TU Delft

Delft University of Technology

Mapping quantum algorithms to multi-core quantum computing architectures

Ovide, Anabel; Rodrigo, Santiago; Bandic, Medina; Van Someren, Hans; Feld, Sebastian; Abadal, Sergi; Alarcon, Eduard; Almudever, Carmen G.

DOI
10.1109/ISCAS46773.2023.10181589

Publication date
2023

Document Version
Final published version

Published in
ISCAS 2023 - 56th IEEE International Symposium on Circuits and Systems, Proceedings

Citation (APA)
Ovide, A., Rodrigo, S., Bandic, M., Van Someren, H., Feld, S., Abadal, S., Alarcon, E., & Almudever, C. G. (2023). Mapping quantum algorithms to multi-core quantum computing architectures. In *ISCAS 2023 - 56th IEEE International Symposium on Circuits and Systems, Proceedings* (Proceedings - IEEE International Symposium on Circuits and Systems; Vol. 2023-May). IEEE. https://doi.org/10.1109/ISCAS46773.2023.10181589

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Green Open Access added to TU Delft Institutional Repository

'You share, we take care!' - Taverne project

https://www.openaccess.nl/en/you-share-we-take-care

Otherwise as indicated in the copyright section: the publisher is the copyright holder of this work and the author uses the Dutch legislation to make this work public.

Mapping quantum algorithms to multi-core quantum computing architectures

Anabel Ovide*, Santiago Rodrigo†, Medina Bandic‡, Hans Van Someren‡, Sebastian Feld‡, Sergi Abadal†, Eduard Alarcon †, and Carmen G. Almudever*
*Technical University of Valencia, Spain
†Technical University of Catalonia, BarcelonaTech, Spain
‡Technical University of Delft, The Netherlands

2023 IEEE International Symposium on Circuits and Systems (ISCAS) | 978-1-6654-5109-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/ISCAS46773.2023.10181589

***Abstract*—Current monolithic quantum computer architectures have limited scalability. One promising approach for scaling them up is to use a modular or multi-core architecture, in which different quantum processors (cores) are connected via quantum and classical links. This new architectural design poses new challenges such as the expensive inter-core communication. To reduce these movements when executing a quantum algorithm, an efficient mapping technique is required. In this paper, a detailed critical discussion of the quantum circuit mapping problem for multi-core quantum computing architectures is provided. In addition, we further explore the performance of a mapping method, which is formulated as a partitioning over time graph problem, by performing an architectural scalability analysis.**

***Index Terms*—scalability quantum computing systems, multi-core quantum computers, mapping of quantum algorithms.**

I. Introduction

Quantum computers are a revolutionary technology that can outperform classical computing in areas such as scientific simulation [1], cryptography [2], machine learning [3], search or optimization [4], thanks to the use of quantum mechanics phenomena like superposition and entanglement. Current quantum computing technologies, commonly called NISQ (Noisy Intermediate-Scale Quantum) [5] devices, are limited in the number of qubits and prone to errors. The most advanced quantum processors consist of a few tens of noisy qubits (e.g. IBM's 433-qubit Osprey processor [6]), meaning that their state can be easily modified due to the interaction with the external environment (decoherence) and that quantum gates and measurements are implemented with imperfect operations. Algorithms for NISQ devices have been developed to leverage their scarce and noisy resources such as Quantum Approximate Optimization Algorithm (QAOA) or Variational Quantum Eigensolver (VQE) [7]. However, to build a universal fault-tolerant quantum computer and achieve the full computational power these machines will provide, it is necessary to scale them up in a way that the number of qubits is increased without incurring much higher error rates. Therefore, the scalability of quantum computing systems is one of the main challenges the quantum community is currently facing.

Nowadays NISQ computers are implemented as single-chip processors, also referred as single-core quantum processors, in which all qubits are integrated within a single chip. This monolithic architecture is hardly scalable due to challenges in the control electronics and wiring [8], an increase of undesired interactions between qubits (i.e. crosstalk) [9] and a decrease of the device uniformity and yield. To overcome these challenges and solve the scaling problem, modular quantum computing architectures have been already proposed for different qubit implementation technologies [10]–[13]. The main idea is to combine multiple quantum processors and connect them via single control systems, classical communication links and ultimately quantum communication technologies [14]–[16]. We refer to the latter, in which both classical and quantum communication channels are incorporated as multi-core quantum computing architectures. They will allow performing distributed multi-core quantum computing in which a large algorithm consisting of more qubits than there are in a single processor, is partitioned into smaller instances and executed on several quantum chips.

