TU Delft

Delft University of Technology

Real-time decoding for fault-tolerant quantum computing

Progress, challenges and outlook

Battistel, F.; Chamberland, C.; Johar, K.; Overwater, R. W.J.; Sebastiano, F.; Skoric, L.; Ueno, Y.; Usman, M.

DOI
10.1088/2399-1984/aceba6

Publication date
2023

Published in
Nano Futures

Citation (APA)
Battistel, F., Chamberland, C., Johar, K., Overwater, R. W. J., Sebastiano, F., Skoric, L., Ueno, Y., & Usman, M. (2023). Real-time decoding for fault-tolerant quantum computing: Progress, challenges and outlook. *Nano Futures*, *7*(3), Article 032003. https://doi.org/10.1088/2399-1984/aceba6

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

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

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

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

TOPICAL REVIEW • OPEN ACCESS

Real-time decoding for fault-tolerant quantum computing: progress, challenges and outlook

To cite this article: F Battistel *et al* 2023 *Nano Futures* **7** 032003

View the article online for updates and enhancements.

You may also like

- Density matrix simulation of quantum error correction codes for near-term quantum devices
Chungheon Baek, Tomohiro Ostuka, Seigo Tarucha et al.
- Magnetic field and microstructural effects on the electrical capacitance and resistance of a quadrupolar electrical capacitor based on cotton fabrics and carbonyl iron microparticles
Ioan Bica and Eugen Mircea Anitas
- Quantum error correction for beginners
Simon J Devitt, William J Munro and Kae Nemoto

This content was downloaded from IP address 154.59.124.113 on 09/11/2023 at 11:19

NANO FUTURES

TOPICAL REVIEW

Real-time decoding for fault-tolerant quantum computing: progress, challenges and outlook

F Battistel[1], C Chamberland[2,3], K Johar[4], R W J Overwater[5], F Sebastiano[5], L Skoric[4], Y Ueno[6,7] and M Usman[8,9,*]

[1] Qblox, Delftechpak 22, 2628 XH Delft, The Netherlands
[2] AWS Center for Quantum Computing, Pasadena, CA 91125, United States of America
[3] IQIM, California Institute of Technology, Pasadena, CA 91125, United States of America
[4] Riverlane Ltd, Cambridge, United Kingdom
[5] QuTech and Department of Quantum and Computer Engineering, Delft University of Technology, 2600 GA Delft, The Netherlands
[6] Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan
[7] Department of Computer Engineering, Technical University of Munich, Garching, Germany
[8] School of Physics, The University of Melbourne, Parkville 3010, Australia
[9] Data61, CSIRO, 3168 Clayton Australia
[*] Author to whom any correspondence should be addressed.

E-mail: musman@unimelb.edu.au

Keywords: quantum technology, nanoscience, quantum error correction, nanotechnology, quantum computing

OPEN ACCESS

RECEIVED
24 February 2023

REVISED
14 July 2023

ACCEPTED FOR PUBLICATION
28 July 2023

PUBLISHED
9 August 2023

Original Content from this work may be used under the terms of the Creative Commons Attribution 4.0 licence.

Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

Abstract

Quantum computing is poised to solve practically useful problems which are computationally intractable for classical supercomputers. However, the current generation of quantum computers are limited by errors that may only partially be mitigated by developing higher-quality qubits. Quantum error correction (QEC) will thus be necessary to ensure fault tolerance. QEC protects the logical information by cyclically measuring syndrome information about the errors. An essential part of QEC is the decoder, which uses the syndrome to compute the likely effect of the errors on the logical degrees of freedom and provide a tentative correction. The decoder must be accurate, fast enough to keep pace with the QEC cycle (e.g. on a microsecond timescale for superconducting qubits) and with hard real-time system integration to support logical operations. As such, real-time decoding is essential to realize fault-tolerant quantum computing and to achieve quantum advantage. In this work, we highlight some of the key challenges facing the implementation of real-time decoders while providing a succinct summary of the progress to-date. Furthermore, we lay out our perspective for the future development and provide a possible roadmap for the field of real-time decoding in the next few years. As the quantum hardware is anticipated to scale up, this perspective article will provide a guidance for researchers, focusing on the most pressing issues in real-time decoding and facilitating the development of solutions across quantum, nano and computer science.

1. Introduction

Quantum computing has the potential to offer a revolutionary impact on both fundamental and applied sciences, leading to the solution of many computationally intensive problems which are currently intractable on classical supercomputers. Recent advances in both quantum hardware and software fronts have brought the practical realization of quantum computing possibly within a decade timeline. In particular, quantum technology and nanotechnology have an expanding overlap which is beneficial in complementary directions, both as quantum speedups in computational power can accelerate the development of new quantum materials, as well as devices and materials at the nanoscale are being used as integral elements of quantum computing, sensing and communication [1, 2]. Early demonstrations of quantum advantage [3, 4] are salient examples indicating the dawn of a quantum revolution. However, there remain serious challenges which must be resolved before a practical quantum advantage can be realized. Among these, the effect of noise is

© 2023 The Author(s). Published by IOP Publishing Ltd

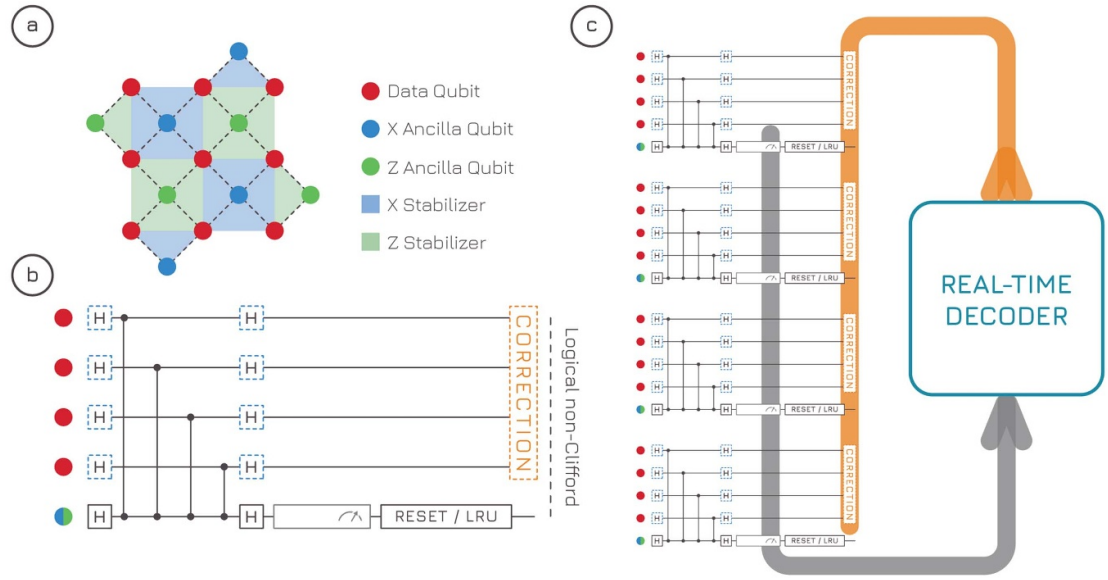

Figure 1. Quantum error correction and the role of real-time decoding. (a) Distance-3 surface code. The logical information is encoded in the data qubits (pink). Ancilla qubits (blue, green) are used to measure error syndromes at every QEC cycle. (b) QEC cycle for one stabilizer. The ancilla qubit performs CZs, is measured and then reset and/or a leakage-reduction unit are applied. The time from the last considered measurement to an impending logical non-Clifford gate is the available time for real-time decoding. (c) The measurement outcomes from all stabilizers (usually including a few previous rounds) are fed to the real-time decoder, which updates the Pauli frame and outputs a correction.

one of the leading issues for the current generation of quantum devices, also known as Noisy Intermediate-Scale Quantum (NISQ) devices [5]. Even with the anticipation of error rates tracking below 1% or less thanks to the development of innovative techniques of fabrication, nanofabrication and control of quantum devices, their cumulative impact on the execution of quantum algorithms requiring deep quantum circuits will be detrimental. Therefore, the development and implementation of sophisticated quantum error correction (QEC) schemes (see [6] for an introduction and [7] for a review) is imperative to achieve practical quantum advantage.

