Overhead in Quantum Circuits with Time-Multiplexed Qubit Control

Featured in Physics

Overhead in Quantum Circuits with Time-Multiplexed Qubit Control

Marvin Richter,* Ingrid Strandberg, Simone Gasparinetti, and Anton Frisk Kockum†
Department of Microtechnology and Nanoscience, Chalmers University of Technology, 412 96 Gothenburg, Sweden

(Received 21 October 2025; revised 23 January 2026; accepted 24 February 2026; published 14 April 2026)

When scaling up quantum processors in a cryogenic environment, it is desirable to limit the number of qubit drive lines going into the cryostat, since fewer lines make cooling of the system more manageable and the need for complicated electronics setups is reduced. However, although time multiplexing of qubit control enables using just a few drive lines to steer many qubits, it comes with a trade-off: fewer drive lines means fewer qubits can be controlled in parallel, which leads to an overhead in the execution time for quantum algorithms. In this article, we quantify this trade-off through numerical and analytical investigations. For standard quantum processor layouts and typical gate times, we show that the trade-off is favorable for many common quantum algorithms—the number of drive lines can be significantly reduced without introducing much overhead. Specifically, we show that couplers for two-qubit gates can be grouped on common drive lines without any overhead up to a limit set by the connectivity of the qubits. For single-qubit gates, we find that the serialization overhead generally scales only logarithmically in the number of qubits sharing a drive line, and the serialization overhead relative to total quantum circuit duration tends to grow only sublinearly or stay nearly constant with the total number of qubits on the quantum processor. These results are promising for continued progress toward large-scale quantum computers.

DOI: 10.1103/82cj-lfzy

I. INTRODUCTION

Solid-state quantum processors, e.g., those using superconducting qubits [1–3] or semiconductor quantum dots [4,5], require operation at millikelvin temperatures to facilitate the systems to be initialized in their ground state and prevent thermal excitations. In current standard architectures, qubit control and readout are performed using microwave pulses generated at room temperature. These pulsed signals are delivered via coaxial lines that connect the quantum processor to the room-temperature electronics. In order to protect the qubits from thermal radiation flowing from the higher temperature stages to the lower temperature stages, the coaxial lines are filtered and thermally anchored at several stages within the cryostat [6].

As quantum processors are being scaled up to larger numbers of qubits, one of several challenges that arise is the cooling [7]. The cooling challenges stem from the fact that every qubit must be individually addressable. In the current small- to medium-scale systems, this addressability is typically achieved by individual connections, with each qubit having a dedicated control line. While this method is straightforward, it has three major limitations that prohibit scaling up. First, as the number of qubits increases, each line adds a thermal load on the cryostat which has a limited cooling power; a recent study showed that the maximum acceptable thermal load for one of the largest commercially available cryostats is reached for less than 200 qubits [8]. Second, there is a limit on the number of coaxial lines that can physically fit in the cryostat. Third, at the scale of 100 qubits and beyond, the cabling inside the cryostat constitutes a majority of the total cost for the system [7]. There are various suggestions for avoiding these bottlenecks, such as using photonic links instead of coaxial cables [9–12] or using single-flux-quantum logic [13–15]. Another option is to perform frequency- or time-domain multiplexing via a controller mounted on the cold stage of the cryostat, close to the quantum chip [7,16–19].

While frequency-multiplexed readout is well established [20,21], frequency-multiplexed control is still in its early stages of development, with ongoing efforts to mitigate detrimental effects such as crosstalk and unwanted interference [22–25]. In contrast, time-multiplexed control has received little direct attention, even though enabling hardware such as complementary metal-oxide semiconductor (CMOS)-based switches have been demonstrated to enable efficient characterization of large numbers of semiconductor quantum dots [26–29], and are also considered

*Contact author: marvin.richter@chalmers.se
†Contact author: anton.frisk.kockum@chalmers.se

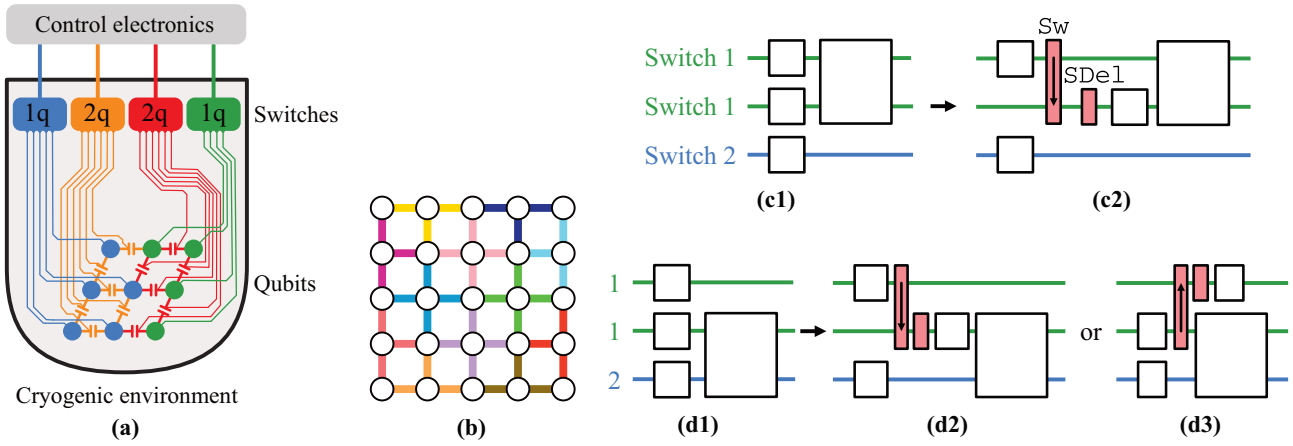

FIG. 1. Time-multiplexed qubit control and compilation considerations. (a) A schematic drawing of control-signal demultiplexing via switches to qubits and coupling elements. (b) Layout for a 5×5 square-grid quantum processor with twelve color-coded coupling control groups for two-qubit gates, which minimizes required control lines without increasing the duration of any quantum circuit. (c) Circuit serialization modeling for single-qubit gates: (c1) target circuit and (c2) serialized circuit with controller switch gates `Sw` (red, arrow) and delay gates `SDel` (red) representing switching time. Quadratic blocks represent single- and two-qubit gates. (d) Impact of gate-execution ordering on overhead: (d1) target circuit, (d2) default ordering by qubit index, and (d3) distance-optimized ordering based on subsequent two-qubit gates.

for the purpose of scalable quantum computing [30–33]. A CMOS device working at 10 mK with low port-to-port-crosstalk, designed for time-multiplexed superconducting qubit control, has also been demonstrated [34]. Additionally, there are significant efforts to create superconducting switches [35–42] and semiconducting-superconducting hybrid devices [43–45] to reduce heat dissipation compared to CMOS technology. While such devices are under active development, a systematic study of their prospected impact in quantum computing systems is missing. Here, we analyze the feasibility of time-multiplexing approaches for qubit control.

In this article, we numerically and analytically study the impact on quantum computing performance when a time-multiplexing approach is adopted to facilitate scaling quantum processors. The approach we consider is one in which the number of lines is reduced by controlling multiple qubits sequentially via a single drive line [46]; see Fig. 1(a). To isolate the effects of time multiplexing, we do not consider the possibility of additional frequency multiplexing. If multiple qubits with different frequencies share a single drive line in this setting, we assume that only one qubit can be driven at a time, meaning the other qubits must idle during that period. Furthermore, the switching itself will introduce some idling time for all qubits. This sequential control thus introduces an overhead in the runtime of a given algorithm. Given the limited qubit coherence time, it is natural to ask how large this overhead is and whether it has a negative impact on the fidelity of quantum algorithms [47].

We develop a compilation algorithm [48] to minimize the runtime overhead from time multiplexing. Although a few compilation methods for similar purposes have been put forward [49–52], they have only been benchmarked on relatively small processors with a focus on the compiler performance itself. Here, using the open-source quantum software stack Qiskit [53] to serialize qubit operations and emulate idling times due to multiplexer hardware constraints, we investigate the resulting runtime overhead when using our compilation algorithm for random circuits and a benchmark set of quantum algorithms [54] on different quantum processor layouts and focus on the consequences of this overhead for multiplexing design. In particular, the processor layouts we consider are a square-grid layout similar to the Google Sycamore and Willow processors [55,56] with up to 121 qubits, and the heavy-hexagon layout of the 127-qubit IBM Eagle-type processor [57].

We find that time multiplexing enables reducing the number of control lines in quantum computers while only paying a low price in circuit runtime overhead. The number of lines controlling couplers for two-qubit gates can even be reduced without overhead by a factor determined by the qubit connectivity of the quantum processor layout, since a single qubit is connected to multiple couplers but can only take part in one two-qubit gate at a time. For single-qubit gates, we observe that reducing the number of control lines by a factor k in most cases only generates an overhead scaling logarithmically with k, and we are able to explain this finding using queueing theory. We also make several more observations about the runtime for implementations of specific standard quantum algorithms. Overall, our main findings are encouraging for the development of larger quantum computers—the need for fewer

control lines does not appear likely to become a limiting factor in that endeavor.

This article is organized as follows. In Sec. II, we present the considered setup, discuss the impact of time multiplexing on two-qubit gates, describe how we model and compile circuits with time multiplexing of single-qubit control, and detail how we benchmark our methods. The extent of the serialization overhead from time multiplexing of single-qubit control depends on many factors, such as the circuit to be executed, gate durations, and the number of qubits sharing a switch. We explore variations of these parameters in Sec. III and make observations about how their impact on the overhead scales. Finally, we summarize our findings and present our conclusions in Sec. IV and discuss avenues for future research in Sec. V.

II. TIME-MULTIPLEXED QUBIT CONTROL

Here, we describe our approach to compilation for time-multiplexed qubit control in more detail and explain how we benchmark this approach. First, we present general considerations and a discussion of related work in Sec. II A. Then, we discuss the impact of time multiplexing on two-qubit gates in Sec. II B and show that these permit a limited degree of multiplexing without overhead, and we therefore focus our study on the impact of single-qubit multiplexing. For single-qubit gates, we explain and motivate our modeling and compilation approach in Sec. II C. Finally, we describe, in Sec. II D, how we benchmark the compilation of time-multiplexed control for single-qubit gates.

A. General considerations and related work

To evaluate the feasibility of time-multiplexed qubit control, we consider a setup with superconducting qubits. A schematic overview of a representative multiplexing setup is shown in Fig. 1(a). Here, control electronics outside the dilution refrigerator connect to (de)multiplexers or switches at the cold stage of the cryostat, which in turn route control signals to selected qubits. Different colors represent different switches and the corresponding connected qubits or coupling elements. Drive signals for single-qubit gates on qubits are separated from drives to couplers that mediate two-qubit gates between qubits. This structure is common [58–60] and used in several high-profile experiments [55,56,61].

The presence of time-multiplexed control introduces a new step in the procedure for mapping a logical quantum circuit onto one physically executable on the quantum processor. This procedure is generally referred to as quantum circuit compilation [48,62]. Two critical steps in quantum circuit compilation without time-multiplexed control are translation into native gates of the hardware in question and qubit routing, which inserts `SWAP` gates to perform arbitrary two-qubit gates on a layout with limited connectivity [63–66]. To emulate the constraint of sequential qubit operations imposed by routing control signals through shared switches, we add a third step: serialization of concurrent gates that share the same switch. When we speak of serialization overhead in the rest of the article, we include both the time added from the serial rather than parallel application of gates and the time delay due to switching between the different control lines to execute the gates in series.

A few related works have previously explored compilation in such settings, though our article differs from these in key aspects. For instance, Ref. [49] presented a compilation method designed for multiplexed quantum control, where the main benchmark was the utilization efficiency of the classical control channels rather than the quantum circuit or algorithm performance. References [50,51] introduced compilers that include constraints imposed by shared classical control electronics, and a constraint functionally similar to shared controls was imposed in Ref. [52], where neighboring qubits were restricted from simultaneous operation due to crosstalk, similarly as if they shared a control line. These methods were tested on between 3 and 21 qubits, while our work here includes benchmarks on more than 1000 qubits. Most importantly, these previous works focused entirely on the performance of the compilers, while our study considers the runtime of the compiled circuits and what consequences that has for time-multiplexing schemes. Hence, we do not perform a fully optimized serialization, but implement a strategic serialization procedure that avoids superfluous switching and utilizes slack time induced by long two-qubit gates. Our serialization procedure is described in Sec. II C.

B. Impact on two-qubit gates

Time-multiplexing control of single-qubit and two-qubit gates presents different considerations, so we need to treat them separately. In our hardware model, which reflects common superconducting processor designs, two-qubit gates are implemented by driving couplers positioned between qubits. To alleviate the hardware overhead from individual drive lines, we propose employing the following time-multiplexing strategy for coupler lines: couplers surrounding alternating qubits are grouped together, as illustrated in Fig. 1(b). For the 5×5 qubit array, this multiplexing approach reduces the required coupler control lines from 40 to 12.

While concurrent two-qubit gates sharing the same control switch would have to be serialized for arbitrary coupler groupings, this proposed grouping avoids scheduling conflicts by exploiting the constraint that each qubit can participate in at most one two-qubit gate per time step, ensuring that control signals within a group are naturally separated in time. This type of natural serialization arises,

e.g., in surface-code stabilizer measurements in quantum error correction [56,67]; see also Appendix E.

We thus see that time-multiplexing control can reduce the number of control lines required for two-qubit gates essentially for free, up to a reduction factor given by the connectivity of the qubits. For a square grid, this reduction factor is four in the asymptotic limit; for a heavy-hexagon layout, it is three (grouping the couplers connected to each hexagon corner). Although one can consider larger groups of couplers, which would result in serialization overhead, we are content with the "free" reduction for now. Accordingly, we assume in the remainder of this article that *serialization of two-qubit gates introduces no overhead* in circuit runtime. Therefore, we focus the rest of our analysis on the serialization of single-qubit gates.

We note that there are other schemes for two-qubit gates, which do not rely on controlling a coupler. A prominent example is the cross-resonance gate [68–72], where one qubit is directly driven at the transition frequency of another qubit. For the purposes of this work, such gates are similar to single-qubit gates, and thus partly covered by the investigation of single-qubit gates in the remainder of this article. There are also cross-resonance three-qubit gates [73,74], where multiple qubits are directly driven. Ensuring that such gates can be performed could be achieved by a suitable grouping of drive lines for single-qubit gates, with drive lines for neighboring qubits assigned to different switches; this type of grouping is one of the options discussed for single-qubit gates below.

We also note here that three-qubit gates can be performed by simultaneously controlling the two couplers in a three-qubit chain [75–77]. For such gates, the two couplers involved would need to belong to different switches. If we use the layout in Fig. 1(b), reducing the number of coupler lines by a factor of four, half of the qubits (those surrounded by four couplers belonging to the same switch) could not act as the middle qubit in these three-qubit gates, but could act as one of the end qubits. Similarly, the other half of the qubits could not act as one of the end qubits in such three-qubit gates, but could act as the middle qubits. The number of possible three-qubit gates is thus halved in this setup. A detailed analysis of this kind of trade-off for three-qubit gates is beyond the scope of the current work.

C. Modeling serialization of single-qubit gates

For single-qubit gates, time multiplexing of control signals will generally introduce an overhead in the circuit duration. This can be easily understood, as only one qubit per switch can be controlled at a time. Effectively, concurrent single-qubit gates applied to qubits of the same switch must be serialized over time.

Numerical analysis of the overhead caused by time-multiplexed single-qubit control requires a circuit representation model of the switch. Therefore, we introduce two dummy gates: a single-qubit switch delay `SDel`, corresponding to the duration of a switching event, and a two-qubit switch gate `Sw` used for bookkeeping active qubits, each acting logically as the identity. While the two-qubit switch gate is an auxiliary construct with zero duration, it adds a dependency edge to the directed acyclic graph (DAG) that enforces the physical constraint that qubits sharing a switch cannot be driven simultaneously. The switch delay gate, in contrast, directly models the finite switching time t_{sw} experienced by the qubit awaiting the drive signal.

We illustrate circuit serialization with these dummy gates in a simple example in Fig. 1(c). First, Fig. 1(c1) shows a circuit with parallel single-qubit gates, where two of the qubits share a switch. The resulting serialized circuit is shown in Fig. 1(c2): after a single-qubit gate is executed on the first qubit, an `Sw` gate is applied between the first and second qubit and then an `SDel` gate is applied on the second qubit to account for nonzero switching time before the actual single-qubit gate is applied on the second qubit.

We determine the circuit execution time by calculating the critical path, i.e., the longest sequence of operations from qubit initialization to final measurement, in the circuit's DAG representation [78], where individual gate durations serve as edge weights. The zero-duration `Sw` gate does not directly contribute to the critical path length; however, it enforces temporal dependencies between previously concurrent single-qubit operations within the same control group, potentially extending the overall execution time. In contrast, the `SDel` gate, applied to the target qubit during control transitions, introduces a brief delay t_{sw} that may directly lengthen the critical path by increasing the cumulative duration of the longest operational sequence.

During the serialization stage, it matters in which order the single-qubit gates are processed. Since two-qubit gates are typically longer than single-qubit gates, single-qubit gates should be prioritized in the order of the smallest distance to the next two-qubit gate applied to the gate's qubit. By applying this heuristic, we bias toward situations like that depicted in Fig. 1(d), where switching occurs during a concurrent two-qubit gate and thereby does not increase the total duration. We provide more details on our optimization strategy for scheduling single-qubit gates in Appendix A. This appendix includes a runtime analysis for our heuristic serialization algorithm, which shows that qubit routing is likely to require more computational resources than serialization in a full transpilation pipeline.

We employ balanced group sizes where qubits are partitioned into groups containing either k or $k-1$ qubits, where k represents the number of qubits per switch. This approach maximizes switch-capacity utilization, avoiding highly uneven group distributions that would lead to inefficient hardware resource allocation.

