Exploring the Potential Benefits of Alternative Quantum Computing Architectures

Arighna Deb *Member, IEEE,* Gerhard W. Dueck *Life Senior Member, IEEE,* and Robert Wille *Senior Member, IEEE*

***Abstract*—*Noisy Intermediate Scale Quantum* (NISQ) computers are becoming a reality thanks to the recent advances made by researchers who physically build such systems. In order to execute corresponding quantum algorithms (usually provided in terms of quantum circuits), certain physical constraints in the architectures need to be satisfied. More precisely, physical constraints restrict the possible interactions between qubits which frequently result in cases where qubits which are supposed to interact in a quantum circuit are not allowed to interact on the physical device. Thus far, this is addressed by dedicated methods that map the logical quantum circuit to a physical realization and satisfy the constraints by inserting further operations. This leads to additional costs which harm the fidelity of the circuit and, hence, urgently need to be avoided.**

Unfortunately, current state-of-the-art approaches for this mapping process take the existing architectures as invariant and only try to reduce the number of additionally needed operations on them. In contrast, (slight) changes in the respectively given architectures (which still keep the underlying physical constraints satisfied) might be possible and may allow for even better (i.e., less costly) mappings. But this potential has not been investigated yet. In this work, we explore this potential. More precisely, we introduce several schemes for generating alternative coupling graphs (and, by this, quantum computing architectures) that still might be able to satisfy physical constraints but, at the same time, allow for a more efficient realization of the desired quantum functionality. Evaluations confirm the potential of those alternative coupling graphs and demonstrate that they can reduce the mapping overhead by up to 60% in the best case and up to almost 40% on average.

I. Introduction

Quantum computers [1] have become a promising alternative to classical computers and show significant superiority over them for certain complex applications. For example, quantum algorithms for large integer factorization [2], [3], searching objects in large unsorted database [4], simulating processes of quantum chemistry, or solving linear equations [5]–[8] have already been proposed and led to substantial speed-ups compared to the existing classical solutions. The supremacy in computational speed is achieved using quantum mechanical phenomena like superposition and entanglement which ultimately provide solutions with substantial speed-up as compared to the classical algorithms.

In recent years, the physical realizations of quantum computers have received significant attention. Several technologies e.g. for ion-traps, superconductors, semiconductor quantum dots, or photonic systems have been used to physically realize real quantum circuits [1]. As of now, the realizations based on superconducting technology [9], [10] stand out since they provide increasing qubit coherence time and better scalability of the technology [11]–[13] compared to other currently considered technologies. Motivated by this, companies such as IBM, Intel, Google, and Microsoft started corresponding developments towards the realization of actual quantum computers for practical purposes [14]–[16]. In fact, IBM introduced its first 5-qubit quantum processor in the beginning of 2017 and later, in mid 2017, further introduced 16-qubit quantum processors. IBM further plans to scale up the number of qubits to 50, while Google and other companies focus on similar developments of quantum chips [15], [17].

Currently, the quantum processors from IBM are widely used since they are accessible to everyone through cloud services [18]. This helps researchers to run their own quantum algorithms (usually represented in terms of circuits) on the IBM quantum computers, known as IBM QX architectures. In order to execute quantum circuits on those architectures, additional steps need to be conducted. These include

1) the decomposition of the initial circuits into elementary quantum operations that are supported by the given architecture and
2) the mapping of the logical qubits from the decomposed quantum circuit to the physical qubits used in the architecture.

For the first step, several solutions exist that decompose arbitrary quantum circuits into a sequence of elementary quantum gates [19], [20]. However, the second step i.e. the mapping of logical qubits used in the given quantum circuit to the physical qubits used in the architecture cannot be done in a one-to-one fashion, because IBM QX architectures have certain physical constraints described by so-called *coupling graphs*.

The physical constraints defined by the coupling graphs allow operations of 2-qubit gates on *selected* qubits only. This means, only certain quantum operations can be applied to the physical qubits of IBM architectures. Besides this, the direction of applying the 2-qubit gate in a pair of qubits is also restricted i.e. the control qubit and target qubit of any 2-qubit gate are also defined by the coupling graph. As a result, the mapping of logical qubits of a quantum circuit to the physical qubits of a quantum architecture is carried out in a fashion such that all operations are conducted and, at the same time, all the physical constraints are satisfied.

Current state-of-the-art methods [21]–[27] address this prob-

lem by inserting additional gates into the circuit in order to re-arrange the qubits and/or to change the control/target connections so that the constraints imposed by the coupling graphs are satisfied. The insertion of additional gates does obviously increase the number of gates in the quantum circuit which results in the reduction of circuit fidelity. This motivates researchers and engineers to focus on developing solutions that aim to derive a proper mapping of logical qubits to physical qubits while, at the same time, keeping the number of additional gates as small as possible.

However, all these approaches have taken the existing architectures as invariant and did not question the corresponding constraints. But, as we will show in more detail in Section IV, there exists further potential. In fact, changing the constraints imposed by the existing quantum computing architectures is a valid option (of course, as long as the underlying physical constrains remain satisfied).

In this work[1], we are exploring this potential. We introduce several schemes for the generation of alternative coupling graphs (and, by this, quantum computing architectures) that still might be able to satisfy physical constraints but, at the same time, allow for a more efficient realization of the desired quantum functionality.

Evaluations confirm the potential benefits of those alternative coupling graphs and demonstrate that they can reduce the mapping overhead by up to 60% in the best case and up to almost 40% on average. This shows that mapping the intended quantum functionality to real quantum devices does not necessarily have to be solely addressed by designers, but can also be supported by the engineers developing the respective architectures. In fact, the results of this work provide motivation for researchers developing quantum computers and architectures to not only take physical constraints into consideration (which of course always have to be satisfied), but also to consider design aspects of the quantum functionality to be executed on the resulting devices.

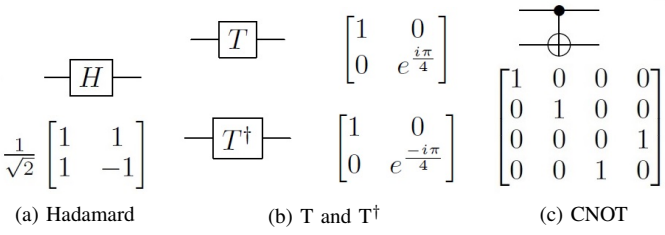
The remainder of this work is structured as follows. In Section II, we review quantum circuits and IBM's QX architectures. In Section III, we discuss the state-of-the-art process of realizing quantum functionality on quantum architectures. This motivates our investigation which is outlined in Section IV. How to explore the potential benefits of alternative quantum architectures is proposed in Section V, while Section VI analyzes the effects the different architectures have on the respective costs. Section VII concludes the paper.

II. Background

In order to make this paper self-contained, this section briefly reviews quantum circuits as well as the quantum architectures commonly used in current NISQ devices.

A. Quantum Circuits

Quantum bits (or *qubits*) are the basic information units in quantum computation [1]. Like bits in traditional computation which can assume states 1 or 0, a qubit can also assume two basis states, $|1\rangle$ or $|0\rangle$. In addition, a qubit can also have a superposition of the basis states i.e. $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ with $|\alpha|^2 + |\beta|^2 = 1$, where α and β are complex values. Any superposition state $|\psi\rangle$ of a qubit also represents a vector $\begin{pmatrix}\alpha\\ \beta\end{pmatrix}$ in a two-dimensional Hilbert space. The tensor product of n such qubits i.e. $\begin{pmatrix}\alpha\\ \beta\end{pmatrix}^{\otimes n}$ allows concurrent representations of 2^n possible basis states. This parallelism greatly reduces the complexity of some algorithms on quantum computers compared to classical ones.



(a) Hadamard (b) T and T† (c) CNOT

Fig. 1: Clifford+T gate library

The manipulations of the qubit states are generally carried out using quantum circuits. A quantum circuit is composed of quantum gates, where each gate represents a quantum operation. A gate can either involve one or two qubits. In the case of two-qubit quantum gates, one qubit is called the target and other is called the control.