QEC is coming to the forefront as one of the key research areas towards the vision of fault-tolerant quantum computation (FTQC) on faulty qubits in the near to long-term hardware systems. To achieve FTQC, a multidisciplinary effort will be required with expertise ranging from quantum hardware, classical hardware design and nanoscience to error-correction protocols, computer science and control systems. Fundamentally, QEC requires two schemes as illustrated in figure 1. Firstly, a code specifies an encoding in which physical degrees of freedom are combined to redundantly represent 'logical' qubit states [6]. Information about the effects of errors on the logical qubits is obtained by cyclically measuring ancilla qubits yielding an error syndrome. Secondly, a classical decoder coprocessor uses a decoding algorithm to determine an appropriate correction to the logical states based on the syndrome data. If the underlying physical noise rate is below a certain threshold, the fault-tolerance threshold theorem states that QEC can lead to an arbitrary suppression of logical errors using a poly-logarithmic overhead in the number of qubits and computation time [8]. For the surface code the error rate has to be below a threshold of around 1%, even though it should track below 0.1% to make the overhead practically manageable. The correction established by the decoder is needed only for the next-in-line logical non-Clifford gate [9] (for example to determine whether a logical S correction has to be applied during magic-state injection), otherwise it can be tracked by the decoder in the Pauli frame (or leakage frame [10]) until needed. A key parameter for the QEC decoders is the delay introduced by the decoder itself. In fact, as the next logical non-Clifford gate requires a decoder-informed correction, the decoder needs to process the syndrome data at the same rate as it is received or close to it, i.e. in real time, to avoid an exponential slowdown of the computation, known as the *backlog problem* [7]. The incoming data rate varies for different qubit technologies and therefore the expected throughput requirements for real-time decoders range from $\lesssim \mathcal{O}(1)$ μs for a surface code using superconducting transmon qubits [11] to $\lesssim \mathcal{O}(1)$ ms for silicon spin-qubits [12–15] and even beyond $\mathcal{O}(100)$ ms for ion traps [16]. Since the widespread transmons pose the tightest constraints, the value of 1 μs is often used as the benchmark for real-time decoding. However, this could be relaxed to a few μs with the use of parallelization [17, 18] or trading time for space with the use of Auto-T gadgets (see figure 17(b) in [19]), where the use of an extra ancilla per T gate allows to postpone the moment when the output of the

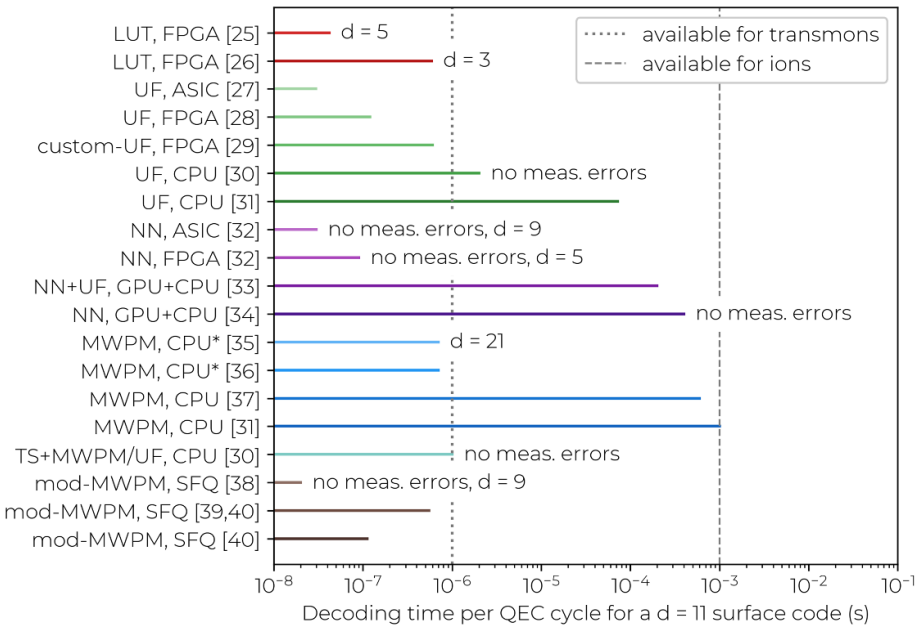

Figure 2. Comparison of decoders: summary of decoders for a distance-11 surface code (unless otherwise specified) [25–40]. For the cases that do not account for measurement errors, the decoding time is given for a $d \times d \times 1$ block, whereas for those that do is given as $1/d$ of the time to decode a $d \times d \times d$ block. We do not divide by d only for the LUT decoders given that the access time of a look-up table is fairly constant. Distance-11 is chosen since it represents a target that is challenging to implement but that at the same time might be attainable within a few years. We include the papers that report the decoding time. If no data point was available for distance-11, we extrapolate it from a fit of the available data if possible (if not, we specify the distance d). Decoder types are marked by similar color shades: look-up table (LUT, red), union find (UF, green), neural network (NN, purple), minimum-weight perfect-matching (MWPM, blue), two-stage (TS, cyan), modified MWPM (brown). The labels on the left report the decoder type and the classical computational resources used in the given paper. SFQ refers to single-flux-quantum logic. The star near CPU refers to the time of the bare computation. By choosing a uniform distance we enable a comparison between decoders, even though the physical error rates vary. Indeed, the speed of especially UF and MWPM depends on the physical error rate, since more detection events require longer to be matched. However, the physical error rates are fairly uniform, being 5%–10% in the case without measurement errors, and 0.1%–0.2% if measurement errors are considered (phenomenological noise). Vertical dotted or dashed lines represent estimates of the available time to compute a correction and feed it back to the qubits on time. For superconducting transmon qubits the estimate (1 μs) is based on [22, 23]. For trapped ions we take 1000 times that (i.e. 1 ms) since the clock speed of ions is much lower than transmons, even though the QEC-cycle time of 200 ms reported in [16] suggests that the bounds for ions might be even much looser.

decoder is needed. Ultimately, the choice of the decoder architecture and its hardware will heavily depend on the target qubit technology. The formalism of QEC codes is oblivious to the underlying physical qubit technology and its length scale [$\mathcal{O}(100\ \mu\text{m})$ for superconducting qubits due to large capacitors but Josephson junctions are $\mathcal{O}(100)$ nm; $\mathcal{O}(100)$ nm for spin qubits/quantum dots; atomic scale with optical control for trapped ions and solid-state qubits like nitrogen-vacancy centers, where in the case of the latter they can control carbon-13 spins within a few nm and they can be separated from one to another by tens of nm to 1 μm]. However, the decoding process is informed by the noise model of the devices and can be made more effective by better characterization of the noise and the physics. Identifying the optimum decoding solution for each technology is an outstanding research question.

So far, the best known experimental demonstrations of real-time decoding have been with ion-trap based qubits [16, 20]. It should be noted that these demonstrations in ion traps have not been based on algorithmic decoding but on a precomputed look-up table, which was sufficient given the small scale of the experiment. On the contrary, experimental demonstrations on superconducting transmon qubits have been based only on offline decoding of errors for a quantum memory [21–24]. However, a few proposals have shown potential to allow for real-time decoding in the near term [25, 26] and up to a fairly-large distance [27–40] (see figure 2 for an overview). Despite these promising results, the experimental implementation of an entirely functional real-time decoding framework fully compatible with a scalable quantum processor still requires substantial development in the next few years.

In this perspective article we provide a succinct summary of the current status with respect to each challenge and a roadmap for the future development, which will serve as a blueprint to guide quantum, nano- and computer-science researchers working in the areas of QEC and real-time decoders. While the real-time decoding field is still in its infancy, we believe that its steady growth makes it timely for a review and perspective to boost a further expansion of the real-time decoding community. Figure 3 plots the flowchart diagram of a real-time decoder framework, highlighting key challenges associated with each

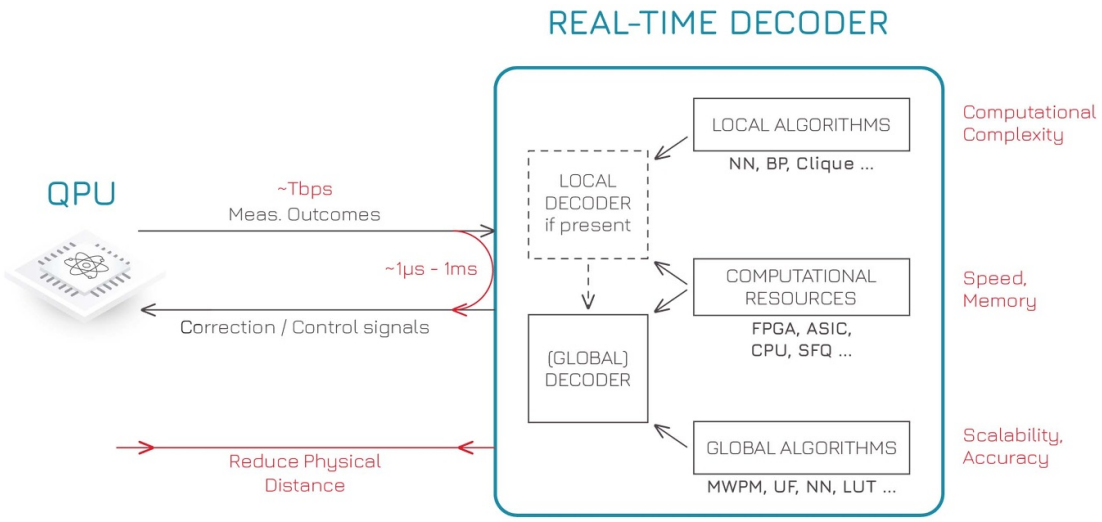

Figure 3. Real-time decoding framework and associated challenges (red). Measurement outcomes from the ancilla qubits in the QPU are passed to a local decoder, if present, otherwise directly to the global decoder. The decoder updates its estimate of the Pauli frame of the qubits (as well as the leakage frame possibly). Physical correction pulses are sent to the QPU when requested because of an impending logical non-Clifford gate (e.g. to apply or not a logical S correction during magic-state injection). A few possible decoding algorithms and computational resources are listed.

