

Protecting expressive circuits with a quantum error detection code

In the format provided by the
authors and unedited

ICEBERG CODES

Here we give additional details on the $[[k+2, k, 2]]$ code, which we have nicknamed the Iceberg code. These supplement the descriptions of the stabilisers, logical operators and universal gate set given in the main text.

Logical operators

One of the useful properties of the Iceberg code is there are global logical operators compiled onto a corresponding local two-qubit physical operator. For simplicity let us label the qubits into the set $[k] = \{1, \dots, k\}$ and the set $[n] = [k] \cup \{t, b\}$. Here we list all logical operators sorted by their two-qubit physical compilation:

$$\overline{X}_i = X_t X_i \quad \forall i \in [k] \quad (1)$$

$$\overline{X}_i \overline{X}_j = X_i X_j \quad \forall i, j \in [k] \quad (2)$$

$$\otimes_{j \in [k] \setminus i} \overline{X}_j = X_b X_i \quad \forall i \in [k] \quad (3)$$

$$\otimes_{j \in [k]} \overline{X}_j = X_t X_b \quad (4)$$

$$\overline{Z}_i = Z_b Z_i \quad \forall i \in [k] \quad (5)$$

$$\overline{Z}_i \overline{Z}_j = Z_i Z_j \quad \forall i, j \in [k] \quad (6)$$

$$\otimes_{j \in [k] \setminus i} \overline{Z}_j = Z_t Z_i \quad \forall i \in [k] \quad (7)$$

$$\otimes_{j \in [k]} \overline{Z}_j = Z_t Z_b \quad (8)$$

$$\overline{Y}_i \overline{Y}_j = Y_i Y_j \quad \forall i, j \in [k] \quad (9)$$

$$\overline{X}_i \otimes_{j \in [k] \setminus i} \overline{Z}_j = -Y_t Y_i \quad \forall i \in [k] \quad (10)$$

$$\overline{Z}_i \otimes_{j \in [k] \setminus i} \overline{X}_j = -Y_b Y_i \quad \forall i \in [k] \quad (11)$$

$$\otimes_{j \in [k]} \overline{Y}_j = (-1)^{1+k/2} Y_t Y_b. \quad (12)$$

We see that both single- and two-qubit logical operators are compiled into a two-qubit physical operator. Besides, in Eqs. (3), (4), (7), (8) and (10)-(12) we see that global logical gates acting on k or $k-1$ logical qubits are also compiled into a two-qubit physical operator.

Fault-tolerant operations

The Iceberg code has fault-tolerant operations for initialisation and syndrome measurement. These circuits require two additional ancilla qubits, which can be reset and reused. Our physical circuits therefore require $k+4$ physical qubits, giving a constant overhead.

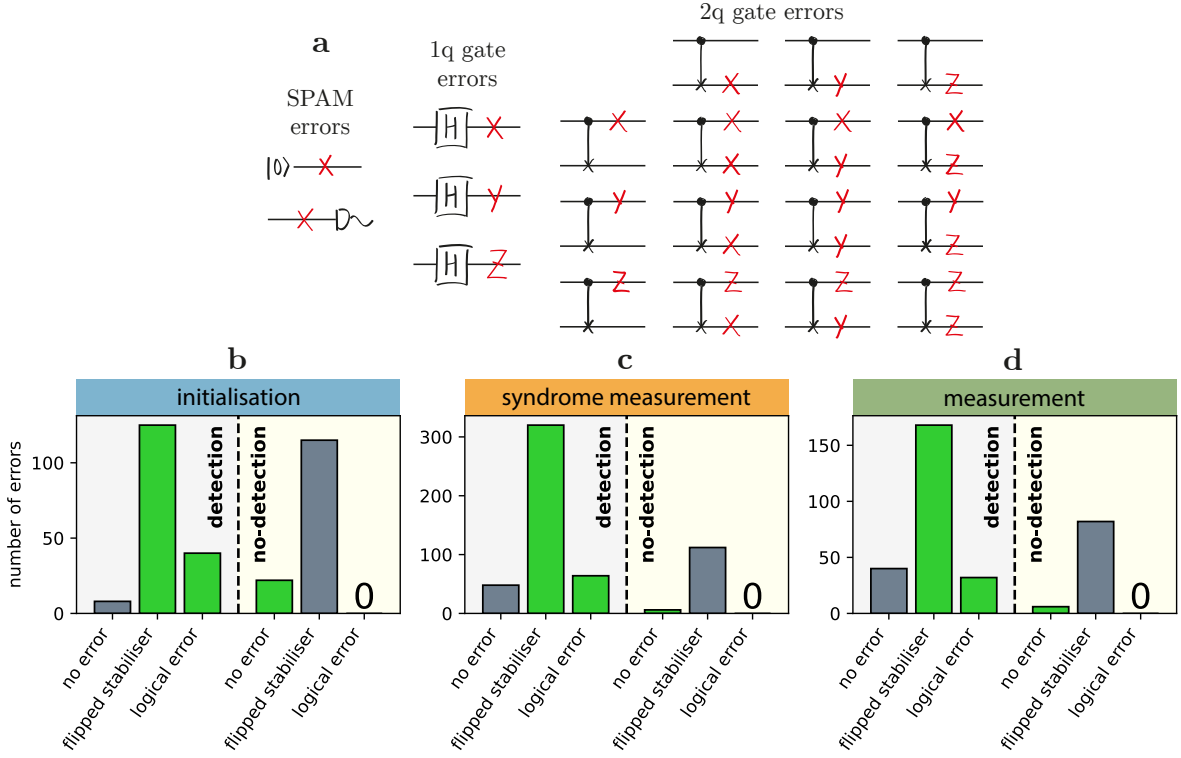
The circuits we implement for these tasks are fault-tolerant in the sense that there are no single faults that give rise to an undetectable logical error. We define single faults to be local Pauli errors arising from the failure of a circuit element. The full set of local errors we consider is shown in Supp. Fig. 1(a).