Several quantum gates such as 1-qubit Pauli *X, Y, Z*, *Hadamard (H)*, *Phase (S, T)* and 2-qubit *controlled NOT (CNOT)*, *square-root-of-not (CV and CV†)*, etc. exist in the literature [29]. Here, the *Clifford+T* gate library [19], [30] composed of the 1-qubit Hadamard (H) gate, T (phase shift by $\frac{\pi}{4}$) gate, and 2-qubit CNOT gates represents a universal gate library, i.e., all quantum operations can be performed by circuits composed of gates from this library. Fig. 1 shows the symbols and corresponding unitary matrices for the gates from the Clifford+T library. In order to realize an efficient quantum circuit, the gate count, i.e., the total number of quantum operations of the circuit, must be kept as low as possible.

Example 1. *Fig. 2 shows an example of a quantum circuit composed of four qubits and six gates. The boxes labeled H and T represent the single qubit gates* H *and* T*, respectively. The control and target qubits of the CNOT gates are denoted by • and ⊕, respectively. First, a T (phase shift by $\frac{\pi}{4}$) operation is applied on qubit q_0. Then, a CNOT operation with control qubit q_0 and target qubit q_1 is performed – followed by a Hadamard (H) operation that is applied on qubit q_1. Finally, three more CNOT gates (denoted as g_4, g_5, g_6) are applied.*

[1] A preliminary version of this work has been presented at [28].

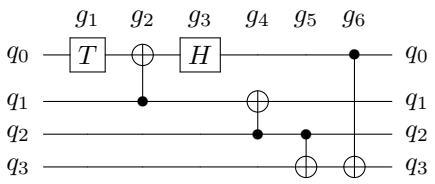


Fig. 2: Quantum circuit

B. Quantum Architectures

In 2017, IBM launched a cloud-based, open-access platform to experimentally execute circuits on quantum processors under the project IBM Q [18]. This platform includes different quantum processors such as 5-qubit quantum processors (IBM QX2 and IBM QX4) and 16-qubit quantum processors (IBM QX3 and IBM QX5) running as backend in the cloud, on which quantum algorithms can be executed. To this end, quantum algorithms to be executed (usually provided in terms of quantum circuits) have to be composed of elementary quantum gates only. IBM's quantum architectures support 1- and 2-qubit elementary gates such as the Hadamard (H) gate, the Pauli X, Y, Z gates, the Phase gates, the square-root of Phase gates, CNOT, and controlled U gates. If any quantum circuit contains non-elementary quantum gates (e.g., the Toffoli gate, Fredkin gate, etc.), then such gates must be replaced by elementary quantum gates supported by the architectures. Several methods for decomposing the desired quantum functionality to an elementary gate library can be found in the literature [19], [20].

Moreover, prior to execution, the circuits have to be mapped to the selected quantum architecture. However, there are some constraints that need to be satisfied before mapping any quantum circuit to the intended architecture. In fact, 2-qubit quantum gates such as CNOT can only be applied between certain pairs of qubits. Furthermore, for each pair of qubits, which qubit will work as the control and which one will work as the target are firmly specified. This restriction is known as *CNOT-constraints*, and are usually described in terms of a *coupling graph* which depicts the layout of the quantum architecture. More formally, a coupling graph $A = (Q, E)$ over physical qubits $Q = \{Q_0, Q_1, \cdots, Q_{n-1}\}$ is a directed graph consisting of a set of vertices Q and a set of edges $E = \{(Q_u, Q_v), Q_u, Q_v \in Q, Q_u \neq Q_v\}$ representing a 2-qubit operation with the qubits Q_u and Q_v being the control and target, respectively.

Example 2. *Fig. 3 shows the coupling graph representing the restrictions of the IBM QX5 architecture. As can be seen, the architecture has 16 physical qubits represented by vertices with labels Q_0 to Q_{15}. The edges e_1 to e_{22} in the graph represent the connections between the qubits. An edge e_1 pointing from physical qubit Q_1 to qubit Q_0 indicates that a CNOT with control qubit Q_1 and a target qubit Q_0 can be applied here. Similarly, all other edges define the other allowed qubit interactions. All remaining interactions are prohibited.*

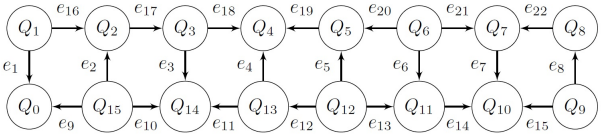


Fig. 3: Quantum architecture IBM QX5

III. Current Design Process for Realizing Quantum Functionality

In order to realize a quantum functionality on real quantum computers, two major steps usually have to be conducted:

1) *Decomposing Quantum Functionality to Elementary Operations:* Most quantum algorithms are first described in high-level quantum programming languages such as Quipper [31], Scaffold [32], or OpenQASM [33]. Since given quantum architectures usually only support 1-qubit and 2-qubit elementary gates/quantum operations such as Hadamard (H) gate, the Pauli X, Y, Z gates, the Phase (S) gate, the square-root of Phase (T) gate, CNOT and controlled U gates, the high-level operations first have to be decomposed/compiled. To this end, several decomposition or synthesis approaches such as those proposed in [19], [20], [34], [35] and [36]–[38] exists, respectively. Corresponding methods are also available in tools for Quipper [31], the ScaffCC compiler for the Scaffold language [32], [39], RevKit [40], or JKQ [41].
2) *Satisfying Restrictions on Qubit Interaction*: Even if the desired quantum functionality is provided in elementary operations, the problem remains that given architectures substantially restrict the allowed interactions between quibts (as reviewed in Section II-B). This frequently yields situations where gates of quantum circuits cannot be executed directly. Current methods address this problem by adding additional gates that re-arrange qubits and/or change control/target connections so that they are eventually in line with the constraints imposed by the quantum architecture/coupling graph. Corresponding approaches first addressed so-called nearest neighbor-constraints (see, e.g., [42]–[49]), but recently also approaches for CNOT-constraints have been proposed (see, e.g., [24], [50], [51]).

In this work, we are focusing on the second step, which is one of the biggest problem in efficiently realizing quantum functionality on real devices. To this end, we have a closer look on how restrictions on qubit interaction are currently handled:

Example 3. *Consider the circuit from Fig. 2 which is to be realized on the IBM QX5 quantum computer. That is, constraints as defined by the coupling graph shown in Fig. 3 have to be satisfied. By directly mapping each logical qubit q_i to a physical qubit Q_i, the first three gates are supported. However, gate g_4 and gate g_6 cannot be realized under the given constraints, because an interaction between Q_2 and Q_1 is only possible if Q_2 is the target and Q_1 is the control (which*

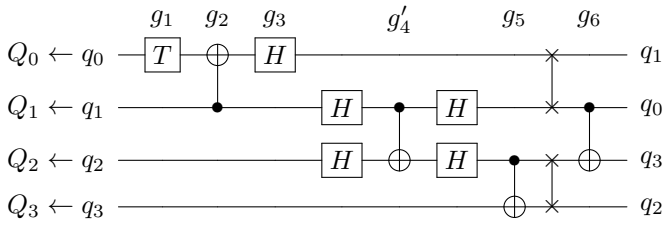


Fig. 4: Mapped circuit (assuming coupling graph from Fig. 3)

is the opposite in g_4) and because no interaction is allowed between Q_0 and Q_3 at all (which is required in g_6).

These issues can be addressed by adding four H gates that flip the respective control/target connections of a gate and, by this, satisfy the constraints for gate g_4 as shown in Fig. 4. Furthermore, SWAP gates (which can be realized by seven elementary gates) can be applied. SWAP gates exchange two qubit values and, by this, allow the "move" of qubit values from one physical position to another. This is used to satisfy the constraint for gate g_6 as also shown in Fig. 4. All these adjustments require 18 additional elementary gates (four H gates and two SWAP gates which need to be realized by seven elementary gates each). Finally, the gate count for the IBM QX5 architecture has increased by a factor of 3.

Obviously, reducing this overhead is a priority during the design process. Accordingly, several methods for realizing quantum functionality under these constraints have been recently proposed (see e.g. [21]–[27], [51]). In [21], the authors partition a given circuit into a number of sub-circuits for each of which they determine a suitable placement for the qubits. Then, they connect the individually placed sub-circuits to form the final circuit using a swapping/permutation technique. The solution presented in [23] generates an initial mapping of logical qubits to physical qubits and then reorders them to satisfy the constraints stated above. The work in [25] partitions the circuit into a number of smaller sub-circuits and, then, applies initial mapping and reordering schemes such that the mapping order remains the same for all the sub-circuits while satisfying all the constraints. In [26], a unique heuristic search technique is introduced, to determine an efficient initial mapping of logical qubits to physical qubits such that the final quantum circuits have less gate overhead. A gate-fidelity aware technology mapping scheme is proposed in [27] which also focuses on the optimization of gate overhead. Also exact solutions that produce a minimum number of SWAP and H operations, i.e., a minimum overhead have been proposed [24], [50]. These solutions employ SAT solvers to generate the best qubit mappings which guarantee a minimal overhead with respect to H and SWAP gates.