We tested different grouping strategies to assign qubits to switches. The "trivial" layout creates assignments based

on qubit indices. Since nearby qubits typically have close indices, the trivial layout resembles the "clustering" layout based on qubit proximity. We also considered other layout strategies, but due to comparatively good performance (low overhead) and ease of reproducibility, we use the trivial layout for all following discussions in the main text. We refer the interested reader to Appendix B for further details and performance of other qubit grouping strategies.

D. Benchmarking

Here, we describe the target hardware and circuits used to quantify the overhead introduced by time multiplexing. This overhead depends on a variety of circuit and hardware parameters, but we identify and focus on a few dominant contributors. The number of qubits per switch, k, is one such principal factor for the overhead. Since single-qubit gates are serialized, the number of single-qubit gates N_1 in a circuit is a relevant factor as well. We also define and consider the gate densities of single- and two-qubit gates in a quantum circuit,

$$\rho_1 = \frac{N_1}{nD}, \quad \rho_2 = \frac{2N_2}{nD}, \quad \rho_{\text{total}} = \rho_1 + \rho_2, \tag{1}$$

with n being the number of qubits, N_2 being the number of two-qubit gates, D being the circuit depth, and ρ_{total} being the total gate density.

1. Hardware platforms

The exact duration overhead depends on several parameters of the target hardware. First, the native gate set determines the number of gates in the translated circuit. Second, the connectivity influences the single- and two-qubit gate densities of the executable circuit through routing. Third, the gate durations naturally affect overall execution time. To cover a range of noisy intermediate-scale quantum (NISQ) [79] devices, we begin with a relatively small 5×5 square grid of 25 qubits and then consider two larger systems of similar size, an 11×11 square grid with 121 qubits and an IBM Eagle device with 127 qubits in a heavy-hexagon layout.

For the square-grid architecture, we divide the native gates into three groups and assume their durations as follows: (i) two-qubit gates with a duration of 200 ns, (ii) single-qubit gates with a duration of 20 ns, and (iii) the virtual `RZ` gate that can be merged into adjacent gates as a phase shift and takes zero time to execute [80]. The first group includes two-qubit gates {`CZ`, `iSWAP`} that can both be executed with a simple flux-tunable coupler configuration [59,81]. The second group consists of single-qubit gates {`RX`, `RY`, `H`} with finite duration, implemented via microwave pulses on the qubit.

These choices are motivated by experimental developments. Google devices use single-qubit gates with durations between 18 and 25 ns (see the Supplementary Material of Ref. [56]). While flux-driven two-qubit gates have decreased their duration from around 200 ns in early demonstrations [59,82] down to 40–50 ns in more recent work [83,84], many alternative approaches remain slower. Since flux-tunable devices are susceptible to decoherence due to flux noise, fully microwave-driven couplers have been developed. These tend to have longer gate times, with recent implementations reporting between 200 and 380 ns [85,86]. Novel cat-transmon hybrid architectures also exhibit comparatively long gate times in the range of 200–400 ns [87,88]. Systems with long-range couplings have recently demonstrated two-qubit gates with durations between 160 and 430 ns [89,90], and a recently introduced multielement coupler similarly reports a duration of about 340 ns [91]. A two-qubit gate duration of 200 ns, therefore, represents a conservative but realistic estimate. Differences in absolute gate durations do not change the qualitative behavior and scaling of the serialization overhead, although the absolute overhead times are affected, as further discussed in Sec. III C.

For IBM's Eagle platform [57] with a heavy-hexagon coupling map, we use the Brisbane quantum processor. This processor has single-qubit gates `SX` and `X` with a fixed duration of 60 ns, and two-qubit gate `ECR` with a duration of 660 ns [92].

2. Benchmarking circuits

We use two different sets of circuits to estimate the overhead introduced by time-multiplexing qubit control: (a) random circuits and (b) examples of quantum algorithms provided in the standard benchmarking set MQT Bench [54].

We generate the random circuits in native gates of the target hardware with single-qubit and two-qubit gate weights 0.7 and 0.3, respectively. At each generation step, a decision between a single- and two-qubit gate is made according to the weights; then, a corresponding gate is drawn from the native gate set and added to the back of the circuit. By this procedure, we generate sparse random circuits with a single-qubit gate density ρ_1 of up to 0.35 for the 25-qubit square grid and up to 0.25 for the larger architectures. We then optimize the random circuits by first using qiskit's default transpiler pass managers on optimization level 2 for the 25-qubit device and on level 1 for the larger devices to receive the target quantum circuit.

For the square-grid architecture, we retrieved the MQT Bench circuits in a hardware-independent form and translated them; for the IBM Eagle architecture, we retrieved the circuits directly in native gates. Afterward, we routed each native-gate circuit according to the hardware coupling map before serialization. We performed the steps separately to measure the overhead arising from both routing and serialization. Results are reported as the median

over multiple transpiler seeds, in order to average out variations due to heuristic qubit routing.

Routing is a well-known source of overhead, and we compare the runtime overheads arising from routing and serialization. Because routing is a necessary step for physically executing a circuit, we analyze both the absolute runtime overhead of serialization and its value relative to the runtime of the routed circuit.

III. RESULTS FOR IMPACT OF TIME MULTIPLEXING ON SINGLE-QUBIT GATES

In this section, we present our results on time multiplexing for single-qubit gates, using the benchmarks described in Sec. II D and the compilation procedure shown in Sec. II C. As discussed in Sec. II D 1, we ran benchmarks on both grid and heavy-hexagon qubit layouts. We focus on the results for grid layouts here and defer the results for the heavy-hexagon layout to Appendix D, since the two layouts yield quite similar behavior for the quantities we are interested in. Some additional results for grid layouts are shown in Appendix C.

We begin with results for random circuits in Sec. III A and then continue with results for quantum algorithms from MQT Bench in Sec. III B. Finally, in Sec. III C, we discuss our observations on how the time-multiplexing overhead scales as a function of different parameters.

A. Random circuits

Figure 2 displays the overhead due to time multiplexing of single-qubit control for random circuits, as a function of the number of gates in the circuit. The panels in the left column of the figure [Figs. 2(a), 2(c), and 2(e)] show results for a 5×5 square grid, while the panels in the right column [Figs. 2(b), 2(d), and 2(f)] show results for an 11×11 square grid. We provide similar results for a heavy-hexagon layout with 127 qubits in Fig. 17 in Appendix D.

In Figs. 2(a) and 2(b), we plot the overhead, in milliseconds, due to serialization of single-qubit gates, for different numbers of qubits per switch. For all cases, we observe that this overhead increases linearly with the number of gates in the circuits, with a steeper slope for more qubits per switch. These behaviors are expected since the number of gates requiring serialization is proportional to the total number of gates, and switches controlling more qubits require more serialization. We return in Sec. III C to the question of how exactly the overhead scales with the number of qubits per switch.

In Figs. 2(c) and 2(d), we show the ratio between the duration of the serialized circuits and the duration of the routed circuits, as a function of the number of gates in the circuit. Again, we find a pattern that holds in all cases: with the exception of some minor variations for a small number of gates in the circuit, this ratio remains constant when the number of gates in the circuit grows. This pattern is consistent with the linear growth of the serialization overhead observed in the top row of Fig. 2, since we also expect the depth of the routed circuit to grow approximately linearly with the number of gates in the original circuit (for a fixed gate density). The value of the overhead ratio scales with the size of the processor, with more overhead for the larger processor. For example, reducing the number of lines by half gives an overhead factor of 1.3 for the 5×5 grid and 1.6 for the 11×11.

In Figs. 2(e) and 2(f), we display examples of the full breakdown of the contributions to the total circuit duration from routing and serialization, for selected numbers of gates and qubits per switch. From these plots, we see that the routing results in a substantial overhead in circuit duration compared to the translated circuit, which helps put the overhead from serialization into perspective. For both the 5×5 and 11×11 square grids, having two qubits per switch results in less serialization overhead than routing overhead. The number of qubits per switch that result in serialization overhead equal to the routing overhead differs for the two grid sizes: for the 5×5 grid, we see from Fig. 2(c) that it is around nine; for the 11×11 grid, we see from Fig. 2(d) that it is around three. We attribute this difference to the scaling for routing overhead with grid size.

To further understand how the overhead behaves as a function of qubit number for larger NISQ devices, we show in Fig. 3(a) the overhead components from routing and serialization for square-grid architectures of sizes between 5×5 and 33×33 qubits. Here, we used random circuits of fixed depth (250), fixed single-qubit gate density ($\rho_1 = 0.20$), and fixed two-qubit density ($\rho_2 = 0.26$). These parameters were chosen such that they match the depth and gate densities of the 5×5 grid in Fig. 2 for 2500 random gates and the 11×11 grid for 10 000 gates. The average duration of a fixed-depth circuit is independent of the number of qubits before routing and serialization since the gate densities are fixed; the simulated duration of the target circuits is seen as a horizontal line in Fig. 3(a).

In Fig. 3(a), we fitted the overhead of routing and serialization for three different k using power-law functions. The fitted exponents, shown in the legend of Fig. 3(a), indicate that the serialization overhead only grows very slightly faster (as $\sim n^{1.4}$–$n^{1.5}$, with n being the number of qubits) than the routing overhead ($\sim n^{1.3}$) for $k \leq 8$ in random circuits with the chosen depth and gate densities. Naturally, the routing of random circuits containing randomly chosen pairs of qubits for two-qubit gates becomes more expensive for larger devices, since these qubit pairs may be separated by larger distances. Similarly, keeping the depth and gate densities constant, a random circuit spanning a larger device will contain more single-qubit gates requiring

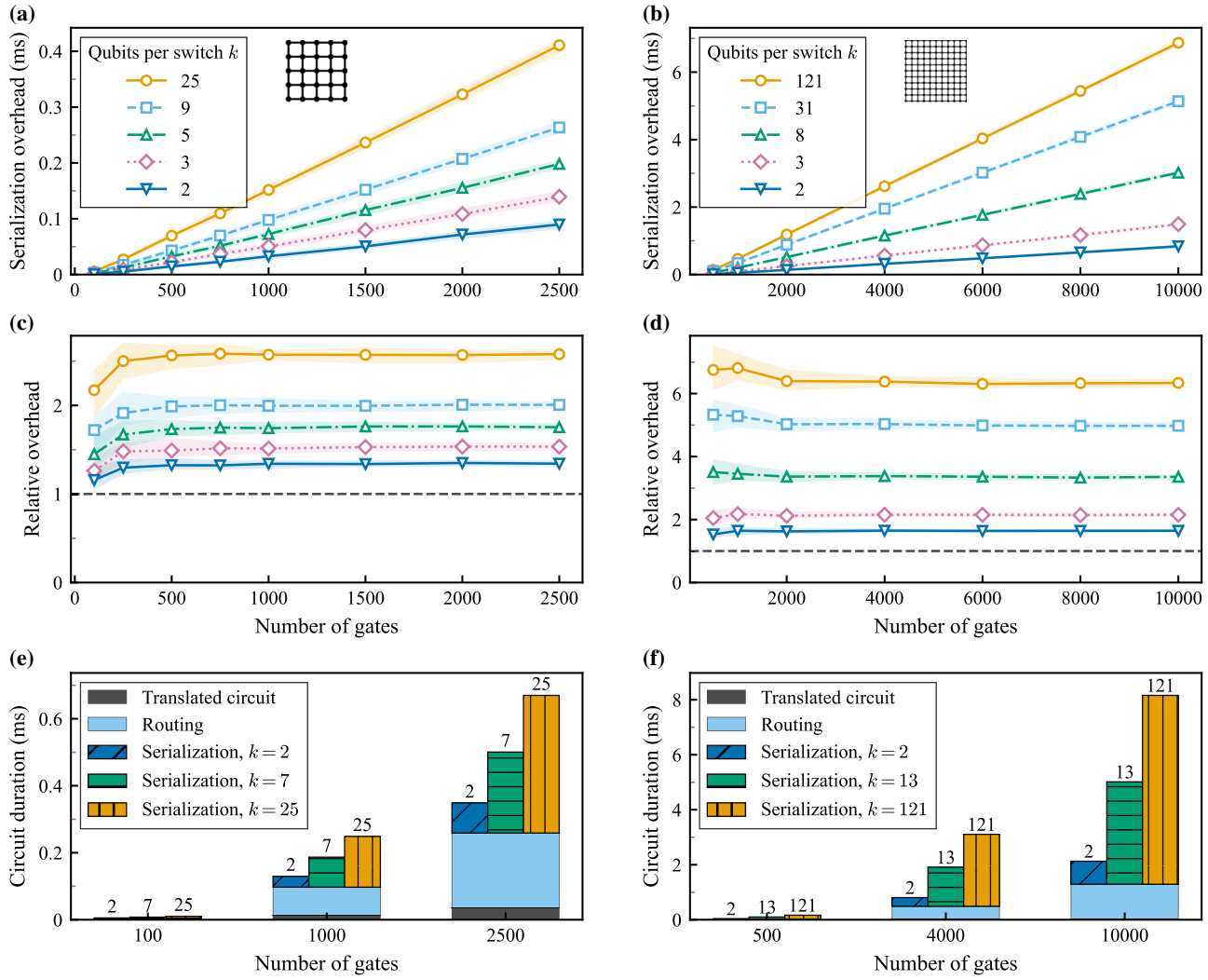

FIG. 2. Overhead from time-multiplexing single-qubit control for square-grid architectures (left column: 5×5 grid; right column: 11×11 grid). Each data point displayed is the median over 100 random circuit seeds; shaded areas show the interquartile range. (a) and (b) Serialization overhead as an increase in circuit duration compared to the routed circuit, as a function of the number of gates in random circuits. (c) and (d) The same serialization overhead as in (a) and (d), but now shown relative to the routed circuit duration, as a function of the number of gates in random circuits. (e) and (f) Breakdown of total circuit duration: translated circuit (dark gray), qubit routing overhead (light blue), and serialization overhead for k qubits per switch (blue, green, yellow), for random circuits with three different gate counts. The procedure for generating and compiling the circuits is detailed in Sec. II.

serialization. The relative overhead by serialization, plotted in Fig. 3(b), displays a slow sublinear growth in system size ($\sim n^{0.1}$–$n^{0.2}$; the difference between the scalings for the serialization and routing overheads). For 1000 qubits, the total circuit duration is under twice the routed duration for $k = 2$ and about three times the routed duration for $k = 4$.

B. Quantum algorithms

In Fig. 4, we show results for time-multiplexing overhead for quantum algorithms in the MQT Bench set, on an 11×11 square grid. We provide similar results for a 5×5 square grid in Fig. 14 in Appendix C and for a heavy-hexagon layout with 127 qubits in Fig. 18 in Appendix D.

In Fig. 4(a), we plot the total circuit duration when using different numbers of qubits per switch (colored markers) and compare it to the duration of the routed circuit (black crosses), for the different algorithms in the MQT Bench set (ordered according to the number of gates in the original circuits, with the fewest gates on the left). The total circuit duration and absolute multiplexing overhead can be compared to qubit coherence times to give insights into the effect of multiplexing on algorithmic fidelity [47]. We observe from the plot that the serialization overhead differs a lot from algorithm to algorithm. For a zoomed-in view of the cases with 2 and 4 qubits per switch, see

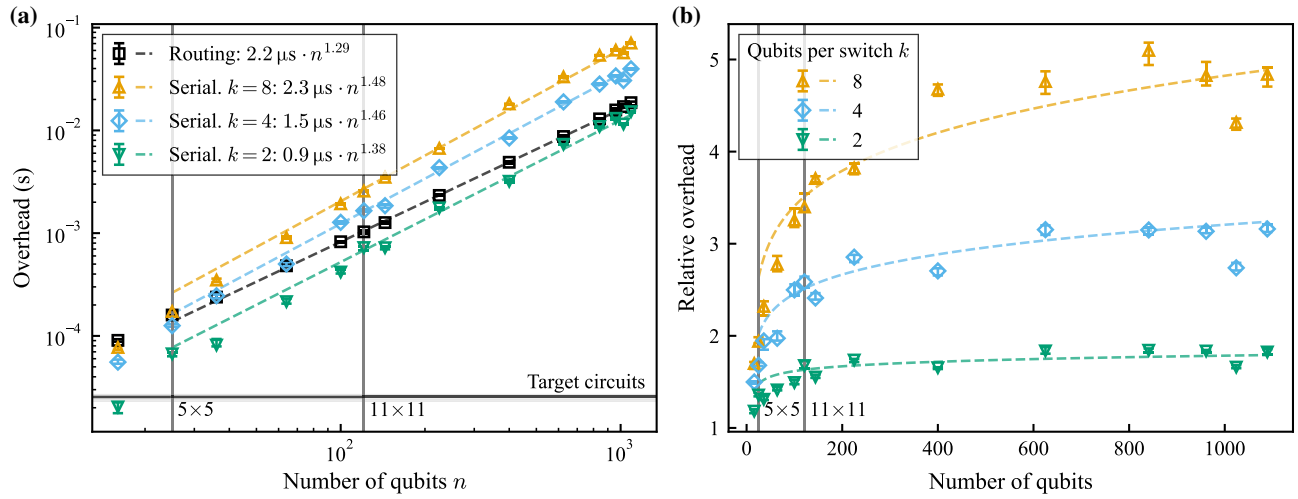

FIG. 3. Serialization and routing overhead for random circuits on square-grid architectures as a function of the number of qubits. Target circuits are randomly generated with fixed depth and gate densities (details in the main text). Vertical gray lines indicate the 5×5 and 11×11 grids from Fig. 2. Each data point is the median over 20 circuit instances; error bars show the interquartile range. (a) Absolute overhead in seconds from routing (black) and serialization for different numbers of qubits per switch k (colored). The horizontal gray line shows the median duration of the unrouted target circuits. The overheads follow power-law scaling with fitted exponents shown in the legend. (b) Relative overhead, defined as the ratio of serialized circuit duration to routed circuit duration. Dashed lines show the estimated scalings from the power-law fits in (a).

Fig. 15(a) in Appendix C; the corresponding plots for a 5×5 square grid and for a heavy-hexagon layout with 127 qubits are given in Fig. 15(b) in Appendix C and in Fig. 19 in Appendix D, respectively.

