Optimized Routing for Shuttling-Based Quantum Processing Architectures

(Extended abstract)

Roy Hermanns and Thomas Noll[0000−0002−1865−1798]

Software Modeling and Verification Group, RWTH Aachen University, Germany
`roy.hermanns@rwth-aachen.de, noll@cs.rwth-aachen.de`

The work presented here is being undertaken within the project *Compiler Optimization for Shuttling-Based Quantum Computing Architectures*, which is part of the *Interdisciplinary Doctoral Program in Quantum Systems Integration* funded by the BMW Group. The goal of this project is to develop systematic support for compiling quantum software to computing architectures featuring qubit shuttling operations, which allow to dynamically reconfigure qubit connectivity between subsequent gate operations. They are common for quantum computing architectures based on semiconductor qubits, trapped ions, or neutral atoms. However, while there exists a large body of work for non-shuttling architectures based on the insertion of SWAP-gates, advanced compilation methods that allow moving qubits are rather scarce. This in particular applies to architectures supporting single electron transport by so-called conveyor-mode shuttling where a quantum dot used to trap the qubit is continuously translated to distant qubit sites.

We aim to close this gap by applying advanced routing methods based on multi-agent path finding algorithms. In summary, our approach comes with the following distinguishing features:

– It provides a flexible architecture-modelling framework that is not only applicable to our setting but to a wider class of quantum hardware systems.
– To the best of our knowledge, it introduces the first formalisation and implementation for qubit routing on conveyor-mode shuttling devices.
– Our routing algorithm computes an optimal collision-free solution for a given quantum circuit.

In the following, we give an overview of our hardware architecture, its formal model, and the routing algorithm.

The SpinBus Architecture. In [6], the *SpinBus architecture* has been introduced as a potential basis for a spin-based quantum processor that meets the scalability requirements for practical quantum computing. It uses electron shuttling driven by sinusoidal voltage signals based on conveyor-mode shuttling lanes to connect qubits and features low operating frequencies and enhanced qubit coherence. Due to its promising features, it has been chosen as the target architecture in our approach. An overview diagram is shown in Figure 1.

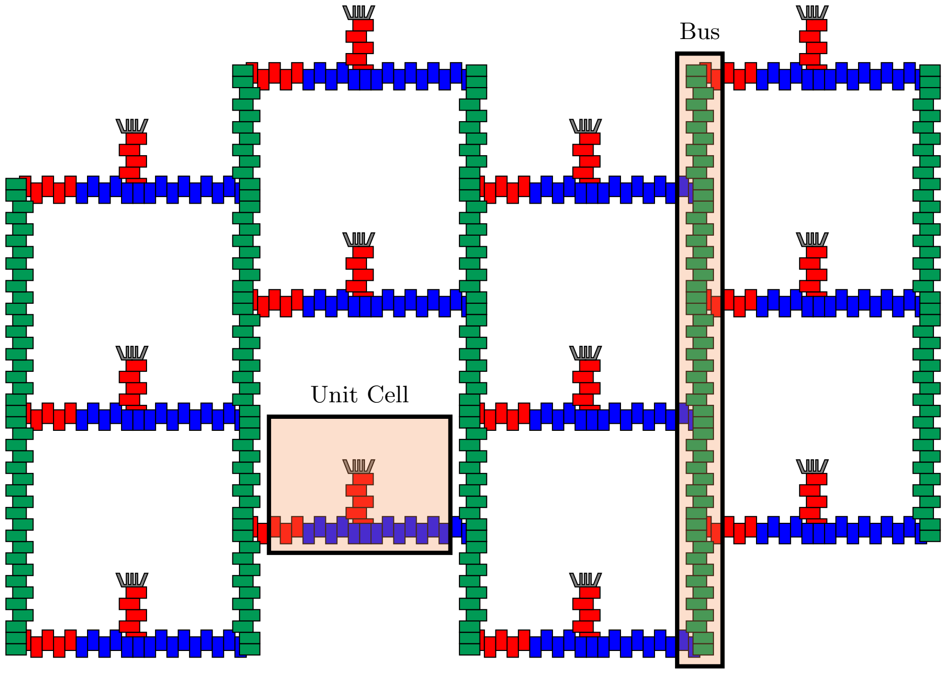


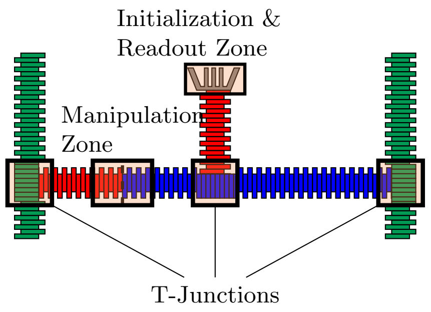
Fig. 1: Overview of SpinBus architecture with 12 unit cells