Initialisation prepares the *all-zeros* code state, which is the state such that $\overline{Z}_i |\overline{0}\rangle^{\otimes [k]} = +1 |\overline{0}\rangle^{\otimes [k]}$ for all $i \in [k]$. From the definition of the stabilisers and logical operators we see that this state is the Greenberger–Horne–Zeilinger (GHZ) state of the n physical qubits:

$$|\overline{0}\rangle^{\otimes [k]} = \frac{1}{\sqrt{2}} \left(|0\rangle^{\otimes [n]} + |1\rangle^{\otimes [n]} \right).$$

The fault-tolerant circuit we use to create this state is shown in the main text in Fig. 1(c).

The syndrome measurement should enable readout of the stabiliser eigenvalue without disturbing the logical information. The syndrome measurement circuit we use is drawn in the main text in Fig. 1(d), where two ancillas are used to separately readout the two stabilisers. The operations are arranged so that the ancillas act as flag qubits to each other.



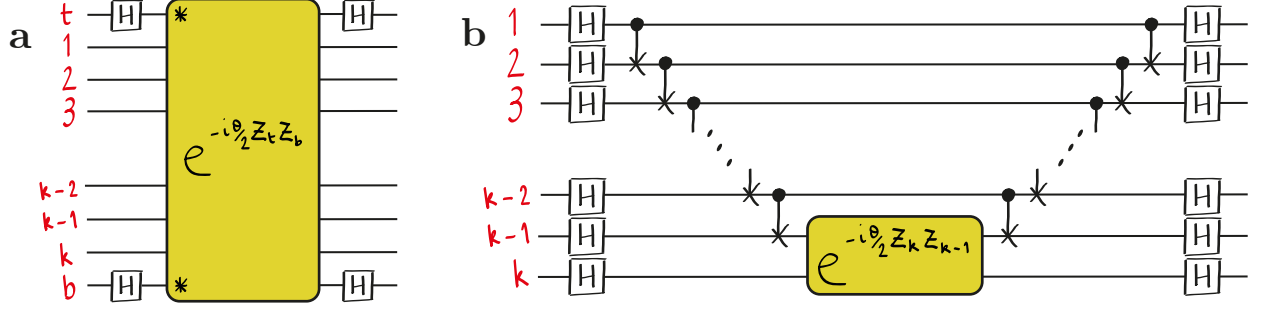
Supplementary Figure 1. Verification that the initialisation, syndrome measurement and measurement circuits of the Iceberg code are fault-tolerant for $k = 16$ logical qubits. **a** The full set of local errors that the initialisation, syndrome measurement and measurement circuits (shown in Fig. 1 of the main text) are fault-tolerant against. **b,c,d** For each circuit, all local errors are classified based on whether they trigger a detection readout (left three bars), or do not (right three bars). Within these groups the error can correspond to no error (for example an error with the form of a stabiliser), a flipped stabiliser, or a logical error. We colour the cases based on whether they are ideal behaviour (green) or non-ideal behaviour that is not a fail (grey). The three circuits are fault-tolerant because the number of undetected logical errors is zero.

We perform a destructive measurement at the end of the circuit by measuring all physical qubits in their computational basis. This allows us to reconstruct the measurement outcome for the $\bar{Z}_i$ logical operators and the S_Z stabiliser. However, we cannot reconstruct the S_X stabiliser from these measurements. To give us this information we extract the S_X stabiliser before the final readout. This is done using a flagged fault-tolerant circuit [1], shown in the main text in Fig. 1(e).

