


# Towards Real-World System-Level Integration of Quantum Accelerators: A Hardware/Software Co-Design Approach

Ralf Ramsauer <sup>1</sup>, Benno Bielmeier <sup>1</sup>, and Wolfgang Mauerer <sup>1,2</sup>

## Abstract:

This work-in-progress explores the architectural and systemic foundations required for future tight integration of quantum accelerators in heterogeneous computing environments. We propose and implement a modular architecture where quantum processing units operate as peripheral devices, supporting pulse-level control interfaces as well as high-level circuit execution offloading, while internally managing tasks such as compilation, transpilation, and scheduling. To enable efficient quantum-classical orchestration, we introduce a Quantum Abstraction Layer (QAL) at the operating system level that facilitates seamless communication, resource management, and smooth integration with established software frameworks.



We follow a two-step design approach: Firstly, we validate our architecture using device models and simulations in virtualised environments. Secondly, we provide a FPGA-based surrogate implementation that supports both, result-accurate and timing-accurate modes, which allows for full-stack emulation, interface validation, and performance evaluation in the absence of physical quantum hardware. The overarching objective is to establish an extensible platform for prototyping, Hardware/Software Co-Design, and investigating the practical quantum advantage under realistic system-level constraints of various use cases. We believe this process will result in a sufficiently complete and useful design that can be immediately adopted by hardware vendors, as the design process can be finalised before real hardware is available.

**Keywords:** Hardware Platforms and Architectures, Hardware/Software Co-Design, Integrated Quantum Systems, Quantum Accelerators, Quantum/Classical Co-Design, System-Level Abstraction, Temporal Modelling and Emulation, Vertical System Integration

## 1 Introduction

Current quantum computing platforms remain largely experimental, comprising loosely integrated hardware and software components that resemble complex physical experiments. They rely on remotely managed, finely controlled signal generators (RF pulses, lasers, masers) to manipulate quantum states. Although these platforms have driven substantial scientific progress, they face inherent challenges in scalability, seamless integration into classical infrastructure, and broad applicability within heterogeneous computing environments. In particular, few existing efforts address vertical system integration, including the interplay

---

<sup>1</sup> Regensburg University of Applied Sciences, Galgenbergstraße 32, 93053 Regensburg,  
ralf.ramsauer@othr.de,  <https://orcid.org/0009-0007-4033-8515>;  
benno.bielmeier@othr.de,  <https://orcid.org/0009-0003-3440-7239>

<sup>2</sup> Siemens AG, Technology, Friedrich-Ludwig-Bauer-Straße 3, 85748 Garching bei München,

with hardware components that—while not yet physically realised—can be meaningfully designed and evaluated through simulation [Zh23].

In our work, we investigate the architectural and systemic foundations required for tightly integrated quantum accelerators into existing classical infrastructure. We model quantum processing units as modular peripheral devices embedded within or tightly connected with classical hosts. Each unit supports heterogeneous interfaces—from low-level, waveform-based pulse control to high-level circuit execution.

At the operating system level, we introduce a kernel-level Quantum Abstraction Layer (QAL). It provides a uniform, technology-agnostic interface for resource management, communication, and error handling between host and accelerator. A complementary user-space control infrastructure offers high-level bindings to established SDKs (e. g., Qiskit, QDMI [Wi]).

To validate and refine our design in the absence of physical hardware, we instantiate virtualised device models within a QEMU-based environment. Our hardware model supports two operational modes. (1.) Result-accurate emulation to provide correct results at the cost of exponential runtime, and (2.) Timing-accurate emulation to run without quantum fidelity, with realistic runtime behaviour. (1.) allows for iterative algorithm tuning and interface validation, while (2.) allows for real-time, latency and run-time analyses.

In our framework, the timing-accurate simulation within the QEMU-based environment primarily serves for early-stage validation of system design, interface behaviour, and software-hardware interaction under realistic assumptions. While this virtualised approach enables rapid prototyping and iterative co-design, it does not fully capture the precise execution characteristics of eventual quantum hardware. Therefore, final timing analyses and more accurate real-time behaviour studies are intended to be conducted on FPGA-based platforms, which offer closer alignment with the physical properties and latencies of future quantum devices.

This progression enables quantitative investigation of performance limits, break-even points, and the practical feasibility of quantum speed-up within tailored [SSM23; WSM22], tightly integrated, heterogeneous computing architectures as they may emerge in future deployments.