For the quantum layer, it employs a layout of tileable *unit cells* (Figure 2a) as building blocks. They provide the means for initializing and reading out qubit values and for performing gate operations in two specialized zones, namely, the *initialization and readout zone (IRZ)* and the *manipulation zone*, respectively. Shuttling lanes connect both the operational zones and adjacent unit cells through *T-junctions*.

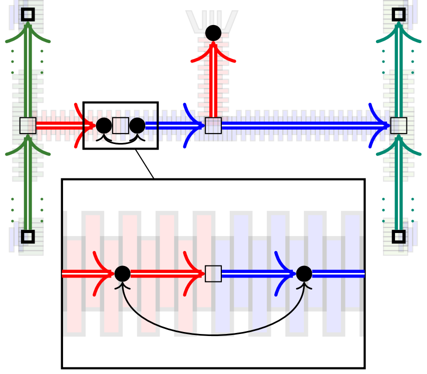
A detailed description of the hardware architecture and physical properties is given in [6]. For the purposes of this presentation, it suffices to give an abstract description and formalization which focuses on the constraints needed for qubit mapping and routing.

Architecture Representation. We encode all the hardware information required for compilation into an attributed, directed graph which we just name the device's *architecture*. The attributes needed for this compilation process will be explained in the following paragraphs. Other kinds of architectures or algorithms can extend the set of possible attributes as appropriate.

Established compilers describe architectures via a *coupling* or *connectivity graph* [4]. Such compilers do not differentiate between a physical qubit and operation zones where quantum gates are applied. Each physical qubit is represented as a vertex in the connectivity graph where the edges determine the ability to perform multi-qubit operations. Such edges can be directed [14], but most ar-



(a) Conceptual view of a unit cell with specific areas marked.



(b) Formalized view of a unit cell as directed and annotated graph.

Fig. 2: Unit cells.

chitectures allow symmetric executions, which we would in turn encode by two symmetric directed edges.

Extending this concept, we model the SpinBus architecture by adding two vertices for each manipulation zone and one vertex for each initialization and readout zone, as shown in Figure 2b as filled circles. Each such vertex is attributed with the single-qubit gate it can execute or, in the case of IRZs, with the ability to perform initializations and readouts. The multi-qubit connectivity is encoded by edges, like in the connectivity graph, with additional attributes declaring that those edges provide two-qubit gates and therefore also traditional connectivity information. This is represented by a solid bidirectional arrow.

To support qubit shuttling, we add routing information to the graph. To this end, each junction is represented by an additional vertex in the architecture, though it does not contain any information on executable gates. Furthermore, edges are established between vertices (including manipulation vertices, IRZs, and junctions) if and only if a direct shuttling operation between those vertices is possible. Such edges are attributed with the ability to perform shuttling and the length of the edge.

Shuttling edges are generally not independent of each other. To reduce wiring, multiple edges can be controlled by the same wire and then represent a *bus*. To formalize this, we add another attribute by associating each shuttling edge with a *control*. A shuttling operation can then be defined by assigning an individual shuttling distance to each control, which can also be negative (shuttling in reverse direction) or zero (no shuttling).

Vertices for shuttling are drawn as hollow squares in Figure 2b, and colored double arrows between them and the operation zones represent shuttling edges. The color denotes the respective control.

Qubit Mapping and Routing. In the SpinBus architecture, two qubits can only be used in a two-qubit operation if they reside in the same manipulation zone. Similarly, single-qubit operations are applied to a position on one side of some manipulation zone. More exactly, at each manipulation zone, two possibly different unitary gates can be executed in parallel on two different qubits. Furthermore, qubit initializations and measurements are performed in the respective IRZs. Therefore, we have to map (logical) qubits to operational zones (or, more generally, to any location on the device).

