Towards the Munich Quantum Software Stack
Enabling Efficient Access and Tool Support for Quantum Computers

Martin Schulz
Chair of Computer Architecture and Parallel Systems
Technical University of Munich
Munich, Germany
schulzm@in.tum.de

Laura Schulz
QCT Department
Leibniz Supercomputing Centre
Garching, Germany
schulz@lrz.de

Martin Ruefenacht
QCT Department
Leibniz Supercomputing Centre
Garching, Germany
Martin.Ruefenacht@lrz.de

Robert Wille
Chair of Design Automation
Technical University of Munich
Munich, Germany
robert.wille@tum.de

***Abstract*—As quantum computing systems mature and move from laboratories to production computing environments, corresponding software stacks are becoming key for their successful utilization. In particular, the expected use of quantum systems as HPC accelerators requires a deep integration with the existing and widely deployed HPC software stacks. Additionally, new requirements such as dynamic compilation and new challenges for tools and programming models must be considered. We tackle these challenges by developing the *Munich Quantum Software Stack*—a comprehensive initiative by the *Munich Quantum Valley* to offer a flexible, efficient, and user-oriented software environment. In this poster, we describe the core components and workflows, and how they will enable this transformation from quantum experiments to quantum accelerators.**

I. The Need for a Unified Software Stack

As quantum computing systems mature and move from laboratories to production computing environments, we must also develop the needed software environments [1]. This is especially important, as it is expected that quantum systems will be part of larger quantum/classic workflows and will serve as accelerators for targeted problems. In such scenarios, the user community will shift from individual physics experts accessing individual quantum computing systems (typically in lab environments) to a wide range of domain users, not intimately familiar with quantum mechanics (as sketched in Figure 1). Such users want to use quantum systems without knowing the physical details and potentially will also want to target multiple different quantum systems.

This transition from single experiments to production-ready classic/quantum hybrid environments can only be accomplished by deploying a comprehensive software stack that addresses the following challenges.

- It must be able to broaden the user community: instead of individual quantum physicists, it must be usable by domain users from varying backgrounds and fields, and it must be able to support novel quantum algorithms.
- It must provide higher-level and easier-to-use abstractions for programming: this will enable easier access for new, non-physics users as well as enable new optimizations.

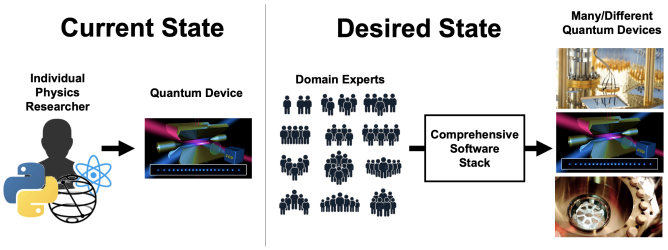

Fig. 1. Transforming access to quantum systems.

- It must be integrated into *High Performance Computing* (HPC) systems: quantum systems are suitable for accelerating very targeted problems, while the remaining computation requires classic HPC systems. Further, as we scale QC systems, we require significant compute power to drive QC systems and their needed compilation, topology optimization, and (ultimately) correction.
- It must be accessible from HPC systems: as the consequence of using QC systems as accelerators, their usage will be through HPC systems, which must be supported with new hybrid programming approaches and tooling. Further, the quantum software stack must match the compute environments of HPC systems.
- It must provide extensive design automation and software tools that enable large scale systems and applications: most tasks are complex and cannot be handled manually anymore—requiring a high degree of automation and corresponding data-structures as well as core methods. In order to get that, we should employ the extensive experience available from classical design automation.

These observations have led to several groups working in this direction, including large scale efforts like IBM Qiskit [2], in the Quantum Delta [3] or the RIKEN software efforts [1]. However, most efforts either focus on quantum computing only and de-emphasize the needed integration with HPC, rely on quick prototyping but non-scalable solutions, or target indi-

