
Supplementary information

Scaling and networking a modular photonic quantum computer

In the format provided by the
authors and unedited

Supplementary information
for
Scaling and networking a modular photonic quantum computer

H. Aghaee Rad, T. Ainsworth, R. N. Alexander, B. Altieri, M. F. Askarani, R. Baby, L. Banchi, B. Q. Baragiola, J. E. Bourassa, R. S. Chadwick, I. Charania, H. Chen, M. J. Collins, P. Contu, N. D’Arcy, G. Dauphinais, R. De Prins, D. Deschenes, I. Di Luch, S. Duque, P. Edke, S. E. Fayer, S. Ferracin, H. Ferretti, J. Gefaell, S. Glancy, C. González-Arciniegas, T. Grainge, Z. Han, J. Hastrup, L. G. Helt, T. Hillmann, J. Hundal, S. Izumi, T. Jaeken, M. Jonas, S. Kocsis, I. Krasnokutska, M. V. Larsen, P. Laskowski, F. Laudenbach, J. Lavoie, M. Li, E. Lomonte, C. E. Lopetegui, B. Luey, A. P. Lund, C. Ma, L. S. Madsen, D. H. Mahler, L. Mantilla Calderón, M. Menotti, F. M. Miatto, B. Morrison, P. J. Nadkarni, T. Nakamura, L. Neuhaus, Z. Niu, R. Noro, K. Papirov, A. Pesah, D. S. Phillips, W. N. Plick, T. Rogalsky, F. Rortais, J. Sabines-Chesterking, S. Safavi-Bayat, E. Sazhaev, M. Seymour, K. Rezaei Shad, M. Silverman, S. A. Srinivasan, M. Stephan, Q. Y. Tang, J. F. Tasker, Y. S. Teo, R. B. Then, J. E. Tremblay, I. Tzitrin, V. D. Vaidya, M. Vasmer, Z. Vernon, L. F. S. S. M. Villalobos, B. W. Walshe, R. Weil, X. Xin, X. Yan, Y. Yao, M. Zamani Abnili, Y. Zhang
Xanadu Quantum Technologies Inc.
(Dated: November 13, 2024)

CONTENTS

I. Outline	2
II. Theory: GKP Stabilizer expectation values and effective squeezing	3
A. GKP stabilizer expectation values and effective squeezing	4
B. Stabilizer expectation values and Gaussian channels	5
C. Effective squeezing for multipeak states	7
D. Gaussian random noise approximation for approximate GKP states	9
III. Theory: Refinery protocols	10
A. Cat state breeding	10
B. Combining multiplexing and breeding	14
C. Measurement-based squeezing and Bell pair generation	18
IV. Theory: Gaussian Boson Sampling state preparation devices	19
A. Two-mode GBS devices to make squeezed cat states	19
B. A family of m -mode GBS devices for making m -peak approximate GKP states	22
C. Loss tolerance and success probabilities	25
V. Theory: State factory simulation techniques	27
A. Loss tolerance simulations	27
B. Simulation details for one state factory	29
1. Simulating GBS devices	31
2. Simulating the MUX-breeder	33

3. Calculating the stabilizer expectation values	34
4. Simulating measurement-based squeezing, and loss and MUX in the QPU	35
VI. Theory: Squeezing threshold for fault tolerance	36
A. Details on inner decoding	38
VII. Experiment: theoretical description	40
A. Two-mode GBS devices for squeezed cat generation	41
B. Cluster state for implementing the distance-2 foliated repetition code	42
1. Foliated repetition codes	42
2. Distance-2 repetition code with GKP states and linear-optical entanglement	43
3. Inner and outer decoding	44
C. Entanglement verification	48
D. Adaptivity demo	50
VIII. Experiment: Full system blueprint	51
A. Module Packaging	54
B. Laser system	55
C. Sources	58
1. GBS modules	58
2. Photon number resolving system	61
3. Non-Gaussian state generation	62
D. Refinery (MUX)	63
1. MUX	64
E. Quantum processing unit	65
1. QPU backend	67
F. Optical interconnects and feedback control	68
1. Stabilized fiber delay lines (SFDL)	68
IX. Experiment: System performance and demonstrations	70
A. Photon number heralding latency	70
B. Optical phase lock architecture	71
1. Phase calibration and initialization	74
C. System transmittance	74
References	75

