Mosca, "Polynomial-time t-depth optimization of clifford+t circuits via matroid partitioning," *IEEE Trans. on CAD*, vol. 33, no. 10, pp. 1476–1489, 2014.

[41] P. Selinger, "Quantum circuits of t-depth one," *Phys. Rev. A*, vol. 87, 2013.

[42] C. Lin, A. Chakrabarti, and N. K. Jha, "Optimized quantum gate library for various physical machine descriptions," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, vol. 21, no. 11, pp. 2055–2068, Nov. 2013.

[43] ARDA, "Quantum computation roadmap," http://qist.lanl.gov/qcomp_map.shtml.

[44] P. Niemann, S. Basu, A. Chakrabarti, N. K. Jha, and R. Wille, "Synthesis of quantum circuits for dedicated physical machine descriptions," in *Conf. on Reversible Computation*, 2015, pp. 248–264.

[45] A. Muthukrishnan and C. R. Stroud, "Multivalued logic gates for quantum computation," *Physical Review A*, vol. 62, p. 052309, 2000.

[46] Z. Sasanian, R. Wille, and D. M. Miller, "Realizing reversible circuits using a new class of quantum gates," in *Design Automation Conf.*, 2012, pp. 36–41.

[47] R. V. Meter and M. Oskin, "Architectural implications of quantum computing technologies," *J. Emerg. Technol. Comput. Syst.*, vol. 2, no. 1, pp. 31–63, 2006.

[48] M. Ross and M. Oskin, "Quantum computing," *Comm. of the ACM*, vol. 51, no. 7, pp. 12–13, 2008.

[49] A. G. Fowler, S. J. Devitt, and L. C. L. Hollenberg, "Implementation of Shor's algorithm on a linear nearest neighbour qubit array," *Quant. Info. and Comput.*, vol. 4, pp. 237–245, 2004.

[50] M. Saeedi, R. Wille, and R. Drechsler, "Synthesis of quantum circuits for linear nearest neighbor architectures," *Quant. Info. Proc.*, vol. 10, no. 3, pp. 355–377, 2011.

[51] Md. Mazder Rahman, Gerhard W. Dueck, Anupam Chattopadhyay and Robert Wille, "Integrated synthesis of linear nearest neighbor ancilla-free mct circuits," in *Int'l Symp. on Multi-Valued Logic*, 2016.

[52] Laxmidhar Biswal, Chandan Bandyopadhyay, Anupam Chattopadhyay, Robert Wille, Rolf Drechsler and Hafizur Rahaman1, "Nearest-neighbor and fault-tolerant quantum circuit implementation," in *Int'l Symp. on Multi-Valued Logic*, 2016.

[53] Y. Hirata, M. Nakanishi, S. Yamashita, and Y. Nakashima, "An efficient method to convert arbitrary quantum circuits to ones on a Linear Nearest Neighbor architecture," *Quantum, Nano and Micro Technologies. ICQNM*, pp. 26–33, 2009.

[54] A. Shafaei, M. Saeedi, and M. Pedram, "Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures," in *Design Automation Conf.*, 2013, pp. 41–46.

[55] R. Wille, A. Lye, and R. Drechsler, "Exact reordering of circuit lines for nearest neighbor quantum architectures," *IEEE Trans. on CAD*, vol. 33, no. 12, 2014.

[56] R. Wille, O. Keszocze, M. Walter, P. Rohrs, A. Chattopadhyay, and R. Drechsler, "Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits," in *ASP Design Automation Conf.*, 2016, pp. 292–297.

[57] A. Shafaei, M. Saeedi, and M. Pedram, "Determining the minimal number of swap gates for multi-dimensional nearest neighbor quantum circuits," in *ASP Design Automation Conf.*, 2014, pp. 495–500.

[58] A. Lye, R. Wille, and R. Drechsler, "Determining the minimal number of swap gates for multi-dimensional nearest neighbor quantum circuits," in *ASP Design Automation Conf.*, 2015, pp. 178–183.

[59] I. Polian and A. G. Fowler, "Design automation challenges for scalable quantum architectures," in *Design Automation Conf.*, 2015, pp. 1–6.

[60] M. Lukac, M. Kameyama, M. Perkowski, P. Kerntopf, and C. Moraga, "Analysis of faults in reversible computing," in *Int'l Symp. on Multi-Valued Logic*, 2014, pp. 115–120.

[61] I. Polian, T. Fiehn, B. Becker, and J. P. Hayes, "A family of logical fault models for reversible circuits," in *Asian Test Symp.*, 2005, pp. 422–427.

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

[63] J. P. Hayes, I. Polian, and B. Becker, "Testing for missing-gate faults in reversible circuits," in *Asian Test Symp.*, Nov 2004, pp. 100–105.

[64] H. Zhang, R. Wille, and R. Drechsler, "Improved fault diagnosis for reversible circuits," in *Asian Test Symp.*, 2011, pp. 207–212.

[65] N. M. Nayeem and J. E. Rice, "Online testable approaches in reversible logic," *Journal of Electronic Testing*, vol. 29, no. 6, pp. 763–778, 2013.

[66] A. Paler, I. Polian, and J. P. Hayes, "Detection and diagnosis of faulty quantum circuits," in *ASP Design Automation Conf.*, 2012, pp. 181–186.

[67] M. Soeken, S. Frehse, R. Wille, and R. Drechsler, "Revkit: A toolkit for reversible circuit design," *Multiple-Valued Logic and Soft Computing*, vol. 18, no. 1, pp. 55–65, 2012, RevKit is available at http://www.revkit.org.

[68] D. Maslov, "Basic circuit compilation techniques for an ion-trap quantum machine," *CoRR*, vol. abs/1603.07678, 2016.

[69] "Quantum Programming Language," https://quantiki.org/wiki/quantum-programming-language, accessed: 2016-04-08.