[1] https://www.riken.jp/en/research/labs/rqc/

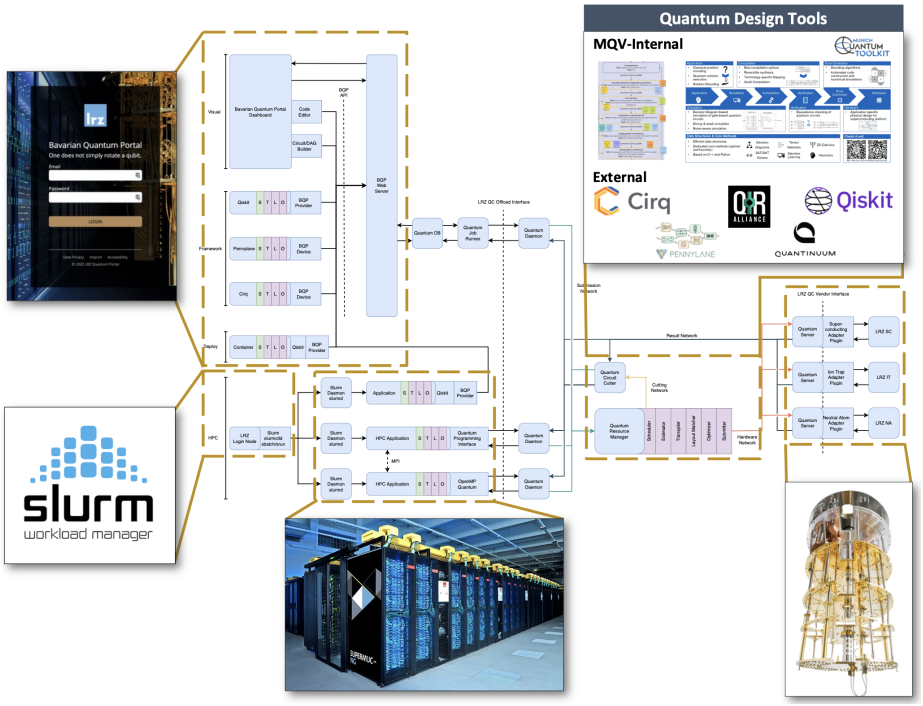

Fig. 2. The Munich Quantum Software Stack at a glance.

vidual systems. Other projects offer approaches for a subpart of these questions, like XACC [4] for hybrid programming, but miss a larger integration towards a true full stack HPCQC workflow from user to backend execution.

II. The Munich Quantum Software Stack

In the *Munich Quantum Valley* (MQV)[2], we are addressing these challenges in a holistic and full-stack manner, by developing a dedicated quantum software stack that supports direct access for experiments, integration with HPC systems, and driving implicit quantum compilation as well as optimization toolkits which, eventually, enable multiple backends.

Our approach is sketched in Figure 2. Users can access the system either via a dedicated portal with multiple language backends (top left) or via HPC systems using traditional schedulers like SLURM (bottom left), which drive hybrid applications capable of offloading parts of their computation to QC backends. The relevant parts of the programs are then processed via a quantum resource manager as well as corresponding quantum design tools (top-right), including those from the *Munich Quantum Toolkit* (MQT)[3]. Finally, the resulting quantum circuits are executed by a suitable backend system (bottom right). This workflow is transparent for the user, hides the particular complexities, and still enables "expert paths" for experimental computations.

III. Status and Next Steps

The presented software stack is currently being developed within the Munich Quantum Valley and deployed at the Leibniz Supercomputing Centre via its *Quantum Integration Centre* (QIC). A first prototype has been tested using LRZ's 5 Qubit superconducting system, which has been installed in a collaboration between LRZ and IQM. Next steps include the integration of further toolkits and compiler optimizations, the development of more sophisticated scheduling techniques, as well as new higher-level programming abstraction. With that, the Munich Quantum Software Stack will form the foundation for efficient, production-level full-stack HPCQC-enabled quantum computing.

[2]https://www.munich-quantum-valley.de/

[3]https://www.cda.cit.tum.de/research/quantum/mqt/

Acknowledgment

This research is part of the *Munich Quantum Valley* (MQV), and is supported by the Bavarian state government with funds from the Hightech Agenda Bayern as well as several projects funded by the Federal Ministries for Economic Affairs and Climate Action and the Education and Research.

References

[1] M. Schulz, M. Ruefenacht, D. Kranzlmüller, and L. B. Schulz, "Accelerating hpc with quantum computing: It is a software challenge too," *Computing in Science & Engineering*, vol. 24, no. 4, pp. 60–64, 2022. [Online]. Available: https://doi.org/10.1109/MCSE.2022.3221845

[2] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, Y. Ben-Haim, D. Bucher, F. J. Cabrera-Hernández, J. Carballo-Franquis, A. Chen, C.-F. Chen *et al.*, "Qiskit: An open-source framework for quantum computing," 2019. [Online]. Available: https://qiskit.org/

[3] X. Fu, L. Riesebos, L. Lao, C. Almudever, F. Sebastiano, R. Versluis, E. Charbon, and K. Bertels, "A heterogeneous quantum computer architecture," 05 2016, pp. 323–330.

[4] A. J. McCaskey, D. I. Lyakh, E. F. Dumitrescu, S. S. Powers, and T. S. Humble, "XACC: a system-level software infrastructure for heterogeneous quantum–classical computing," *Quantum Science and Technology*, vol. 5, p. 024002, 2020. [Online]. Available: https://doi.org/10.1088/2058-9565/ab6bf6