However, even if all these solutions are able to reduce the overhead, they are still bounded by the restrictions imposed by the given architecture. As a result, many resulting quantum circuits still have a substantial overhead caused by a large number of additionally required gates. This is a serious drawback since the total number of gates significantly affects the fidelity of the result. In fact, studies have shown that, if the gate overhead gets too large, the intended result cannot be determined anymore because of the noise introduced by them [52]. Hence, further solutions to address this problem are needed.

IV. Motivation and General Idea

In this work, we propose an alternative direction to satisfy restrictions on qubit interactions which goes beyond currently considered schemes. In fact, we observe that, thus far, the realization of quantum functionality onto real quantum computers has been conducted by simply taking the existing architectures as invariant and not questioning the corresponding constraints. This does not only yield a significantly more complex design process (in fact, realizing a given quantum functionality to a given architecture has been proven to be $\mathcal{NP}$-hard [53]), but also leads to the substantial overhead discussed in the previous section.

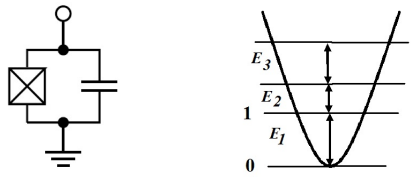
In the following, we present an alternative idea which aims at changing the constraints themselves so that a better realization, i.e., a realization with less overhead, can be determined. Since arbitrarily changing constraints imposed by physical requirements are obviously not an option, we first discuss the physical constraints that are actually imposed by current quantum architectures. Afterwards, we outline the potential benefits that can be exploited when some degree of freedom on the design of quantum architectures is exploited (while the physical constraints remain satisfied). Finally, we discuss possible concerns which might come with this idea.

A. Physical Constraints

In order to consider the physical constraints of quantum computing technology, we briefly discuss the physical realization of quantum computers. For this purpose, we choose quantum superconducting (cf. [9], [10]) as a representative technology. Quantum superconducting is used in many quantum computers accessible today. Furthermore, the corresponding constraints are similar in other technologies. Hence, choosing it as the representative allows for valid conclusions for several quantum computing technologies.

Here, each qubit is realized as an artificial atom using a non-linear inductor-capacitor circuit as depicted in Fig. 5(a). While the linear inductor-capacitor circuit results in harmonic potential which ultimately produces equally spaced energy levels, in contrast, the non-linear elements lead to anharmonicity which results in unequally spaced energy levels [54]. Typically, the two lowest energy levels are considered to define the computational basis state of a qubit- ground state represents logic 0 and the next excited state represents logic 1 as shown in Fig. 5(b). Besides that, the different anharmonic oscillators also allow the addressing of individual qubits. As a result, in a multi-qubit quantum computer, each qubit has a unique frequency [55].

The two qubit interactions between fixed frequency qubits are realized by one or more microwave pulses [56], [57] for which the resonance couplers are used. In other words, the couplers establish a cross-resonance coupling between



(a) Circuit model (b) Energy levels

Fig. 5: Physical model of a superconducting qubit

two qubits – allowing to realize 2-qubit quantum operations. Hence, each edge in the coupling graph could, in principle, be realized by such resonance couplers. Note that the coupling graph for IBM QX5 as shown in Fig. 3 has 22 edges, i.e., describes 22 qubit interactions for which 22 couplers are required.

Typically, in the case of 2-qubit gates, the qubit with high frequency is used as the control qubit and the qubit with low frequency is used as the target qubit to achieve high fidelity [58], [59]. Exceptions to this high frequency control and low frequency target arise when the qubits are degenerated or there is an interference between coupling qubits and other qubits with low frequency.

This establishes couplings between two qubits and, by this, supports operations on the target qubit based on the state of the control qubit [60] and thus realizing 2-qubit operations such as CNOT. However, such a strong coupling is only possible between two qubits and can only be established if the qubits are next to each other (otherwise, the qubits may degenerate which results in a gate operation with very low fidelity). Eventually, this led to quantum computer architectures with constraints expressed by coupling graphs.

In general, the coupling constraints are not implicitly defined. In other words, the coupling graphs, to some extent, are defined arbitrarily while defining the quantum architectures. As a result, there exists some degree of freedom on the design of quantum architectures which we discuss next.

B. Potential Benefits

Reducing the gate overhead caused by the need of satisfying the constraints from physical realizations obviously is the main objective of all solutions introduced thus far for quantum circuit realization. However, even if minimal overheads can be achieved, their impact on the reliability of the resulting computations remains substantial. Hence, to further improve realizations, more avenues need to be explored. Changing the constraints imposed by the existing quantum computer architectures (and described by the coupling graphs) seems to be a promising direction. Since those constraints resulted from physical requirements, they have been taken as invariant and were not questioned thus far. Herein, we show that, even if we recognize that physical constraints have to be satisfied, some degree of freedom exists. This allows for the design of valid alternative quantum architectures onto which certain quantum circuits can be realized with much less gate overhead than before.

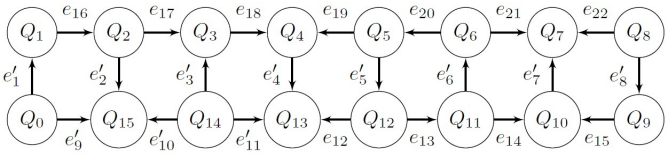


Fig. 6: Coupling graph of an alternative architecture

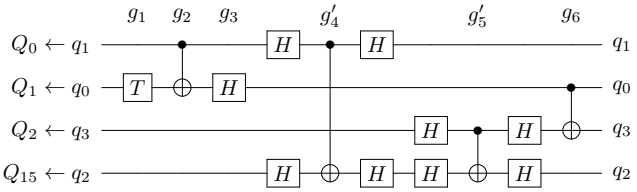


Fig. 7: Mapped circuit (assuming coupling graph from Fig. 6)

However, it is obvious that those characteristics do not necessarily have to lead to quantum architectures as available thus far. In fact, a coupling between qubits that follow these characteristics can be established in numerous fashions. This allows one to determine architectures with coupling graphs that are much more suited for quantum circuits to be executed on them. An example illustrates the idea.

Example 4. *Fig. 6 shows the coupling graph for an alternative architecture that also satisfies the physical constraints discussed above. In fact, this coupling graph is almost identical to the coupling graph for the IBM QX5 architecture shown before in Fig. 3, but differs in the directions of the edges. Despite these minimal differences (which should not pose any obstacles with respect to a physical realization), this allows one to map the quantum circuit from Fig. 2 with much less overhead as shown in Fig. 7. In fact, rather than 18 additional gates, only eight additional gates are needed – a reduction of the overhead by 55%.*

This example sketches the potential benefits in the design of quantum architectures: Rather than only satisfying physical constraints (which, of course, always remains a primary objective), it should also be considered how good/bad a derived architecture is able to realize quantum functionality. However, thus far, those potential benefits have not been investigated and it remains unknown whether such an additional consideration indeed will lead to improvements.

C. Possible Concerns

Motivated by the idea and example from above, the remainder of this work provides methods and results exploring the illustrated potential benefits. Before doing so, we briefly discuss some possible concerns that may stem from this idea.

First, the most obvious issue is whether it is too early for the proposed idea. In fact, the development of quantum architectures is intensely considered and several advancements have been made. This can be observed in the emergence of different (and constantly improving) physical realizations. But the question remains whether it is better to wait until an established

quantum realization eventually crystallizes (from which coupling graphs and corresponding architectures can be derived). While this certainly is a valid argument, it puts designers of those architectures in a kind of "chicken-and-egg"-situation: Either first clarify all physical options or first perform the evaluation on the possible impact and use this as motivation for physicists to investigate further towards this direction. In the first case, the full impact of the design choice may not be known. Whereas in the second case, some dead end paths may have been explored. We propose a compromise between these two approaches and start evaluating the potential benefits from a design automation perspective, while, at the same time, constantly keeping the current (and possible future) limitations in mind. Eventually, this leads to first approaches and results (summarized in the remainder of this paper) which might not perfectly address all concerns, but already start an evaluation of the potential.