To better understand how different algorithms give rise to different serialization overhead, we plot the data from Fig. 4(a) in terms of the relative overhead factor, defined as the ratio of serialized (and routed) circuit duration to routed circuit duration in Fig. 4(b), and display, for each quantum algorithm, the single- and two-qubit gate densities ρ_1 and ρ_2 along with the total gate density ρ_{total} [see Eq. (1)]. Here, a pattern emerges: higher gate densities generally result in higher serialization overhead. This is expected since sparse circuits (with low gate density) have little need for serialization.

In Fig. 5, we provide a different perspective on the data in Fig. 4(b). Here, we display the distribution of serialization overhead relative to the routed circuit duration, for six different numbers of qubits per switch, ranging from 2 to 121. As the number of qubits per switch increases, large serialization overheads become increasingly common. We provide the corresponding results for a 5×5 square grid in Fig. 16 in Appendix C and for a heavy-hexagon layout with 127 qubits in Fig. 20 in Appendix D.

We also examine the routing and serialization overheads for a prominent algorithm, the quantum Fourier transform (QFT), for larger numbers of qubits, to study how the overhead scales with qubit count and compare it to the behavior observed for random circuits in Fig. 3. In contrast to the random circuits with fixed depth and fixed gate densities, the depth of QFT circuits grows linearly with the number of qubits for the translated (but unrouted) circuit, as shown by the orange line in Fig. 6(a). Due to the circuit structure, the gate densities decrease for larger device sizes (not plotted). This results in a different overhead scaling than for random circuits.

For the QFT, the cost for routing and serialization is predominantly linear up to around 100 qubits. Then, for larger devices, starting from the 121-qubit square-grid to dismiss finite-size effects for small circuits, we fit a power law analogously to the random circuits in Fig. 6(a). We observe exponents very close to one for both routing and serialization overheads. As a consequence, we see in Fig. 6(b) that the relative overhead by serialization compared to the routed circuit duration is almost constant for $k = 2$ and $k = 4$ for larger n, and growing very slowly for $k = 8$. This illustrates that the QFT exhibits more favorable scaling of serialization overhead with the number of qubits compared to random circuits and that this scaling is dependent on the selected circuit type. We also emphasize that these are worst-case scalings since our scheduling described in Appendix A does not claim to be optimal.

C. Scaling behavior

We further explore the functional dependence of the serialization overhead on the dominant parameters. As could be expected, we observe that the overhead increases linearly with the number of single-qubit gates N_1 in the routed circuit and the single-qubit gate time t_{1q}. We saw

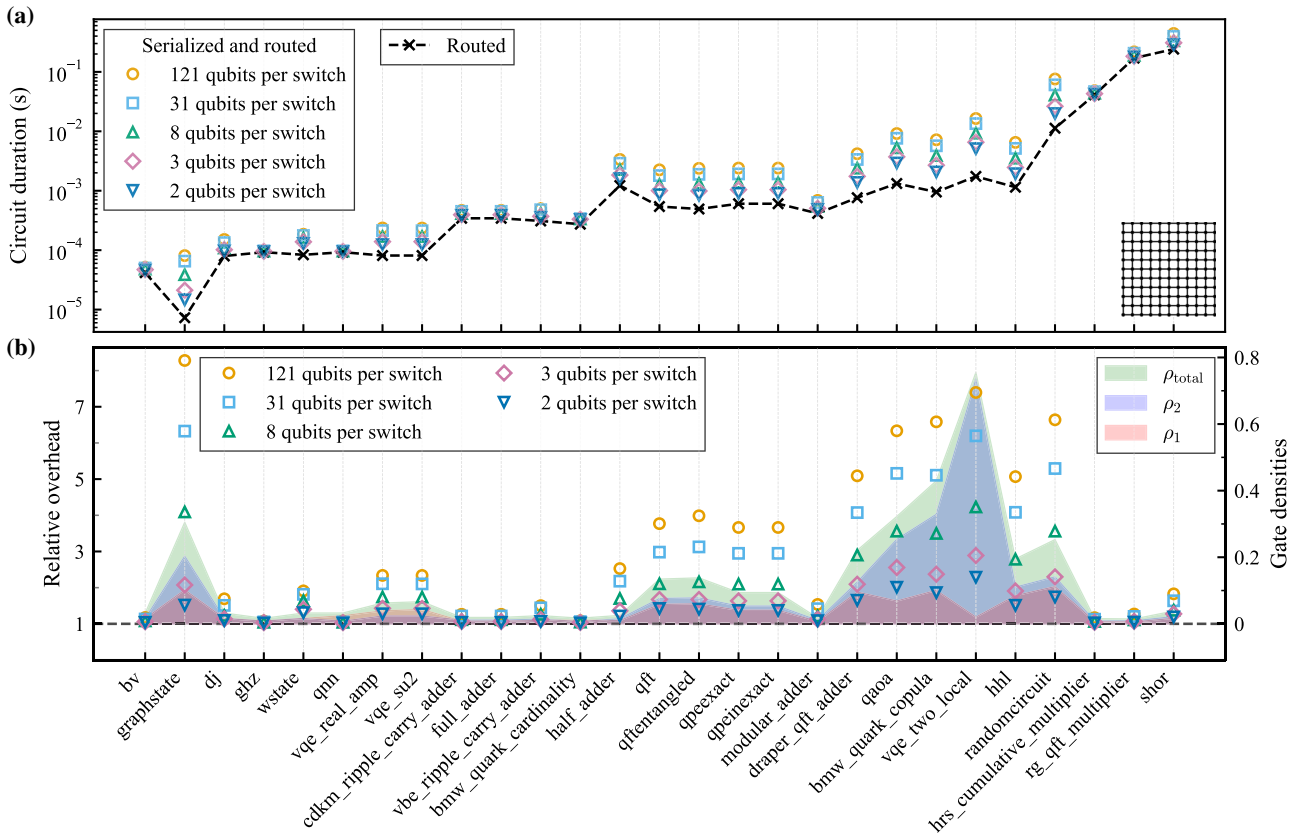

FIG. 4. Impact of time multiplexing on execution of quantum algorithms from the MQT Bench set for an 11×11 grid. (a) Total circuit duration for serialized routed circuits (colored markers) and routed circuits without serialization (black crosses and dashed line) across different quantum algorithms, ordered by increasing gate count. (b) Relative overhead of serialization compared to the routed circuit duration (markers, left axis), with shaded areas indicating gate densities (ρ_1 pink, ρ_2 blue, ρ_{total} green; right axis) and different marker colors indicating different numbers of qubits per switch. The data points are medians over 20 transpiler seeds. The full names of the quantum algorithms are listed in Table II in Appendix F.

this behavior in Figs. 2(a) and 2(b) for 5×5 and 11×11 grids, respectively. We further show that this linear dependence holds for the 11×11 grid with random circuits in Fig. 7 and with circuits from MQT Bench in Fig. 8, where data points are compared to the predicted linear function (solid lines) for different numbers of qubits per switch.

Naively, one might also expect a linear scaling of the serialization overhead in the number of qubits per switch k, since each of the k qubits must be addressed sequentially (illustrated in Fig. 22 in Appendix F). However, this turns out to not hold in general. For all benchmark circuits, we observe a logarithmic scaling in k, as shown by the solid lines in Figs. 7 and 8. For the MQT Bench circuits in Fig. 8, the solid lines are fits to the function $pN_1t_{1q}\log(k)$ with fitting parameter p. The rest of the benchmarks in the MQT Bench set follow the same scaling, as we show in detail in Appendix F.

The reduction from linear to logarithmic scaling in k arises from the interplay between single- and two-qubit gates. Two-qubit gates impose a structure on the circuit, introducing idle time slots where some qubits are naturally inactive. In contrast, a circuit composed solely of single-qubit gates exhibits no dependencies, allowing all gates to be executed as soon as possible, making the circuit automatically dense. In this case, the overhead indeed scales linearly $\sim N_1t_{1q}k$, as confirmed by benchmarks including only single-qubit gates. Besides this structural effect related to the gate density, two-qubit gates introduce another important mechanism that was mentioned in the discussion of efficient serialization in Sec. II C: during a two-qubit gate, sequences of single-qubit gates on other qubits can be performed without increasing the overall circuit duration. Together, these two effects can account for the observed logarithmic scaling in k of the serialization overhead. A simple numerical toy model reproduces this behavior, as we show in Appendix F, although not all parameter regimes fit the analytical reference equally well. Moreover, the dependence on two-qubit gate duration does not appear to be universal across all benchmarks, and it seems likely that the overall circuit density should be included in a complete model.

While these observations explain a sublinear scaling in k, the logarithmic form is not immediately obvious. It can be rationalized using arguments from queueing theory,

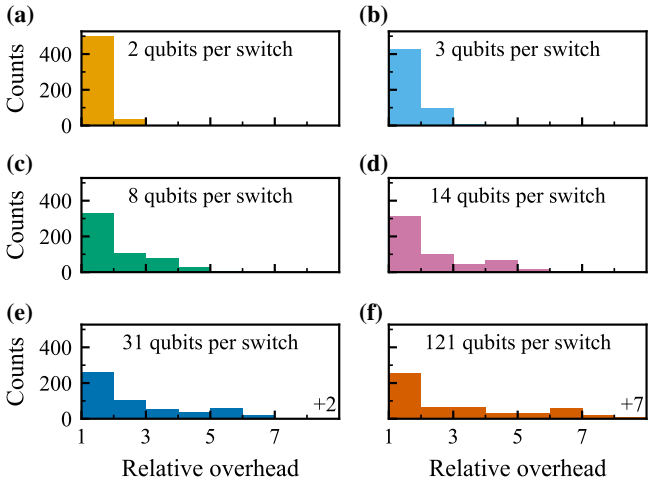

FIG. 5. Distribution of time-multiplexing overhead across switch sizes on an 11×11 grid, for the quantum algorithms from MQT Bench shown in Fig. 4, using 20 different transpilation seeds. The histograms show the relative overhead (the ratio of serialized to routed circuit duration) for different numbers of qubits per switch. The $+n$ notation to the right in the bottom histograms indicates n additional counts beyond the histogram limits.

where extreme-value methods are applied to analyze maximum waiting times. In queueing models, the waiting time or queue length is treated as a random variable, and in many standard settings, the asymptotic behavior of the queue-length distribution has an exponentially decaying tail [93]. These methods are also used to analyze multiplexing and other types of signal transmission in classical communication networks [94,95]. Practically, these properties of queues mean that the probability of encountering a very long queue decreases exponentially with its length, making long delays rare. In Appendix F, we present a heuristic derivation of how this leads to logarithmic scaling. Alternatively, in queueing-theory terms, each switch in our time-multiplexing setup can be viewed as a server that sequentially processes single-qubit gates (jobs) from multiple qubits (clients). The serialization of gates on a given switch generates waiting times, and the total circuit duration is determined by the maximum waiting time across all switches. It can be calculated that the expectation value of the maximum waiting time is logarithmic in the number of clients (qubits) [96].

D. Effect of gate and switch durations

For the simulations described above, we selected a ratio between single- and two-qubit gate durations $t_{2q} = 10\, t_{1q}$ (see Sec. II D 1). In Fig. 9, we present the relative overhead for different t_{2q}/t_{1q} ratios for Shor's algorithm, Quantum approximate optimization algorithm (QAOA), and graph-state preparation. The plots show that larger t_{2q}/t_{1q} ratios result in lower overhead, likely because more single-qubit gates can be executed serially during the longer two-qubit

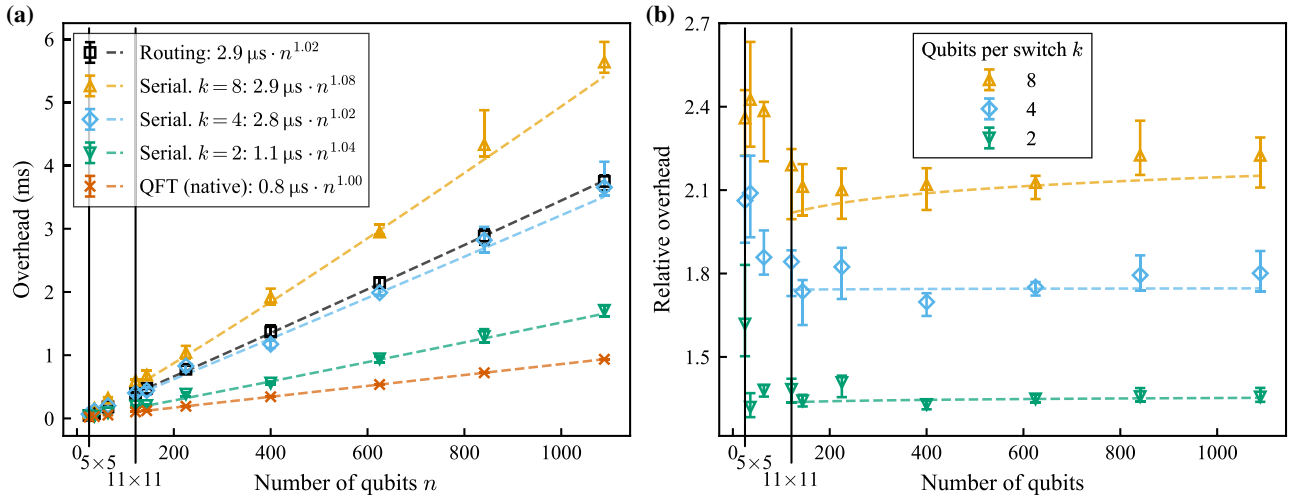

FIG. 6. Scaling of serialization and routing overhead for the quantum Fourier transform on square-grid architectures with different numbers of qubits. Vertical gray lines indicate the 11×11 and 5×5 grids from Figs. 4 and 14. Each data point is the median over 20 different transpiler seeds resulting in different routed crcuits; shaded regions show the interquartile range. (a) Absolute overhead from routing (black) and serialization for different numbers of qubits per switch k (colored), along with the native circuit duration (orange), which shows the increase in execution time for the target quantum Fourier transform circuit translated to the native gate set, before routing. The overheads follow power-law scaling with fitted exponents shown in the legend. We dismiss data dominated by finite-size effects from devices smaller than 11×11 for the fit to capture asymptotic scaling. (b) Relative overhead, defined as the ratio of serialized circuit duration to routed circuit duration. For $k = 2$ and $k = 4$, the predominantly linear growth of the overheads results in an almost constant relative overhead by serialization.

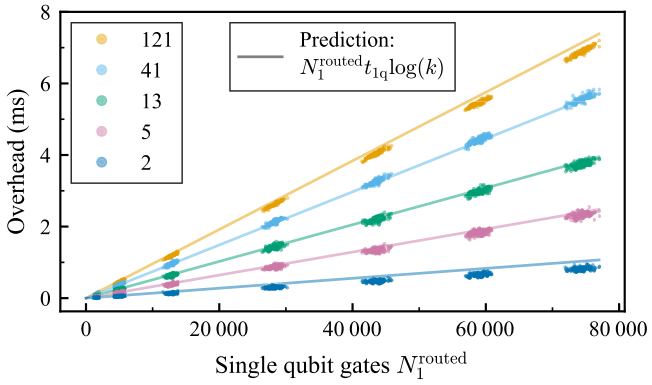

FIG. 7. Scaling of serialization overhead for random circuits on an 11 × 11 square grid. For each gate count, 100 random circuits were generated. The number of single-qubit gates N_1^{routed} is retrieved after routing, with slight variations due to optimizations and routing. The serialization overhead scales linearly with N_1^{routed} and the single-qubit gate duration t_{1q}, but logarithmically with qubits per switch k. Predictions (solid lines) align well with the data without additional scaling factors.

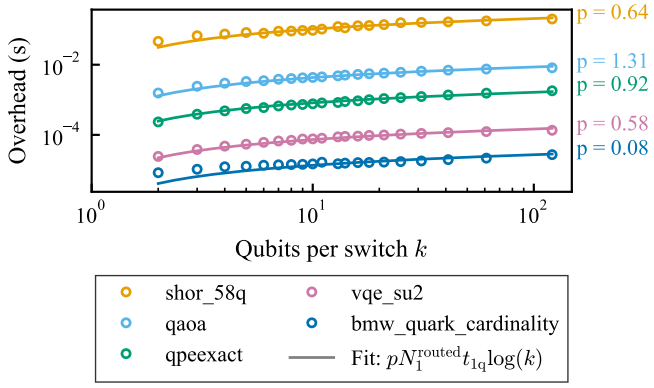

FIG. 8. Scaling of serialization overhead for selected quantum algorithms from the MQT Bench set on an 11 × 11 grid. The overhead is fitted using an additional (compared to Fig. 7) scaling factor p (values shown next to each fit line) and scales logarithmically with qubits per switch k. The complete data for all MQT Bench algorithms is shown in Fig. 24 in Appendix F.

gates, effectively hiding the serialization cost. The functional scaling with qubits per switch k does not appear to be affected by different ratios; it remains approximately logarithmic, but with different multiplicative factors.

All circuits considered so far have used a switch delay time $t_{sw} = 2$ ns, corresponding to the time it takes for a switching event to occur. This is fast but realistic for state-of-the-art devices [34]. The impact of longer switch times of up to 100 ns, modeled by the `SDel` gate in the serialized circuit, is shown in Fig. 10, again for Shor's algorithm, QAOA, and graph-state preparation on the 11 × 11 square grid. From Fig. 10, we see that the overhead depends linearly on the switch time, with a steeper increase when there are more qubits per switch. The scaling of the overhead with the number of qubits per switch remains approximately logarithmic.