The *mapping problem* can be formalized as an optimization problem that, given an architecture, a quantum circuit, a subset of its gates, and a current qubit mapping, aims to find a new mapping which enables the execution of those gates and minimizes the expected error of executing the remaining circuit.

To establish new mappings, qubits have to be moved through the device, which is referred to as the *routing problem*. For established architectures with stationary physical qubits, this is done by inserting SWAP-gates, thereby exchanging the logical information between physical qubits. Architectures that employ shuttling operations are capable of physically moving qubits and thus allow reconfigurations while avoiding SWAP-gates and their resulting errors. Thus, given an architecture and two mappings, the routing problem can be defined as finding a finite sequence of shuttling operations that accordingly remaps the qubits and that is minimal with respect to combined errors induced by the operations.

This problem can be generalized to a class of problems named *multiple-agent path finding (MAPF)* [11], in which one has to determine collision-free paths for multiple agents in a shared environment. In our case, the agents are physical qubits. A common category of algorithms for solving MAPF problems is based on *constraint-based search (CBS)* [10]. The idea of this approach is to first individually find an isolated path for each agent. Afterwards, the combined paths are evaluated, and it is checked whether conflicts occur. Such conflicts are then transformed into two constraints. Each agent involved in a conflict generates a constraint for the other involved agent such that both constraints on their own prohibit the conflict. The algorithm now proceeds by branching on both constraints and creating sets of constraints including the newly generated ones. Then the agents are routed again in isolation but additionally obeying the assigned constraints. This repeats until a conflict-free solution is found. The CBS algorithm thus performs a search over the search space consisting of sets of constraints that internally uses an adapted single-agent shortest-path algorithm.

For routing with shuttling operations, we identified two kinds of conflicts and two corresponding types of constraints:

Distance Conflicts: To prevent crosstalk, we require a minimum distance between two qubits. More specifically, we require twice the period of the cosinusoidal shuttling signals, thus leaving a free "pocket" between the qubits. On junctions where two controls meet, it is possible to insert qubits too close to each other.

Shuttling Conflicts: Since the qubits are routed in isolation, it can happen that two qubits sharing the same control require shuttling over different distances or in different directions. However, at each time step the shuttling distance and direction must be unique for each bus.

Occupancy Constraints: If a distance conflict occurs, at least two occupancy constraints are generated and assigned to the involved qubits. An occupancy constraint consists of a connected subgraph around some qubit q and a time t. If this constraint is assigned to some other qubit q', it prohibits q' from entering the subgraph at time t when routing q'. Symmetrically, qubit q is assigned an occupancy constraint around the position of q' at time t.

Shuttling Constraints: A shuttling conflict must be due to two qubits sharing the same control. To avoid shuttling conflicts, a shuttling constraint is generated that forces both into the same direction.

CBS provides optimal solutions to the routing problem. Moreover, the MAPF problem is known to be NP-complete [2] (just as the general qubit routing problem [3,8]). Therefore, CBS is only applicable to device architectures and circuits of moderate size (such as those that are currently implementable). However, it can serve as a baseline and reference for assessing heuristic methods that will be required for more complex systems. Moreover, due to our flexible modeling approach, it provides a general framework for handling shuttling architectures.

Ongoing and Future Work. An essential aspect that has largely been ignored so far in our work is robustness against errors. High-level quantum algorithms require error-free qubits and logic gates. However, due to interaction with the environment, qubits suffer from decoherence effects that impact the durability of the information stored. Another source of noise is gate infidelities, which are due to the fact that the user-specified gate operations do not precisely correspond to their physical implementation [1]. A quantum compiler thus has to translate ideal quantum gate operations used in quantum algorithms into machine-level operations and to mitigate the loss of quantum information over time.

On a lower level, quantum systems can be protected against and recover from such errors by using quantum error correction (QEC) and fault-tolerant computations [9]. At a higher level of abstraction, further improvements are enabled by making the quantum circuits themselves resilient to noise. This can be achieved by noise-aware compilation methods for quantum circuits employing symbolic or numerical planning and optimization algorithms [5]. We are planning to apply and develop such techniques also in our setting.