We verify that initialisation, syndrome measurement and measurement circuits are fault-tolerant by exhaustively checking all code sizes up to $k = 16$. For each possible local error, we insert the corresponding Pauli operator, propagate it to the end of the circuit, and finally assess its effect. We confirm that there are no errors that give rise to an undetectable logical error. This is illustrated for the case of $k = 16$ in Supp. Fig. 1(b-d), which shows a classification of all local errors.

Compiling target unitaries into a logical circuit

A generic approach to compiling a target unitary $\bar{U}$ into a logical circuit is to first compile $\bar{U}$ into a gate set that only contains single-qubit rotations, $\bar{R}_{\bar{X}}(\theta) = \exp(-i\theta\bar{X}/2)$ and $\bar{R}_{\bar{Z}}(\theta) = \exp(-i\theta\bar{Z}/2)$, and two-qubit logical rotations, $\bar{R}_{\bar{X}\bar{X}}(\theta) = \exp(-i\theta\bar{X}\bar{X}/2)$, $\bar{R}_{\bar{Y}\bar{Y}}(\theta) = \exp(-i\theta\bar{Y}\bar{Y}/2)$ and $\bar{R}_{\bar{Z}\bar{Z}}(\theta) = \exp(-i\theta\bar{Z}\bar{Z}/2)$. This can be done using the TKET compiler [2, 3]. Once expressed using single and two-qubit logical gates, each gate in the circuit can be directly compiled into its encoded physical form using $MS_{ij}(\theta)$ gates along with up to four single-qubit Clifford gates.



Supplementary Figure 2. Compilations of the rotation $R_{Xtb}(\theta) = \exp(-i\theta \bar{P}_{Xtb}/2)$, generated by $\bar{P}_{Xtb} = \otimes_{i \in [k]} \bar{X}_i = X_t X_b$. **a** With the Iceberg code, the rotation is realised by a single two-qubit $MS_{tb}(\theta)$ gate, as well as four Hadamard gates. **b** In the unencoded circuit, the physical circuit for the same rotation utilises a phase gadget. Since the logical generator of the rotation is a global operator, $\otimes_{i \in [k]} \bar{X}_i$, a ladder of CNOT gates is used to compute the parity before applying a single $MS_{k(k-1)}(\theta)$ gate. Then another CNOT ladder is used to uncompute the parity. Additionally, Hadamard gates on all qubits at the start and end apply a basis change.

MIRROR CIRCUIT EXPERIMENTS

Our mirror circuit experiments use 8 logical qubits encoded into 10 physical qubits. Experiments were executed on the Quantinuum H1-2 trapped-ion quantum computer between the 13th September and 4th October 2022.

In the following we discuss the experiments in more detail. First, we give a full discussion of the construction of the random mirror circuits, followed by additional practical details of the experiment. Next we give an in-depth account of the numerical simulations presented in the main text including the simplified noise model employed and the simulation method. Finally, we present numerical evidence that the random circuits used in our experiments are typical of the random circuit families we construct.

Random circuit construction

As described in the main text, random unitaries are constructed starting from the encoded physical circuit. The unitaries are built in layers. Within each layer physical qubits are paired (i, j) for $i \neq j \in [n]$ to give $\frac{k}{2} + 1$ pairings. For each pairing we uniformly choose $\sigma \in \{X, Y, Z\}$, uniformly sample an angle $\theta \in (0, 2\pi)$, and apply a two-qubit rotation to the pair, of the form:

$$R_{\sigma ij}(\theta) = \exp(-i\theta \sigma_i \sigma_j / 2). \quad (13)$$

Logical gates are all physically compiled using a $MS_{ij}(\theta) = \exp(-i\theta Z_i Z_j / 2)$ gate along with up to four single-qubit Clifford gates. An example of the physical compilation of $R_{Xtb}(\theta)$ is shown in Supp. Fig. 2(a).

Once the physical random unitary is constructed we map it to the corresponding logical unitary. All physical operators $\bar{P}_{\sigma ij} = \sigma_i \sigma_j$ represent a logical Pauli operator, the full set of relations is given in Eqs. (1)-(12). Logical operators can be single- and two-qubit rotations or rotations with generators that act on k or $k-1$ qubits. Single-qubit rotations are compiled directly. Two-qubit rotations are compiled using the two-qubit rotation $MS_{ij}(\theta)$ along with up to four single-qubit Clifford gates. Global rotations on k or $k-1$ qubits are compiled with a phase gadget whose kernel is a $MS_{ij}(\theta)$ gate. Supplementary Figure 2(b) shows an example of the compiled $R_{\otimes_{i \in [k]} \bar{X}_i}(\theta)$ in the unencoded circuit.