Following this argument, another issue might be whether the physical constraints we are assuming in this work (cf. Section IV-A) are indeed valid and whether the changes in physical architectures as illustrated, e.g., in Example 4, can indeed be implemented. While this certainly is a physical and not a design question, current developments show that various interactions between qubits can physically be realized. However, this does not mean that *arbitrary* interactions always become possible. But, there is a degree of freedom and exploiting this freedom while, at the same time, trying to restrict the generation of alternative coupling graphs as much as possible is worthwhile to eventually determine an architecture that does not not only satisfy physical constraints but also reduces the costs.

Finally, the question might arise whether the resulting coupling graphs (and corresponding architectures) are circuit-specific, i.e., whether the changes in the coupling graph would be beneficial only for some selected quantum circuits and not for arbitrary quantum circuits. While this indeed might be the case, we have the same situation in the classical realm, where, designers often choose architectures depending on the applications to be executed on them. Still, this is much better than just taking an "arbitrary" architecture. In fact, many design evaluations involve benchmark circuits, i.e., it is important to see how the architecture performs on a set of circuits. The idea proposed in this work at least allows one to evaluate the possible effects—which is the main motivation of our work. Moreover, our evaluations (summarized later in Section VI) show that, even for a broad variety of quantum circuits, best case, average case, and worst case reductions in gate overhead for the alternative architectures are better as compared to the existing architecture.

Hence, despite these concerns, we believe that it is worthwhile to start evaluating the potential of alternative quantum computing architectures. To that end, we present and evaluate some possible strategies in the following. While our elaboration might not provide a comprehensive view taking all physical constraints into consideration, it showcases the potential of the proposed idea and provides a motivation for researchers developing quantum computers and architectures to not only take physical constraints into consideration (which of course always have to be satisfied), but also to consider design aspects of the quantum functionality.

V. Exploring the Potential Benefits

In a naive fashion, exploring the potential benefits sketched above is easy. One just needs to generate alternative coupling graphs and map the respective quantum circuits to it in order to see whether this yields more efficient results than when, e.g., IBM's QX5 is considered as the coupling graph. However, exploring the potential benefits using "arbitrary" coupling graphs is meaningless (in this case, a complete graph where all qubits may arbitrarily interact with each other will be the best but also physically most unrealistic solution). Hence, we consider alternative schemes for coupling graph generation that, on the one hand, allow one to explore the potential benefits while, on the other hand, remain as close to the characteristics of existing quantum computing architectures (and, by this, most likely will also be physically possible). In the following, the considered schemes are described. Afterwards, Section VI summarizes the exploration conducted with graphs obtained by these schemes.

A. Flipping Edges of Existing Coupling Graphs

The first approach to generating alternative coupling graphs involves the minor modification of the existing coupling graph in order to still satisfy the physical constraints/restrictions discussed in Section IV-A. To this end, we consider an existing coupling graph (namely the one for IBM QX5) as a basis for generating a modified one[2]. The modification is done by randomly reversing the directions of selected edges that exist in the given coupling graphs. More precisely, given an existing coupling graph $A = (Q, E)$, we randomly choose an edge $e_{i,j} \in E$ pointing from qubit Q_i to qubit Q_j ($Q_i, Q_j \in Q$) and flip its direction which results in an edge $e_{j,i}$, now pointing from qubit Q_j to Q_i. In a similar fashion, the directions of the other edges in the graph can also be reversed. The choice of the edges to be flipped is done in a purely random fashion.

Example 5. *Consider the coupling graph for the IBM QX5 architecture as shown in Fig. 3. Applying the scheme described above may lead to an alternative coupling graph as shown in Fig. 6. As already discussed in Example 4, this reduces the overhead by 55% from 18 to eight additional gates for the case of the circuit from Fig. 2.*

In the following, this scheme is denoted by *Flipped Edges*. Architectures described by corresponding coupling graphs can most likely be physically realized. In fact, reversing an edge between two qubits requires no additional hardware except for some minimal adjustments. More precisely, since qubit interactions can physically be realized using resonance couplers (cf. Section IV-A), each edge in an (alternative) coupling graph could, in principle, be realized by such resonance couplers. In the case of the example from above, the same number of interactions need to be realized than in the original architecture

[2]Note that we considered the IBM QX5 architecture as a baseline to sketch the general ideas of the proposed schemes. However, the methods proposed here can easily be extended to other architectures as well.

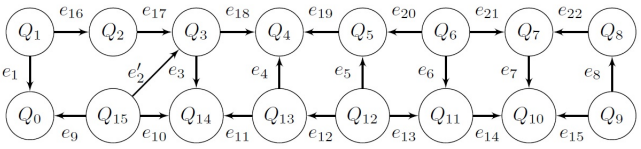


Fig. 8: Coupling graph determined by random modifications

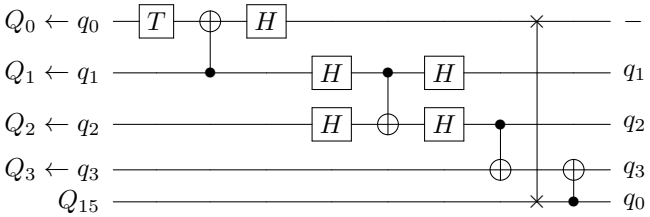


Fig. 9: Mapped circuit (assuming coupling graph from Fig. 8)

and, hence, the same number of couplers are needed – only the direction of some of them needs to be changed. As far as physical realization is concerned, this requires a small adjustment in the placement of the couplers which also seems quite pragmatic since (1) the qubit arrangements are such that the changes result in a graph with no other qubit in between (allowing the possibility of qubit-qubit interaction between them) and (2) there exists no connection between other diagonally located qubits (which removes the possibility of crossing of connections). This means that the physical realizations require no additional hardware, but only slight adjustments.

B. Random Modifications

While the coupling graphs generated by the above scheme differ from the existing graphs with respect to the directions of the edges, i.e., only minor modifications are made, more substantial modifications can be made by a random approach which is proposed as a second scheme to generate alternative coupling graphs. We again consider an existing coupling graph (such as the IBM QX5) as a basis for generating an alternative one. The modification is done by randomly adding and removing edges that exist in the considered coupling graph. More precisely, given an existing coupling graph $A = (Q, E)$, we randomly select a qubit Q_i, its adjacent qubit Q_j followed by a qubit Q_k which is adjacent to Q_j ($\{Q_i, Q_j, Q_k\} \in Q$). Based on the edges with outward direction, we order the nodes as control qubit to target qubit. Without the loss of generality, assume that an edge points from Q_i to Q_j, while another edge points from Q_k to Q_j resulting the order $Q_i > Q_j < Q_k$. Next we remove an existing edge between Q_i and Q_j and add an edge either pointing from Q_i to Q_k or vice versa.

Example 6. *Consider again the coupling graph from Fig. 3. Applying the scheme sketched above, we choose qubits Q_{15}, Q_2, and Q_3 which are adjacent to each other (see Fig. 3). Now, we remove the edge pointing from Q_{15} to Q_2 and add an edge pointing from Q_{15} to Q_3. This leads to an alternative coupling graph as shown in Fig. 8. Using this coupling graph, the circuit from Fig. 2 can be realized as shown in Fig. 9. Rather than 18 additionally gates (needed in the case of the IBM QX5 architecture), this requires only eleven additional gates – a reduction of the overhead by 39%.*

In the following, this scheme is denoted by *Moved Edges*. Architectures described by correspondingly obtained coupling graphs can also most likely be physically realized. In fact, the physical realizations of the resulting architectures obtained from the scheme Moved Edges require a small adjustment in the placement of the couplers which also seems quite pragmatic since (1) the order of the qubits allows the possibility of interaction between diagonally located qubits and (2) there exists no connection between other diagonally located qubits which removes the possibility of crossing of connections. In fact, the required changes are very similar to the *Flipped Edges* scheme and means that the physical realizations of proposed architectures (for e.g., random modifications) require no additional hardware as the qubit arrangement and the number of qubit interactions remain the same as for the original architecture. That is, the resulting graphs have the same number of edges/connectivity as that of other (existing) architectures. Therefore, the resulting coupling graphs satisfy the constraints discussed in Section IV-A, which make these architectures likely physically realizable.