With this novel architectural approach, new challenges emerge as pointed out in [17] that include: i) the implementation of input/output communication ports for each core (processor) as well as the definition of the ratio of qubits devoted to computation and communication; ii) the development of the technology required for communicating quantum information between chips and corresponding communication protocols; and iii) compilation techniques, including placement and routing of qubits and scheduling of quantum operations, that allow for an efficient distributed multi-core quantum computation, which will be the central topic of this paper.

Executing an algorithm on a NISQ processor, requires to perform some modifications on the corresponding quantum circuit such that all quantum hardware constraints are satisfied. This process of adapting the quantum circuit to the quantum processor restrictions is usually called mapping or transpiling. Whereas several quantum circuit mapping techniques have been proposed for single-core quantum architectures [18]–[23] only recently, the first compilation techniques for mapping quantum algorithms onto connectivity-simplified multi-core quantum architectures have been proposed [17], [24]. In [24], the authors propose a method for mapping quantum programs to a modular quantum architecture based on graph partitioning techniques. However, this approach is only tested on a relatively small and fixed quantum computing multi-core architecture in which the number of cores and qubits per core are both constant (i.e. 10 cores $\times$ 10 qubits per core)

Authorized licensed use limited to: TU Delft Library. Downloaded on August 23,2023 at 07:59:54 UTC from IEEE Xplore. Restrictions apply.

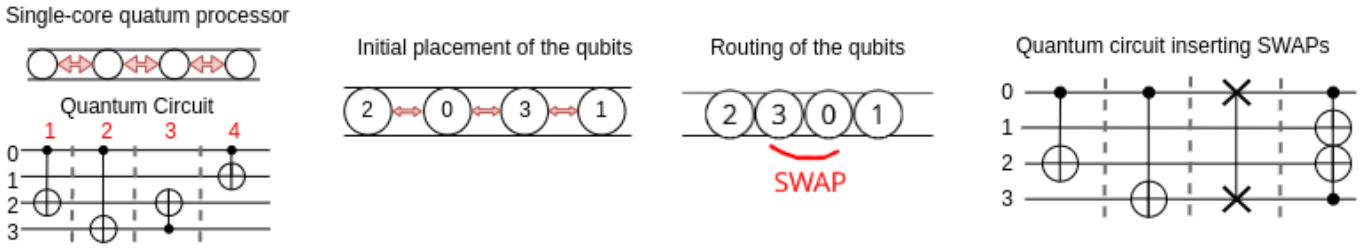

Fig. 1: An illustrative example of the quantum circuit mapping procedure. We consider a 1-D linear array quantum processor shown at the top-left, where only adjacent qubits (circles) can interact. Next, an optimal initial placement of qubits is performed based on the quantum circuit to be executed. Note that, the first two CNOT gates (CNOTS in time steps 1 and 2) can be directly performed as qubits 0 and 2 as well qubits 0 and 3 are adjacent. However, an extra SWAP gate has to be inserted for allowing the execution of the other two CNOT gates between qubit 2 and qubit 3 and between quit 0 and qubit 1. A SWAP gate exchanges the state of the two involved qubits. In addition, note that scheduling the quantum gates saves one time step as the last two CNOT gates can be performed in parallel.

irrespective of the width (i.e number of qubits) of the circuit to be executed.

This paper focuses on the very new field of compilation techniques for scalable multi-core quantum computer architectures with the aim of performing distributed quantum computing. To this purpose, the challenges of mapping quantum algorithms to these modular architectures are discussed in Section II, emphasizing the main differences with single-core mapping methods. In section III, we introduce one of the most recent and advanced works on mapping for modular architectures [24]. In Section IV, we further explore the performance of this mapping approach by performing an architectural scalability analysis. Finally, conclusions are presented in Section V.