Additional experimental details

Circuits are initially constructed in the gateset: $\{H, S, S^\dagger, \text{CNOT}, MS(\theta)\}$, using the construction of the initialisation, syndrome measurement and final measurement circuits given in Fig. 1 of the main text as well as the construction of the random circuits described previously. These circuits are then compiled into the native gateset of the Quantinuum hardware [4] using the TKET compiler [3].

We consider from 4 random layers up to a maximum of 256. When reporting the number of layers we count the layers in both the unitary and its inverse, in order to count the total number of layers between initialisation and

final measurement. In our experiments on circuits encoded with the Iceberg code we insert one round of syndrome measurement between the unitary and its inverse for 32, 64, 128 & 256 layers, while we perform no intermediate syndrome measurements for 4, 8 & 16 layers.

Circuits are run for different numbers of repeats, here we report the total number of repeats (measurement shots) run in each case. For unencoded circuits with global rotations all cases have 300 shots; without global rotations 4, 8 & 16 layers run with 300 shots, 32 layers with 400, 64 layers with 600, 128 layers with 800 and 256 layers with 500. For the encoded circuits both with and without global rotations use the same shot numbers: 4, 8, 16 & 32 layers run with 200 shots, 64 layers with 400, 128 layers with 800 and 256 layers with 500. The approximate total experimental runtimes, including both encoded and unencoded circuits, is 7 hours for circuits containing global rotations and 8 hours for circuits not containing global rotations.

Numerical simulations

The numerical simulations in the main text cover $k = 10, 12, 14, 16$ logical qubits. For each k , we consider increasingly deep circuits, from $l = 4$ layers up to a maximum of 128. At each l , we generate 32 random mirror circuits allowing global logical rotations. Syndrome measurement rounds are inserted after every 16 layers. The noisy output of these circuits is simulated using a Monte-Carlo statevector approach described below. For each circuit we draw 32 statevector samples from the error model. We simulate both the unencoded and encoded version of each random circuit and compute the survival probabilities exactly for each statevector. Averaging the survival probabilities over these 32 statevector samples gives the estimated survival probability of the random circuit.

We simulate a simplified noise model that includes state preparation and measurement (SPAM) errors, as well as depolarising errors acting after gates. SPAM errors are represented with a bit-flip channel applied after qubit initialisation and before measurements,

$$\mathcal{E}_{\text{bitflip}}(\rho) = (1 - p_b)\rho + p_b X\rho X.$$

After each single (1q) and two-qubit (2q) gate we apply a depolarising error channel to the qubits acted on by the gate. The operator-sum expressions for these depolarising channels are

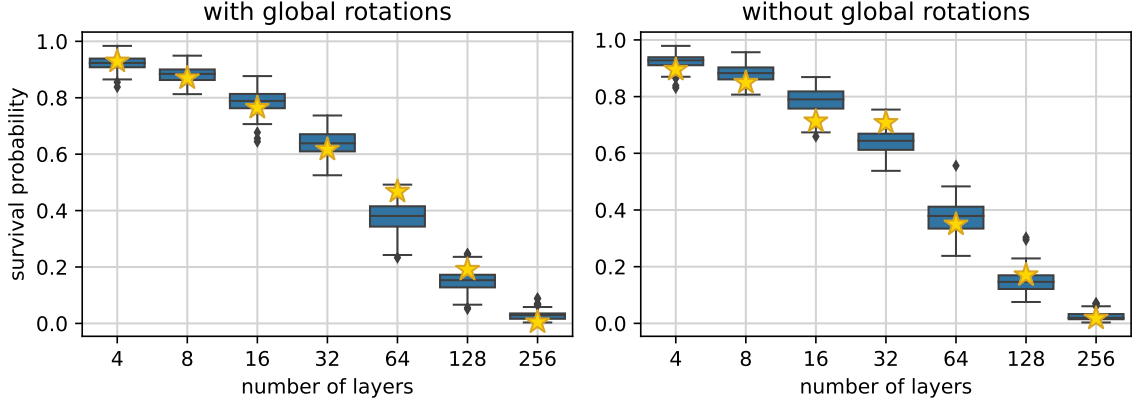
$$\begin{aligned} \mathcal{E}_{1q}(\rho) &= (1 - p_{1q})\rho + \frac{p_{1q}}{3} (X\rho X + Y\rho Y + Z\rho Z) \\ \mathcal{E}_{2q}(\rho) &= (1 - p_{2q})\rho + \frac{p_{2q}}{15} \left(IX\rho IX + IY\rho IY + IZ\rho IZ \right. \\ &\quad + XI\rho XI + XX\rho XX + XY\rho XY + XZ\rho XZ \\ &\quad + YI\rho YI + YX\rho YX + YY\rho YY + YZ\rho YZ \\ &\quad \left. + ZI\rho ZI + ZX\rho ZX + ZY\rho ZY + ZZ\rho ZZ \right) \end{aligned}$$

where I is the identity and X, Y, Z are the Pauli matrices.