component. The decoding algorithm must be scalable with respect to the number of qubits and have a limited computational complexity, while the underlying classical computational resources need to have enough speed and memory, possibly requiring the development of new compute technology (see sections 3.2, 3.3 and 4.3). Not only the decoding algorithm itself has to run fast, but the deployment of measurement outcomes to the decoding module and feedback in the quantum control stack must keep a fast pace, even though this requirement could be mitigated using selective teleportation [41] at the cost of slowing down the quantum computation and of a higher error rate per logical gate unless an increased code distance is employed. Hence, the communication latency, bandwidth and location of the decoding module are crucial. Furthermore, the decoder must be applicable not just to quantum memories but also to logical operations such as lattice surgery. We first discuss the requirements for real-time decoding and hard system integration in section 2. We then cover the challenges and perspectives related to the decoding algorithms in section 3, and to the computational resources, their location and latency in section 4. We conclude with an outlook on the next few years in section 5.

2. Requirements for (hard) real-time decoding

Different platforms are going to require custom-made decoding algorithms to address individual requirements such as decoding rate, noise profiles, and device topologies, as well as cost. For superconducting and photonic qubits, a high operational rate implies that the decoding throughput and latency are paramount [21, 42]. Throughput (defined as the time it takes the decoder coprocessor to perform the decoding once it receives the measurement outcomes) is even more important than latency (defined as the time it takes to send the measurement outcomes to the decoder coprocessor) [41, 43]. In recent superconducting-qubit experiments, QEC rounds were performed every $\sim$1 μs [21–24], leading to an estimate that a utility-scale quantum computer would generate a few tens of Mbps of syndrome data per logical qubit, for a total of many Tbps [44]. For each logical qubit, the decoder needs to process that data at the acquisition rate or close to it to avoid an exponential slowdown due to an ever-growing data backlog.

In the case of ion-trap and neutral-atom quantum computing, the QEC-cycle times are usually in excess of many milliseconds [16, 20, 45, 46], leaving ample time for real-time decoding. We note that relatively slow operations in these platforms are compensated by high connectivity and long coherence times. These features allow to choose across a wide landscape of QEC codes and their decoders, beyond the surface code. For example, color codes have been studied for this purpose due to lower qubit overheads and the ease of applying fault-tolerant operations [16, 47]. While polynomially-efficient decoders for color codes have been proposed [48, 49], further work is required in order to improve performance in realistic conditions in the presence of circuit-level noise [50]. Furthermore, the increased connectivity can be exploited to implement quantum low-density parity-check codes (LDPCs) which outperform surface and color codes in terms of protection against errors and the ratio between logical and physical qubits [51–53]. However, quantum

LDPC codes may require lower physical error rates due to lower thresholds than surface codes. While classical LDPC codes have fast decoders, these do not generalize well to their quantum counterparts. With a few algorithms starting to investigate this space [54, 55], we expect that over the coming years this gap is going to be closed, bringing quantum LDPC codes closer to practical applications.

The first experiments with error-corrected silicon [13] and solid-state [56] qubits are also beginning to take place. We expect that the encoding and decoding requirements for these platforms are going to be defined over the coming years as the systems mature.

2.1. Hard versus soft real-time

The feedback loop of altering the direction of the program depending on the measured error syndromes can only be possible if the decoder is active and running during the execution of the quantum program. In addition to decoding being real-time, it is also important to establish the subtlety of *hard* versus *soft* real-time. In a hard real-time system [57], each computational piece must complete within a deterministic and precise amount of time. The sequencing of the individual events must be time-bounded and it is often statically determined by performing worst-case execution times analysis (WCETs) [58]. On the contrary, soft real-time systems are those that can tolerate some variation in how long the program takes to execute. Performing WCET analysis is generally extremely tedious, complex and highly sub-optimal for systems that have a probabilistic behavior. Even if the analysis can be performed, allocating fixed time to individual tasks and scheduling them statically could lead to overallocation and therefore lead to performance degradation. In the case of quantum computing, the lost time will also affect qubit coherence.

A fault-tolerant quantum stack will likely need every piece to precisely execute within a fixed timing budget. These pieces range from the pulses that are generated to control the QPU, to repeatedly reading out intermediate states and correcting the direction of future pulses based on the outcome of decoding errors, to magic-state distillation [59]. As such, a fault-tolerant quantum computer seems to constitute a hard real-time system, even though some softness may have to be allowed for a highly non-deterministic process such as magic-state distillation, while QEC cycles are repeated in a hard real-time fashion. Hard real-time decoding requires tighter system integration to remove any unnecessary communication overhead and latencies. This sometimes implies that specific customizations will need to be made to every part of the stack, which increases development time and costs. Furthermore, tight integration will have to be conjugated with scalability of the system. As explained above, the determination of an optimal schedule for a hard real-time system is tedious and complex. Therefore, high-level execution models for various parts of the stack will need to be put together to determine its schedule and time allocation. The use of WCET tools and methodologies will help in reducing or removing any slack in the execution and help improve the performance of the quantum program while reducing the chances of decohering.

3. Decoding algorithms and hardware implications

In principle, every code can be decoded with a maximum-likelihood decoder by assigning a probability to each set of errors that would generate the given syndrome. These can then be separated into equivalence classes based on their effect on the logical degrees of freedom. The class with the greater total likelihood is then taken as the decoding result. While this provides the optimal logical fidelity, in general it has been shown to be a $\sharp P$-complete problem [60]. Tensor network decoders reduce the problem of optimal decoding to contractions of tensor networks that can be made more efficient by arbitrarily-precise approximate contractions [61–63], but nevertheless require substantial computational resources. Therefore, as much as designing qubit-efficient and robust codes, the challenge of QEC consists of finding less-accurate decoders that achieve the required trade-offs between logical fidelity and classical computing requirements.

One general solution is to pre-compute the corrections for a given syndrome in a look-up table, such as LILLIPUT [25]. While extremely fast and relatively easy to implement, the memory requirements of look-up tables scale exponentially, making them impractical but for the smallest demonstrations. A more scalable approach is to design fast algorithms that are tailored to a smaller family of codes and that can be efficiently implemented. For the widespread topological codes like the surface code, these include minimum-weight perfect matching (MWPM) [64, 65], union find (UF) [31, 66], renormalization group [67] and belief propagation [54, 68, 69] decoders (the latter cannot be used as a standalone decoder but can supplement e.g. MWPM or UF for improved performance). While significant engineering effort is required to realize these implementations, recent improvements in parallelization of real-time decoders [17, 18] warrant optimism that scalable decoding hardware is going to be demonstrated in the forthcoming years.

To date, all hardware demonstrations of QEC have fallen short of unambiguously achieving logical lifetimes better than physical lifetimes, though several come close [16, 20, 21, 23, 24, 56, 70]. As near-term hardware is likely going to suffer considerable error rates, to unequivocally demonstrate this next big

technological milestone, accurate decoders that are closely tuned to the physical noise are going to be necessary. Maintaining high fidelity in the presence of correlated errors [71], leakage [22, 72, 73], erasure [74], device inhomogeneities [23], and even rare high-energy events [75] necessitates modifications to standard decoders. This also means that any decoding hardware is going to have to be adaptable to the characteristics of a particular device, and tuned as the noise profile changes with time.

3.1. Minimum weight perfect matching (MWPM) and union find (UF)

Topological surface codes have been the focus of QEC research as a promising architecture for scalable quantum computation [64, 76] due to a low connectivity requirement and high tolerance to errors. Moreover, there is an abundance of literature on how to efficiently perform FTQC in a surface-code architecture [19, 77]. Local fault mechanisms in surface codes can be decomposed into errors that trigger either a pair of changes or a single change in the syndrome, referred to as defects or detection events. This makes them suitable for decoding with an MWPM algorithm [64, 78]. MWPM finds the smallest error consistent with the syndrome in polynomial time using the blossom algorithm [65, 79]. While MWPM does not take into account the degeneracy present in QEC codes [80], it nevertheless achieves good performance even against circuit-level noise. Fast implementations of MWPM have been recently released, namely fusion-blossom [35] and sparse-blossom (PyMatching v2) [36, 81, 82]. Both exploit the sparse decoding graph to solve MWPM more efficiently. Sparse-blossom uses further optimization strategies to grow alternating trees and fusion-blossom allows parallelization over many spacetime chunks that are then fused together. The two approaches may be combined together to achieve further speedup [82].