IV. SUMMARY AND CONCLUSION

We have quantified an important trade-off that occurs when scaling up quantum computers: the impact on quantum circuit duration (and hence performance) from reducing the number of control lines. For quantum computers to outperform modern supercomputers, a large number of qubits will be required, even for problems where quantum advantage is theoretically achievable. The strength of classical computing is in many ways due to the invention of the integrated circuit, which enabled the integration of

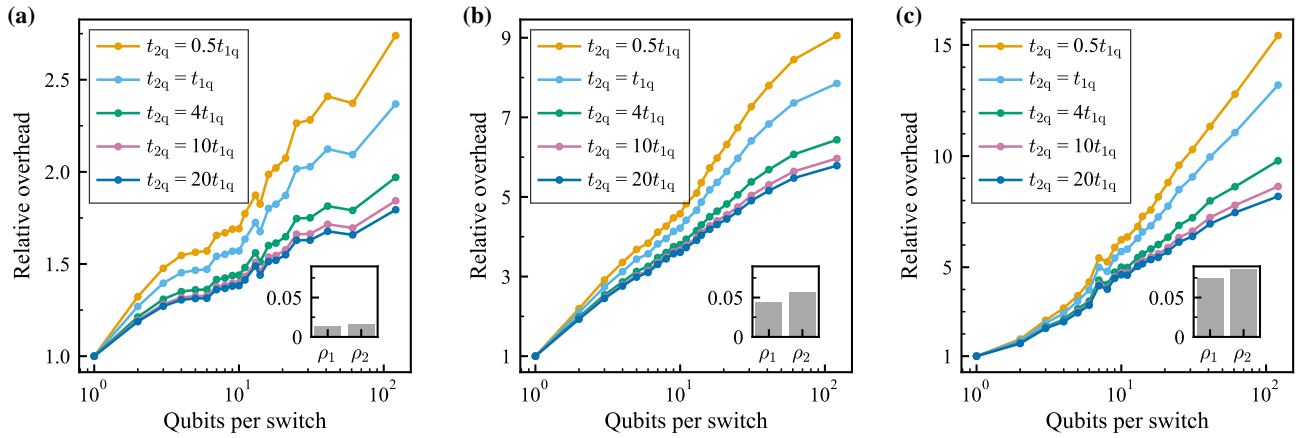

FIG. 9. Relative overhead by serialization compared to the routed circuit duration, as a function of qubits per switch, for different ratios of two-qubit to single-qubit gate durations, t_{2q} and t_{1q}, respectively, for selected algorithms on an 11 × 11 grid. (a) Shor's algorithm. (b) Quantum approximate optimization algorithm. (c) Graph-state preparation. For increasing t_{2q}, the overhead by serialization decreases as serialized single-qubit gates are more likely to be concurrent with a two-qubit gate. The insets show the single- and two-qubit gate densities of the circuit. Note the logarithmic x axis. Each marker represents the median over 30 seeds.

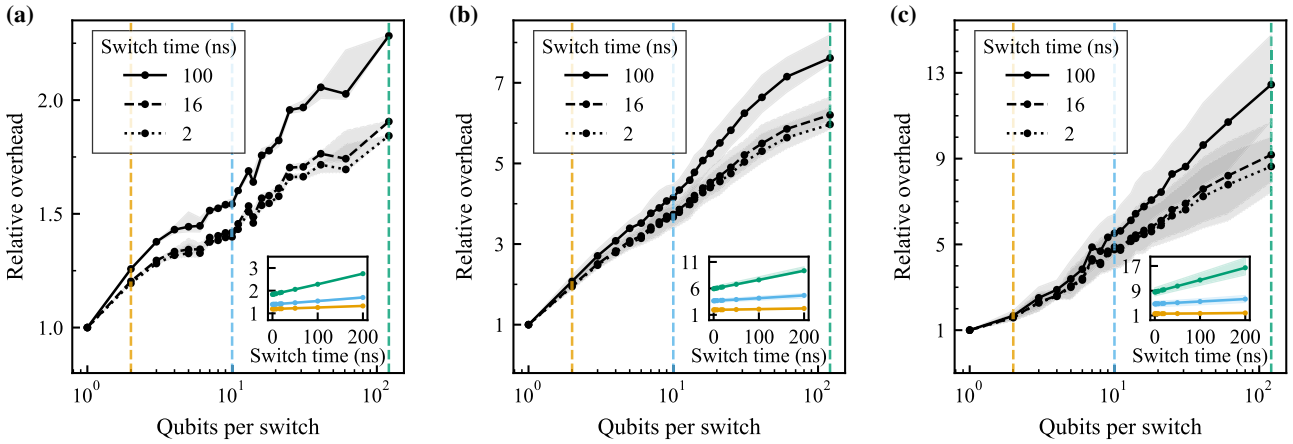

FIG. 10. Relative overhead by serialization compared to the routed circuit duration, as a function of qubits per switch, for different switch delay times t_{sw}, for selected algorithms on an 11×11 grid with $t_{2q} = 10\, t_{1q}$. (a) Shor's algorithm. (b) Quantum approximate optimization algorithm. (c) Graph-state preparation. Note the logarithmic x axis. Each marker represents the median over 30 seeds and shaded areas correspond to the inter-quartile range. The insets show serialization overhead as a function of t_{sw} for 2, 10, and 121 qubits per switch, color corresponding to the vertical dashed lines in the main plot. These plots confirm a linear dependence of the relative overhead on the switch delay time.

millions of transistors on a small chip surface while significantly reducing the number of external connections. Similarly, scaling up quantum computers requires more efficient control schemes than the current approach, where the number of control terminals at room temperature scale linearly with the number of qubits. To address this challenge, on-chip signal routing or multiplexing can be used to reduce wiring complexity and enable practical scaling of both superconducting- and semiconductor-based quantum computing platforms [97,98].

The trade-off we quantified occurs since a scalable architecture with shared control lines limits the possibility of gate parallelization. This leads to increased circuit runtimes, increasing exposure to decoherence, and therefore the impact on algorithm execution needs to be assessed. To this end, we quantified the serialization overhead for a range of standard benchmark quantum algorithms and random quantum circuits on three NISQ devices: a 5×5-qubit square grid, an 11×11-qubit square grid, and IBM's 127-qubit Eagle heavy-hexagon layout. In some cases, we also explored larger qubit numbers, up to 33×33-qubit square grids. We assumed a hierarchy of time scales common to such devices: two-qubit gates take longer to execute than single-qubit gates, which in turn are slower than the switching delay in multiplexed control. We verified that our scaling results were robust to variations of the specific times when they respected this hierarchy.

For two-qubit gates, strategic placement of controlled couplers mediating these gates on switches enables serialization of operations without incurring any duration overhead, eliminating the trade-off, up to a limit of couplers per switch set by the connectivity between qubits. For a square-grid layout, the number of control lines can be reduced by a factor of four without affecting the circuit, since each qubit is connected to four other qubits, but only can be involved in a two-qubit gate with one of them at a time. Although more couplers per switch will lead to serialization overhead, affected also by the density of two-qubit gates in the quantum circuit to be executed, we deemed the "free" reduction sufficient to warrant more focus on single-qubit gates. Thus, we only studied overhead from serialization of single-qubit gates in the rest of the article.

For the serialization overhead for single-qubit gates, we found a surprisingly benign, logarithmic scaling with the number k of qubits per switch. Only when the quantum circuits are close to fully saturated with just single-qubit gates does the scaling become linear with k. We understand the logarithmic scaling with k from queueing theory: the duration overhead corresponds to the maximum waiting time for all gates to be executed and that waiting time scales logarithmically. Furthermore, we found that the scaling of the serialization overhead with qubit number n also is benign. The exact scaling will depend on the circuit, but in the examples we tested, the relative serialization overhead either stayed constant or only grew weakly sublinearly in n.

Together, these findings for single- and two-qubit gates constitute the key results of our work: the number of control lines in a quantum computer can be significantly reduced without introducing much overhead in execution time for quantum algorithms. In other words, the trade-off we studied turns out to be manageable, which opens up for scaling quantum computers to large sizes. Additionally, multiplexing can be combined with moving signal generation into the cryostat, where further reduced

wiring would make integration even more practical [16,99,100].

We can make several further observations for specific quantum algorithms, which can be used as input for co-design of quantum processors and algorithms. Among the most well-known benchmark algorithms, GHZ state generation and quantum neural networks showed the lowest serialization overhead for single-qubit gates. For these algorithms, the overhead is small when using two qubits per switch, which halves the number of drive lines and provides a significant hardware reduction. Specifically, the circuit duration is extended only between 2 and 3 μs on the 121-qubit grid and 127-qubit heavy-hexagon architectures, while it is completely negligible for the 25-qubit grid.

Graph-state preparation adds 5 μs for the 121-qubit grid and nearly an order of magnitude more, 60 μs, for the Eagle architecture, while remaining minor on the 25-qubit grid (165 ns). These numbers can be compared with state-of-the-art coherence times for superconducting qubits, which are in the range of hundreds of microseconds.

Several other prominent algorithms exhibit overheads that could be problematic with short coherence times. For the larger devices, QFT adds between 0.25 and 1.2 ms, and the additional duration for serialized QAOA exceeds 1 ms on the 121-qubit grid and approaches 9 ms on Eagle.

In general, the 127-qubit Eagle generally shows larger overhead than the 121-qubit grid, and both are higher than for the small grid. This is likely because running algorithms on more qubits requires more gates, and the overhead scales linearly with the number of gates. Eagle's overhead exceeds the 121-qubit grid on all benchmarks except W-state preparation, where both are of the order of tens of microseconds. Since Eagle only has six more qubits than the large grid, it remains unclear whether the difference is due primarily to the layout difference or the qubit count, but the lower connectivity leads to larger routing overhead for the Eagle layout.

V. OUTLOOK

There are several directions for future work that could be explored. One direction is to use queueing-theory methods and other analytical tools to more accurately model and optimize serialization of both single- and two-qubit gates. Further numerical studies could also help further elucidate the scaling of circuit duration overhead as a function of the number of control lines, the layout of the quantum processor, the gate densities in the quantum circuit, the switching time, the single-qubit gate time, and the two-qubit gate time. All of these factors can vary between hardware implementations, so a more detailed quantification of their impact would be valuable for choosing quantum processor architectures to scale up.

When scaling up with the goal of building a fault-tolerant quantum computer, the impact of time multiplexing on quantum error correction should be studied further using the framework we have introduced here. For a fixed number of control lines, there would be a trade-off between how large code distance one could achieve by using many qubits and how much the serialization overhead would degrade the performance of such a longer-distance code; see Appendix E. The analysis of such a trade-off would depend heavily on details of the hardware, e.g., gate times for different types of gates, the error budget for these gates, and other error sources. For particular hardware layouts and error-correction codes, there can be highly optimized arrangements of operations in time and space [101].

Another natural direction for future work is to integrate serialization directly into routing and compilation algorithms, enabling global optimization of both qubit placement and control-line allocation. Here, factors such as coherence times varying across a quantum processor could be taken into account. The setup we studied could also be combined with frequency multiplexing to further reduce the number of lines [102].

ACKNOWLEDGMENTS

We thank David Fitzek for taking part in the initial project discussions. We acknowledge support from the Knut and Alice Wallenberg Foundation through the Wallenberg Centre for Quantum Technology (WACQT). M.R. and A.F.K. also acknowledge support from the Swedish Foundation for Strategic Research (Grant No. FUS21-0063). A.F.K. is also supported by the Swedish Foundation for Strategic Research (Grant No. FFL21-0279) and the Horizon Europe programme HORIZON-CL4-2022-QUANTUM-01-SGA via the project 101113946 OpenSuperQPlus100. I.S. acknowledges financial support from the European Union via Grant No. 101057977 SPECTRUM. S.G. acknowledges financial support from the European Research Council (Grant No. 101041744 ESQuAT). Numerical results were obtained using HPC resources at the Chalmers Centre for Computational Science and Engineering (C3SE). External interest disclosure: S.G. is a co-founder and equity holder in Sweden Quantum AB.

DATA AVAILABILITY

The data that support the findings of this article are openly available [103].

APPENDIX A: COMPILATION STRATEGIES FOR MINIMIZING SERIALIZATION OVERHEAD FOR SINGLE-QUBIT GATES

Here, we provide further details about the algorithms we considered and used for minimizing serialization overhead for single-qubit gates. We can reduce the overhead

from switch delays significantly by leveraging the temporal overlap between switching operations and concurrent two-qubit gates. Since two-qubit gates have durations equivalent to multiple single-qubit operations plus switching time in most situations we consider in this article (see Sec. II D 1), the transition of control between qubits can often be scheduled during these longer operations without extending the total circuit duration. Our serialization model always places a zero-duration two-qubit switch gate between the source qubit and the target qubit to enforce temporal dependencies. When an actual two-qubit gate already exists between these qubits, the switch gate does not introduce an additional constraint. More importantly, if a two-qubit gate involves one qubit in the switching pair but not the other, the temporal dependency created by `Sw` does not cause serialization since the coupler control is distinct from the qubit control, and the zero duration of the switch gate ensures that there is no increase in total execution time.

The switch delay gate `SDel` would normally increase the circuit duration by the switch response time. However, this delay can be hidden when switching occurs during concurrent two-qubit operations. We implement two optimization strategies to imitate this effect: (a) omit the delay gate when both qubits involved in the switch are simultaneously engaged in two-qubit gates and (b) omit the delay gate when the temporal position of a neighboring two-qubit gate exceeds the temporal position of the single-qubit predecessor plus the switching duration. These optimizations ensure that control transitions are scheduled within existing idle periods, minimizing the impact on overall circuit execution time and representing a realistic switch scheduling.

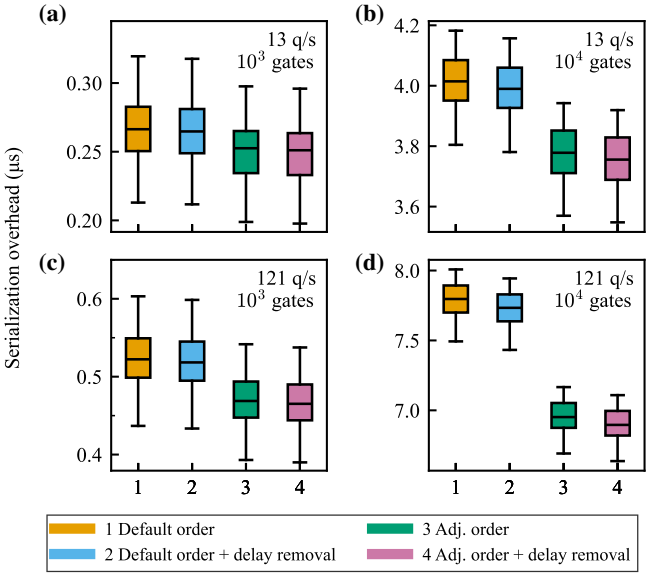

FIG. 11. Comparison of optimization strategies for serialization overhead on an 11 × 11 square grid with [(a) and (c)] 1000 and [(b) and (d)] 10 000 random gates, using [(a) and (b)] 13 and [(c) and (d)] 121 qubits per switch. Strategies 1 and 2 use default gate ordering, while 3 and 4 prioritize single-qubit gates based on layer distance to the next two-qubit gate. Strategies 2 and 4 employ delay-gate removal when switching occurs during two-qubit gate execution.

In Fig. 11, we show examples of the impact on serialization overhead of using these strategies for improving the compilation. We compare results from using the default gate order (orange) to either using optimized delay removal (blue), adjusting gate order (green), or doing both (purple). From the plots, we see that both optimizations reduce the serialization overhead, with the major contribution coming from adjusting the gate order.

The adjusted gate order via a topological sort of the nodes (gates) in the DAG dominates the overall runtime of the presented serialization algorithm. Using a priority-queue-based algorithm, this sort imposes a total runtime complexity of the serialization stage as $\mathcal{O}(N \log N)$ in the number of gates N, or, alternatively, $\mathcal{O}(D \cdot n \cdot (\log D + \log n))$ in the depth D and the number of qubits n, since $N = \mathcal{O}(D \cdot n)$. This quasilinear scaling adds a negligible classical runtime overhead to the total transpilation pipeline, where heuristic qubit routing with LightSABRE alone requires $\mathcal{O}(D \cdot n^{5/2})$ time [104,105].

APPENDIX B: STRATEGIES FOR GROUPING QUBITS

Here, we provide details on and comparisons between the four strategies we have explored for grouping qubits, i.e., to assign groups of k qubits to switches for control of single-qubit gates. In Fig. 12, we illustrate three of these strategies: trivial, clustered, and dispersed layouts. As discussed in Sec. II C, we are solely using the trivial layout, which is built by assigning qubits to groups following their index. Since qubit indices are typically assigned by row-major counting, such layouts are relatively clustered. The clustered layouts are created using a heuristic balanced clustering algorithm with swap refinements to improve the number of fully connected clusters and the connectedness of each cluster subgraph over the trivial layouts. The dispersed layouts are based on an algorithm that finds the $d_{\max}$-distance coloring using a d-coloring algorithm and using the maximal d for which such a layout can be found. Finally, the fourth grouping strategy is a random layout, i.e., distributing the qubits randomly among the switches.

In Fig. 13, we compare the performance of the described layouts through their influence on the serialization overhead, using random circuits and a few selected quantum algorithms from the MQT Bench set. For most algorithms, the overhead is similar for the different layouts. In general, the influence of the chosen layout decreases for increasing k until the same value is reached for $k = n$, where n

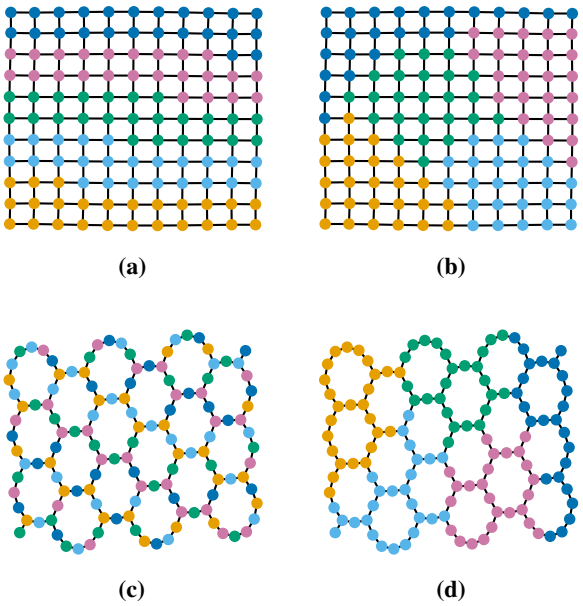

FIG. 12. Different strategies for grouping qubits on switches for single-qubit control, illustrated for 26 qubits per switch. (a) Trivial layout for an 11×11 grid. (b) Clustered layout for an 11×11 grid. (c) Dispersed layout for a 127-qubit heavy-hexagon architecture. (d) Clustered layout for a 127-qubit heavy-hexagon architecture.