II. From single-core to multi-core mapping

Quantum circuit mapping techniques have been developed for single-chip NISQ processors, as part of the compilation process, to deal with their constraints and allow to successfully execute quantum algorithms [18]–[23], [25], [26]. More precisely, quantum circuit mapping is about transforming hardware-agnostic quantum circuit descriptions into a hardware-compliant version that considers all physical restrictions of a given quantum processor. One of the main constraints in current quantum devices is the reduced connectivity between physical qubits, which usually limits their possible interactions to only nearest-neighbour requiring qubits to be moved to adjacent positions to execute the desired two-qubit operation (e.g. CNOT gate). The circuit mapping procedure consists of different steps: i) **gate decomposition**, in which gates of the circuit are decomposed into a series of native gates implementable in the quantum processor; ii) **initial placement** of qubits, where quantum circuit qubits, i.e. virtual qubits, are assigned to the physical qubits of the device. This process helps to minimize the (movement) operations needed in the routing stage; iii) **routing** of qubits, in which non-neighboring qubits that need to perform a two-quit gate are moved to adjacent physical qubits (which share a connection) by means of, for instance, SWAP gates; and iv) **scheduling** of operations to leverage parallelism while respecting their dependencies and quantum hardware constraints. An example of the quantum circuit mapping process is shown in Figure 1.

As mentioned in the previous section, multi-core quantum computing architectures are a promising approach to scale up current single-core quantum computers. Existing proposals agree on an architecture based on interconnecting multiple NISQ processors [12], [14] consisting of tens to hundreds of qubits, increasing the total qubit count without losing that much fidelity and improving isolation. In this architectural design, NISQ processors will be ultimately interconnected through short-range quantum-coherent links and classical links in the form of a so-called ‘quantum intranet’ [14]. Quantum coherent links will be responsible for transporting qubits (or quantum states) from core to core, for instance, by means of shuttling o quantum teleportation. Several challenges arise with this new architecture, being the most relevant for this work the need for exchanging quantum information between cores. Note that these inter-core communications are more expensive and error prone than those performed in single-core architectures. Therefore, multi-core quantum computing architectures require the development of a new breed of compilation techniques that will have to consider the following fundamental different aspects:

Inter-core communication: Similar to the single-chip case in which qubits need to be adjacent for interacting, qubits placed in different cores cannot directly perform a two-qubit gate. To do so, they have to make use of entanglement-based quantum communication protocols that require the generation of the so-called *Bell pairs* allowing to perform, for instance, remote CNOTs between distant qubits or to teleport quantum states from one core to another [27], [28]. This comes with an overhead of resources needed for creating and distributing entangled pairs. In addition, the entanglement generation is a non-deterministic process making the scheduling task more complex.

Not all qubits have the same functionality: In each of the quantum cores there will be qubits devoted to computation and

Authorized licensed use limited to: TU Delft Library. Downloaded on August 23,2023 at 07:59:54 UTC from IEEE Xplore. Restrictions apply.

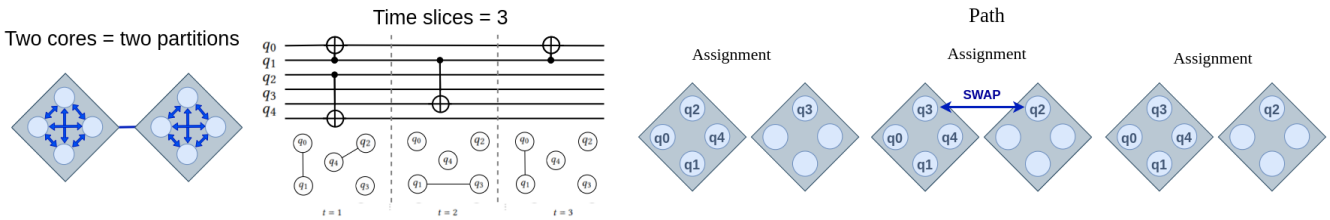

Fig. 2: Example of the quantum circuit mapping technique proposed in [24]. A multi-core quantum architecture composed of 2 cores and 4 qubits per core with all-to-all connectivity is shown on the left. Next, the quantum circuit to be mapped with its respective time slices is presented. Note that each time slice can be represented by a qubit interaction graph in which virtual qubits correspond to the nodes and edges are the interactions between them (i.e. two-qubit gates). For each of the time slices, a valid assignment is returned by using a relaxed version of the so-called Overall Extreme Exchange (rOEE) algorithm. To achieve it, qubits are exchanged between cores by means of SWAP gates, until a valid assignment is found.