Based on randomised benchmarking of the Quantinuum H1-2 hardware we use the following values for the parameters of the error channels: initialisation errors have a bitflip error rate $p_b = 4 \times 10^{-4}$, measurement errors have a bitflip error rate $p_b = 3 \times 10^{-3}$, single qubit gate errors have a depolarising rate $p_{1q} = 4 \times 10^{-4}$ and two qubit gate errors have a depolarising rate $p_{2q} = 3 \times 10^{-3}$.

Our noisy simulations are Monte-Carlo statevector simulations. For each noise channel we randomly select one of the Kraus operators of the channel with the correct probabilities. The noise channel is replaced with this randomly chosen Kraus operator. Once this is done for all error channels this turns the noisy quantum evolution back into a unitary operator that can be studied with a statevector simulator. By repeatedly sampling different Kraus operators of the noise channels we can approximate the full noisy evolution. This makes larger system sizes accessible by avoiding working with the, exponentially larger, density matrices.

In our quantum error detection circuits we carry out mid-circuit measurements and post-selection. These are implemented in our state-vector simulations with projectors in the computational basis. This allows us to compute both the probability of discard at that mid-circuit measurement and the normalised post-selected statevector (assuming the probability of discard is less than one).



Supplementary Figure 3. Distribution of the survival probabilities at each circuit depth for 100 random instances (left) including global rotations and (right) without global rotations, using box plots. The centre line of the box plots shows the median, with the lower and upper box edges corresponding to the 1st and 3rd quartile respectively. The whiskers of the plot extend to show data up to $1.5 \times$ the interquartile range below the 1st quartile and above the 3rd quartile. Outliers beyond these limits are plotted individually. Star icons indicate the performance of the experiments we present in the main text. Results are obtained from noisy simulations using a simplified noise model.

The numerical results presented in the main text are obtained using a Qiskit [5] statevector simulator. The circuits are prepared in the gateset: $\{H, S, S^\dagger, \text{CNOT}, \text{MS}(\theta)\}$, following the construction of the initialisation, syndrome measurement and final measurement circuits given in Fig. 1 of the main text along with the random circuits described previously. We consider the noise model acting on the circuits in this gateset.

In both the unencoded and encoded cases the survival probability is computed exactly from each statevector. In the unencoded case, this is simply the probability amplitude of the all-zeros computational basis state. In the encoded case, we must first extract the full probability distribution of outcomes in the computational basis, we then decode each computational basis state and add up the contributions to the logical all-zeros state.

Typicality of the instances used in experiments

In our experimental results we consider single instances of the random circuit families we describe. For a fixed number of layers, we generate one random circuit for the ‘with global rotations’ case and another random circuit for the ‘without global rotations’ version. The circuits generated at different numbers of layers are independent of each other. Here we give numerical evidence that these instances are typical instances of the random circuit families.

At each depth we generate 100 instances of the two random circuit families and Supp. Fig. 3 plots the distribution of the survival probabilities we observe as a box plot. Outliers are identified as points that lie more than 1.5 times the interquartile range above(below) the upper(lower) quartile. Each instance is numerically simulated using the simplified error model and statevector simulation method described previously under “Numerical Simulations”. For each circuit we average over 100 statevector samples from the error model.

The random instances that we use in experiments are indicated in Supp. Fig. 3 with star icons. These cases correspond to fixed seeds for random generators. We see that none of these instances correspond to outliers and they are spread above and below the median.

LOGICAL QUANTUM VOLUME EXPERIMENTS

We have carried out a logical Quantum Volume (QV) test of 8 qubits encoded in 10 physical qubits using the Iceberg code running on Quantinuum’s H1-2 trapped-ion quantum computer. The data presented in the main text was obtained between the 27th July and the 8th October 2022.

Here we give more details of these experiments – beginning with a short review of the QV test, followed by a discussion of the preparation of the logical circuits including the optimisations we apply. Next we discuss the encoding of the QV circuits into the Iceberg code. Finally, we give further practical details of the experiment and present additional experimental results.

QV test summary

The QV test is a sample efficient, randomised sampling scheme to verify that non-classical processing is happening inside a noisy quantum computer [6, 7]. The test applies random square circuits of the form shown in Supp. Fig. 4(a) to k qubits. These circuit have k layers, within each layer the qubits are randomly paired and a random $SU(4)$ unitary is applied to that pair. Finally, all of the qubits are measured in the computational basis.