Another algorithm is UF, which has close-to-linear runtime [31, 66], a distributed FPGA realization [28] and a proposed micro-architecture [27]. Furthermore, the three major steps in the algorithm could be handled by three different hardware units, some of which may be faster than the others. In a quantum computer with many logical qubits, a proposal [27] to be demonstrated in practice is to have multiple copies of the slower units but share the faster units across many decoders to reduce hardware overhead. This approach could also be applied to other types of decoders.

3.2. Neural-network decoders

Besides being fast, a decoder must also be accurate, scalable with respect to the number of physical qubits, able to tackle complex noise models and compatible with lattice surgery. In those respects, neural-network decoders are promising candidates for real-time decoding, thanks to their constant inference time, the inherent ability to learn any error model, the scalability to large code distances [34, 43, 83] and compatibility with lattice surgery [34, 43, 84]. Nevertheless, several challenges must be overcome before seeing neural-network hardware decoders in a practical quantum computer, such as finding the optimal neural-network architecture able to address complex error models, and quantifying the trade-offs between the hardware costs and the decoder performance. Furthermore, as of yet, the performance against time-like errors during lattice surgery (see section 3.5) when using pre-decoders (see section 3.4), such as neural networks, has not been analyzed. The adaptability to change in surface-code patch shapes through time has to be demonstrated as well.

The training strategy is also a crucial choice for neural-network decoders. Supervised learning is the straightforward choice when the training data is synthetically generated using a software-implemented error model as both the decoder input and its expected output for the full training dataset are known, e.g. [32–34, 83, 84]. Nevertheless, reinforcement learning has also shown promise [85, 86], and is particularly interesting when training on data from real quantum hardware, for which only the syndrome is known but the errors on the data qubit are not. Furthermore, reinforcement learning can be applied to automatically discover new QEC schemes and their corresponding decoders by exploiting the feature of a specific quantum platform to gain efficiency [87–89] and also leading to a recent real-time implementation for superconducting qubits [90].

Hardware-based decoder implementations have been proposed as alternative to software decoders to keep the (expected) decoding times below the 1 μs target, e.g. employing FPGAs [25, 26, 28, 29, 32, 43], ASICs [27, 32] or single-flux-quantum logic (SFQ) [38–40, 84]. In the case of neural-network-based decoders, implementing them in hardware can exploit the advances in the field of hardware accelerators for neural networks for classical applications, such as image and voice recognition. These hardware-accelerator ASICs have been recently the subject of a fast-evolving research direction, where the main emphasis lies on improving the underlying hardware fabric while also exploiting the sparsity of the network [91], e.g. by co-optimizing the quantization in the digital signal representation and pruning [92, 93]. In terms of hardware costs, state-of-the-art fully-digital accelerators currently achieve power efficiency up to 1000 TOPS/W with area efficiency of 400 TOPS/mm^2 in integrated-circuit implementations [94]. Since the main limitations with these architectures are the bandwidth and power bottleneck in fetching and

transferring data between the memory and the computational unit, in-memory computing has been proposed for a better scaling in energy, bandwidth and delay [95]. Additional improvement in efficiency for in-memory-computing neural networks can in principle be achieved by moving from fully-digital solutions to analog and mixed-signal ones, which already demonstrate up to 500 TOPS/W and are quickly closing the gap with the digital solutions [96]. However, these analog solutions are still lagging behind in area efficiency, although non-CMOS (nano)technologies, such as flash- and memristor-based architectures, promise significant enhancements in that direction. Coming years will be crucial to demonstrate whether such progress can be applied also to neural-network decoders for QEC.

On the way to 2025, quantum-computing systems are quickly growing to a scale where experiments with useful QEC schemes can be performed. The hardware optimization will then become more relevant and several questions must be addressed in the next few years towards the goal of a real-time neural-network-based decoder. The best network architecture must be identified: for instance, while convolutional neural networks (CNN) seem amenable to scalability and lattice surgery, which architecture can best address measurement errors: a 3D CNN or a recurrent 2D CNN? Following the examples set by state-of-the-art classical neural-network hardware, what are the trade-offs between decoding performance (accuracy, delay) and hardware costs (power, area)? Can neural-network decoders be efficiently adopted for emerging QEC schemes? Which role can reinforcement learning play? Which role can nanotechnologies play? The resulting tight constraints in area and power will push towards advanced techniques in neural-network hardware, such as analog computing and in-memory computing, possibly in a cryo-CMOS chip hosting qubits, interface electronics and QEC decoders as well (see section 4.3 for more about this).

3.3. SFQ-based decoders

Single-flux-quantum (SFQ) logic is a digital circuit composed of superconductor devices, and it has the potential for utilization in next-generation computers because of its ultra-fast and low-power performance compared to CMOS [97, 98]. Information processing in SFQ circuits is performed with magnetic flux quanta stored in superconductor rings. The presence or absence of an SFQ in the ring represents a logical '1' or '0', respectively. The pulse-driven nature of information processing in SFQ circuits enables both their fast switching ($\sim 10^{-12}$ s) and low-energy consumption ($\sim 10^{-19}$ J per switching). Hence, with this technology it is feasible to increase the device clock frequency to $\mathcal{O}(10\text{–}100)$ GHz while keeping its power consumption low enough to operate in a cryogenic environment. On the downside, SFQ-based decoders need to be designed and developed by hand (called custom design). SPICE simulations are then needed to prove that the output is as expected for given input conditions. To scale this development methodology to complex algorithms, SFQ logic needs to be fully supported by commercial EDA tools (Electronic Design Automation) that do not exist yet.

Several SFQ-based peripherals have been proposed for fundamental operations of quantum computers, such as measurement and manipulation of superconducting qubits [99–101]. In addition, SFQ circuits have been proposed for decoding QEC codes [38–40, 84]. However, memory-intensive decoding algorithms, such as MWPM or UF, are unsuitable for SFQ circuits because a large amount of RAM is expensive in SFQ implementations. Thus, memory-efficient decoding algorithms are required for SFQ circuit execution.

Holmes *et al* [38] proposed a new decoding algorithm for the surface code, named AQEC, and designed the first SFQ chip implementing such algorithm, while the concept of using SFQ circuits for QEC had already been proposed [102]. This decoder consists of multiple units corresponding to each data and ancilla qubit to detect and correct errors by propagating simple signals between the units in a distributed processing scheme. However, this decoder focuses only on correcting Pauli errors on data qubits and is incapable of addressing measurement errors.

Ueno *et al* [39] proposed the QECOOL algorithm, which is an extension of AQEC to deal with measurement errors, designing an SFQ-based chip which achieves even lower power consumption than the AQEC decoder. Furthermore, QECOOL is an online-QEC or sliding-window decoder, which in general seems more appropriate to keep the decoding time bounded for realistic QEC, where new syndrome data comes in as input at every QEC cycle. Ueno *et al* [40] proposed the QULATIS decoder by extending QECOOL to deal with lattice surgery (see section 3.5) and [84] proposed the NEO-QEC decoder by combining a binarized neural-network-based SFQ decoder in a two-stage process with QECOOL/QULATIS.

3.4. Two-stage decoders

Many proposals [30, 33, 34, 43, 69, 84, 103, 104] have been put forward where decoding occurs in two stages, with a first, computationally-simple local decoder and a second, more-complex global decoder. The first stage can either be a pre-processing stage whose outcome is passed anyway to the global decoder [69], or it can try to correct all errors (if the pattern is 'simple' enough) and call the global decoder only when it fails. In

Table 1. Comparison of Cryo-CMOS and SFQ decoders. The area, power consumption and throughput are per distance-9 logical qubit.

	NN [32]	AQEC [38]	QECOOL [39]	QULATIS [40]	NEO-QEC [84]	Clique [103]
Platform	CMOS	SFQ	SFQ	SFQ	SFQ	SFQ
Meas. errors			✓	✓	✓	✓
Lattice surgery				✓	✓	
Area (mm^2)	10	369	183	16.4	N/A	14.4
Power consumpt. (μW)	20 000	3780	400.3	417.4	614.9	99
Throughput Max/Avg. (ns)	28	19.2/3.8	364/9.15	82/2.12	N/A	0.24

the latter case it can provide either the original [30] or a pre-processed syndrome [33, 34, 43, 84, 103, 104] to the global decoder.