storage and qubits used for communication. Communication qubits will handle inter-core communications, whereas storage qubits will perform local operations. Mapping techniques will have to include information about qubit 'types' and which ones are being used as well as the resources available for communication. Note that the more qubits are dedicated to communication, the higher the number of inter-core communications that can be performed in parallel.

A two-step quantum circuit mapping process: An initial qubit placement and routing should be done at the quantum core level, placing qubits that need to interact on the same core and use efficient routing techniques to reduce inter-communication operations, but also within the quantum processors to reduce the overhead created due to their limited qubit connectivity as in the single-chip case.

Multi-core or modular architectures for scaling up quantum devices share a lot of similarities with the quantum networks that are being deployed for a future quantum internet [11], [29], [30]. The main difference resides in the fact that communications links, instead of being short-range, are long-range [31], resulting in the need for a more complicated infrastructure to move qubits between quantum devices, i.e. quantum repeaters. Due to this quantum network infrastructure, moving qubits among devices would be more complex, needing to perform entanglement swaps [27], which increases latency considerably as its duration grows exponentially with the distance between devices. One possible application of quantum networks is to perform distributed quantum computing, for which compilation techniques have been already proposed [27], [32], [33]. However, not so much attention has been paid so far to the development of compilers for multi-core quantum computing architectures [17], [24]. In the next sections, we will focus on the mapping technique proposed in [24].

III. Quantum Circuit Partitioning for Distributed Quantum Computing

In [24] a technique for mapping quantum algorithms on multi-core architectures based on graph partitioning has been proposed. The goal is to place qubits in the different quantum processors such that inter-core movements are minimized. An illustrative example of this mapping technique is shown in Figure 2. Note that in the proposal presented in [24] the following assumptions that simplify the quantum circuit mapping problem are made: i) all-to-all connectivity between cores and among physical qubits within the cores. This means that there is no need for qubit routing inside the core, nor for optimal initial placement. Regarding inter-core routing and qubit placement at the core level, all qubits are at a one-hop distance, and therefore when two qubits have to interact and cannot be placed from the beginning on the same quantum core it is enough to place one of them on any other core; ii) SWAP operations (i.e. exchange of quantum states) are used for inter-core communication that makes simpler the management of resources as it not required to check if there is space (i.e. qubits that do not have any information) for exchanging qubits between cores; iii) only a fixed modular architecture is considered consisting of 10 cores with 10 qubits per core, which is not enough for analyzing the performance of the quantum circuit mapper. In the following section, different architectures will be used to further analyze this proposed mapping procedure.

IV. Results

As previously mentioned, in this work we further analyze the performance of the quantum circuit mapping approach in [24] by considering different multi-core architecture designs with still all-to-all qubit and cores connectivity. For this purpose, several quantum benchmarks have been used. In this paper, results for the Cuccaro and the QFT adder are presented as they have very dissimilar circuit characteristics. The Cuccaro adder is a well parallelizable and easy to scale algorithm with a low number of two-qubit gates and circuit depth. In contrast, the QFT adder is a more sequential algorithm with a huge number of two-qubit gates and large depth. In addition, two performance metrics are used: the number of non-local communications (i.e. inter-core movements) and the execution time (i.e. time it takes to calculate all valid assignments).

Figure 3a compares how the mapping algorithm behaves when a fixed and a variable number of qubits per core is

Authorized licensed use limited to: TU Delft Library. Downloaded on August 23,2023 at 07:59:54 UTC from IEEE Xplore. Restrictions apply.