I. OUTLINE

This document contains numerical, theoretical and experimental details relating to a) the structure of the photonic architecture (Secs. III, IV), b) the experimental demonstration (Secs. VII, VIII, IX), and the analysis of the tolerable loss to reach the fault-tolerance threshold (Secs. II, V, VI).

We begin with our theoretical results by describing the effective squeezing, a figure of merit used to relate the quality of Gottesman-Kitaev-Preskill (GKP) states (including loss throughout all optical paths) to the threshold for fault tolerance in Sec. II. GKP state generation is a two-stage process involving both non-Gaussian outputs of GBS devices, and breeding protocols. We introduce the theory behind breeding in Sec. III—which is well

studied for squeezed-cat state inputs—and discuss its application to more general non-Gaussian states. Then in Sec. IV, we describe how to choose GBS circuit parameters to generate non-Gaussian precursor states that are useful for breeding into GKP states. Armed with the figure of merit, set of refinery circuits, and set of GBS circuits, we describe the details of our numerical simulation in Sec. V that sweeps over different architecture configurations and computes the quality of the average GKP states produced as a function of the loss in each path. We compare this to an improved squeezing threshold of 9.75 dB (relative to 10.1 dB of previous work [1]), describing the details of how it arises by taking additional correlations account in the decoding in Sec. VI.

Next, we describe the details of Aurora, our scale-model architecture demonstration. In Sec. VII we provide the theoretical details of the state sources, refineries, and the details of the cluster state generated and how it implements the distance 2 foliated repetition code. We also provide theory for the inseparability measurements and our protocol for next-time-step adaptive quadrature measurements, conditioned on the outcome of a real-time decoder. In Sec. VIII we describe in detail the different subsystems and control systems that make up Aurora, including packaging, the laser system, sources, photon-number resolving detection system, refinery, QPU, and optical interconnects. We expand on the overall performance of the system and give details of unwanted imperfections in Sec. IX.

II. THEORY: GKP STABILIZER EXPECTATION VALUES AND EFFECTIVE SQUEEZING

Many of the key theoretical results presented in this work require a way to quantify how suitable different non-Gaussian states are for the purpose of GKP error correction (including concatenating with a higher qubit code). It is not feasible to do an exact simulation of the non-Gaussian approximate GKP states generated by our candidate GKP state generation systems, which we refer to here as “state factories” (including both the GBS and refinery subsystems), being entangled into a 3D cluster state operating in memory mode, as is often studied for threshold calculations. Such numerical calculations inevitably rely on a simplified noise model. For instance, one can approximate the noise of the initial GKP states as a Gaussian random noise channel applied to ideal GKP states [1–3], which captures the average peak width of the GKP states (see Appendix A of Ref. [3] and Sec. VI for details). Then, one can efficiently track how the covariance matrix of the Gaussian noise updates through the Gaussian cluster state construction, and sample homodyne outcomes from it, representative of sampling from the noisy peaks in the final homodyne distribution of the cluster.

To decide whether a particular configuration of our architecture is a promising candidate for fault-tolerant operation, we therefore desire a way of mapping the quality of the average output states to the variance of the Gaussian distribution of displacement errors. Then, we can extrapolate the large-scale performance using the assumption of the Gaussian random noise model. Here, we use the effective squeezing [4] as a figure of merit for characterizing approximate GKP states produced by GBS devices and breeding. Given a state, the effective squeezing measures the variance of its quadrature distribution from ideal periodic lattices. We use this parameter to set the variance in the Gaussian random noise model, when comparing to scalable fault-tolerance simulations. We begin by reviewing the effective squeezing, explore its useful properties, and then describe in more detail how we relate it to the variances of the Gaussian noise model used in threshold simulations.