The purpose of two-stage decoding can be to enhance the accuracy and/or speed of the global decoder, in which case the two stages can be co-located. For example, the techniques in [43] improve the speed of MWPM for pure memory by 10^6 for circuit-level noise. However it is required that such local decoders are implemented in times less than $\mathcal{O}(1)$ μs (for superconducting-qubit architectures) to be useful. In general, the main purpose of two-stage decoding is to reduce bandwidth on the cryostat I/O, as well as the power consumption inside the cryostat, by placing the first stage at cryogenic temperature and the second stage at room temperature. In particular, the first-stage decoder must be computationally-inexpensive so that its power requirements are minimal. Provided low-enough physical error rates, low-weight and sparse error configurations will be the most common and thus can be handled by the local decoder alone, without having to often send signals to the top of the cryostat. For example, the Clique decoder [103] has a hierarchical structure where a simple SFQ decoder in the 4 K environment and a complex decoder in the room-temperature environment are combined. In this implementation, the SFQ part decodes only the case where all weight-1 errors are isolated, and the more difficult error signatures are sent to the second-stage decoder. Furthermore, the SFQ part is simple enough to achieve a much lower resource overhead than global SFQ-based decoders (see table 1).

3.5. Lattice surgery

Several systems are now at the size where implementing multi-qubit logical operations in a fault-tolerant manner is becoming feasible. In surface codes, these are performed by merging and splitting code patches in a scheme known as lattice surgery [19, 40, 43, 77]. Lattice-surgery decoders take the full syndrome information into account over a given syndrome history window. Corrections are applied in a manner identical to what would be done with a decoder used for pure memory with a few exceptions. First, to correctly address boundary effects obtained by merging and splitting surface code patches, logical representatives must be defined with care, since corrections can always be viewed as flipping the sign of a set of logical operators. As an example, when performing an $X \otimes X$ measurement via lattice surgery, ancilla qubits in the routing space region will initially be prepared in Z basis eigenstates. As such, logical Z representatives for the surface code patches can be taken to have support on qubits in the routing space before merging the patches and after the split. Such re-definitions of the logical operators avoid ambiguities due to boundary effects. Second, certain lattice operations require measuring extended rectangles (such as weight-4 checks which are longer range or require more ancilla qubits) and potentially higher weight stabilizers (for instance if twists are used to measure Y). In such cases, the information from the additional ancilla qubits must be used with care to reconstruct the stabilizer measurement outcomes. With the above points, error correction can then proceed exactly the same way as what would be done with pure memory. For instance, if a MWPM decoder is used, matching would be performed over the syndrome history window, with each edge in the graph specifying which logical is flipped. The same holds for any other graph-based decoder, such as UF, as long as logical representatives are correctly specified through the full syndrome history of the computation.

While performing lattice-surgery operations, the code patches are changing in size and shape as they move, merge, and split. This means that the decoding hardware is going to have to be reconfigurable on-the-fly, working in a low-latency feedback loop with the control system to implement complex fault-tolerant circuits. This process also often requires decoding excessively large codes during the merge stage. Therefore, the decoding will likely have to be parallelized across space, as well as time, in order to prevent a slowdown of the logical clock speed [17, 18].

High thresholds for space-like as well as time-like errors when performing lattice surgery still need to be demonstrated. Recall that time-like errors arise as an error in the parity of the multi-qubit Pauli measurement outcome when performing lattice surgery. Temporal encoding of lattice surgery (TELS) [59, 77] allows one to correct logical time-like failures which occur during lattice surgery using measurement outcomes from a redundant amount of lattice-surgery measurements. The decoding of TELS uses classical

error correcting codes. This could be used to alleviate some of the decoding requirements, potentially in conjunction with dedicated control hardware for TELS.

4. Computational resources

4.1. CPUs, FPGAs, ASICs

The challenges faced around computational resources depend on the choice of the platform. Here we consider all three popular choices (CPUs, FPGAs, and ASICs) and highlight the relevant progress and challenges.

CPUs. The first challenge facing the execution of decoding algorithms on CPUs is that these are currently benchmarked on desktop/server class CPUs. The typical power consumed by a desktop computer is in the order of several watts. While it would be possible to use multiple threads on the machine to decode in parallel, there are significant challenges in scaling this to several thousands of logical qubits. The problem gets exacerbated if the control electronics moves into cryogenic environments, which have strict power budgets in the order of 1 mW per qubit at 4 K. On the contrary, there are CPUs that are already part of the room-temperature quantum control stack. These CPUs are on FPGA boards which are used to control the sequences of pulses to the QPU. In most cases these CPUs are underutilized and therefore could be good candidates to use for real-time decoding. However, it must be noted that these CPUs are generally several generations old, are clocked at lower frequencies and therefore are at least an order of magnitude lower in performance compared to a typical laptop/server grade CPU.

Another challenge that CPUs would face is around *hard* real-time decoding (see section 2.1). Typical usage of CPUs in the stack makes the system non-deterministic. CPUs are designed to run operating systems with memory-management units and address translation. They also have multiple levels of caching from level 1 to off-chip memory. The combination of using a non-real-time operating system such as Linux, memory management, memory hierarchies and out-of-order execution makes the system non-deterministic. This implies that it is almost impossible to determine the worst-case execution time of any computation (such as decoding). As explained in section 2.1, for hard real-time systems having a bounded worst-case execution time is vital. The above problems can be worked around by using CPUs that are specifically designed for hard real-time execution, running bare metal without any operating system and avoiding the use of caches. However, this will negatively impact the performance of the system since real-time CPUs typically cater to a lower-performance category. Furthermore, in order to operate a tight loop between control systems and CPUs for decoding, it would be necessary to transport the data as close to the CPU caches as possible. A good number of high-performance CPUs have accelerated coherency ports that allow pre-loading of caches, but there might be additional coherency-port overheads that lead to slowdown in the execution of the decoding algorithm.

FPGAs. There have been several proposals of building decoders onto an FPGA [25, 26, 28, 29, 32, 43]. The primary advantage of using an FPGA decoder in the near term is to allow for its straightforward integration with existing control systems for low-level pulse control, which are also typically built on FPGAs. The vast majority of the control systems already have direct access to readout of qubits at the FPGA level and therefore that information can be readily transported to the decoding FPGA via an appropriate bus protocol. This protocol has to be compatible with a distributed control-stack architecture and has to have extremely low latency, taking at most a couple hundreds of nanoseconds to not overburden the decoding budget. Another advantage is that FPGAs have relatively short development cycles for developing algorithms and are easily reconfigurable. It is also possible to get execution frequency in the order of 100 s MHz, even though this is still lower than with CPUs or ASICs in principle [105]. Furthermore, FPGAs have recently demonstrated the execution of algorithms specifically designed for parallel hardware for a fairly large code distance [28]. On the downside, high-end FPGAs used to control qubits are generally expensive, which would make it cost-prohibitive to scale to millions of physical qubits.

ASICs. The landscape around the development of physical qubits and QEC is constantly evolving. It is expected to take several years before industry-wide consensus is reached on the precise requirements of a truly useful real-time decoder, including choices of decoding algorithms, QEC codes, interfaces with the control system, etc. While the industry moves forward and active research converges on practical solutions, the use of ASICs would be premature due to the complex and cost-prohibitive nature of ASIC development. This constitutes the primary obstacle for ASIC adoption for decoding systems, temporarily outweighing the benefits of low per-part costs, drastically reduced power consumption, high execution frequencies in the order of 1–2 GHz and very tight integration for scaling. However, if the control electronics moves into the cryogenic environment, these advantages become natural arguments for complete decoder and control-electronics integration using ASICs, and a path forward to large-scale systems.

4.2. Location of the decoder module

The location of the decoder module and its subsequent latency and bandwidth will depend on the overall system stack. The system stack as it stands today is primarily built to improve the quality of the qubits, rapidly prototype improvements to the control systems and updates to the software stack. In the NISQ era, it is likely that the decoder will be integrated in different parts of the stack (see section 3.4). However, this picture is likely to be very different in the fault-tolerant era. A parallel is often drawn between classical and quantum computers: in fact, one of the highlights of classical computers has been the progress through Moore's law and the very large scale integration. The two together have contributed to significant improvements in the computational power, large reduction in size, energy requirements and most importantly bringing the cost down, making the technology accessible to almost everyone. A similar evolution is also envisaged for quantum computers and it is therefore natural that large-scale integration will enable quantum computers to scale. Given this paradigm shift, the ideal location for the decoder would be alongside the control electronics, integrated with very low latency in the order of 10 s of nanoseconds with fast decoder execution and fast feedback into control for fault-tolerant logical operations.