C. Function-specific Generation with Restrictions

The schemes proposed above generate various coupling graphs and, by this, allow us to evaluate the effects coupling graphs have, in general, on the final quantum circuits. Nevertheless, in order to work towards the development of coupling graphs/architectures, schemes considering the quantum functionality to be executed in the resulting architecture can be of interest. To this end, we first determine how often which qubit pairs interact in the quantum functionality to be executed on the architecture (this can be obtained from a representative quantum circuit or defined by the designer). Based on the number of interactions between the pairs of qubits, edges are added between the corresponding pairs, which ultimately generate a coupling graph. More precisely, for n logical qubits $\{q_0, q_1, \cdots, q_{n-1}\}$ of a given quantum circuit G, we add n physical qubits Q_i (where, $i = \{0, 1, \cdots, n-1\}$) in the coupling graph $A = (Q, E)$. Then, an edge $e_{i,j}$ is added to E with a point of direction from physical qubit Q_i to Q_j depending on the 2-qubit gate $g_l(q_i, q_j) \in G$ where, q_i and q_j denote control and target qubits respectively. In a similar manner, the other edges are applied between physical qubits based on the remaining 2-qubit gates $g_m(c, t) \in G$ resulting in a coupling graph $A = (Q, E)$.

However, adding an edge between qubits based on every 2-qubit gates in a given circuit leads to a coupling graph where all qubits may interact with each other. According to the current physical constraints, this is unrealistic. To avoid such unrealistic coupling graphs, we enforce two restrictions to the graph: (1) only one edge exists between two qubits Q_i and Q_j

and (2) each qubit Q_i has an out-degree of 2. The choice of the 2 target qubits for any qubit Q_i is made depending on how frequently 2-qubit operations occur between Q_i and the corresponding target qubits. Without loss of generality, assume that the quantum gates $\{g_{l_1}(q_i, q_j), g_{l_2}(q_i, q_k), \cdots, g_{l_k}(q_i, q_m)\} \in G$ occur T_1, T_2, T_3 times ($T_1 \geq T_2 \geq T_3$) respectively, then we only add edges between qubits Q_i to Q_j and Q_i to Q_k as T_1 and T_2 are higher than that of T_3 resulting in no edge pointing from Q_i to Q_m.

Overall, the above idea yields the quantum architectures through the generation of coupling maps as follows:

1) Traverse a quantum circuit $G = \{g_1(c,t), g_2(c,t), \cdots, g_p(c,t)\}$ over n logical qubits $q = \{q_0, q_1, \cdots, q_{n-1}\}$ from left to right.
2) Create a list L.
3) For each 2-qubit gate $g_l(c,t) \in G$ with $c, t \in \{q_0, q_1, \cdots, q_{n-1}\}$ and $c \cap t = \emptyset$, store control qubit, target qubit and number of interactions in $L.control, L.target$ and $L.count$ respectively.
4) Sort the list L in descending order of the number of interactions.
5) Add n physical qubits $Q_0, \cdots, Q_{n-1}$ in a graph $A = (Q, E)$.
6) By traversing list L, for each $L.control$ check if the vertex Q_i stored in $L.control$ has out-degree less than 2. If this check fails, continue with the next $L.control$ in the list L. Otherwise, add an edge $e_{i,j}$ ($e_{i,j} \in E(A)$) between vertices Q_i and Q_j stored in $L.control$ and $L.target$ respectively, where, Q_i and Q_j act as control and target qubits respectively. Increment the current out-degree count of Q_i by 1.

Example 7. *Consider the circuit from Fig. 2 for which a coupling graph is to be generated. Applying the scheme described above, for each logical qubit q_i, we consider a physical qubit Q_i. Now, we add an edge pointing from Q_1 to Q_0 by considering the gate g_2 with q_1 as control and q_0 as target. In a similar manner, edges from Q_2 to Q_1 and to Q_3, and Q_0 to Q_3 are added according to the gates g_4, g_5 and g_6 respectively. This leads to the coupling graph depicted in Fig. 10*[3]*. Using this coupling graph, the circuit from Fig. 2 can be realized with no additional gates*[4]*.*

In the following, this scheme is denoted by *Function Specific*. Architectures described by correspondingly obtained coupling graphs can also most likely be physically realized. In fact, the restrictions on out-degrees of the qubits allow limited qubit interactions. In addition to this, only one-way interactions between the pairs of qubits are allowed which is commonly the case in other existing architectures. Hence, using the function-specific scheme, if significant reductions in the overhead can be achieved, the resulting alternative architectures may provide better solutions than the architectures available thus far.

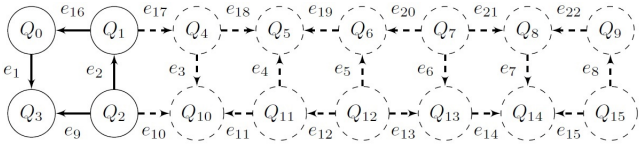


Fig. 10: Dedicated coupling graph for circuit from Fig. 2

Please note that, although the architectures determined by following this scheme are optimized for a single (representative) function, it does not mean that no other quantum functions can be executed better on this alternative. In fact, right now, all existing architectures are designed without taking *any* quantum circuit (or information about the application to be executed on the desired device) into consideration. The scheme proposed here at least allows to evaluate the possible effects. Moreover, as the evaluations summarized later in Section VI confirm, that this frequently leads to architectures that work well for the function used as representative, but also for other functions.

VI. Experimental Evaluation

Using the schemes described above, we evaluated the potential benefits of different quantum architectures. To this end, we generated alternative coupling graphs/architectures (using the proposed schemes) and mapped different quantum functionality on them. Afterwards, we compared the resulting costs with those of an existing architecture, namely IBM's QX5. By this, we were able to evaluate the effects the alternative coupling graphs have on the respective costs and the potential benefits which might be available when a coupling graph is generated and when additionally taking the desired functionality into consideration. In this section, we report and discuss the results obtained by our evaluations. We first review the setup of the case study and, afterwards, we discuss the obtained results.

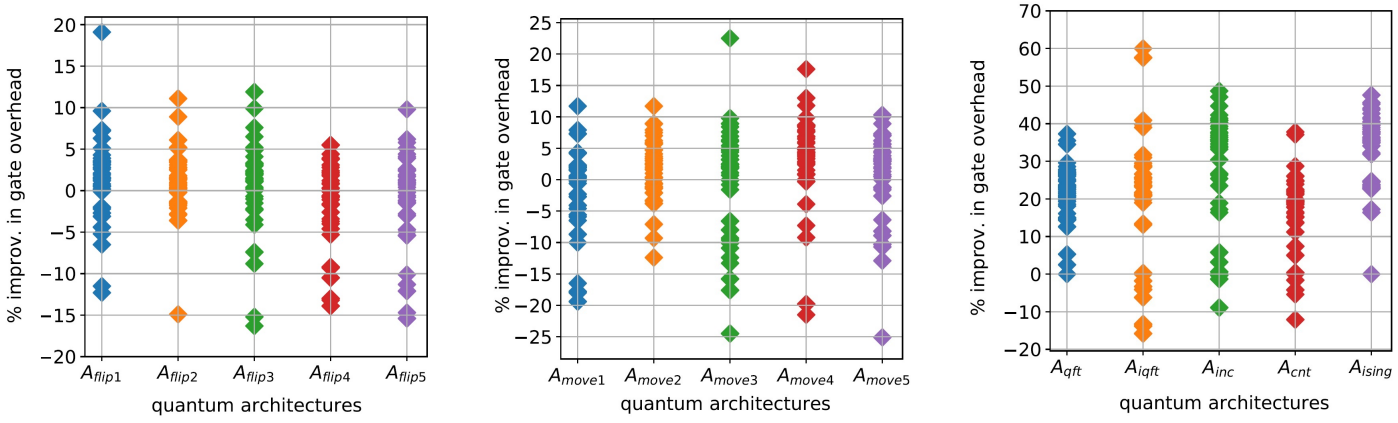
A. Setup and Results

We used IBM's QX5 (denoted as *QX5*) as a baseline and generated a total of five alternative coupling graphs using scheme *Flipped Edges* (resulting architectures are denoted $A_{\texttt{flip1}}$ to $A_{\texttt{flip5}}$), a total of five alternative coupling graphs using scheme *Moved Edges* (resulting architectures are denoted $A_{\texttt{move1}}$ *to* $A_{\texttt{move5}}$), and a total of five alternative coupling graphs using Scheme *Function Specific*. For the latter scheme, the benchmarks *qft_16*, *iqft_16*, *inc_237*, *cnt3-5_180*, and *ising_16* (taken from [1], [61]) have been used as representative quantum functionality (accordingly, the resulting architectures are denoted $A_{\texttt{qft}}$, $A_{\texttt{iqft}}$, $A_{\texttt{inc}}$, $A_{\texttt{cnt}}$, and $A_{\texttt{ising}}$, respectively). We have chosen these five representative functions, because, out of 50 benchmarks with the number of qubits ranging from 10 to 16, only these five functions are composed of 16-qubits – allowing one to realize all the other considered benchmarks.