## 2 State of the Art

Integration of quantum hardware into classical infrastructures operates across multiple abstraction levels. At the highest level, software developers design quantum algorithms expressed as quantum circuits [Kh]. At the lowest level, quantum hardware uses specialised instruments such as arbitrary-waveform generators (AWGs), radio-frequency generators, and FPGAs to manipulate physical qubits precisely. Bridging these abstraction layers necessitates homogeneous interfaces and standardised formats, as well as clearly defined

abstraction layers that enable efficient communication, reduce complexity, and facilitate seamless integration.

Prior research has explored low-level representations of quantum programs by encoding quantum instructions as binary streams of Quantum Intermediate Representation (QIR) or Quantum-ISA commands, which are parsed and dispatched by a quantum control processor to analogue signal generators [Fu; GQS; MN; SBW]. Standardisation efforts have further advanced with the introduction of the Quantum Device-agnostic middleware Interface (QDMI) within the Munich Quantum Software Stack, providing a uniform abstraction of hardware characteristics to facilitate the adaptation of high-level tools to heterogeneous quantum platforms [Wi].

Unified, layered architectural approaches addressing multiple abstraction levels have been identified as promising for hybrid quantum-classical computing systems. Full-stack frameworks have been proposed to span these abstraction layers and integrate quantum resources into high-performance computing environments [EGS24; Gi; Zh23]. Despite these advances, the field still lacks comprehensive end-to-end implementations that also combine accurate hardware simulation with vertically integrated software and hardware stacks.

### 3 Quantum Architecture Framework

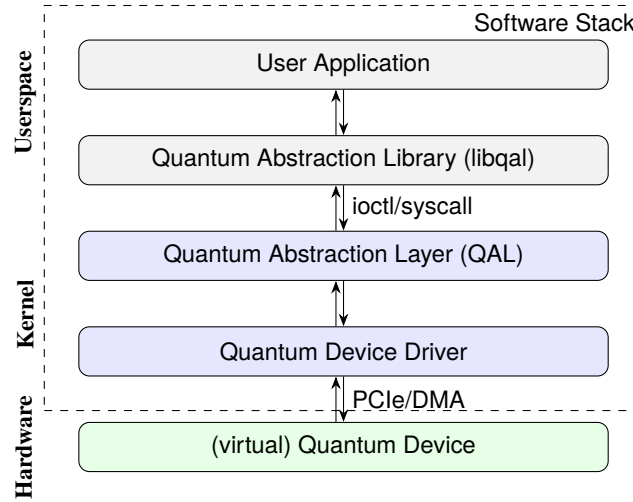


Fig. 1: Software Stack of Quantum Architecture comprising five conceptual components that reside in userspace, kernel space and the (virtual) hardware.

The proposed architecture integrates five principal components into a cohesive and layered system, as depicted in Figure 1. To foster interoperability across current computing

ecosystems, it builds on established (operating-system) interfaces and strives to provide seamless compatibility with existing software and hardware stacks by adhering to proven standards. Its foundation is a kernel-level Quantum Abstraction Layer (QAL), which provides a streamlined system call interface for device configuration, job scheduling and submission, and monitoring. A lightweight user-space library encapsulates high-level quantum programs into compact binary representations and provides access to kernel interfaces. Initially, we explicitly avoid instruction set architecture (ISA) extensions, acknowledging prior research indicating that quantum ISA developments are promising but presently impractical due to current hardware limitations [Cr22; SBW]. Instead, we propose an intermediary solution utilising established hardware communication protocols and intermediate representation. This library offers to implement backend support for established frameworks such as Qiskit and PennyLane and submit jobs after compilation and transpilation to the hardware. In kernel space, specific device drivers facilitate interaction with the actual control hardware while exposing a uniform kernel API. A central research question addressed in this work concerns the appropriate level of abstraction for interfacing quantum hardware within heterogeneous computing systems. Our architecture is deliberately designed to remain flexible and extensible regarding (yet unknown) abstraction boundaries, recognising that different applications and user expertise necessitate access at varying semantic levels. While the framework aims to expose low-level control capabilities—such as direct manipulation of pulse-level sequences for fine-grained hardware tuning under expert supervision—it equally supports higher-level representations, including quantum circuits and algorithmic workflows, to facilitate broader usability and integration with established quantum software ecosystems.

Throughout the project, we investigate how these layers of abstraction can be coherently structured to balance hardware accessibility, system efficiency, and user programmability, to ensure that the platform accommodates both specialist requirements and future *general-purpose* quantum computing workloads [SWM23].