4.3. Opportunities at cryogenic temperature

In many quantum-computing platforms qubits are operated in a cryogenic environment, such as inside of a cryostat, to reduce noise. Superconducting quantum computers require many high-frequency coaxial cables connecting qubits and peripherals for manipulation, measurement, and QEC to a room-temperature environment. The heat inflow and the occupied space by cabling in the cryostat seriously limit the scalability of superconducting quantum computers. The same problem applies to other types of solid-state qubits that operate in a cryogenic environment, e.g. spin qubits in donor silicon and MOS-type double quantum dots. To scale up these platforms, controlling qubits in a cryogenic environment is mandatory in the future [102]. Co-integration of qubits and control electronics would circumvent relevant bottlenecks in scalability, such as the cabling and the communication overhead between qubits and their electronic interface [106]. Unfortunately, co-locating the electronics may be limited in cryogenic qubit platforms, where for typical dilution refrigerators the cooling power is $\mathcal{O}(1)$ mW below 100 mK and it is several watts at 1-4 K. In a system with thousands of qubits, even the latter leaves at most a few mW of cooling power per qubit. Note that placing qubits and electronics/decoding at two different temperature stages, for example at 20 mK and 4 K respectively, does not really solve the scaling issue related to cabling since the main bottleneck remains across those two stages. Furthermore, so far locating the electronics at 20 mK on top of the qubits causes quasi-particle poisoning, which requires new technological developments to avoid impacting the coherence time of the qubits [107].

In terms of hardware costs for decoding, power consumption and die area are the most relevant metrics (see table 1), especially when operating the decoder very close to the quantum processor, i.e. at the same operating temperature or even on the same die or package. We discuss challenges and opportunities for complementary metal oxide semiconductor (CMOS) and SFQ.

Cryo-CMOS. Cryogenic CMOS (cryo-CMOS) technologies for peripherals of quantum computers in a cryostat have been actively studied recently [108–110]. For example, [108] proposed a cryo-CMOS qubit controller fabricated in Intel 22-nm FinFET technology, and its power consumption is several hundred mW. However, as superconducting transmon qubits operate below 100 mK, dilution refrigerators offer insufficient cooling power for these electronic circuits at that temperature (as discussed above), although recent efforts propose an electronic interface for transmons operating at 4 K [111, 112]. Alternatively, semiconductor spin qubits can operate at temperatures above 1 K [113], where the power budget is enough for system on a chip (SoCs) [108–110], making them in principle compatible with standard CMOS processing.

SFQ. Although an SFQ decoder can in principle operate at the millikelvin stage, all of the SFQ decoders proposed so far are assumed to operate in a 4 K environment since the existing SFQ process technologies are designed for that. In particular, these technologies do not allow to meet the required power consumption at the millikelvin layer, as even the lowest-power SFQ-based decoder, namely the Clique decoder [103], uses 99 μW for a distance-9 logical qubit (see table 1). While satisfactory in the near-term, this is insufficient in a large FTQC architecture with thousands of logical qubits. To realize such architecture using SFQ circuits, it is thus essential to reduce their power consumption.

One solution is a new SFQ process technology targeting a millikelvin environment. If targeting such environments, it is theoretically possible to reduce the switching energy of an SFQ circuit by two orders of magnitude compared to the 4 K environment case. However, careful parameter selection according to the fabrication size of Josephson junctions and to the target operating frequency is required. Recently, the adiabatic quantum flux parametron (AQFP) [114] and the reversible QFP (RQFP) [115] using AQFP have been proposed as ultra-low-power SFQ circuits. The RFQP logic achieves its low power consumption by using logically and physically reversible logic gates so that the entropy of information does not change during

operation. Since the switching energy of AQFP and RFQP is more than two orders of magnitude less than that of typical SFQ circuits, they are expected to be helpful in building a large-scale FTQC architecture in the millikelvin layer.

5. Outlook

Many quantum-computing platforms are now getting to the size and fidelities required for QEC demonstrations. While a number of challenges lay ahead, the dream of FTQC is closer than ever. We expect to first see demonstrations for a single logical qubit, likely in FPGA and with sliding-window decoding, and then for logical two-qubit operations. The earliest demonstrations will still make use of look-up table decoders, but these will be quickly supplanted by scalable algorithmic decoding. The initial code distance will be 3 but will soon scale to 5 and beyond depending on experimental progress. By 2025 we expect exciting experimental and theoretical results that address many of these challenges: the development of new decoding algorithms for topological, color and LDPC codes, improved suppression of realistic noise, demonstrations of decoders tightly integrated with control systems, and real-time decoding of logical operations.

We hope that by 2025 the community will have undeniably demonstrated that utility-scale FTQC is achievable in practice, and a clear path will be paved to realize such systems in the second half of the decade. All of this will likely be demonstrated on a variety of qubit technologies, each of which is likely to face different challenges throughout the stack. For real-time decoding this means that different solutions will be explored, from running on CPUs to tighter integration on FPGAs or ASICs. Time will tell if FPGA solutions can offer scalability and speed comparable with ASIC solutions, eventually co-integrated with qubit control and readout electronics. Following the current trend towards more compact large-scale computing systems, the electronics, including the decoder, will eventually operate close to the qubits (at cryogenic temperature for the solid-state qubit platforms). We may also see some experiments performed with ASIC-based decoders within a cryogenic environment, which may exploit developments in nanotechnologies as well. While ASICs may not be essential to the experimental success as of 2025 yet, they could be used to showcase the benefits of reducing the overall power consumption of the system and pave the way forward for the fault-tolerant era of highly integrated, scalable and cost-effective quantum systems.

Data availability statement

This is a review/perspective article and all data is already published. The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgments

Qblox is supported by the European Commission, Grant Agreement ID: 969201. Y U is supported by Grant-in-Aid for JSPS Research Fellow Grant Number JP21J10882.

ORCID iDs

F Battistel https://orcid.org/0000-0003-4800-2518
C Chamberland https://orcid.org/0000-0003-3239-5783
R W J Overwater https://orcid.org/0000-0002-8101-8434
F Sebastiano https://orcid.org/0000-0002-8489-9409
M Usman https://orcid.org/0000-0003-3476-2348

References

[1] Laucht A *et al* 2021 Roadmap on quantum nanotechnologies *Nanotechnology* **32** 162003
[2] Finocchio G *et al* 2023 Roadmap for unconventional computing with nanotechnology (arXiv:2301.06727)
[3] Arute F *et al* 2019 Quantum supremacy using a programmable superconducting processor *Nature* **574** 505–10
[4] Zhong H-S *et al* 2020 Quantum computational advantage using photons *Science* **370** 1460–3
[5] Preskill J 2018 Quantum computing in theNISQ era and beyond *Quantum* **2** 79
[6] Gottesman D 1997 Stabilizer codes and quantum error correction (arXiv:quant-ph/9705052)
[7] Terhal B M 2015 Quantum error correction for quantum memories *Rev. Mod. Phys.* **87** 307–46
[8] Kitaev A Y 2003 Fault-tolerant quantum computation by anyons *Ann. Phys., NY* **303** 2–30
[9] Riesebos L, Fu X, Varsamopoulos S, Almudever C G and Bertels K 2017 Pauli frames for quantum computer architectures *Proc. 54th Annual Design Automation Conf. 2017 (Association for Computing Machinery) (DAC '17)*
[10] Suchara M, Cross A W and Gambetta J M 2015 Leakage suppression in the toric code *Quantum Info. Comput.* **15** 997–1016