Moreover, we are planning to use extensions of MAPF algorithms to not only cover routing but also other circuit compilation tasks. For example, *Combined Task Assignment and Path Finding (TAPF)* [12,13] could be employed to first let qubits themselves determine their target locations and then do path planning, i.e., to integrate qubit mapping and routing. Another promising approach is *Lifelong Multi-Agent Path Finding (L-MAPF)* [7], which enables the dynamic integration of new planning tasks over time, and thus also addresses the scheduling problem.

References

1. Almudever, C.G., Lao, L., Fu, X., Khammassi, N., Ashraf, I., Iorga, D., Varsamopoulos, S., Eichler, C., Wallraff, A., Geck, L., Kruth, A., Knoch, J., Bluhm, H., Bertels, K.: The engineering challenges in quantum computing. In: Design, Automation & Test in Europe. pp. 836–845. IEEE (2017). https://doi.org/10.23919/DATE.2017.7927104
2. Barer, M., Sharon, G., Stern, R., Felner, A.: Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem. In: Symposium on Combinatorial Search (2014), https://api.semanticscholar.org/CorpusID:14537415
3. Botea, A., Kishimoto, A., Marinescu, R.: On the complexity of quantum circuit compilation. In: 11th Int. Symp. on Combinatorial Search (SoCS). pp. 138–142. AAAI Press (2018). https://doi.org/10.1609/SOCS.V9I1.18463
4. Cowtan, A., Dilkes, S., Duncan, R., Krajenbrink, A., Simmons, W., Sivarajah, S.: On the qubit routing problem. In: 14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC). LIPIcs, vol. 135, pp. 5:1–5:32. Schloss Dagstuhl – Leibniz-Zentrum für Informatik (2019). https://doi.org/10.4230/LIPIcs.TQC.2019.5
5. Kusyk, J., Saeed, S.M., Uyar, M.U.: Survey on quantum circuit compilation for noisy intermediate-scale quantum computers: Artificial intelligence to heuristics. IEEE Transactions on Quantum Engineering **2**, 1–16 (2021). https://doi.org/10.1109/TQE.2021.3068355
6. Künne, M., Willmes, A., Oberländer, M., Gorjaew, C., Teske, J.D., Bhardwaj, H., Beer, M., Kammerloher, E., Otten, R., Seidler, I., Xue, R., Schreiber, L.R., Bluhm, H.: The SpinBus architecture: Scaling spin qubits with electron shuttling. arXiv (2023). https://doi.org/10.48550/arXiv.2306.16348
7. Madar, N., Solovey, K., Salzman, O.: Optimal and bounded-suboptimal multi-goal task assignment and path finding. In: 15th International Symposium on Combinatorial Search (SoCS2022). pp. 118–126 (2022). https://doi.org/10.1609/socs.v15i1.21759
8. Molavi, A., Xu, A., Diges, M., Pick, L., Tannu, S., Albarghouthi, A.: Qubit mapping and routing via MaxSAT. In: 55th IEEE/ACM International Symposium on Microarchitecture (MICRO). pp. 1078–1091 (2022). https://doi.org/10.1109/MICRO56248.2022.00077
9. Roffe, J.: Quantum error correction: an introductory guide. Contemporary Physics **60**(3), 226–245 (2019). https://doi.org/10.1080/00107514.2019.1667078
10. Sharon, G., Stern, R., Felner, A., Sturtevant, N.R.: Conflict-based search for optimal multi-agent pathfinding. Artificial Intelligence **219**, 40–66 (2015). https://doi.org/10.1016/j.artint.2014.11.006
11. Stern, R.: Multi-Agent Path Finding – An Overview, pp. 96–115. Springer (2019). https://doi.org/10.1007/978-3-030-33274-7_6
12. Tang, Y., Ren, Z., Li, J., Sycara, K.: Solving multi-agent target assignment and path finding with a single constraint tree (2023). https://doi.org/10.48550/arXiv.2307.00663
13. Zhong, X., Li, J., Koenig, S., Ma, H.: Optimal and bounded-suboptimal multi-goal task assignment and path finding. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 10731–10737 (2022). https://doi.org/10.1109/ICRA46639.2022.9812020
14. Zulehner, A., Wille, R.: Compiling SU(4) quantum circuits to IBM QX architectures (2018). https://doi.org/10.48550/arXiv.1808.05661