A. GKP stabilizer expectation values and effective squeezing

Ideal GKP states are defined as states stabilized by two displacement operators [5]. That is, they are the +1 eigenstates of displacement operators,

$$\hat{D}(\alpha) |\psi_{\text{GKP}}\rangle = \hat{D}(\beta) |\psi_{\text{GKP}}\rangle = |\psi_{\text{GKP}}\rangle, \quad (1)$$

where $\hat{D}(\alpha), \hat{D}(\beta)$, $\alpha, \beta \in \mathbb{C}$, are two displacement operators such that $\hat{D}(\alpha)\hat{D}(\beta) = e^{2\pi i n} \hat{D}(\beta)\hat{D}(\alpha)$, $n \in \mathbb{Z}$, with $\hat{D}(\alpha) = e^{\alpha \hat{a}^\dagger - \alpha^* \hat{a}}$. To specify a 1D Hilbert space (e.g. a square-grid quanaught state [6, 7]), we require $\text{Im}(\alpha^* \beta) = \pi$, and to specify a qubit codespace (i.e. a 2D Hilbert space spanned by state vectors that are stabilized by the two displacement operators), we require $\text{Im}(\alpha^* \beta) = 2\pi$. This commutation of displacement operators defines a unit cell in phase space with area $2\pi n$. Also, since the ideal GKP states obey a discrete translational symmetry based on their stabilizers being displacements, we can use this translational symmetry to define a lattice in phase space.

For an ideal GKP state, the expectation value of its codespace stabilizers (displacement operators) are:

$$\text{Tr} [\hat{D}(\alpha) |\psi_{\text{GKP}}\rangle \langle \psi_{\text{GKP}}|] = \text{Tr} [\hat{D}(\beta) |\psi_{\text{GKP}}\rangle \langle \psi_{\text{GKP}}|] = \langle \psi_{\text{GKP}} | \psi_{\text{GKP}} \rangle. \quad (2)$$

While $|\psi_{\text{GKP}}\rangle$ is formally not normalizable, we can always consider this equation for e.g. Fock damped GKP states $e^{-\Delta \hat{n}} |\psi_{\text{GKP}}\rangle / \mathcal{N}_\Delta$, where $\mathcal{N}_\Delta$ is a normalization constant, in the limit as the damping constant $\Delta \rightarrow 0$ and the stabilizer expectation values approach 1. This is the first motivating factor for employing stabilizer expectation values as a figure of merit, since the higher the quality of (normalizable) GKP, the closer the expectation value is to 1.

Note that if a state is displaced, then its stabilizer expectation value will simply be multiplied by a complex number of the form $e^{i\phi}$, where ϕ is a function of the amount the state has been displaced [8]:

$$\text{Tr} [\hat{D}(\alpha) \hat{D}(\gamma) \rho \hat{D}(\gamma)^\dagger] = \text{Tr} [\hat{D}(\gamma)^\dagger \hat{D}(\alpha) \hat{D}(\gamma) \rho] = e^{2i \text{Im}[\alpha \gamma^*]} \text{Tr} [\hat{D}(\alpha) \rho], \quad (3)$$

which implies $\phi = 2\text{Im}[\alpha \gamma^*]$. This motivates using the absolute values of the expectation values of the GKP stabilizers as figures of merit for any approximate GKP states, since they are invariant under displacements [6, 8–10]. Having a figure of merit that is invariant under heralded displacement is valuable since all the measurements on the cluster state in our architecture are homodyne measurements, so heralded displacements can always be accounted for in software, and hence do not change the quality of the states in the cluster state. This is in contrast to e.g. fidelity as a figure of merit, which is sensitive to whether a state has been displaced.