[11] van Dijk J P G, Charbon E and Sebastiano F 2019 The electronic interface for quantum processors *Microprocess. Microsyst.* **66** 90–101
[12] O'Gorman J, Nickerson N H, Ross P, Morton J J L and Benjamin S C 2016 A silicon-based surface code quantum computer *npj Quantum Inf.* **2** 15019
[13] Takeda K, Noiri A, Nakajima T, Kobayashi T and Tarucha S 2022 Quantum error correction with silicon spin qubits *Nature* **608** 682–6
[14] Barthel C, Kjærgaard M, Medford J, Stopa M, Marcus C M, Hanson M P and Gossard A C 2010 Fast sensing of double-dot charge arrangement and spin state with a radio-frequency sensor quantum dot *Phys. Rev.* B **81** 161308
[15] Vandersypen L M K, Bluhm H, Clarke J S, Dzurak A S, Ishihara R, Morello A, Reilly D J, Schreiber L R and Veldhorst M 2017 Interfacing spin qubits in quantum dots and donors-hot, dense and coherent *npj Quantum Inf.* **3** 1–10
[16] Ryan-Anderson C *et al* 2021 Realization of real-time fault-tolerant quantum error correction *Phys. Rev.* X **11** 041058
[17] Skoric L, Browne D E, Barnes K M, Gillespie N I and Campbell E T 2022 Parallel window decoding enables scalable fault tolerant quantum computation (arXiv:2209.08552)
[18] Tan X, Zhang F, Chao R, Shi Y and Chen J 2022 Scalable surface code decoders with parallelization in time (arXiv:2209.09219)
[19] Litinski D 2019 A Game of surface codes: large-scale quantum computing with lattice surgery *Quantum* **3** 128
[20] Egan L *et al* 2021 Fault-tolerant control of an error-corrected qubit *Nature* **598** 281–6
[21] Egan L, Debroy D M, Noel C, Risinger A, Zhu D, Biswas D, Newman M, Muyuan Li, Brown K R, Cetina M and Monroe C 2022 Suppressing quantum errors by scaling a surface code logical qubit (arXiv:2207.06431)
[22] Marques J F *et al* 2021 Logical-qubit operations in an error-detecting surface code *Nat. Phys.* **18** 80–86
[23] Krinner S *et al* 2022 Realizing repeated quantum error correction in a distance-three surface code *Nature* **605** 669–74
[24] Sundaresan N, Yoder T J, Kim Y, Muyuan Li, Chen E H, Harper G, Thorbeck T, Cross A W, Córcoles A D and M Takita Matching and maximum likelihood decoding of a multi-round subsystem quantum error correction experiment (arXiv:2203.07205)
[25] Das P, Locharla A and Jones C 2022 LILLIPUT: a lightweight low-latency lookup-table decoder for near-term quantum error correction *Proc. 27th ACM Int. Conf. on Architectural Support for Programming Languages and Operating Systems, ASPLOS 2022 (Association for Computing Machinery) (New York, USA)* pp 541–53
[26] Ristè D, Govia L C G, Donovan B, Fallek S D, Kalfus W D, Brink M, Bronn N T and Ohki T A 2019 Real-time decoding of stabilizer measurements in a bit-flip code (arXiv:1911.12280)
[27] Das P, Christopher A Pattison S M, Carmean D M, Svore K M, Qureshi M and Delfosse N 2022 AFS: accurate ,fast and scalable error-decoding for fault-tolerant quantum computers *2022 IEEE Int. Symp. On High-Performance Computer Architecture (HPCA)* pp 259–73
[28] Liyanage N, Yue W, Deters A and Zhong L 2023 Scalable quantum error correction for surface codes using FPGA (arXiv:2301.08419)
[29] Riverlane 2022 Deltaflow *Decode Technical White paper* (available at: www.riverlane.com/media/nz2dvqmi/deltaflow_decode_technical_white_paper_september_2022.pdf)
[30] Delfosse N 2020 Hierarchical decoding to reduce hardware requirements for quantum computing (arXiv:2001.11427)
[31] Huang S, Newman M and Brown K R 2020 Fault-tolerant weighted union-find decoding on the toric code *Phys. Rev.* A **102** 012419
[32] Overwater R W J, Babaie M and Sebastiano F 2022 Neural-network decoders for quantum error correction using surface codes: a space exploration of the hardware cost-performance tradeoffs *IEEE Trans. Quantum Eng.* **3** 1–19
[33] Meinerz K, Park C-Y and Trebst S 2022 Scalable neural decoder for topological surface codes *Phys. Rev. Lett.* **128** 080505
[34] Gicev S, Hollenberg L C L and Usman M 2023 A scalable and fast artificial neural network syndrome decoder for surface codes *Quantum* **7** 1058
[35] Wu Y 2022 Fusion blossom (available at: https://github.com/yuewuo/fusion-blossom)
[36] Higgott O and Gidney C 2023 Sparse blossom: correcting a million errors per core second with minimum-weight matching (arXiv:2303.15933 [quant-ph])
[37] Fowler A G, Whiteside A C and Hollenberg L C L 2012 Towards practical classical processing for the surface code: timing analysis *Phys. Rev.* A **86** 042313
[38] Holmes A, Reza Jokar M, Pasandi G, Ding Y, Pedram M and Chong F T 2020 NISQ+: Boosting quantum computing power by approximating quantum error correction *2020 ACM/IEEE 47th Annual Int. Symp. on Computer Architecture (ISCA)* (IEEE) pp 556–69
[39] Ueno Y, Kondo M, Tanaka M, Suzuki Y and Tabuchi Y 2021 QECOOL: On-line quantum error correction with a superconducting decoder for surface code *2021 58th ACM/IEEE Design Automation Conf. (DAC)* (IEEE) pp 451–6
[40] Ueno Y, Kondo M, Tanaka M, Suzuki Y and Tabuchi Y 2022 QULATIS: a quantum error correction methodology toward lattice surgery *2022 IEEE Int. Symp. on High-Performance Computer Architecture* pp 274–87
[41] Fowler A G 2013 Time-optimal quantum computation (arXiv:1210.4626)
[42] Bartolucci S *et al* 2021 Fusion-based quantum computation (arXiv:2101.09310)
[43] Chamberland C, Goncalves L, Sivarajah P, Peterson E and Grimberg S 2022 Techniques for combining fast local decoders with global decoders under circuit-level noise (arXiv:2208.01178)
[44] Bacon D 2022 Software of QIP, by QIP, and for QIP (available at: www.youtube.com/watch?v=9CHvTfWZyls)
[45] Webber M, Elfving V, Weidt S and Hensinger W K 2022 The impact of hardware specifications on reaching quantum advantage in the fault tolerant regime *AVS Quantum Sci.* **4** 013801
[46] Henriet L, Beguin L, Signoles A, Lahaye T, Browaeys A, Reymond G-O and Jurczak C 2020 Quantum computing with neutral atoms *Quantum* **4** 327
[47] Landahl A J, Anderson J T and Rice P R 2011 Fault-tolerant quantum computing with color codes (arXiv:1108.5738)
[48] Sarvepalli P and Raussendorf R 2012 Efficient decoding of topological color codes *Phys. Rev.* A **85** 022317
[49] Kubica A and Delfosse N 2019 Efficient color code decoders in $d \geqslant 2$ dimensions from toric code decoders (arXiv:1905.07393)
[50] Chamberland C, Kubica A, Yoder T J and Zhu G 2020 Triangular color codes on trivalent graphs with flag qubits *New J. Phys.* **22** 023019
[51] Breuckmann N P and Niklas Eberhardt J 2021 Quantum low-density parity-check codes *PRX Quantum* **2** 040101
[52] Panteleev P and Kalachev G 2022 Asymptotically good quantum andlocally testable classical LDPC codes (arXiv:2111.03654)
[53] Gottesman D 2013 Fault-tolerant quantum computation with constant overhead (arXiv:1310.2984)
[54] Roffe J, White D R, Burton S and Campbell E T 2020 Decoding across the quantum LDPC code landscape *Phys. Rev. Res.* **2** 043423