For the Scheme *Flipped Edges* and the Scheme *Moved Edges*, we randomly choose edges of the original architecture

[3]To stay in line with the goal of generating a 16 qubit architecture, we have randomly generated the rest of the qubits Q_4 to Q_{15} which are shown with dashed circles/lines. However, we do not do that for larger circuits and the evaluations in Section VI later considers only 16-qubit functionality from which "full" 16-qubit architectures are generated.

[4]This is an ideal case. Usually, also the coupling graphs resulting from this scheme lead to additional gate overhead (see Section VI).



(a) Flipped Edges (b) Random (c) Function Specific

Fig. 11: Obtained results

to flip or move, respectively. Since there are 22 edges in the original architecture, the total number of alternatives would be 22!, i.e., significantly large. Hence, rather than checking out this exponential number of possibilities, we randomly create few alternative architectures by either flipping or moving the edges and, then, apply them to realize the quantum circuits. More precisely, we randomly generated 10 alternative architectures. As an example, Fig. 6 shows an alternative architecture which is obtained by flipping 11 edges ($e_1^{'}$ to $e_{11}^{'}$) of the QX5 architecture shown in Fig. 3. This alternative architecture is named $A_{\texttt{flip1}}$. In a similar manner, the other architectures are obtained using *Flipped Edges* and *Moved Edges* schemes.

On all architectures (the original architecture *QX5* and all newly generated ones), we realized a total of 50 quantum functions which have been taken from [61] and were used in previous work to evaluate NISQ mapping methods. The considered benchmarks are composed of 1- and 2-qubit quantum gates such as Hadamard (H), T, CNOT (i.e., they are composed of gates from the Clifford+T library). Each benchmark is mapped to the quantum architectures using the mapping scheme reported in [25] (which is also publicly available and anyone can re-produce the results). The resulting realizations have been generated on a computer with an Intel(R) Core(TM) i5-6200U CPU 2.40 GHz and 8 GB RAM. The run-times range from less than a second up to 3 minutes in the worst case[5]. Afterwards, we determined the improvement or degradation of the realizations from the newly generated coupling graphs compared to the baseline QX5.

The results are summarized in the plots shown in Fig. 11 as well as in Table I. The plots in Fig. 11 provide the relative improvement/degradation in the gate count (y-axis) which was obtained when realizing the benchmarks on the architectures $A_{\texttt{flip1}}$, ..., $A_{\texttt{flip5}}$ (x-axis of Fig. 11(a)), the architectures $A_{\texttt{move1}}$, ..., $A_{\texttt{move5}}$ (x-axis of Fig. 11(b)), as well as the architectures $A_{\texttt{qft}}$, $A_{\texttt{iqft}}$, $A_{\texttt{inc}}$, $A_{\texttt{cnt}}$ and $A_{\texttt{ising}}$ (x-axis of Fig. 11(c)). Table I reports the best improvement, average improvement/degradation, and worst improvement/degradation for each architecture.

Note that we compare the improvement/degradation with respect to the gate count. Other metrics such as T-count/T-depth or fidelity exists as well, but do not provide substantially further insights for our evaluations here. In fact, the T-count/T-depth does not change since no additional T gates are required when an initial quantum circuit is mapped to any given quantum architecture. Similarly, the fidelity depends on the gate count as well (i.e., the larger the number of gates, the less the resulting fidelity). That is, although there might be slight differences, the general conclusions would remain the same. Hence, for sake of simplicity we focus on gate count only.

B. Discussion

The results clearly confirm the thesis of this work: Coupling graphs and, by this, the respectively defined quantum architectures have significant impacts on the costs of the quantum circuits realized for them. Both, small changes (as in Scheme *Flipped Edges*; cf. Fig. 11(a)) as well as larger changes (as in Scheme *Moved Edges*; cf. Fig. 11(b)) may lead to improvements but also degradation in the costs of up to approx. 25%. More precisely, Table I demonstrates that, out of 10 alternative architectures, seven architectures ($A_{flip1}, A_{flip2}, A_{flip5}, A_{move2}, A_{move3}, A_{move4}, A_{move5}$) are better than the QX5 architecture, i.e., these architectures, on average, realize better quantum circuits with less gate overhead as compared to the QX5 architecture . In the best cases, these architectures lead to a reduction in gate overhead by up to 22%, however, in the worst cases, an increase in the gate overhead by up to 25% is obtained as well. This is a significant effect and clearly supports the idea that not only physical aspects should be considered (which of course have to be satisfied and,

[5]Note that, due to page limitations and since the run-time efficiency of the respectively applied mapping method is no concern of this case study, we omitted a detailed documentation of the run-times. All results, however, will be made publicly available in an online repository in a possible final version.

TABLE I: Best, average, and worst case performance

architecture name	% improvement in gate overhead		
	Best case	Avg. case	Worst case
$A_{\texttt{flip1}}$	19.1%	2.7%	-12.3%
$A_{\texttt{flip2}}$	11.1%	1.1%	-14.9%
$A_{\texttt{flip3}}$	11.9%	-1.1%	-16.3%
$A_{\texttt{flip4}}$	5.5%	-0.7%	-13.9%
$A_{\texttt{flip5}}$	9.8%	1.1%	-15.4%
$A_{\texttt{move1}}$	11.7%	-0.5%	-19.4%
$A_{\texttt{move2}}$	11.7%	4.5%	-12.4%
$A_{\texttt{move3}}$	22.5%	2.4%	-24.5%
$A_{\texttt{move4}}$	11.8%	5.1%	-21.5%
$A_{\texttt{move5}}$	10.3%	1.6%	-25.1%
$A_{\texttt{qft}}$	37.3%	25.9%	2.5%
$A_{\texttt{iqft}}$	60.0%	24.4%	-15.8%
$A_{\texttt{inc}}$	48.7%	34.3%	-8.9%
$A_{\texttt{cnt}}$	37.8%	21.0%	-12.1%
$A_{\texttt{ising}}$	47.6%	39.3%	16.4%

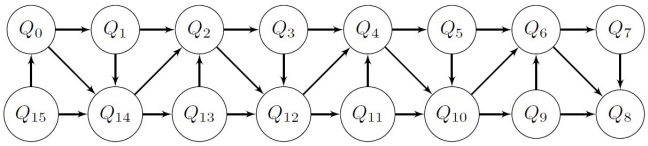


Fig. 12: Coupling graph derived from benchmark *ising_16*

hence, constitute a first priority), but that it is also worthwhile to take the effect of an architecture in the realized quantum functionality into account (e.g., as a second priority).

Moreover, the results obtained by Scheme *Function Specific* (cf. Fig. 11(c)) show the potential benefits that are possible by this. Even if only a single representative quantum functionality has been considered when generating the respective architectures, substantial improvements for almost all benchmarks are achieved (on average, approx. 21% to 39%, in the best case up to 60% less overhead are reported). In particular, the architecture $A_{\texttt{ising}}$ stands out. Although again only one benchmark is used as a representative (namely *ising*), this architecture allows for better realizations for *all* considered benchmarks. This clearly shows the importance of considering alternative architectures and their effect. Since the architecture $A_{\texttt{ising}}$ is special, we show the resulting coupling in Fig. 12.

Of course, all these results are under the reservation that the architectures generated by the proposed schemes indeed can satisfy all physical constraints and can actually be physically realized. Ensuring this requires collaboration between researchers from the quantum computing community and the design automation community. But the results summarized in Fig. 11 and Table I which are already obtained by heavily restricted architectures/coupling graphs show substantial potential benefits for this and provide a strong motivation towards that.