Each random circuit is simulated noiselessly to obtain the ideal output distribution and from the ideal distribution we determine the set of heavy outputs. These are computational basis states that occur with higher probability than the median. The circuit is executed on a noisy quantum processor with a small number of shots (e.g. 100). The heavy output frequency of the noisy output is computed by counting up the number of shots that measure a computational basis state from the set of heavy outputs.

Heavy output frequency is averaged over the random circuits and the QV test is passed if the average is $> 2/3$. If a device passes a QV test of k qubits, it has quantum volume 2^k . Average heavy output frequency is not a fidelity measure, for noiseless circuits it is $\approx 85\%$ and for completely depolarised states it is 50% .

Preparation of the logical QV circuits

We use quantum volume circuits generated from Qiskit [5] and implement the ‘medium’ optimisation method described in Ref. [7]. Using this method $SU(4)$ blocks are combined together when allowed. An example of when this would occur is shown in Supp. Fig. 4(b).

Following these optimisations, the $SU(4)$ blocks are decomposed into single-qubit rotations – $R_{\bar{X}}(\theta)$ and $R_{\bar{Z}}(\theta)$ – and two-qubit logical rotations – $R_{\bar{X}\bar{X}}(\theta)$, $R_{\bar{Y}\bar{Y}}(\theta)$, $R_{\bar{Z}\bar{Z}}(\theta)$. This is done using the TKET compiler [2, 3]. At this stage we further optimise by squashing together single-qubit logical rotations as much as possible. For example, a sequence of gates $R_{\bar{Z}}(a)R_{\bar{X}}(b)R_{\bar{Z}}(c)R_{\bar{Z}}(d)R_{\bar{X}}(e)R_{\bar{Z}}(f)$ can be replaced with a shorter sequence $R_{\bar{Z}}(\alpha)R_{\bar{X}}(\beta)R_{\bar{Z}}(\gamma)$ that applies the same logical rotation. Additionally, we can remove Pauli-Z rotations that act immediately after initialisation and immediately before measurement.

Encoding the QV circuits

Once expressed using single and two-qubit logical rotations the QV circuit can be directly translated into its physical form, using Eqs. (1)-(12). This logical circuit is then combined with initialisation and measurement instruction sets. We note that due to the structure of the Iceberg code, where all single-qubit logical X rotations involve the physical qubit t and all single-qubit logical Z rotations involve the physical b , the compiled physical circuit does no longer present the parallel form of the logical QV circuit.

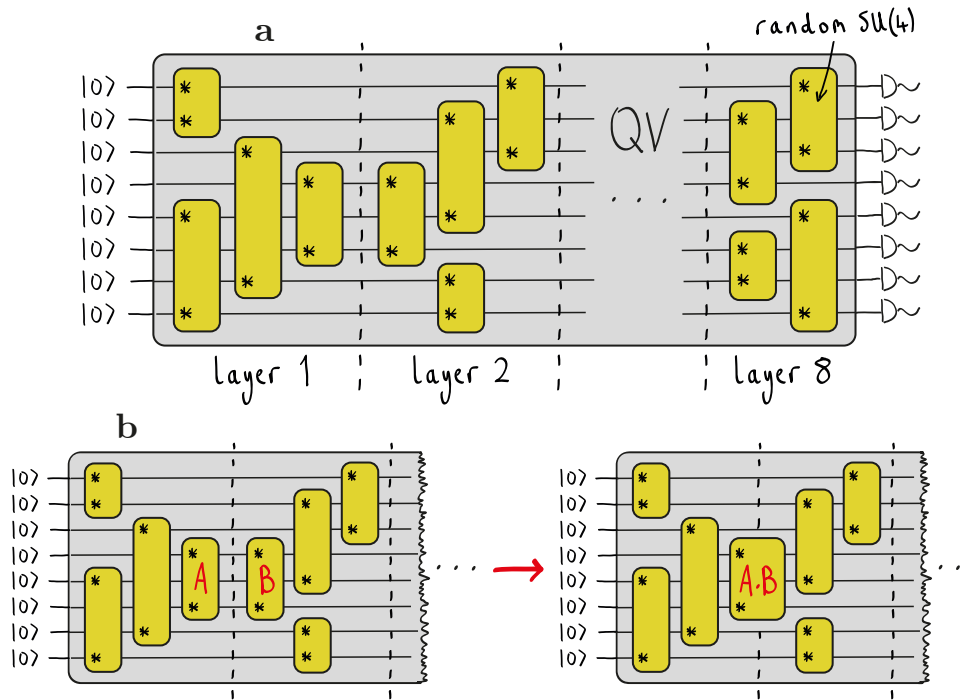
Syndrome measurements are inserted between equally sized chunks of the QV instruction set. When doing this we do not respect the layered structure of the QV circuit, meaning the syndrome measurement circuits may be inserted inside a QV layer. However, we must be careful not to break up logical operations, e.g. to insert a syndrome measurement between the Hadamards and $MS_{ij}(\theta)$ gate of $\exp(-iX_iX_j/2) = (H_iH_j)MS_{ij}(\theta)(H_iH_j)$.