[55] Shouzhen G, Pattison C A and Tang E 2022 An efficient decoder for a linear distance quantum LDPC code (arXiv:2206.06557)
[56] Abobeih M H, Wang Y, Randall J, Loenen S J H, Bradley C E, Markham M, Twitchen D J, Terhal B M and Taminiau T H 2022 Fault-tolerant operation of a logical qubit in a diamond quantum processor *Nature* **606** 884–9
[57] Burns A 1991 Scheduling hard real-time systems: a review *Softw. Eng. J.* **6** 116–28
[58] Wilhelm R *et al* 2008 The worst-case execution-time problem-overview of methods and survey of tools *ACM Trans. Embed. Comput. Syst.* **7** 1–53
[59] Prabhu P and Chamberland C 2022 New magic state distillation factories optimized by temporally encoded lattice surgery (arXiv:2210.15814)
[60] Iyer P and Poulin D 2013 Hardness of decoding quantum stabilizer codes (arXiv:1310.3235)
[61] Ferris A J and Poulin D 2014 Tensor networks and quantum error correction *Phys. Rev. Lett.* **113** 030501
[62] Christopher T C 2021 General tensor network decoding of 2D Pauli codes (arXiv:2101.04125)
[63] Bravyi S, Suchara M and Vargo A 2014 Efficient algorithms for maximum likelihood decoding in the surface code *Phys. Rev.* A **90** 032326
[64] Dennis E, Kitaev A, Landahl A and Preskill J 2002 Topological quantum memory *J. Math. Phys.* **43** 4452–505
[65] Edmonds J and Johnson E L 1973 Matching, Euler tours and the Chinese postman *Math. Program.* **5** 88–124
[66] Delfosse N and Nickerson N H 2021 Almost-linear time decoding algorithm for topological codes *Quantum* **5** 595
[67] Duclos-Cianci G and Poulin D 2010 Fast decoders for topological quantum codes *Phys. Rev. Lett.* **104** 050504
[68] Criger B and Ashraf I 2018 Multi-path summation for decoding 2D topological codes *Quantum* **2** 102
[69] Higgott O, Bohdanowicz T C, Kubica A, Flammia S T and Campbell E T 2022 Fragile boundaries of tailored surface codes and improved decoding of circuit-level noise (arXiv:2203.04948)
[70] Zhao Y *et al* 2022 Realization of an error-correcting surface code with superconducting qubits *Phys. Rev. Lett.* **129** 030501
[71] Paler A and Austin G F 2022 Pipelined correlated minimum weight perfect matching of the surface code (arXiv:2205.09828)
[72] Varbanov B M, Battistel F, Tarasinski B M, Petrovych Ostroukh V, O'Brien T E, DiCarlo L and Terhal B M 2020 Leakage detection for a transmon-based surface code *npj Quantum Inf.* **6** 102
[73] Battistel F, Varbanov B M and Terhal B M 2021 Hardware-efficient leakage-reduction scheme for quantum error correction with superconducting transmon qubits *PRX Quantum* **2** 030314
[74] Delfosse N and Zémor G 2020 Linear-time maximum likelihood decoding of surface codes over the quantum erasure channel *Phys. Rev. Res.* **2** 033042
[75] McEwen M *et al* 2021 Resolving catastrophic error bursts from cosmic rays in large arrays of superconducting qubits *Nat. Phys.* **18** 107–11
[76] Fowler A G, Mariantoni M, Martinis J M and Cleland A N 2012 Surface codes: Towards practical large-scale quantum computation *Phys. Rev.* A **86** 032324
[77] Chamberland C and Campbell E T 2022 universal quantum computing with twist-free and temporally encoded lattice surgery *PRX Quantum* **3** 010331
[78] Fowler A G 2014 Minimum weight perfect matching of fault-tolerant topological quantum error correction in average $O(1)$ parallel time (arXiv:1307.1740)
[79] Kolmogorov V 2009 BlossomV: a new implementation of a minimum cost perfect matching algorithm *Math. Prog. Comp.* **1** 43–67
[80] Beverland M E, Brown B J, Kastoryano M J and Marolleau Q 2019 The role of entropy in topological quantum error correction *J. Stat. Mech.* **2019** 073404
[81] Higgott O 2021 PyMatching: a Python package for decoding quantum codes with minimum-weight perfect matching (arXiv:2105.13082)
[82] Higgott O and Gidney C 2022 Pymatching v2 (available at: https://github.com/oscarhiggott/PyMatching)
[83] Xiaotong N 2020 Neural network decoders for large-distance 2D toric codes *Quantum* **4** 310
[84] Ueno Y, Kondo M, Tanaka M, Suzuki Y and Tabuchi Y 2022 NEO-QEC: neural network enhanced online superconducting decoder for surface codes (arXiv:2208.05758)
[85] Andreasson P, Johansson J, Liljestrand S and Granath M 2019 Quantum error correction for the toric code using deep reinforcement learning *Quantum* **3** 183
[86] Colomer L D, Skotiniotis M and Muñoz-Tapia R 2020 Reinforcement learning for optimal error correction of toric codes *Phys. Lett.* A **384** 126353
[87] Fösel T, Tighineanu P, Weiss T and Marquardt F 2018 Reinforcement learning with neural networks for quantum feedback *Phys. Rev.* X **8** 031084
[88] Nautrup H P, Delfosse N, Dunjko V, Briegel H J and Friis N 2019 Optimizing quantum error correction codes with reinforcement learning *Quantum* **3** 215
[89] Zeng Y, Zhou Z-Y, Rinaldi E, Gneiting C and Nori F 2022 Approximate autonomous quantum error correction with reinforcement learning (arXiv:2212.11651)
[90] Sivak V V *et al* 2023 Real-time quantum error correction beyond break-even *Nature* **616** 50–55
[91] Murmann B 2020 Mixed-signal processing opportunities for AI *Edge AI Workshop*
[92] Han S, Mao H and Dally W J 2015 Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding (arXiv:1510.00149)
[93] Esser S K, McKinstry J L, Bablani D, Appuswamy R and Modha D S 2019 Learned step size quantization (arXiv:1902.08153)
[94] Knag P C *et al* 2020 A 617-TOPS/W all-digital binary neural network accelerator in 10-nm FinFET CMOS *IEEE J. Solid-State Circuits* **56** 1082–92
[95] Verma N, Jia H, Valavi H, Tang Y, Ozatay M, Chen L-Y, Zhang B and Deaville P 2019 In-memory computing: Advances and prospects *IEEE Solid-State Circuits Mag.* **11** 43–55
[96] Yin S, Zhang B, Kim M, Saikia J, Kwon S, Myung S, Kim H, Kim S J, Seok M and Seo J-Sun 2021 PIMCA: A 3.4-Mb programmable in-memory computing accelerator in 28 nm for on-chip DNN inference *2021 Symp. on VLSI Technology* (IEEE) pp 1–2
[97] Likharev K K and Semenov V K 1991 RSFQ logic/memory family: a newJosephson-junction technology for sub-terahertz-clock-frequency digital systems *IEEE Trans. Appl. Supercond.* **1** 3–28
[98] Kirichenko D E, Sarwana S and Kirichenko A F 2011 Zero static power dissipation biasing ofRSFQ circuits *IEEE Trans. Appl. Supercond.* **21** 776–9
[99] Leonard E *et al* 2019 Digital coherent control of a superconducting qubit *Phys. Rev. Appl.* **11** 014009
[100] Liebermann P J and Wilhelm F K 2016 Optimal qubit control using single-flux quantum pulses *Phys. Rev. Appl.* **6** 024022

[101] Reza Jokar M, Rines R and Chong F T 2021 Practical implications of SFQ-based two-qubit gates *2021 IEEE Int. Conf. on Quantum Computing and Engineering (QCE)* pp 402–12
[102] Tannu S S, Myers Z A, Nair P J, Carmean D M and Qureshi M K 2017 Taming the instruction bandwidth of quantum computers via hardware-managed error correction *Proc. 50th Annual IEEE/ACM Int. Symp. on Microarchitecture (Association for Computing Machinery) (MICRO-50 '17)* pp 679–91
[103] Subramanian Ravi G, Baker J M, Fayyazi A, Fuhui Lin S, Javadi-Abhari A, Pedram M and Chong F T 2022 Better than worst-case decoding for quantum error correction (arXiv:2208.08547)
[104] Smith S C, Brown B J and Bartlett S D 2022 A local pre-decoder to reduce the bandwidth and latency of quantum error correction (arXiv:2208.04660)
[105] Kuon I and Rose J 2009 *Quantifying and Exploring the Gap Between Fpgas and Asics* 1st edn (Springer Publishing Company, Incorporated)
[106] Sebastiano F, Homulle H, Patra B, Incandela R, van Dijk J, Song L, Babaie M, Vladimirescu A and Charbon E 2017 Cryo-CMOS electronic control for scalable quantum computing *Proc. 54th Annual Design Automation Conf. 2017* pp 1–6
[107] Liu C-H *et al* 2023 Single flux quantum-based digital control of superconducting qubits in a multi-chip module (arXiv: 2301.05696)
[108] Patra B *et al* 2020 19.1 a scalable cryo-CMOS 2-to-20GHz digitally intensive controller for 4 × 32 frequency multiplexed spin qubits/transmons in 22nmFinFET technology for quantum computers *2020 IEEE Int. Solid-State Circuits Conf.-(ISSCC)* (IEEE) pp 304–6
[109] Park J *et al* 2021 A fully integrated cryo-CMOS SoC for state manipulation, readout and high-speed gate pulsing of spin qubits *IEEE J. Solid-State Circuits* **56** 3289–306
[110] Bardin J C *et al* 2019 Design and characterization of a 28-nm bulk-CMOS cryogenic quantum controller dissipating less than 2 mW at 3 K *IEEE J. Solid-State Circuits* **54** 3043–60
[111] Frank D J *et al* 2022 A cryo-CMOS low-power semi-autonomous qubit state controller in 14nmFinFET technology *2022 IEEE Int. Solid- State Circuits Conf. (ISSCC)* vol 65 pp 360–2
[112] Kang K, Minn D, Bae S, Lee J, Kang S, Lee M, Song H-J and Sim J-Y 2022 A 40-nm cryo-CMOS quantum controller IC for superconducting qubit *IEEE J. Solid-State Circuits* **57** 3274–87
[113] Petit L *et al* 2018 Spin lifetime and charge noise in hot silicon quantum dot qubits *Phys. Rev. Lett.* **121** 076801
[114] Takeuchi N, Ozawa D, Yamanashi Y and Yoshikawa N 2013 An adiabatic quantum flux parametron as an ultra-low-power logic device *Supercond. Sci. Technol.* **26** 035010
[115] Takeuchi N, Yamanashi Y and Yoshikawa N 2014 Reversible logic gate using adiabatic superconducting devices *Sci. Rep.* **4** 1–4