VII. Conclusion

In this paper, we explored the potential benefits of alternative quantum architectures. To this end, we proposed approaches for the generation of alternative coupling graphs (and, by this, quantum computing architectures) that should still be able to satisfy the physical constraints but, at the same time, allow for a more efficient realization of the desired quantum functionality. Moreover, we have confirmed through evaluations the potential benefits of those alternative coupling graphs which can reduce the mapping overhead by up to almost 40% on average (while, in the best case, up to 60% reduction in the mapping overhead is obtained). The results of this work provide a motivation to researchers developing quantum computers and architectures to not only take physical constraints into consideration (which of course have always to be satisfied), but also to consider design aspects of the quantum functionality to be executed on the resulting devices.

In this regard, an obvious step for future work is the physical realization of alternative architectures generated by the proposed schemes. Here, a work recently reported in [62] does provide a proper follow-up for the methods proposed here. There, physical design steps for realizing alternative quantum architectures are proposed. That is, while the consideration of our work has been done from a design perspective to explore the potential benefit of alternative quantum architectures, the following step of realizing the resulting alternatives can be tackled following the discussions of [62]. Moreover, considering that the methods proposed in this work are *initial* considerations to show the potential benefits, further refining/improving upon the schemes proposed here remains an further task for designers and tool developers. In this regard, also the exploitation of other quantum computing concepts such as teleportation (as initially proposed in [63]) remains an issue for future work as well.

VIII. Acknowledgments

This work has partially been supported by the LIT Secure and Correct Systems Lab funded by the State of Upper Austria as well as by BMK, BMDW, and the State of Upper Austria in the frame of the COMET Programme managed by FFG.

References

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

[2] P. W. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring," *Foundations of Computer Science*, pp. 124–134, 1994.

[3] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM review*, vol. 41, no. 2, pp. 303–332, 1999.

[4] L. K. Grover, "A Fast Quantum Mechanical Algorithm for Database Search," in *Theory of computing*, 1996, pp. 212–219.

[5] S. Hallgren, "Polynomial-time quantum algorithms for Pell's equation and the principal ideal problem," *Journal of the ACM (JACM)*, vol. 54, no. 1, p. 4, 2007.

[6] A. Aspuru-Guzik, A. D. Dutoi, P. J. Love, and M. Head-Gordon, "Simulated quantum computation of molecular energies," *Science*, vol. 309, no. 5741, pp. 1704–1707, 2005.

[7] A. Peruzzo, J. McClean, P. Shadbolt, M. H. Yung, X. Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'brien, "A variational eigenvalue solver on a photonic quantum processor," *Nature communications*, vol. 5, p. 4213, 2014.

[8] J. Preskill, "Quantum Computing in the NISQ era and beyond," *Quantum*, vol. 2, p. 79, 2018.

[9] M. H. Devoret and R. J. Schoelkopf, "Superconducting Circuits for Quantum Information: An Outlook," *Science*, vol. 339, no. 6124, pp. 1169–1174, 2013.

[10] M. Kjaergaard, M. E. Schwartz, J. Braumüller, P. Krantz, J. I. Wang, S. Gustavsson, and W. D. Oliver, "Superconducting Qubits: Current State of Play," ArXiV, Tech. Rep. arXiv:1608.02792 [cs.IT], 2019.

[11] H. Paik *et al.*, "Observation of High Coherence in Josephson Junction Qubits Measured in a Three-Dimensional Circuit QED Architecture," *Phys. Rev. Lett.*, vol. 107, p. 240501, 2011.

[12] R. Barends, J. Kelly, A. Megrant, D. Sank, E. Jeffrey, Y. Chen, Y. Yin, B. Chiaro, J. Mutus, C. Neill *et al.*, "Coherent Josephson qubit suitable for scalable quantum integrated circuits," *Phys. Rev. Lett.*, vol. 111, no. 8, p. 080502, 2013.

[13] Y. Chen *et al.*, "Qubit Architecture with High Coherence and Fast Tunable Coupling," *Phys. Rev. Lett.*, vol. 113, p. 220502, 2014.

[14] IBM Q Devices, https://quantumexperience.ng.bluemix.net/qx/devices.

[15] R. Courtland, "Google aims for quantum computing supremacy [news]," *IEEE Spectrum*, vol. 54, pp. 9–10, 2017.

[16] F. Arute *et al.*, "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[17] L. Gomes, "Quantum computing: Both here and not here," *IEEE Spectrum*, vol. 55, no. 4, pp. 42–47, 2018.

[18] IBM Q, https://www.research.ibm.com/ibm-q.

[19] 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 (TCAD)*, vol. 32, no. 6, pp. 818–830, 2013.

[20] D. M. Miller, R. Wille, and Z. Sasanian, "Elementary quantum gate realizations for multiple-control Toffolli gates," in *Intl. Symp. on Multi-Valued Logic (ISMVL)*, 2011, pp. 288–293.

[21] D. Maslov, S. M. Falconer, and M. Mosca, "Quantum circuit placement," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 4, pp. 752–763, 2008.

[22] IBM Quantum Information Software Kit (QISKit), https://qiskit.org/.

[23] M. Y. Siraichi, V. F. D. Santos, S. Collange, and F. M. Q. Pereira, "Qubit Allocation," in *Intl. Symp. on Code Generation and Optimization*, 2018, pp. 113–125.

[24] A. Matsuo, W. Hattori, and S. Yamashita, "Reducing the Overhead of Mapping Quantum Circuits to IBM Q System," in *Intl. Symp. on Circuits and Systems (ISCAS)*, 2019, pp. 1–5.

[25] A. Zulehner, A. Paler, and R. Wille, "An Efficient Methodology for Mapping Quantum Circuits to the IBM QX Architectures," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 38, no. 7, pp. 1226–1236, 2019.

[26] G. Li, Y. Ding, and Y. Xie, "Tackling the qubit mapping problem for NISQ-era quantum devices," in *Proc. of International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2019, pp. 1001–1014.

[27] D. Bhattacharjee, A. A. Saki, M. Alam, A. Chattopadhyay, and S. Ghosh, "MUQUT: Multi-Constraint Quantum Circuit Mapping on Noisy Intermediate-Scale Quantum Computers," *arXiv preprint arXiv:1911.08559*, 2019.

[28] A. Deb, G. W. Dueck, and R. Wille, "Towards Exploring the Potential of Alternative Quantum Computing Architectures," in *Design, Automation Test in Europe (DATE)*, 2020.

[29] A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, "Elementary gates for quantum computation," *Physical review A*, vol. 52, no. 5, p. 3457, 1995.

[30] P. Selinger, "Quantum circuits of T-depth one," *Physical Review A*, vol. 87, no. 4, p. 042302, 2013.

[31] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron, "Quipper: a scalable quantum programming language," in *Proc. of ACM SIGPLAN conference on Programming language design and implementation*, 2013, pp. 333–342.

[32] A. J. Abhari, A. Faruque, M. J. Dousti, L. Svec, O. Catu, A. Chakrabati, C. F. Chiang, S. Vanderwilt, J. Black, and F. Chong, "Scaffold: Quantum programming language," PRINCETON UNIV NJ DEPT OF COMPUTER SCIENCE, Tech. Rep., 2012.

[33] A. W. Cross, L. S. Bishop, J. A. Smolin, and J. M. Gambetta, "Open quantum assembly language," *arXiv preprint arXiv:1707.03429*, 2017.

[34] K. Matsumoto and K. Amano, "Representation of Quantum Circuits with Clifford and $\pi/8$ Gates," *arXiv preprint arXiv:0806.3834*, 2008.

[35] R. Wille, M. Soeken, C. Otterstedt, and R. Drechsler, "Improving the mapping of reversible circuits to quantum circuits using multiple target lines," in *Proc. of Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2013, pp. 145–150.

[36] D. M. Miller, D. Maslov, and G. W. Dueck, "A transformation based algorithm for reversible logic synthesis," in *Proc. of Design Automation Conference (DAC)*, 2003, pp. 318–323.

[37] A. Zulehner and R. Wille, "One-Pass Design of Reversible Circuits: Combining Embedding and Synthesis for Reversible Logic," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 37, no. 5, pp. 996–1008, 2018.