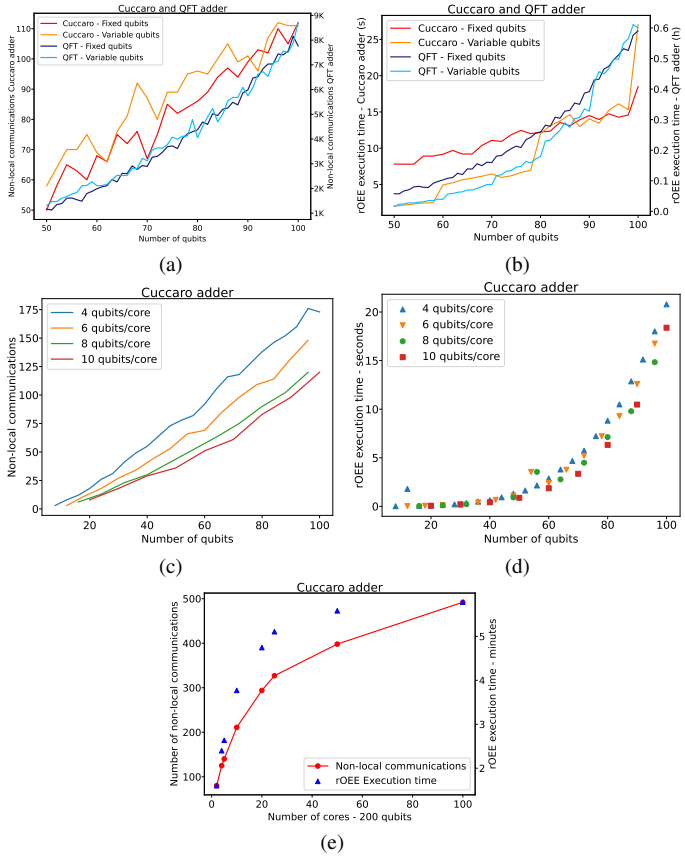

Fig. 3: Non-local communications (SWAPs) and execution time for different multi-core architectures. (a) and (b) for a fixed and a variable number of qubits per core. (c) and (d) when a strong scaling of the architecture is performed. (e) Weak scaling of the multi-core architecture.

assumed, both architectures counting with ten cores. In other words, in the first case cores always consist of 10 qubits per core independently of the circuit width, whereas in the second case, the minimum (even) number of qubits per core is used based on the algorithm requirements. Similar behavior with respect to non-local communications can be observed for a fixed and variable number of qubits for both quantum circuits. However, note that the difference between both cases is much more pronounced for the Cuccaro adder due to its circuit characteristics. This means that the relevance of the number of physical qubits in the architecture regarding non-local communications depends on the algorithm to be executed. In contrast, the total number of physical qubits in the architecture is crucial for the rOEE runtime. As shown in Figure 3b, a lower execution time is required for the variable case. The reason is that the rOEE algorithm computes over physical qubits to find a valid assignment, and therefore the more physical qubits, the more iterations are needed, increasing the execution time. Furthermore, note the large difference in non-local communications as well as in execution time between the Cuccaro and the QFT adder.

In addition, two more architectural scalability experiments have been performed, named *weak and strong* scaling. In weak scaling, the total number of physical qubits is fixed, whereas the number of cores and qubits per core varies, increasing the number of cores while decreasing the number of qubits per core. In Figure 3e the weak scaling results are shown; both, the rOEE runtime as well as the non-local communications increase when more cores are added to the architecture. The more cores and fewer qubits per core, the more computations will be performed until the rOEE algorithm finds a valid assignment and the higher the inter-core movements are.

In strong scaling, the number of qubits per core is fixed but we increase the number of cores and therefore the total number of qubits in the device. Four different architectures have been used with 4, 6, 8, and 10 qubits per core, starting with 2 cores and increasing them until a total number of 100 qubits is reached. As shown in Figures 3c and 3d, non-local communications increase as more cores are added. Moreover, on architectures with fewer qubits per core, a higher number of non-local communications is observed due to higher constraints to find a valid assignment. The execution time in relation to the total number of qubits is similar for all four cases since, as mentioned before, the most crucial parameter concerning execution time is the total number of physical qubits and not how they are distributed.