Complementing the software stack, both a virtual device model and an FPGA-based prototype are employed to support iterative development, interface validation, and performance evaluation in advance of the availability of physical quantum processing units (QPUs).

In summary, the proposed architecture establishes a flexible experimental platform to explore the interplay of hardware and software in quantum-classical systems. It enables the systematic evaluation of architectural choices, control interfaces, and application-specific workflows under realistic operational conditions. By supporting both virtual and physical prototyping, it provides the means to assess performance boundaries and identify practical quantum benefits in diverse computational contexts.

## References

- [Cr22] Cross, A.; Javadi-Abhari, A.; Alexander, T.; De Beaudrap, N.; Bishop, L. S.; Heidel, S.; Ryan, C. A.; Sivarajah, P.; Smolin, J.; Gambetta, J. M.; Johnson, B. R.: OpenQASM 3: A Broader and Deeper Quantum Assembly Language. *ACM Transactions on Quantum Computing*, 2022.
- [EGS24] Elsharkawy, A.; Guo, X.; Schulz, M.: Integration of Quantum Accelerators into HPC: Toward a Unified Quantum Platform. In: *IEEE International Conference on Quantum Computing and Engineering (QCE)*. 2024, DOI: 10.48550/arXiv.2407.18527.
- [Fu] Fu, X.; Rieseboos, L.; Rol, M.; Van Straten, J.; Van Someren, J.; Khammassi, N.; Ashraf, I.; Vermeulen, R.; Newsum, V.; Loh, K., et al.: eQASM: An Executable Quantum Instruction Set Architecture. In: *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. DOI: 10.1109/HPCA.2019.00040.
- [Gi] Giortamis, E.; Romão, F.; Tornow, N.; Bhatotia, P.: QOS: A Quantum Operating System, DOI: 10.48550/arXiv.2406.19120.
- [GQS] Guo, X.; Qin, K.; Schulz, M.: HiSEP-Q: A Highly Scalable and Efficient Quantum Control Processor for Superconducting Qubits, DOI: 10.48550/arXiv.2308.16776.
- [Kh] Khan, A. A.; Ahmad, A.; Waseem, M.; Liang, P.; Fahmideh, M.; Mikkonen, T.; Abrahamsson, P.: Software Architecture for Quantum Computing Systems - A Systematic Review. *SSRN Journal*, DOI: 10.2139/ssrn.4040490.
- [MN] McCaskey, A.; Nguyen, T.: A MLIR Dialect for Quantum Assembly Languages. In: *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*. DOI: 10.1109/QCE52317.2021.00043.
- [SBW] Stade, Y.; Burgholzer, L.; Wille, R.: Towards Supporting QIR: Thoughts on Adopting the Quantum Intermediate Representation, DOI: 10.48550/arXiv.2411.18682.
- [SSM23] Schönberger, M.; Scherzinger, S.; Mauerer, W.: Ready to Leap (by Co-Design)? Join Order Optimisation on Quantum Hardware. In: *Proceedings of ACM SIGMOD/PODS International Conference on Management of Data*. 2023, DOI: 10.1145/3588946.
- [SWM23] Safi, H.; Wintersperger, K.; Mauerer, W.: Influence of HW-SW-Co-Design on Quantum Computing Scalability. In: *2023 IEEE International Conference on Quantum Software (QSW)*. 2023, DOI: 10.1109/QSW59989.2023.00022.
- [Wi] Wille, R.; Schmid, L.; Stade, Y.; Echavarria, J.; Schulz, M.; Schulz, L.; Burgholzer, L.: QDMI - Quantum Device Management Interface: Hardware-Software Interface for the Munich Quantum Software Stack. In: *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)*. DOI: 10.1109/QCE60285.2024.10411.
- [WSM22] Wintersperger, K.; Safi, H.; Mauerer, W.: QPU-System Co-Design for Quantum HPC Accelerators. In: *Proceedings of the 35th GI/ITG International Conference on the Architecture of Computing Systems*. Gesellschaft für Informatik, 2022, DOI: 10.1007/978-3-031-21867-5\_7.
- [Zh23] Zhang, F.; Zhu, X.; Chao, R.; Huang, C.; Kong, L.; Chen, G.; Ding, D.; Feng, H.; Gao, Y.; Ni, X., et al.: A Classical Architecture For Digital Quantum Computers. *ACM Transactions on Quantum Computing*, 2023.