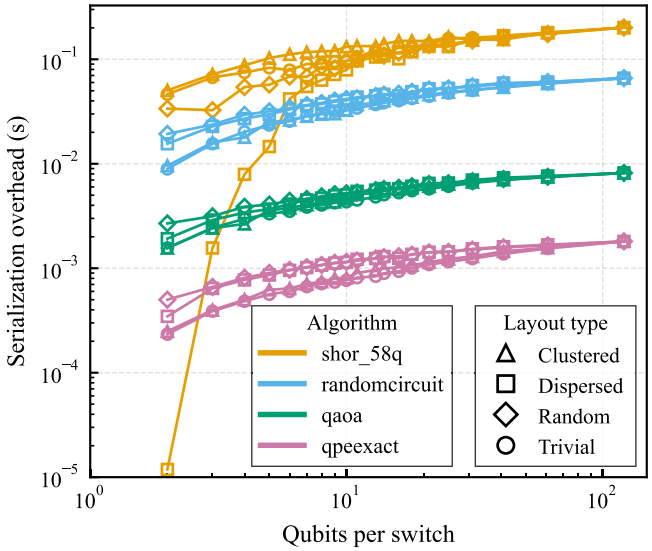

FIG. 13. Comparison of the performance of different qubit-to-switch layouts (different markers) on an 11×11 grid. The plot shows serialization overhead for four types of algorithms (different colors). With the notable exception of Shor's algorithm, the clustered and trivial layouts show similarly low overhead, below that of the random and dispersed layouts.

is the total number of qubits on the processor. Among the selected quantum algorithms, only the circuit that implements Shor's algorithm shows large deviations for small k. In this case, the dispersed layout adds only 10 µs serialization overhead for $k = 2$ to the circuit execution while other layouts add up to 30 ms, which is related to the structure of the circuits in this algorithm. For the other algorithms, including random circuits, we see that the trivial and clustered layouts tend to yield somewhat lower overhead. This observation is the reason we selected the trivial layout for all other investigations in this article.

APPENDIX C: ADDITIONAL RESULTS FOR SQUARE-GRID QUBIT LAYOUTS

In this appendix, we present additional results for the 5×5 and 11×11 square-grid qubit layouts, complementing the plots shown in Sec. III.

First, in Fig. 14, we show the MQT Bench results for the 5×5 grid, in the same style as for the 11×11 grid in Fig. 4. Figure 14(a) shows the total circuit duration for different numbers of qubits per switch for the quantum algorithms in MQT Bench. Figure 14(b) shows the relative increase in circuit duration due to serialization (the ratio of serialized to routed circuit duration). The results for the 5×5 grid display the same patterns as those for the 11×11 grid: serialization overhead varies widely between algorithms, but tends to be larger for algorithms with higher gate densities. However, we note that the absolute and relative serialization overhead is smaller for the 5×5 grid than for the 11×11 grid, even for the same number of qubits per switch. We attribute the difference in absolute serialization overhead to the fact that the benchmark circuits are longer for more qubits; for the difference in relative serialization overhead, we note that it is consistent with the pattern seen in the bottom row in Fig. 2, as discussed at the end of Sec. III A.

Next, we zoom in on the results in Fig. 4(a) and plot the duration of only the serialization overhead for the cases of 2 and 4 qubits per switch in Fig. 15(a). We provide the same type of plot for the 5×5 grid by zooming in on Fig. 14(a) to plot the serialization overhead for 2 and 4 qubits per switch in Fig. 15(b). These low numbers of qubits per switch are the most likely to be used in the first implementations of time-multiplexed qubit control, which makes extracting the duration of the serialization overhead particularly important here, allowing comparison to state-of-the-art qubit coherence time to judge the feasibility of using switches.

Finally, in the same style as Fig. 5, we present the data from Fig. 14(b) in another form in Fig. 16. Here, we display histograms showing the distribution of serialization overhead relative to the routed circuit duration, for six different numbers of qubits per switch, ranging from 2 to 25. These results for the 5×5 grid follow the same pattern as those for the 11×11 grid in Fig. 5: as the number of qubits per switch increases, large serialization overheads gradually become increasingly common.

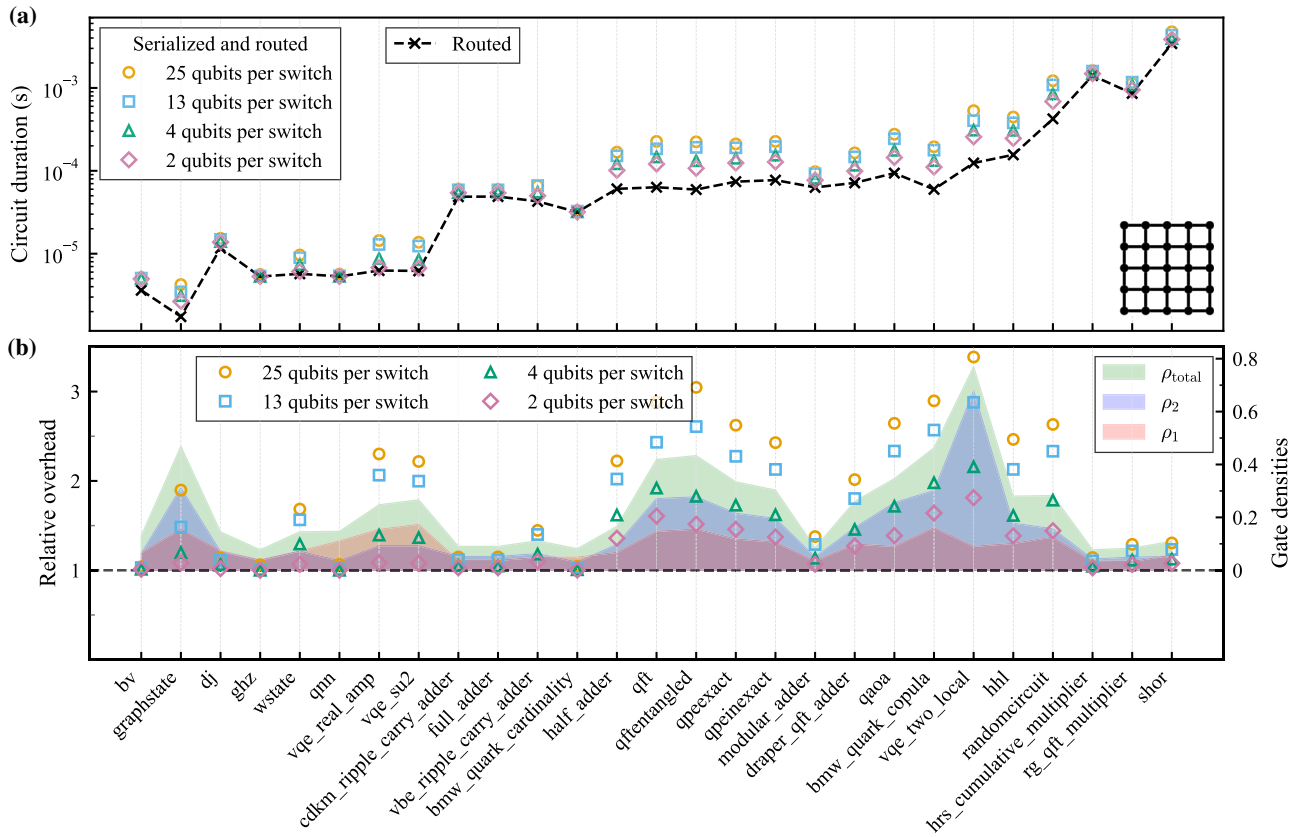

FIG. 14. Impact of time multiplexing on execution of quantum algorithms from the MQT Bench set for a 5×5 grid. (a) Total circuit duration for serialized routed circuits (colored markers) and routed circuits without serialization (black crosses and dashed line) across different quantum algorithms, ordered by increasing gate count. (b) Relative overhead of serialization compared to the routed circuit duration (markers, left axis), with shaded areas indicating gate densities (ρ_1 pink, ρ_2 blue, ρ_{total} green; right axis) and different marker colors indicating different numbers of qubits per switch. The data points are medians over 20 transpiler seeds. The full names of the quantum algorithms are listed in Table II in Appendix F.

APPENDIX D: RESULTS FOR A HEAVY-HEXAGON QUBIT LAYOUT

In this appendix, we present results for serialization overhead with 127 qubits in a heavy-hexagon layout, corresponding to those shown for 5×5 and 11×11 square-grid layouts in Sec. III and in Appendix C. Note that the number of qubits is almost the same for this heavy-hexagon layout as for the 11×11 square-grid layout.

In Fig. 17, we show the overhead due to time multiplexing for random circuits, in the way as we did for square grids in Fig. 2. We observe the same patterns for the heavy-hexagon layout in Fig. 17 as for the square grids in Fig. 2: in Fig. 17(a), we see that the serialization overhead increases linearly with the number of gates in the circuits; in Fig. 17(b), we see that the ratio between the duration of the serialized circuits and the duration of the routed circuit remains constant as a function of the number of gates in the circuits (after some minor variation at the start); in Fig. 17(c), we see that routing itself gives a substantial overhead compared to the uncompiled circuit, but that having more than 2 qubits per switch gives an even larger overhead. Compared to the results for the 11×11 square-grid layout in the right column of Fig. 2, the absolute serialization overhead is longer here for the 127-qubit heavy-hexagon layout, but the relative overhead compared to routing is quite similar. The reason for the increased routing overhead is the sparser connectivity of the heavy-hexagon layout compared to the square-grid layout.

Next, in Fig. 18, we show the MQT Bench results for the heavy-hexagon layout, in the same style as for the 11×11 square grid in Fig. 4 and for the 5×5 square grid in Fig. 14 in Appendix C. Figure 18(a) shows the total circuit duration for different numbers of qubits per switch for the quantum algorithms in MQT Bench, while Fig. 18(b) shows the serialization overhead for the same cases, relative to the duration of the routed circuit. The results for the heavy-hexagon layout display the same patterns as those for the 5×5 and 11×11 square grids: serialization overhead varies widely between algorithms, but tends to be larger for algorithms with higher gate densities. Compared to the 11×11 square grid in Fig. 4, we note that

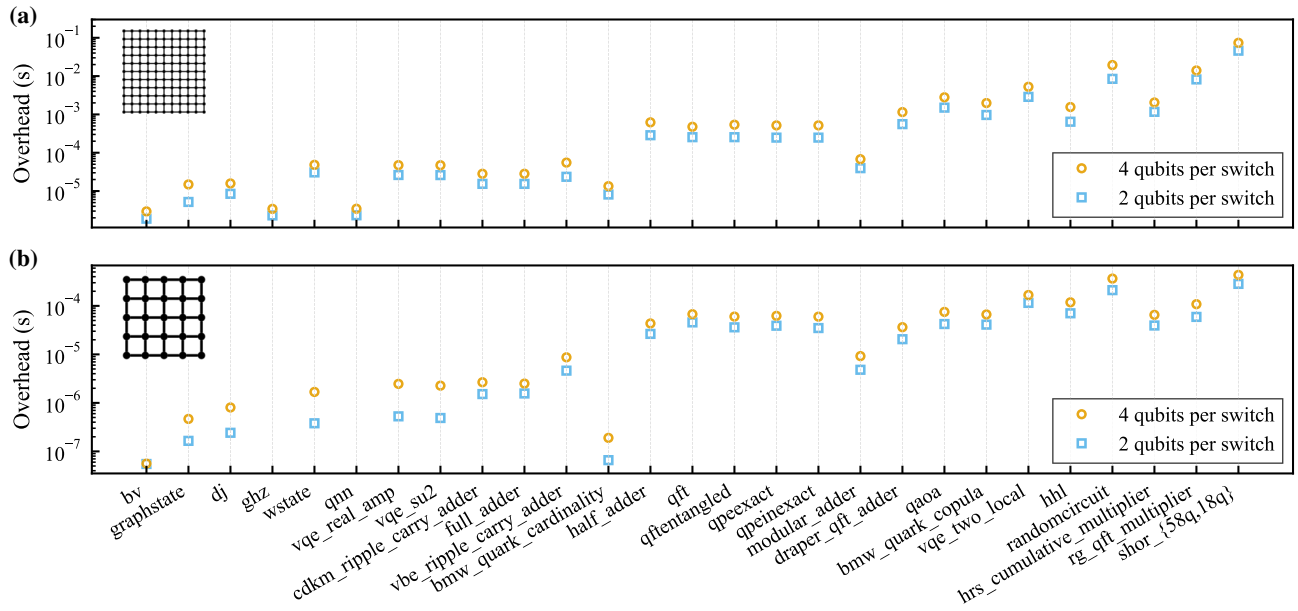

FIG. 15. Serialization overhead, in seconds, for the cases of 2 (blue squares) and 4 (orange circles) qubits per switch, for quantum algorithms from the MQT Bench set on an (a) 11×11 and (b) 5×5 square grid of qubits. Exact numerical values for $k = 2$ are shown in Table I. The full names of the quantum algorithms are listed in Table II in Appendix F.

TABLE I. Serialization overhead for different platforms and algorithms with $k = 2$ qubits/switch. All values in ms. The full names of the quantum algorithms are listed in Table II in Appendix F.

Algorithm	5×5	11×11	127 Hex
bmw_quark_cardinality	0.000	0.008	0.011
bmw_quark_copula	0.047	1.166	5.299
bv	0.000	0.002	0.018
cdkm_ripple_carry_adder	0.002	0.016	0.045
dj	0.000	0.008	0.024
draper_qft_adder	0.020	0.542	1.501
full_adder	0.002	0.016	0.045
ghz	0.000	0.002	0.002
graphstate	0.000	0.005	0.060
half_adder	0.028	0.286	1.726
hhl	0.072	0.767	2.173
hrs_cumulative_multiplier	0.038	1.169	6.996
modular_adder	0.023	0.443	1.531
qaoa	0.044	1.561	9.028
qft	0.041	0.227	0.908
qftentangled	0.050	0.259	1.199
qnn	0.000	0.002	0.003
qpeexact	0.039	0.240	1.042
qpeinexact	0.042	0.240	0.989
randomcircuit	0.225	8.916	59.625
rg_qft_multiplier	0.070	8.696	33.883
shor	0.275	46.323	168.742
vbe_ripple_carry_adder	0.005	0.023	0.086
vqe_real_amp	0.004	0.024	0.081
vqe_su2	0.004	0.024	0.090
vqe_two_local	0.107	2.827	15.829
wstate	0.002	0.029	0.014

the absolute serialization overhead is larger for the heavy-hexagon layout, while the relative serialization overhead is more or less the same across algorithms for both cases. Just as in the paragraph above, we attribute the difference in absolute serialization overhead to the increased routing overhead due to the sparser connectivity of the heavy-hexagon layout.

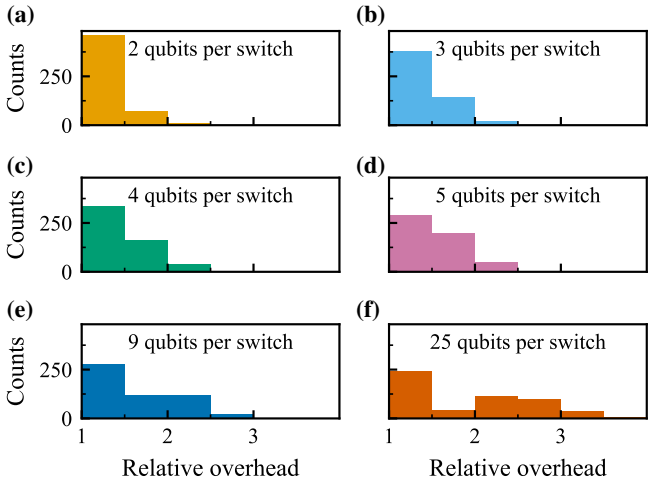

FIG. 16. Distribution of time-multiplexing overhead across switch sizes on a 5×5 grid, for the quantum algorithms from MQT Bench shown in Fig. 14. The histograms show the relative overhead (the ratio of serialized to routed circuit duration) for different numbers of qubits per switch. The $+n$ notation to the right in the bottom right histogram indicates n additional counts beyond the histogram limits.