The final executable circuits for the $k = 8$ logical QV test act on 12 physical qubits and have circuit depths of ~ 700 up to over 1000 depending on the number of syndrome measurements, including up to ~ 350 two-qubit gates.

Additional experimental details and results

Circuits are initially constructed in the gateset: $\{H, S, S^\dagger, \text{CNOT}, MS(\theta)\}$, following the construction of the initialisation, syndrome measurement and final measurement circuits given in Fig. 1 of the main text as well as the construction of the encoded QV circuits described previously. These circuits are then compiled into the native gateset of the Quantinuum hardware [4] using the TKET compiler [3].

Our experiments use 0x, 1x, 2x, 3x syndrome measurements. For 0x rounds of syndrome measurement we carry out 75 repeats of the experiment (measurement shots), for 1x we use 100 shots and for 2x and 3x we perform 150 shots. We vary the number of shots with the goal of retaining 10-20 shots of each circuit after discards, while the discard rate varies with the number of syndrome measurements. For the 0x, 1x and 3x cases we run 45 random circuits, while for 2x we run 100 circuits. The approximate total experimental runtimes is 5 hours for 0x rounds of syndrome measurement, 6 hours for 1x rounds, 20 hours for 2x rounds and 11 hours for 3x rounds.



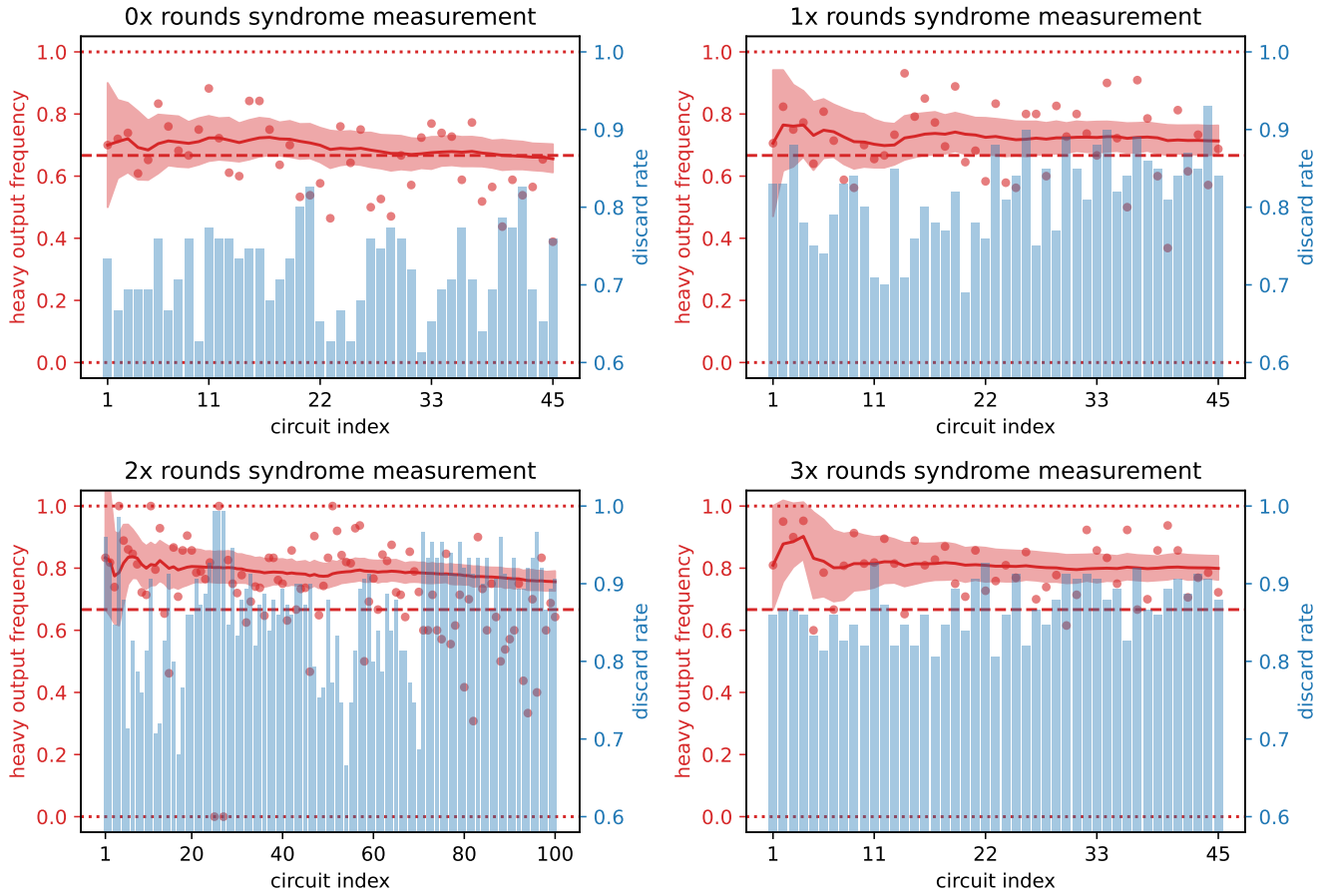
Supplementary Figure 4. **a** Schematic example of a Quantum Volume (QV) circuit for $N = 8$ qubits. The circuit consists of N layers, within each layer the qubits are randomly paired and the pairs are acted on by a random $SU(4)$. **b** Compression of $SU(4)$ blocks in QV circuits. We show a schematic of the first two layers of a six qubit QV test, where A and B are random $SU(4)$ unitaries. Qubits 4 and 6 are randomly paired in both layers meaning A and B act on the same qubits. They can be combined into a single $SU(4)$ unitary. This will generally result in a shorter circuit depth than applying A and B separately.