[38] P. Niemann, R. Wille, and R. Drechsler, "Advanced Exact Synthesis of Clifford+T Circuits," *Quantum Information Processing*, 2020.

[39] A. J. Abhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong, and M. Martonosi, "Scaffcc: a framework for compilation and analysis of quantum computing programs," in *Computing Frontiers Conference*, 2014, pp. 1:1–1:10.

[40] 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.

[41] R. Wille, S. Hillmich, and L. Burgholzer, "JKQ: JKU Tools for Quantum Computing," in *Proc. of International Conference on Computer-Aided Design (ICCAD)*, 2020.

[42] R. Wille, A. Lye, and R. Drechsler, "Exact Reordering of Circuit Lines for Nearest Neighbor Quantum Architectures," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 33, no. 12, pp. 1818–1831, 2014.

[43] M. Saeedi, R. Wille, and R. Drechsler, "Synthesis of quantum circuits for linear nearest neighbor architectures," *Quantum Information Processing*, vol. 10, no. 3, pp. 355–377, 2011.

[44] R. Wille, O. Keszöcze, M. Walter, P. Rohrs, A. Chattopadhyay, and R. Drechsler, "Look-ahead schemes for nearest neighbor optimization of 1D and 2D quantum circuits," in *Proc. of Asia and South Pacific Design Automation Conference ASP-DAC*, 2016, pp. 292–297.

[45] A. Shafaei, M. Saeedi, and M. Pedram, "Optimization of quantum circuits for interaction distance in linear nearest neighbor architectures," in *Proc. of Design Automation Conference (DAC)*, 2013, pp. 41:1–41:6.

[46] A. Shafaei, M. Saeedi, and M. Pedram, "Qubit placement to minimize communication overhead in 2D quantum architectures," in *Proc. of Asia and South Pacific Design Automation Conference ASP-DAC*, 2014, pp. 495–500.

[47] R. Wille, N. Quetschlich, Y. Inoue, N. Yasuda, and S. Minato, "Using πDDs for Nearest Neighbor Optimization of Quantum Circuits," in *Proc. of International Conference on Reversible Computation*, 2016, pp. 181–196.

[48] A. Zulehner, S. Gasser, and R. Wille, "Exact Global Reordering for Nearest Neighbor Quantum Circuits Using A*," in *Proc. of International Conference on Reversible Computation*, 2017, pp. 185–201.

[49] D. Bhattacharjee and A. Chattopadhyay, "Depth-Optimal Quantum Circuit Placement for Arbitrary Topologies," *CoRR*, vol. abs/1703.08540, 2017. [Online]. Available: http://arxiv.org/abs/1703.08540

[50] R. Wille, L. Burgholzer, and A. Zulehner, "Mapping Quantum Circuits to IBM QX Architectures Using the Minimal Number of SWAP and H Operations," in *Design Automation Conference (DAC)*, 2019.

[51] A. Zulehner and R. Wille, "Compiling SU(4) quantum circuits to IBM QX architectures," in *Proc. of Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2019.

[52] M. Sisodia, A. Shukla, A. A. A. de Almeida, G. W. Dueck, and A. Pathak, "Circuit optimization for IBM processors: A way to get higher fidelity and higher values of nonclassicality witnesses," ArXiV, Tech. Rep. arXiv:1812.11602 [quant-ph], 2019.

[53] A. Botea, A. Kishimoto, and R. Marinescu, "On the complexity of quantum circuit compilation," in *Intl Symp. on Combinatorial Search*, 2018, pp. 138–142.

[54] M. Steffen, D. P. DiVincenzo, J. M. Chow, T. N. Theis, and M. B. Ketchen, "Quantum computing: An IBM perspective," *IBM Journal of Research and Development*, vol. 55, no. 5, p. 13, 2011.

[55] S. S. Tannu and M. K. Qureshi, "Not All Qubits Are Created Equal: A Case for Variability-Aware Policies for NISQ-Era Quantum Computers," in *Intl. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2019, pp. 987–999.

[56] A. Blais, J. Gambetta, A. Wallraff, D. I. Schuster, S. M. Girvin, M. H. Devoret, and R. J. Schoelkopf, "Quantum-information processing with circuit quantum electrodynamics," *Phys. Rev. A*, vol. 75, p. 032329, 2007.

[57] J. M. Chow, J. M. Gambetta, A. W. Cross, S. T. Merkel, C. Rigetti, and M. Steffen, "Microwave-activated conditional-phase gate for superconducting qubits," *New Journal of Physics*, vol. 15, no. 11, p. 115012, 2013.

[58] S. Sheldon, E. Magesan, J. M. Chow, and J. M. Gambetta, "Procedure for systematically tuning up cross-talk in the cross-resonance gate," *Phys. Rev. A*, vol. 93, p. 060302, 2016.

[59] H. Paik, A. Mezzacapo, M. Sandberg, D. T. McClure, B. Abdo, A. D. Córcoles, O. Dial, D. F. Bogorin, B. L. T. Plourde, M. Steffen, A. W. Cross, J. M. Gambetta, and J. M. Chow, "Experimental Demonstration of a Resonator-Induced Phase Gate in a Multiqubit Circuit-QED System," *Phys. Rev. Lett.*, vol. 117, p. 250502, 2016.

[60] A. D. Patterson *et al.*, "Calibration of the cross-resonance two-qubit gate between directly-coupled transmons," ArXiV, Tech. Rep. arXiv:1905.05670 [quant-ph], 2019.

[61] R. Wille, D. Große, L. Teuber, G. W. Dueck, and R. Drechsler, "RevLib: An Online Resource for Reversible Functions and Reversible Circuits," in *Intl. Symp. on Multi-Valued Logic (ISMVL)*, 2008, pp. 220–225.

[62] G. Li, Y. Ding, and Y. Xie, "Towards Efficient Superconducting Quantum Processor Architecture Design," in *Proc. of International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020, pp. 1031–1045.

[63] S. Hillmich, A. Zulehner, and R. Wille, "Exploiting quantum teleportation in quantum circuit mapping," in *Proc. of Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2021.

Arighna Deb is currently an Assistant Professor at the School of Electronics Engineering, KIIT University, India. He received the PhD (Engineering) from Jadavpur University, India, and Dr.rer.nat. from the University of Bremen, Germany, in 2017 and 2018, respectively. His research interests include the logic synthesis for conventional technologies and emerging technologies such as quantum computing, optical computing, quantum-dot cellular automata (QCA) based computing.

Gerhard W Dueck (M'89–LSM'17) was born in Montevideo, Uruguay. He received the BSc, Master, and PhD degrees in computer science form the University of Manitoba, Winnipeg, Manitoba, Canada, in 1983, 1986, and 1988, respectively. He is currently a professor in the Faculty of Computer Science at the University of New Brunswick. After completing his PhD he joined St. Francis Xavier University in Antigonish, Nova Scotia. In 1991 he spent a year at the Naval Postgraduate School in Monterey, CA, as a research associate. In 1999 he joined the Faculty of Computer Science at the University of New Brunswick. He has been actively involved in the IEEE Computer Society Technical Committee on Multiple-Valued Logic, where he severed as chair. Four times he has been program chair of the IEEE International Symposium on Multiple-Valued Logic and twice symposium chair. For more than 40 years, his research interests include reversible logic, Reed Muller expansions, multiple-valued logic, and digital design. More recently, his research has included runtime support systems for virtual machine, dealing with dynamic memory management. His research supported by government and industrial grants.

Robert Wille (M'06–SM'15) is Full Professor at the Johannes Kepler University Linz, Austria, and Chief Scientific Officer at the Software Competence Center Hagenberg, Austria. He received the Diploma and Dr.-Ing. degrees in Computer Science from the University of Bremen, Germany, in 2006 and 2009, respectively. Since then, he worked at the University of Bremen, the German Research Center for Artificial Intelligence (DFKI), the University of Applied Science of Bremen, the University of Potsdam, and the Technical University Dresden. Since 2015, he is working in Linz/Hagenberg. His research interests are in the design of circuits and systems for both conventional and emerging technologies. In these areas, he published more than 300 papers in journals and conferences and served in editorial boards and program committees of numerous journals/conferences such as TCAD, ASP-DAC, DAC, DATE, and ICCAD. For his research, he was awarded, e.g., with a Best Paper Award at ICCAD, a DAC Under-40 Innovator Award, a Google Research Award, and more.