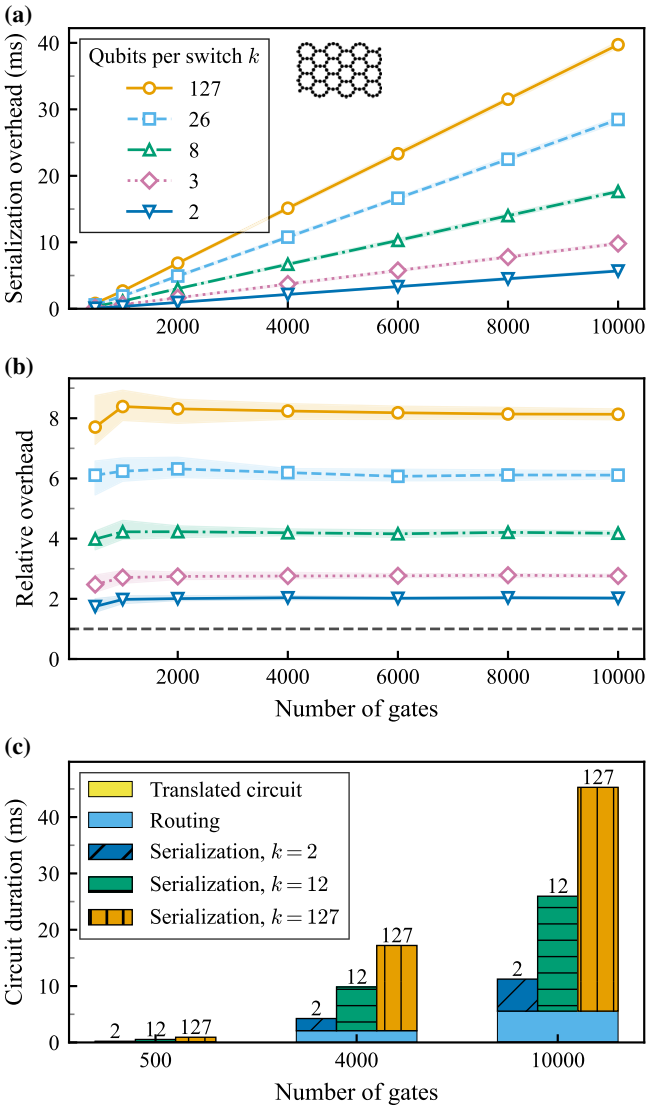

FIG. 17. Overhead from time-multiplexing single-qubit control for a 127-qubit heavy-hexagon architecture. Each data point displayed is the median over 100 seeds creating random circuits; shaded areas show the interquartile range. (a) Serialization overhead (increase in circuit duration compared to the routed circuit), as a function of the number of gates in random circuits. (b) The same serialization overhead as in (a), but now shown relative to the duration of the routed circuit, as a function of the number of gates in random circuits. (c) Breakdown of total circuit duration: translated circuit (black), qubit routing overhead (light blue), and serialization overhead (blue, green, yellow), for three different numbers of gates in random circuits. The procedure for generating and compiling the circuits is detailed in Sec. II.

We now zoom in on the results in Fig. 18(a) and plot the duration of only the serialization overhead for the cases of 2 and 4 qubits per switch in Fig. 19, in the same way as we did for the 11×11 grid in Fig. 15(a) and for the 5×5 grid in Fig. 15(b). We see that the results for the 127-qubit heavy-hexagon layout are quite similar to those for the 11×11 grid, with slightly longer serialization overheads in the heavy-hexagon case.

Finally, in the same style as Fig. 5 for the 11×11 grid and Fig. 16 in Appendix C for the 5×5 grid, we present the data from Fig. 18(b) in another form in Fig. 20. Here, we display histograms showing the distribution of serialization overhead relative to the routed circuit duration, for six different numbers of qubits per switch, ranging from 2 to 127. These results for the heavy-hexagon layout follow the same pattern as those for the 11×11 grid in Fig. 5 and the 5×5 grid in Fig. 16: as the number of qubits per switch increases, large serialization overheads gradually become increasingly common. The distributions in the different histograms are very similar to those in the corresponding histograms for the 11×11 grid in Fig. 5.

APPENDIX E: COMPATIBILITY OF SWITCH-BASED MULTIPLEXING WITH THE SURFACE CODE

The surface code is a stabilizer code with a relatively simple realization on a two-dimensional lattice. It operates on a square-grid layout of alternating data qubits and measurement qubits, each with nearest-neighbor couplings used for two-qubit gates. The measurement qubits are used to read out the stabilizers. In the surface code, the four two-qubit gates required for a single stabilizer measurement are already performed one after the other, as they involve the same measurement qubit [67]. As such, one can reduce coupler drive lines by a factor of four without incurring any overhead in execution time; compare Fig. 1(b).

Additionally, if data qubits in a distance-d rotated surface code [107,108] are connected to d shared switches in the pattern illustrated in Fig. 21, logical X_L and Z_L gates can be executed without additional duration.

If one increases the number of qubits per switch beyond what can be done without overhead as discussed above, there will be a trade-off between the increase in code distance one achieves and how much additional error the serialization introduces, as mentioned in Sec. V. For the surface code, the logical error rate

$$P_L \propto \left(\frac{p}{p_{\text{th}}}\right)^{d/2}, \tag{E1}$$

where p is the single-qubit error rate, p_{th} is the error threshold below which quantum error becomes beneficial, and d is the code distance [67]. The code distance d grows as $\sqrt{n}$, where n is the number of qubits. So if some physical limit prevents us from adding more control lines for the qubits, adding switches will enable increasing the number of qubits by some factor s, and the code distance will increase by a factor of $\sqrt{s}$. At the same time, adding switches will increase p, since the serialization will increase the time required to run one step of the error-correction cycle. If the

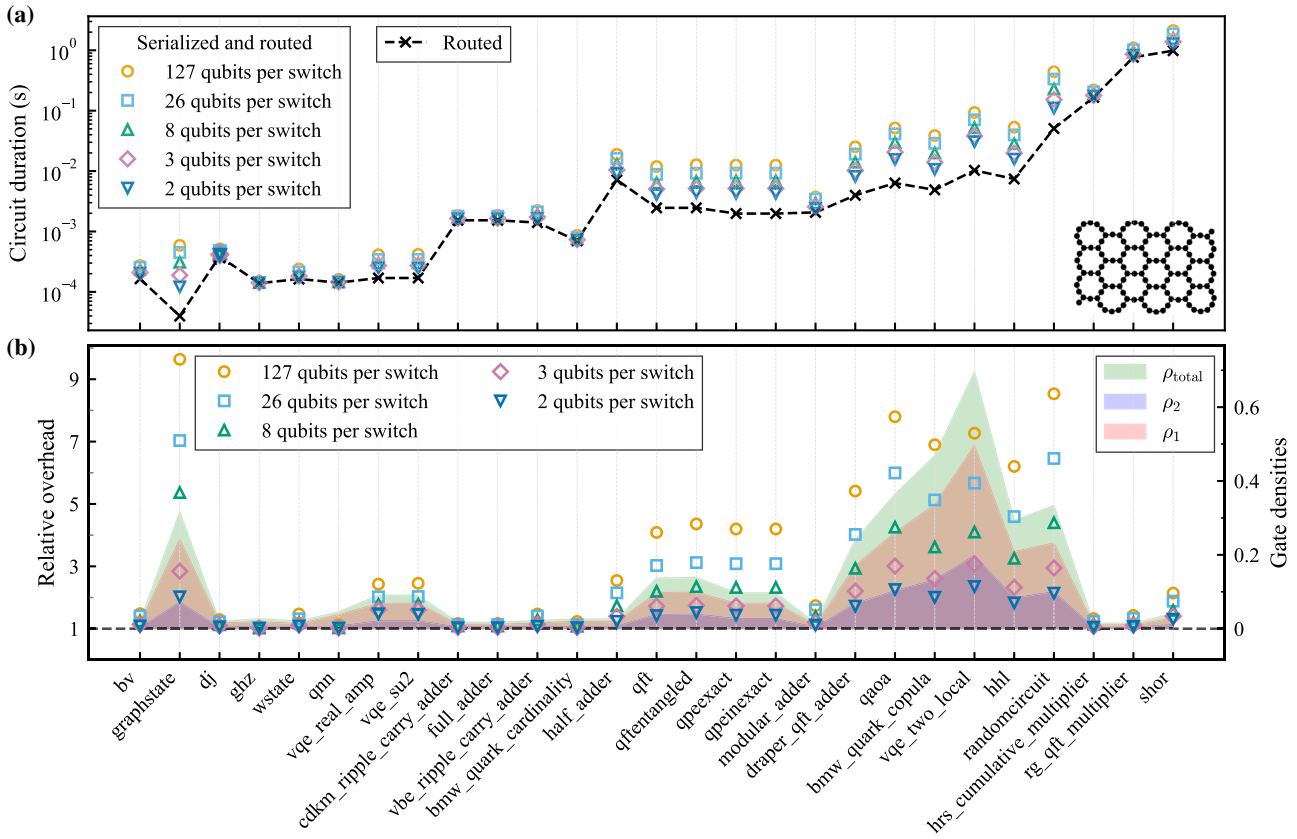

FIG. 18. Impact of time multiplexing on execution of quantum algorithms from the MQT Bench set for a 127-qubit heavy-hexagon layout. (a) Total circuit duration for serialized routed circuits (colored markers) and routed circuits without serialization (black crosses and dashed line) across different quantum algorithms, ordered by increasing gate count. (b) Relative overhead of serialization compared to the routed circuit duration [(markers, left axis)], with shaded areas indicating gate densities (ρ_1 pink, ρ_2 blue, ρ_{total} green; right axis) and different marker colors indicating different numbers of qubits per switch. The data points are medians over 20 transpiler seeds. The full names of the quantum algorithms are listed in Table II in Appendix F.

serialization overhead is low enough, as our results in this work indicate that it may very well be, and p is sufficiently

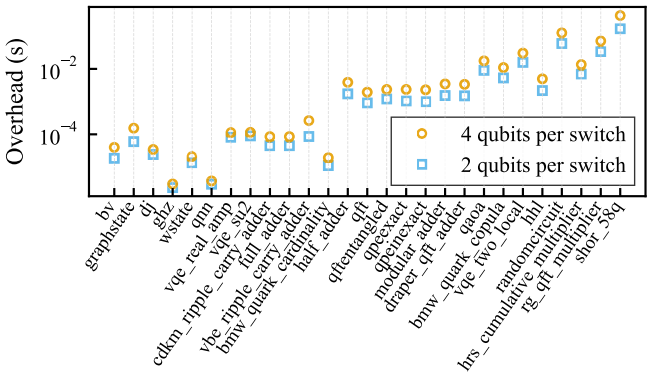

FIG. 19. Serialization overhead, in seconds, for the cases of 2 (blue squares) and 4 (orange circles) qubits per switch, for quantum algorithms from the MQT Bench set on a 127-qubit heavy-hexagon architecture. Exact numerical values for $k = 2$ are shown in Table I. The full names of the quantum algorithms are listed in Table II in Appendix F.

far below p_{th} to begin with, the increase in code distance will dominate, such that the trade-off becomes favorable and adding switches reduces the logical error rate P_L.

To be more specific, we note that a rotated surface code with code distance d on a square grid of qubits requires d^2 data qubits and $d^2 - 1$ measurement qubits [107,108]. Without switches, $\sim 4d^2$ control lines are then needed for the two-qubit gates between neighboring data and measurement qubits, and $2d^2 - 1$ control lines are needed for the single-qubit gates on data and measurement qubits. Adding switches that collect k control lines for single-qubit gates per switch or K control lines for two-qubit gates per switch, and keeping the number of control lines constant, we then find that we can increase the number of qubits from $\sim 2d^2$ to $\sim 2D^2$, where the new code distance

$$D = d\sqrt{\frac{3}{\frac{2}{K} + \frac{1}{k}}}. \tag{E2}$$

So if we only reduce the number of coupler drive lines by a factor $K = 4$ (which, as we showed above, does not incur

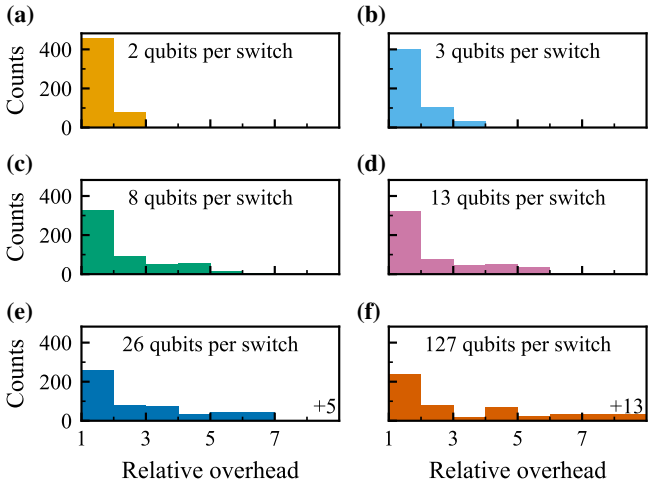

FIG. 20. Distribution of time-multiplexing overhead across switch sizes on a 127-qubit heavy-hexagon architecture, for the quantum algorithms from MQT Bench shown in Fig. 18. The histograms show the relative overhead (the ratio of serialized to routed circuit duration) for different numbers of qubits per switch. The $+n$ notation to the right in the same histograms indicates n additional counts beyond the histogram limits.

any serialization overhead) and do not add any switches for single-qubit gates (i.e., set $k = 1$), we can increase the code distance by a factor $\sqrt{2}$. Increasing K and k further beyond that yields even larger code distances, but also some serialization overhead, so the benefit will depend on the parameters in Eq. (E1) and the impact these changes have on some of these parameters.

APPENDIX F: MODELING THE OVERHEAD SCALING

Here, we present the numerical toy model and the analytical derivation we use to understand the scaling of serialization overhead with the number of qubits per switch

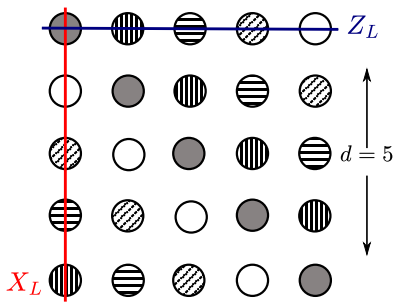

FIG. 21. The circles correspond to data qubits in a distance $d = 5$ rotated surface code. The five different patterns represent five different switches to which the qubit drive lines are connected. Compare Fig. 1 in Ref. [106]. Logical X_L and Z_L gates can be executed with concurrent single-qubit gates as participating qubits are allocated to different switches.

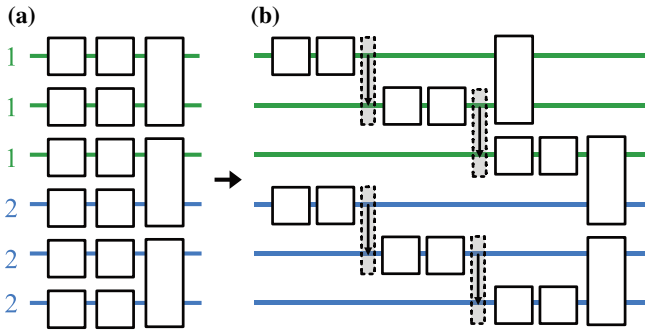

FIG. 22. Serialization of a dense circuit. (a) Circuit with unit gate density. (b) The serialized circuit shows extended duration linear in qubits per switch and single-qubit gates.

k, discussed in Sec. III C. We also provide further data to complement Fig. 8 in that section.

1. Numerical modeling of overhead scaling

For a dense circuit with many single-qubit gates, as illustrated in Fig. 22, not much optimization is possible and we expect a linear scaling in k for the serialization overhead. However, for lower gate densities, more optimization of gate ordering and placement becomes possible. To illustrate the difference between these cases, we put together a simple numerical model.

We model the circuit generation and overhead calculation as follows. First, we set a circuit depth and loop through layers of the circuit. For each qubit in each layer,

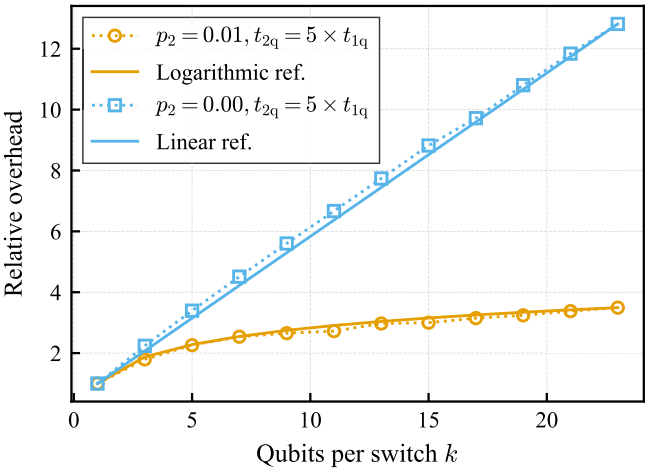

FIG. 23. Overhead scaling with number of qubits per switch k for a simple numerical model on a 5×5 grid. The single-qubit gate probability is fixed to $p_1 = 0.2$, roughly corresponding to the gate density. For a nonzero two-qubit gate probability $p_2 = 0.01$ and two-qubit gate duration t_2 larger than single-qubit gate duration t_1, the overhead scales as $\log k$ (round markers, orange). For $p_2 = 0$ (squares, blue), the scaling is linear in k. The results are averaged over 1000 random samples. The reference curves are scaled to match the data.