All circuits run to the end, with the outcome of mid-circuit error detection measurements being saved to classical syndrome registers. When the measurement results are collected we discard results where an error is detected. The results that are not discarded are post-processed to reconstruct the values of the logical operators $\bar{Z}_i$ as well as S_Z . If S_Z detects an error we discard the result. The remaining decoded results are processed to compute the heavy output frequency.

The heavy output frequencies for each individual QV circuit with 0x, 1x, 2x and 3x syndrome measurements are shown in Supp. Fig. 5, as well as the discard rates. The reported cumulative averages are calculated using Qiskit [5]. The bounds shown are calculated using the bootstrapped approach described in Ref. [7].

SUPPLEMENTARY REFERENCES

- [1] Chao, R. & Reichardt, B. W. Quantum error correction with only two extra qubits. *Phys. Rev. Lett.* **121**, 050502 (2018). URL <https://link.aps.org/doi/10.1103/PhysRevLett.121.050502>.
- [2] Sivarajah, S. *et al.* t|ket>: a retargetable compiler for NISQ devices. *Quantum Science and Technology* **6**, 014003 (2020). URL <https://doi.org/10.1088/2058-9565/ab8e92>.
- [3] Pytket Python Package (v1.4.1, v1.6.0, v1.7.3) <https://pypi.org/project/pytket/>, Pytket-Quantinuum extension (v0.4.0, v0.7.0, v0.9.0) <https://pypi.org/project/pytket-quantinuum/>, Pytket-Qiskit extension (v0.26.0, v0.28.0, v0.29.0) <https://pypi.org/project/pytket-qiskit/> (2022).
- [4] Pino, J. M. *et al.* Demonstration of the trapped-ion quantum ccd computer architecture. *Nature* **592**, 209–213 (2021). URL <https://doi.org/10.1038/s41586-021-03318-4>.
- [5] Qiskit Python Package (v0.37.0, v0.37.2, v0.38.0) <https://pypi.org/project/qiskit/> (2022).
- [6] Cross, A. W., Bishop, L. S., Sheldon, S., Nation, P. D. & Gambetta, J. M. Validating quantum computers using randomized model circuits. *Phys. Rev. A* **100**, 032328 (2019). URL <https://link.aps.org/doi/10.1103/PhysRevA.100.032328>.
- [7] Baldwin, C. H., Mayer, K., Brown, N. C., Ryan-Anderson, C. & Hayes, D. Re-examining the quantum volume test: Ideal distributions, compiler optimizations, confidence intervals, and scalable resource estimations. *Quantum* **6**, 707 (2022). URL <https://doi.org/10.22331/q-2022-05-09-707>.



Supplementary Figure 5. Additional experimental results showing the heavy output frequency of individual Quantum Volume (QV) circuits for 0x, 1x, 2x and 3x syndrome measurements. In each panel, the left axes (drawn in red) plots the heavy output frequency of each circuit after discarding, as well as the cumulative mean and its bootstrapped bounds. The passing threshold $2/3$ is indicated with a dashed line, as well as 0 and 1 being highlighted with dotted lines. The right axes (plotted as blue bars) shows the number of shots retained after discarding. Experiments with 0x syndrome measurements used 75 shots, with 1x syndrome measurement used 100 shots, and with 2x and 3x syndrome measurements used 150 measurement shots.