V. Conclusions

Multi-core or modular quantum computing architectures are a promising approach to overcome the scaling difficulties encountered in monolithic or single-core quantum processors. However, this new architectural design comes with a set of challenges such as qubit interactions across cores. Inter-core communications can be minimized through the process of mapping, as proposed in [24]. In this paper, we have further analyzed the performance of this quantum circuit mapping technique by performing several experiments in which different architectures with all-to-all connectivity are considered. The most important findings can be summarized as follows: i) Non-local communications and execution time increase with the circuit width. ii) The number of physical qubits is the most important factor regarding execution time and therefore, using a variable number of qubits per core is more efficient. iii) The higher the number of cores, the longer the execution time and the higher the non-local communications. Note also, that the performance highly depends on the quantum algorithm to be executed.

Acknowledgments

We acknowledge support from EU, grant HORIZON-ERC-101042080 (S.A.) and grant HORIZON-EIC-2022-PATHFINDEROPEN-01-101099697 (S.A., E.A and C.G.A.), from project PRG 946 funded by the Estonian Research Council (A.O.), from MICIIN and European ERDF under grant PID2021-123627OB-C51 (C.G.A.) and from MICIIN with funding from European Union NextGenerationEU(PRTR-C17.I1) and by Generalitat de Catalunya (E.A.)

Authorized licensed use limited to: TU Delft Library. Downloaded on August 23,2023 at 07:59:54 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] M. Martonosi and M. Roetteler, "Next steps in quantum computing: Computer science's role," *arXiv preprint arXiv:1903.10541*, 2019.

[2] V. Mavroeidis, K. Vishi, M. D., and A. Jøsang, "The impact of quantum computing on present cryptography," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 3, 2018.

[3] T. M. Khan and A. Robles-Kelly, "Machine learning: Quantum vs classical," *IEEE Access*, vol. 8, pp. 219275–219294, 2020.

[4] A. Montanaro, "Quantum algorithms: an overview," *npj Quantum Information*, vol. 2, jan 2016.

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

[6] J. Gambetta, "Quantum-centric supercomputing: The next wave of computing," Nov 2022. https://research.ibm.com/blog/next-wave-quantum-centric-supercomputing.

[7] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, "Noisy intermediate-scale quantum algorithms," *Reviews of Modern Physics*, vol. 94, feb 2022.

[8] F. Sebastiano, J. Van Dijk, B. Patra, J. van Staveren, X. Xue, C. Almudever, G. Scappucci, M. Veldhorst, L. Vandersypen, A. Vladimirescu, *et al.*, "Cryo-cmos interfaces for large-scale quantum computers," in *2020 IEEE International Electron Devices Meeting (IEDM)*, pp. 25–2, IEEE, 2020.

[9] Y. Ding, P. Gokhale, S. F. Lin, R. Rines, T. Propson, and F. T. Chong, "Systematic crosstalk mitigation for superconducting qubits via frequency-aware compilation," in *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, IEEE, oct 2020.

[10] C. Monroe, R. Raussendorf, A. Ruthven, K. R. Brown, P. Maunz, L.-M. Duan, and J. Kim, "Large-scale modular quantum-computer architecture with atomic memory and photonic interconnects," *Physical Review A*, vol. 89, feb 2014.

[11] N. LaRacuente, K. N. Smith, P. Imany, K. L. Silverman, and F. T. Chong, "Short-range microwave networks to scale superconducting quantum computation," 2022.

[12] H. Jnane, B. Undseth, Z. Cai, S. C. Benjamin, and B. Koczor, "Multicore quantum computing," *arXiv preprint arXiv:2201.08861*, 2022.

[13] K. N. Smith, G. S. Ravi, J. M. Baker, and F. T. Chong, "Scaling superconducting quantum computers with chiplet architectures," 2022.

[14] J. M. Chow, "Quantum intranet," *IET Quantum Communication*, vol. 2, no. 1, pp. 26–27, 2021.

[15] IBM, "Ibm roadmap," 2022. https://www.ibm.com/quantum/roadmap.

[16] IBM, "Ibm roadmap 2025," 2022. https://research.ibm.com/blog/ibm-quantum-roadmap-2025.