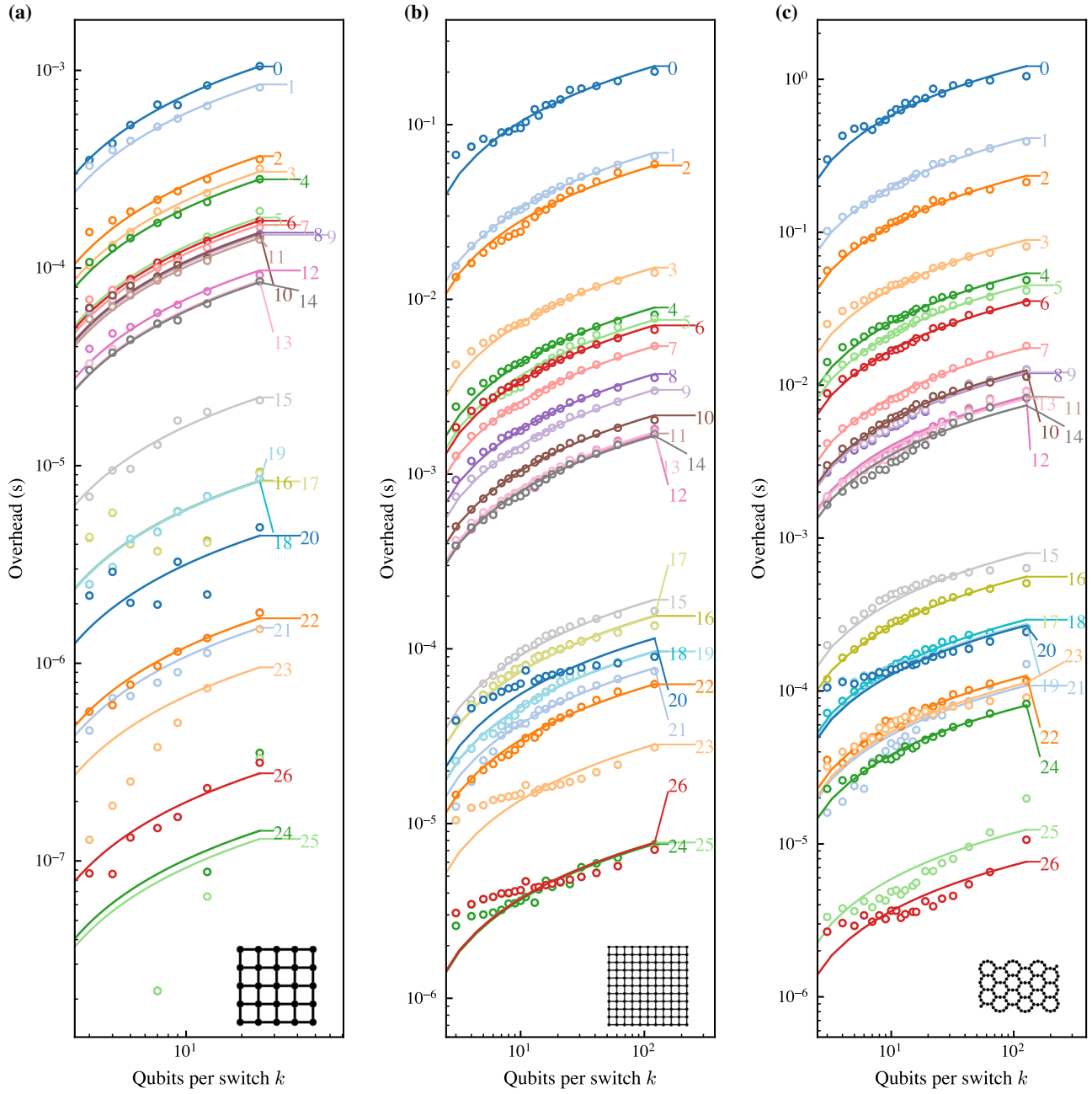

FIG. 24. Scaling of serialization overhead for the quantum algorithms from the MQT Bench set on (a) a 5×5 grid, (b) an 11×11 grid, and (c) a 127 qubit heavy-hexagon layout. Numbers label for different quantum algorithms, as listed in Table II. The fit function is $T_\Delta(k) = pN_1^{\text{routed}} t_{1q} \log(k)$. The fit parameters p for all fitted lines (solid curves) are listed in Table II.

we add a single-qubit gate with probability p_1. Assuming a square grid, we then add a two-qubit gate on nearest-neighbor pairs with probability p_2. We normalize units of time such that $t_1 = 1$.

For each layer in the circuit, we perform two calculations. First, we calculate the ideal runtime, which is unity if there are only parallel single-qubit gates, and t_2 if there is a two-qubit gate. Second, we calculate the serialized time per layer. If there is a two-qubit gate, we take the maximum of t_2 and the number of single-qubit gates on a switch. If there is no two-qubit gate, the serialized runtime is the number of single-qubit gates. We sum over layers to obtain the serialized runtime over ideal runtime (overhead factor).

In Fig. 23, we show scaling behaviors for selected parameters using this simple numerical model. We see from the plot that when there are only single-qubit gates, the serialization overhead grows linearly with k, but when there are some two-qubit gates present, and the density of

TABLE II. Full names of the quantum algorithms in the MQT Bench set used in this article, along with the indices used to label them in Fig. 24 and the values for the fitting parameter p for the fits in that figure.

Index	Label	Quantum algorithm	5×5 grid	11×11 grid	Heavy hexagon
0	shor_58q	Shor's factorization, 58 qubits	0.005(0)	0.643(10)	3.587(64)
1	randomcircuit	Random circuit ($2n$ depth)	0.019(0)	1.052(5)	6.230(46)
2	vqe_two_local	VQE two-local ansatz	0.050(2)	1.389(15)	8.049(109)
3	rg_qft_multiplier	Ruiz-Garcia QFT multiplier	0.003(0)	0.337(4)	1.337(15)
4	hhl	HHL linear system solver	0.080(1)	1.034(5)	3.318(30)
5	hrs_cumulative_multiplier	HRS cumulative multiplier	0.007(0)	0.197(3)	1.152(10)
6	qaoa	QAOA optimization	0.038(0)	1.308(15)	7.764(93)
7	qftentangled	QFT on GHZ state	0.133(4)	0.940(5)	4.534(83)
8	qft	Quantum Fourier transform	0.130(3)	0.960(6)	4.222(96)
9	qpeexact	QPE (exact phase)	0.118(3)	0.920(10)	4.356(89)
10	bmw_quark_copula	QUARK copula model	0.043(2)	1.356(11)	6.705(45)
11	qpeinexact	QPE (inexact phase)	0.114(2)	0.920(10)	4.500(73)
12	half_adder	Half adder	0.054(1)	0.811(6)	4.618(46)
13	modular_adder	Modular adder	0.042(1)	0.994(5)	3.928(48)
14	draper_qft_adder	Draper QFT adder	0.037(0)	1.108(8)	3.514(51)
15	vbe_ripple_carry_adder	VBE ripple carry adder	0.087(2)	0.503(8)	2.077(50)
16	vqe_real_amp	VQE real amplitudes	0.051(6)	0.634(9)	1.090(33)
17	vqe_su2	VQE efficient SU(2)	0.047(6)	0.578(9)	1.021(32)
18	cdkm_ripple_carry_adder	CDKM ripple carry adder	0.034(1)	0.265(3)	0.795(10)
19	full_adder	Full adder	0.034(1)	0.265(3)	0.795(10)
20	wstate	W state preparation	0.040(5)	0.700(27)	0.486(6)
21	graphstate	Graph state preparation	0.030(2)	1.043(14)	7.447(80)
22	dj	Deutsch-Jozsa algorithm	0.018(0)	0.441(4)	0.877(11)
23	bmw_quark_cardinality	QUARK cardinality model	0.004(1)	0.080(3)	0.302(14)
24	qnn	Quantum neural network	0.002(1)	0.066(3)	0.105(6)
25	ghz	GHZ state preparation	0.002(1)	0.097(4)	0.094(4)
26	bv	Bernstein-Vazirani algorithm	0.010(0)	0.176(4)	2.591(73)

single-qubit gates is not so high, the serialization overhead instead grows logarithmically with k.

2. Further supporting data

In Fig. 24, we expand on the selection in Fig. 8, showing the full data for all algorithms from the MQT Bench set, and for all three quantum processor layouts considered in this article, for scaling of serialization overhead with the number of qubits per switch k. From these plots and the fits there (solid curves), we see that the logarithmic scaling with k is prevalent. We provide the fitting parameter for each case, along with the full names for the quantum algorithms considered, in Table II.

3. Queueing-theory-inspired derivation of the logarithmic scaling

We sketch a derivation of how the worst-case overhead duration scales with the number of qubits per switch k, where the overhead corresponds to the maximum waiting time. We use results from extreme-value theory [109] and queueing theory, in particular, the standard result that steady-state waiting-time distributions in stable queues often exhibit exponential tail decay [96,110,111].

Let $W_1, \ldots, W_k$ be waiting times for k qubits on a switch, each with an exponential tail,

$$\Pr(W > x) \sim e^{-\eta x}, \quad x \to \infty, \tag{F1}$$

where η is a constant called the asymptotic decay rate. For simplicity, we set $\eta = 1$. Define the maximum waiting time

$$M_k = \max\{W_1, \ldots, W_k\}. \tag{F2}$$

For a given circuit layer, the serialization overhead is determined by the maximum waiting time across all switches, since the layer cannot complete before the last queued gate is executed. As such, we let M_k represent the waiting time for the worst-case scenario where all k qubits assigned to a switch are queued. We have

$$\begin{aligned} \Pr(M_k < x) &= \Pr(M_k \leq x) \\ &= \Pr(W \leq x)^k \\ &= [1 - \Pr(W > x)]^k \\ &= [1 - e^{-x}]^k. \end{aligned} \tag{F3}$$

It can be shown that

$$\Pr(M_k < x + \log k) = \left(1 - \frac{e^{-x}}{k}\right)^k \underset{k\to\infty}{\to} \exp\{-e^{-x}\} = \Lambda(x), \tag{F4}$$

where $\Lambda(x)$ is the Gumbel distribution, one of the standard extreme-value distributions [109]. As such, we have

$$M_k \overset{d}{=} \log k + Z, \tag{F5}$$

where the equality denotes convergence in distribution, and Z is a random variable of the Gumbel distribution. As such, the expectation value of the maximum waiting time is

$$E[M_k] = \log k + E[Z], \tag{F6}$$

with the expectation value of the Gumbel random variable $E[Z] = 0.5772$ (Euler's constant) [96]. This implies that in the asymptotic limit, the expected serialization overhead grows logarithmically with the number of qubits per switch k.

[1] X. Gu, A. F. Kockum, A. Miranowicz, Y.-X. Liu, and F. Nori, Microwave photonics with superconducting quantum circuits, Phys. Rep. **718–719**, 1 (2017).

[2] P. Krantz, M. Kjaergaard, F. Yan, T. P. Orlando, S. Gustavsson, and W. D. Oliver, A quantum engineer's guide to superconducting qubits, Appl. Phys. Rev. **6**, 021318 (2019).

[3] A. Blais, A. L. Grimsmo, S. M. Girvin, and A. Wallraff, Circuit quantum electrodynamics, Rev. Mod. Phys. **93**, 025005 (2021).

[4] A. Chatterjee, P. Stevenson, S. De Franceschi, A. Morello, N. P. de Leon, and F. Kuemmeth, Semiconductor qubits in practice, Nat. Rev. Phys. **3**, 157 (2021).

[5] G. Burkard, T. D. Ladd, A. Pan, J. M. Nichol, and J. R. Petta, Semiconductor spin qubits, Rev. Mod. Phys. **95**, 025003 (2023).

[6] S. Krinner, S. Storz, P. Kurpiers, P. Magnard, J. Heinsoo, R. Keller, J. Lütolf, C. Eichler, and A. Wallraff, Engineering cryogenic setups for 100-qubit scale superconducting circuit systems, EPJ Quantum Technol. **6**, 2 (2019).

[7] M. Mohseni *et al.*, How to build a quantum supercomputer: Scaling from hundreds to millions of qubits, arXiv:2411.10406.

[8] N. Raicu, T. Hogan, X. Wu, M. Vahidpour, D. Snow, M. Hollister, and M. Field, Cryogenic thermal modeling of microwave high density signaling, EPJ Quantum Technol. **12**, 124 (2025).

[9] F. Lecocq, F. Quinlan, K. Cicak, J. Aumentado, S. A. Diddams, and J. D. Teufel, Control and readout of a superconducting qubit using a photonic link, Nature **591**, 575 (2021).

[10] A. Youssefi, I. Shomroni, Y. J. Joshi, N. R. Bernier, A. Lukashchuk, P. Uhrich, L. Qiu, and T. J. Kippenberg, A cryogenic electro-optic interconnect for superconducting devices, Nat. Electron. **4**, 326 (2021).

[11] M. J. Weaver, P. Duivestein, A. C. Bernasconi, S. Scharmer, M. Lemang, T. C. V. Thiel, F. Hijazi, B. Hensen, S. Gröblacher, and R. Stockill, An integrated microwave-to-optics interface for scalable quantum computing, Nat. Nanotechnol. **19**, 166 (2024).

[12] M. Shen, J. Xie, Y. Xu, S. Wang, R. Cheng, W. Fu, Y. Zhou, and H. X. Tang, Photonic link from single-flux-quantum circuits to room temperature, Nat. Photonics **18**, 371 (2024).

[13] R. McDermott, M. G. Vavilov, B. L. T. Plourde, F. K. Wilhelm, P. J. Liebermann, O. A. Mukhanov, and T. A. Ohki, Quantum–classical interface based on single flux quantum digital logic, Quantum Sci. Technol. **3**, 024004 (2018).

[14] C. H. Liu, A. Ballard, D. Olaya, D. R. Schmidt, J. Biesecker, T. Lucas, J. Ullom, S. Patel, O. Rafferty, A. Opremcak, K. Dodge, V. Iaia, T. McBroom, J. L. DuBois, P. F. Hopkins, S. P. Benz, B. L. T. Plourde, and R. McDermott, Single flux quantum-based digital control of superconducting qubits in a multichip module, PRX Quantum **4**, 030310 (2023).

[15] O. Mukhanov, A. Kirichenko, C. Howington, J. Walter, M. Hutchings, I. Vernik, D. Yohannes, K. Dodge, A. Ballard, B. L. T. Plourde, A. Opremcak, C.-H. Liu, and R. McDermott, in *2019 IEEE International Electron Devices Meeting (IEDM)* (IEEE, San Francisco, CA, USA, 2019), pp. 31.2.1–31.2.4.

[16] R. Huang, X. Geng, X. Wu, G. Dai, L. Yang, J. Liu, and W. Chen, Cryogenic multiplexing control chip for a superconducting quantum processor, Phys. Rev. Appl. **18**, 064046 (2022).

[17] J. M. Hornibrook, J. I. Colless, I. D. Conway Lamb, S. J. Pauka, H. Lu, A. C. Gossard, J. D. Watson, G. C. Gardner, S. Fallahi, M. J. Manfra, and D. J. Reilly, Cryogenic control architecture for large-scale quantum computing, Phys. Rev. Appl. **3**, 024010 (2015).

[18] J. P. G. van Dijk, B. Patra, S. Pellerano, E. Charbon, F. Sebastiano, and M. Babaie, Designing a DDS-based soC for high-fidelity multi-qubit control, IEEE Trans. Circuits Syst. I Regul. Pap. **67**, 5380 (2020).

[19] P. Zhao, A multiplexed control architecture for superconducting qubits with row-column addressing, arXiv:2403.03717.

[20] Y. Chen, D. Sank, P. O'Malley, T. White, R. Barends, B. Chiaro, J. Kelly, E. Lucero, M. Mariantoni, A. Megrant, C. Neill, A. Vainsencher, J. Wenner, Y. Yin, A. N. Cleland, and J. M. Martinis, Multiplexed dispersive readout of superconducting phase qubits, Appl. Phys. Lett. **101**, 182601 (2012).

[21] J. Heinsoo, C. K. Andersen, A. Remm, S. Krinner, T. Walter, Y. Salathé, S. Gasparinetti, J.-C. Besse, A. Potočnik, A. Wallraff, and C. Eichler, Rapid high-fidelity multiplexed readout of superconducting qubits, Phys. Rev. Appl. **10**, 034040 (2018).

[22] S. Asaad, C. Dickel, N. K. Langford, S. Poletto, A. Bruno, M. A. Rol, D. Deurloo, and L. DiCarlo, Independent, extensible control of same-frequency superconducting qubits by selective broadcasting, npj Quantum Inf. **2**, 16029 (2016).

[23] Z. H. Yang, R. Wang, Z. T. Wang, P. Zhao, K. Huang, K. Xu, Y. Tian, H. F. Yu, and S. P. Zhao, Mitigation of microwave crosstalk with parameterized single-qubit gate in superconducting quantum circuits, Appl. Phys. Lett. **124**, 214001 (2024).

[24] R. Ohira, R. Matsuda, H. Shiomi, K. Ogawa, and M. Negoro, Optimizing multi-tone microwave pulses via phase selection for quantum computing applications, J. Appl. Phys. **136**, 114402 (2024).

[25] R. Matsuda, R. Ohira, T. Sumida, H. Shiomi, A. Machino, S. Morisaka, K. Koike, T. Miyoshi, Y. Kurimoto, Y. Sugita, Y. Ito, Y. Suzuki, P. A. Spring, S. Wang, S. Tamate, Y. Tabuchi, Y. Nakamura, K. Ogawa, and M. Negoro, Selective excitation of superconducting qubits with a shared control line through pulse shaping, Phys. Rev. Appl. **8**, L012003 (2026).

[26] D. R. Ward, D. E. Savage, M. G. Lagally, S. N. Coppersmith, and M. A. Eriksson, Integration of on-chip field-effect transistor switches with dopantless Si/SiGe quantum dots for high-throughput testing, Appl. Phys. Lett. **102**, 213107 (2013).

[27] H. Al-Taie, L. W. Smith, B. Xu, P. See, J. P. Griffiths, H. E. Beere, G. A. C. Jones, D. A. Ritchie, M. J. Kelly, and C. G. Smith, Cryogenic on-chip multiplexer for the study of quantum transport in 256 split-gate devices, Appl. Phys. Lett. **102**, 243102 (2013).

[28] B. Paquelet Wuetz, P. L. Bavdaz, L. A. Yeoh, R. Schouten, H. van der Does, M. Tiggelman, D. Sabbagh, A. Sammak, C. G. Almudever, F. Sebastiano, J. S. Clarke, M. Veldhorst, and G. Scappucci, Multiplexed quantum transport using commercial off-the-shelf CMOS at sub-kelvin temperatures, npj Quantum Inf. **6**, 43 (2020).

[29] E. J. Thomas, V. N. Ciriano-Tejel, D. F. Wise, D. Prete, M. de Kruijf, D. J. Ibberson, G. M. Noah, A. Gomez-Saiz, M. F. Gonzalez-Zalba, M. A. I. Johnson, and J. J. L. Morton, Rapid cryogenic characterization of 1,024 integrated silicon quantum dot devices, Nat. Electron. **8**, 75 (2025).

[30] M. F. Gonzalez-Zalba, S. de Franceschi, E. Charbon, T. Meunier, M. Vinet, and A. S. Dzurak, Scaling silicon-based quantum computing using CMOS technology, Nat. Electron. **4**, 872 (2021).

[31] H. Bohuslavskyi, A. Ronzani, J. Hätinen, A. Rantala, A. Shchepetov, P. Koppinen, J. S. Lehtinen, and M. Prunnila, Scalable on-chip multiplexing of silicon single and double quantum dots, Commun. Phys. **7**, 323 (2024).

[32] P. L. Bavdaz, H. G. J. Eenink, J. van Staveren, M. Lodari, C. G. Almudever, J. S. Clarke, F. Sebasatiano, M. Veldhorst, and G. Scappucci, A quantum dot crossbar with sub-linear scaling of interconnects at cryogenic temperature, npj Quantum Inf. **8**, 86 (2022).

[33] S. K. Bartee, W. Gilbert, K. Zuo, K. Das, T. Tanttu, C. H. Yang, N. Dumoulin Stuyck, S. J. Pauka, R. Y. Su, W. H. Lim, S. Serrano, C. C. Escott, F. E. Hudson, K. M. Itoh, A. Laucht, A. S. Dzurak, and D. J. Reilly, Spin-qubit control with a milli-kelvin CMOS chip, Nature **643**, 382 (2025).

[34] R. Acharya, S. Brebels, A. Grill, J. Verjauw, T. Ivanov, D. P. Lozano, D. Wan, J. Van Damme, A. M. Vadiraj, M. Mongillo, B. Govoreanu, J. Craninckx, I. P. Radu, K. De Greve, G. Gielen, F. Catthoor, and A. Potočnik, Multiplexed superconducting qubit control at millikelvin temperatures with a low-power cryo-CMOS multiplexer, Nat. Electron. **6**, 900 (2023).

[35] M. Pechal, J.-C. Besse, M. Mondal, M. Oppliger, S. Gasparinetti, and A. Wallraff, Superconducting switch for fast on-chip routing of quantum microwave fields, Phys. Rev. Appl. **6**, 024009 (2016).

[36] B. J. Chapman, B. A. Moores, E. I. Rosenthal, J. Kerckhoff, and K. W. Lehnert, General purpose multiplexing device for cryogenic microwave systems, Appl. Phys. Lett. **108**, 222602 (2016).

[37] O. Naaman, M. O. Abutaleb, C. Kirby, and M. Rennie, On-chip Josephson junction microwave switch, Appl. Phys. Lett. **108**, 112601 (2016).

[38] G. De Simoni, F. Paolucci, P. Solinas, E. Strambini, and F. Giazotto, Metallic supercurrent field-effect transistor, Nat. Nanotechnol. **13**, 802 (2018).

[39] A. Wagner, L. Ranzani, G. Ribeill, and T. A. Ohki, Demonstration of a superconducting nanowire microwave switch, Appl. Phys. Lett. **115**, 172602 (2019).

[40] A. L. Graninger, J. M. Cochran, A. A. Pesetski, J. D. Strand, R. E. Zimmerman, and N. L. Mungo, Microwave switch architecture for superconducting integrated circuits using magnetic field-tunable Josephson junctions, IEEE Trans. Appl. Supercond. **33**, 1501605 (2023).

[41] J. Zotova, A. Semenov, R. Wang, Y. Zhou, O. Astafiev, and J.-S. Tsai, Tunable compact on-chip superconducting switch, Phys. Rev. Appl. **21**, 024059 (2024).

[42] Y.-H. Huang, Q.-Y. Zhao, H. Hao, N.-T. Liu, Z. Liu, J. Deng, F. Yang, S.-Y. Ru, X.-C. Tu, L.-B. Zhang, X.-Q. Jia, J. Chen, L. Kang, and P.-H. Wu, Monolithic integrated superconducting nanowire digital encoder, Appl. Phys. Lett. **124**, 192601 (2024).

[43] A. Paghi, L. Borgongino, S. Tortorella, G. De Simoni, E. Strambini, L. Sorba, and F. Giazotto, Supercurrent time division multiplexing with solid-state integrated hybrid superconducting electronics, Nat. Commun. **16**, 8442 (2025).

[44] A. Paghi, L. Borgongino, S. Battisti, S. Tortorella, G. Trupiano, G. De Simoni, E. Strambini, L. Sorba, and F. Giazotto, Josephson field effect transistors with InAs on insulator and high permittivity gate dielectrics, ACS Appl. Electron. Mater. **7**, 3756 (2025).

[45] F. Barati, J. P. Thompson, M. C. Dartiailh, K. Sardashti, W. Mayer, J. Yuan, K. Wickramasinghe, K. Watanabe, T. Taniguchi, H. Churchill, and J. Shabani, Tuning supercurrent in Josephson field-effect transistors using h-BN dielectric, Nano Lett. **21**, 1915 (2021).

[46] Technically, this describes demultiplexing, but we simply refer to it as multiplexing for convenience.

[47] T. Abad, J. Fernández-Pendás, A. F. Kockum, and G. Johansson, Universal fidelity reduction of quantum operations from weak dissipation, Phys. Rev. Lett. **129**, 150504 (2022).

[48] Y. Ge, W. Wenjie, C. Yuheng, P. Kaisen, L. Xudong, Z. Zixiang, W. Yuhan, W. Ruocheng, and Y. Junchi, Quantum circuit synthesis and compilation optimization: Overview and prospects, arXiv:2407.00736.

[49] Z. Shan, Y. Zhu, and B. Zhao, A high-performance compilation strategy for multiplexing quantum control architecture, Sci. Rep. **12**, 7132 (2022).

[50] L. Lao, H. van Someren, I. Ashraf, and C. G. Almudever, Timing and resource-aware mapping of quantum circuits to superconducting processors, IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. **41**, 359 (2022).

[51] C.-Y. Huang and W.-K. Mak, in *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)* (IEEE, Incheon, Republic of Korea, 2024), pp. 22–25.

[52] K. Booth, M. Do, J. Beck, E. Rieffel, D. Venturelli, and J. Frank, in *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS 2018)* (AAAI Press, Delft, The Netherlands, 2018), Vol. 28, pp. 366–374.

[53] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, Quantum computing with Qiskit, arXiv:2405.08810.

[54] N. Quetschlich, L. Burgholzer, and R. Wille, MQT bench: Benchmarking software and design automation tools for quantum computing, Quantum **7**, 1062 (2023).

[55] F. Arute *et al*., Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[56] R. Acharya *et al*., Quantum error correction below the surface code threshold, Nature **638**, 920 (2025).

[57] J. Chow, O. Dial, and J. Gambetta, IBM Quantum breaks the 100-qubit processor barrier (2021), IBM Quantum Computing Blog (accessed 23 August 2025), https://www.ibm.com/quantum/blog/127-qubit-quantum-processor-eagle.

[58] Y. Chen *et al.*, Qubit architecture with high coherence and fast tunable coupling, Phys. Rev. Lett. **113**, 220502 (2014).

[59] D. C. McKay, S. Filipp, A. Mezzacapo, E. Magesan, J. M. Chow, and J. M. Gambetta, Universal gate for fixed-frequency qubits via a tunable bus, Phys. Rev. Appl. **6**, 064007 (2016).

[60] F. Yan, P. Krantz, Y. Sung, M. Kjaergaard, D. L. Campbell, T. P. Orlando, S. Gustavsson, and W. D. Oliver, Tunable coupling scheme for implementing high-fidelity two-qubit gates, Phys. Rev. Appl. **10**, 054062 (2018).

[61] R. Acharya *et al*., Suppressing quantum errors by scaling a surface code logical qubit, Nature **614**, 676 (2023).

[62] J. Kusyk, S. M. Saeed, and M. U. Uyar, Survey on quantum circuit compilation for noisy intermediate-scale quantum computers: Artificial intelligence to heuristics, IEEE Trans. Quantum Eng. **2**, 2501616 (2021).

[63] G. Li, Y. Ding, and Y. Xie, in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems* (ACM, New York, NY, 2019). pp. 1001–1014.

[64] A. Cowtan, S. Dilkes, R. Duncan, A. Krajenbrink, W. Simmons, and S. Sivarajah, in *14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019)*, edited by W. van Dam and L. Mancinska (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, 2019), Vol. 135, pp. 5:1–5:32.

[65] G. Nannicini, L. S. Bishop, O. Günlük, and P. Jurcevic, Optimal qubit assignment and routing via integer programming, ACM Trans. Quantum Comput. **4**, 1 (2023).

[66] T. Ito, N. Kakimura, N. Kamiyama, Y. Kobayashi, and Y. Okamoto, in *Lecture Notes in Computer Science*, edited by P. Morin and S. Suri (Springer, Cham, 2023), Vol. 14079, pp. 533–546.

[67] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, Phys. Rev. A **86**, 032324 (2012).

[68] G. S. Paraoanu, Microwave-induced coupling of superconducting qubits, Phys. Rev. B **74**, 140504(R) (2006).

[69] C. Rigetti and M. Devoret, Fully microwave-tunable universal gates in superconducting qubits with linear couplings and fixed transition frequencies, Phys. Rev. B **81**, 134507 (2010).

[70] J. M. Chow, A. D. Córcoles, J. M. Gambetta, C. Rigetti, B. R. Johnson, J. A. Smolin, J. R. Rozen, G. A. Keefe, M. B. Rothwell, M. B. Ketchen, and M. Steffen, Simple all-microwave entangling gate for fixed-frequency superconducting qubits, Phys. Rev. Lett. **107**, 080502 (2011).

[71] 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 **93**, 060302(R) (2016).

[72] K. Heya and N. Kanazawa, Cross-cross resonance gate, PRX Quantum **2**, 040336 (2021).

[73] Y. Kim, A. Morvan, L. B. Nguyen, R. K. Naik, C. Jünger, L. Chen, J. M. Kreikebaum, D. I. Santiago, and I. Siddiqi, High-fidelity three-qubit iToffoli gate for fixed-frequency superconducting qubits, Nat. Phys. **18**, 783 (2022).

[74] T. Itoko, M. Malekakhlagh, N. Kanazawa, and M. Takita, Three-qubit parity gate via simultaneous cross-resonance drives, Phys. Rev. Appl. **21**, 034018 (2024).

[75] X. Gu, J. Fernández-Pendás, P. Vikstål, T. Abad, C. Warren, A. Bengtsson, G. Tancredi, V. Shumeiko, J. Bylander, G. Johansson, and A. F. Kockum, Fast multiqubit gates through simultaneous two-qubit gates, PRX Quantum **2**, 040348 (2021).

[76] C. W. Warren, J. Fernández-Pendás, S. Ahmed, T. Abad, A. Bengtsson, J. Biznárová, K. Debnath, X. Gu, C. Križan, A. Osman, A. Fadavi Roudsari, P. Delsing, G. Johansson, A. Frisk Kockum, G. Tancredi, and J. Bylander, Extensive characterization and implementation of a family of three-qubit gates at the coherence limit, npj Quantum Inf. **9**, 44 (2023).

[77] H.-T. Liu *et al.*, Direct implementation of high-fidelity three-qubit gates for superconducting processor with tunable couplers, Phys. Rev. Lett. **135**, 050602 (2025).

[78] A. M. Childs, E. Schoute, and C. M. Unsal, in *14th Conference on the Theory of Quantum Computation, Communication and Cryptography (TQC 2019)*, Leibniz International Proceedings in Informatics (LIPIcs), edited by W. van Dam and L. Mančinska (Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, 2019), Vol. 135, pp. 3:1–3:24.

[79] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).

[80] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta, Efficient Z gates for quantum computing, Phys. Rev. A **96**, 022330 (2017).

[81] C. Križan *et al.*, Quantum SWAP gate realized with CZ and iSWAP gates in a superconducting architecture, New J. Phys. **27**, 074507 (2025).

[82] S. A. Caldwell *et al*., Parametrically activated entangling gates using transmon qubits, Phys. Rev. Appl. **10**, 034050 (2018).

[83] R. Li, K. Kubo, Y. Ho, Z. Yan, Y. Nakamura, and H. Goto, Realization of high-fidelity CZ gate based on a double-transmon coupler, Phys. Rev. X **14**, 041050 (2024).

[84] L. Ding, M. Hays, Y. Sung, B. Kannan, J. An, A. Di Paolo, A. H. Karamlou, T. M. Hazard, K. Azar, D. K. Kim, B. M. Niedzielski, A. Melville, M. E. Schwartz, J. L. Yoder, T. P. Orlando, S. Gustavsson, J. A. Grover, K. Serniak, and W. D. Oliver, High-fidelity, frequency-flexible two-qubit fluxonium gates with a transmon coupler, Phys. Rev. X **13**, 031035 (2023).

[85] S. Shirai, Y. Okubo, K. Matsuura, A. Osada, Y. Nakamura, and A. Noguchi, All-microwave manipulation of superconducting qubits with a fixed-frequency transmon coupler, Phys. Rev. Lett. **130**, 260601 (2023).

[86] P. Xu, H. Zhang, and S. Wu, Protocols for $\sqrt{i\mathrm{SWAP}}$ gates using a fixed coupler driven by two microwave pulses, Phys. Rev. Appl. **23**, 034036 (2025).

[87] H. Putterman *et al*., Hardware-efficient quantum error correction via concatenated bosonic qubits, Nature **638**, 927 (2025).

[88] C. T. Hann, K. Noh, H. Putterman, M. H. Matheny, J. K. Iverson, M. T. Fang, C. Chamberland, O. Painter, and F. G. S. L. Brandão, Hybrid cat-transmon architecture for scalable, hardware-efficient quantum error correction, PRX Quantum **6**, 030305 (2025).

[89] X. Deng, W. Zheng, X. Liao, H. Zhou, Y. Ge, J. Zhao, D. Lan, X. Tan, Y. Zhang, S. Li, and Y. Yu, Long-range *ZZ* interaction via resonator-induced phase in superconducting qubits, Phys. Rev. Lett. **134**, 020801 (2025).

[90] J. Song, S. Yang, P. Liu, H.-L. Zhang, G.-M. Xue, Z.-Y. Mi, W.-G. Zhang, F. Yan, Y.-R. Jin, and H.-F. Yu, Realization of high-fidelity perfect entanglers between remote superconducting quantum processors, Phys. Rev. Lett. **135**, 050603 (2025).

[91] G. B. P. Huber *et al.*, Parametric multielement coupling architecture for coherent and dissipative control of superconducting qubits, PRX Quantum **6**, 030313 (2025).

[92] Qiskit GitHub, Properties JSON for the Brisbane backend in qiskit-ibm-runtime, commit 7a99d12, 2025, https://github.com/Qiskit/qiskit-ibm-runtime/blob/main/qiskit_ibm_runtime/fake_provider/backends/brisbane/props_brisbane.json

[93] S. Asmussen, *Applied Probability and Queues* (Springer, New York, NY, 2003).

[94] G. L. Choudhury, D. M. Lucantoni, and W. Whitt, Squeezing the most out of ATM, IEEE Trans. Commun. **44**, 203 (1996).

[95] F. Ciucu and F. Poloczek, in *2015 IEEE Conference on Computer Communications (INFOCOM)* (IEEE, Hong Kong, China, 2015), p. 1122–1130.

[96] A. W. Berger and W. Whitt, Maximum values in queueing processes, Probab. Eng. Inf. Sci. **9**, 375 (1995).

[97] D. P. Franke, J. S. Clarke, L. M. K. Vandersypen, and M. Veldhorst, Rent's rule and extensibility in quantum computing, Microprocess. Microsyst. **67**, 1 (2019).

[98] F. Borsoi, N. W. Hendrickx, V. John, M. Meyer, S. Motz, F. van Riggelen, A. Sammak, S. L. de Snoo, G. Scappucci, and M. Veldhorst, Shared control of a 16 semiconductor quantum dot crossbar array, Nat. Nanotechnol. **19**, 21 (2024).

[99] B. Patra, R. M. Incandela, J. P. G. van Dijk, H. A. R. Homulle, L. Song, and M. Shahmohammadi, Cryo-CMOS circuits and systems for quantum computing applications, IEEE J. Solid-State Circuits **53**, 309 (2018).

[100] S. J. Pauka, K. Das, R. Kalra, A. Moini, Y. Yang, M. Trainer, A. Bousquet, C. Cantaloube, N. Dick, G. C. Gardner, M. J. Manfra, and D. J. Reilly, A cryogenic CMOS chip for generating control signals for multiple qubits, Nat. Electron. **4**, 64 (2021).

[101] R. Versluis, S. Poletto, N. Khammassi, B. Tarasinski, N. Haider, D. J. Michalak, A. Bruno, K. Bertels, and L. DiCarlo, Scalable quantum circuit and control for a superconducting surface code, Phys. Rev. Appl. **8**, 034021 (2017).

[102] P. Shi, J. Yuan, F. Yan, and H. Yu, Multiplexed control scheme for scalable quantum information processing with superconducting qubits, arXiv:2312.06911.

[103] M. Richter, I. Strandberg, S. Gasparinetti, and A. Frisk Kockum, marvin-richter/overhead-time-multiplexing, v1.0.0—publication release, 2026, https://doi.org/10.5281/zenodo.18864897

[104] H. Zou, M. Treinish, K. Hartman, A. Ivrii, and J. Lishman, in *2025 IEEE International Conference on Quantum Computing and Engineering (QCE)*, Vol. 1 (Albuquerque, NM, USA, 2025), p. 749.

[105] E. Russo, E. Vinciguerra, M. Palesi, D. Patti, G. Ascia, and V. Catania, TeleSABRE: Layout synthesis in multi-core quantum systems with teleport interconnect, arXiv:2505.08928.

[106] S. Bravyi, M. Englbrecht, R. Köinig, and N. Peard, Correcting coherent errors with surface codes, npj Quantum Inf. **4**, 55 (2018).

[107] H. Bombin and M. A. Martin-Delgado, Optimal resources for topological two-dimensional stabilizer codes: Comparative study, Phys. Rev. A **76**, 012305 (2007).

[108] A. R. O'Rourke and S. Devitt, Compare the pair: Rotated versus unrotated surface codes at equal logical error rates, Phys. Rev. Res. **7**, 033074 (2025).

[109] P. Embrechts, C. Klüppelberg, and T. Mikosch, *Modelling Extremal Events* (Springer, Berlin, 1997).

[110] J. Abate, G. L. Choudhury, and W. Whitt, Exponential approximations for tail probabilities in queues, I: Waiting times, Oper. Res. **43**, 885 (1995).

[111] P. W. Glynn and W. Whitt, Logarithmic asymptotics for steady-state tail probabilities in a single-server queue, J. Appl. Probab. **31**, 131 (1994).