[17] S. Rodrigo, S. Abadal, E. Alarcón, M. Bandic, H. Van Someren, and C. G. Almudéver, "On double full-stack communication-enabled architectures for multicore quantum computers," *IEEE micro*, vol. 41, no. 5, pp. 48–56, 2021.

[18] C. G. Almudever, L. Lao, R. Wille, and G. G. Guerreschi, "Realizing quantum algorithms on real quantum computing devices," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 864–872, IEEE, 2020.

[19] L. Lao, H. Van Someren, I. Ashraf, and C. G. Almudever, "Timing and resource-aware mapping of quantum circuits to superconducting processors," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 2, pp. 359–371, 2021.

[20] R. Wille, S. Hillmich, and L. Burgholzer, "Efficient and correct compilation of quantum circuits," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, 2020.

[21] M. Walter, R. Wille, F. Sill Torres, and R. Drechsler, *Scalable Placement and Routing*, pp. 79–97. Cham: Springer International Publishing, 2022.

[22] D. Venturelli, M. Do, E. Rieffel, and J. Frank, "Compiling quantum circuits to realistic hardware architectures using temporal planners," *Quantum Science and Technology*, vol. 3, p. 025004, feb 2018.

[23] P. Murali, J. M. Baker, A. Javadi-Abhari, F. T. Chong, and M. Martonosi, "Noise-adaptive compiler mappings for noisy intermediate-scale quantum computers," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS '19, (New York, NY, USA), p. 1015–1029, Association for Computing Machinery, 2019.

[24] J. M. Baker, C. Duckering, A. Hoover, and F. T. Chong, "Time-sliced quantum circuit partitioning for modular architectures," in *Proceedings of the 17th ACM International Conference on Computing Frontiers*, CF '20, (New York, NY, USA), p. 98–107, Association for Computing Machinery, 2020.

[25] M. Y. Siraichi, V. F. d. Santos, C. Collange, and F. M. Q. Pereira, "Qubit allocation," in *Proceedings of the 2018 International Symposium on Code Generation and Optimization*, CGO 2018, (New York, NY, USA), p. 113–125, Association for Computing Machinery, 2018.

[26] P. Murali, N. M. Linke, M. Martonosi, A. J. Abhari, N. H. Nguyen, and C. H. Alderete, "Full-stack, real-system quantum computer studies: Architectural comparisons and design insights," in *2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture (ISCA)*, pp. 527–540, 2019.

[27] D. Ferrari, A. S. Cacciapuoti, M. Amoretti, and M. Caleffi, "Compiler design for distributed quantum computing," *IEEE Transactions on Quantum Engineering*, vol. 2, pp. 1–20, 2021.

[28] S. Rodrigo, D. Spanò, M. Bandic, S. Abadal, H. Van Someren, A. Ovide, S. Feld, C. G. Almudéver, and E. Alarcón, "Characterizing the spatio-temporal qubit traffic of a quantum intranet aiming at modular quantum computer architectures," in *Proceedings of the 9th ACM International Conference on Nanoscale Computing and Communication*, pp. 1–7, 2022.

[29] R. Van Meter and S. J. Devitt, "The path to scalable distributed quantum computing," *Computer*, vol. 49, no. 9, pp. 31–42, 2016.

[30] A. S. Cacciapuoti, M. Caleffi, F. Tafuri, F. S. Cataliotti, S. Gherardini, and G. Bianchi, "Quantum internet: Networking challenges in distributed quantum computing," *IEEE Network*, vol. 34, no. 1, pp. 137–143, 2020.

[31] W. Kozlowski and S. Wehner, "Towards large-scale quantum networks," in *Proceedings of the Sixth Annual ACM International Conference on Nanoscale Computing and Communication*, ACM, sep 2019.

[32] D. Cuomo, M. Caleffi, K. Krsulich, F. Tramonto, G. Agliardi, E. Prati, and A. S. Cacciapuoti, "Optimized compiler for distributed quantum computing," 2021.

[33] C. Cicconetti, M. Conti, and A. Passarella, "Resource allocation in quantum networks for distributed quantum computing," in *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*, IEEE, jun 2022.

Authorized licensed use limited to: TU Delft Library. Downloaded on August 23,2023 at 07:59:54 UTC from IEEE Xplore. Restrictions apply.