The stabilizer expectation values can also be related to a quantity called the effective squeezing of the approximate GKP state via [6, 8]:

$$\sigma_\alpha^2 = -\frac{1}{|\alpha|^2} \log \left(|\text{Tr}[\rho \hat{D}(\alpha)]|^2 \right). \quad (4)$$

We often consider two stabilizers, one of which is real-valued and the other is imaginary-valued. In this case, we use the phrases ‘effective squeezing in p ’ for $\alpha \in \mathbb{R}$ and ‘effective squeezing in q ’ for $i\alpha \in \mathbb{R}$, which, for convenience, we denote σ_p^2 and σ_q^2 respectively. The effective squeezing has a physical interpretation as the average per-peak variance of the distribution of homodyne measurement results in the quadrature that commutes with $\hat{D}(\alpha)$, where per-peak refers to the fact that we are computing the variance of the homodyne

distribution on a lattice with spacing $\frac{\sqrt{2}\pi}{\alpha}$. Note that the effective squeezing σ_α^2 is unitless, the implication being it is a ratio relative to vacuum variance; to add units, e.g. to relate to the ϵ variance in the GRN model discussed later in Sec. VI, we multiply by $\frac{1}{2}$. We can also express the effective squeezing in dB units via:

$$\sigma_\alpha^2[\text{dB}] = -10 \log_{10} \sigma_\alpha^2. \quad (5)$$

More precisely, if we compute $\text{Tr}[\rho \hat{D}(\alpha)]$, where $\hat{D}(\alpha) = e^{i\sqrt{2}|\alpha|\hat{q}_\phi}$, in the basis that diagonalizes $\hat{q}_\phi = \hat{q} \cos \phi + \hat{p} \sin \phi$:

$$\begin{aligned} \text{Tr}[\rho \hat{D}(\alpha)] &= \int dq_\phi P(q_\phi) e^{i\sqrt{2}|\alpha|q_\phi}, \quad P(q_\phi) = \langle q_\phi | \rho | q_\phi \rangle_{\hat{q}_\phi} \\ &= \sum_{n=-\infty}^{\infty} \int_{-\frac{\pi}{\sqrt{2}|\alpha|}}^{\frac{\pi}{\sqrt{2}|\alpha|}} dq_\phi e^{i\sqrt{2}|\alpha|(q_\phi - \frac{\sqrt{2}\pi n}{|\alpha|})} P\left(q_\phi - \frac{\sqrt{2}\pi n}{|\alpha|}\right) \\ &= \int_{-\frac{\pi}{\sqrt{2}|\alpha|}}^{+\frac{\pi}{\sqrt{2}|\alpha|}} dq_\phi e^{i\sqrt{2}|\alpha|q_\phi} \sum_{n=-\infty}^{\infty} P\left(q_\phi - \frac{\sqrt{2}\pi n}{|\alpha|}\right) \\ &\equiv e^{i|\alpha|\mu_\alpha - |\alpha|^2\sigma_\alpha^2/2}. \end{aligned} \quad (6)$$

We see that this expression for $\text{Tr}[\rho \hat{D}(\alpha)]$ is the equation for the first moment of the wrapped distribution

$$P_{\text{wrapped}}(q_\phi) = \sum_{n=-\infty}^{\infty} P\left(q_\phi - \frac{\sqrt{2}\pi n}{|\alpha|}\right), \quad q_\phi \in \left[-\frac{\pi}{\sqrt{2}|\alpha|}, +\frac{\pi}{\sqrt{2}|\alpha|}\right) \quad (7)$$

for the circular variable $e^{i\sqrt{2}|\alpha|q_\phi}$, where the last line of Eq. 6 is taken as the equation defining the moments of the wrapped distribution $(\mu_\alpha, \sigma_\alpha)$. We can think of $P_{\text{wrapped}}(q_\phi)$ as chopping up the distribution $P(q_\phi)$ into bins set by $|\alpha|$, then computing the average distribution within a bin over all the bins. Were we to approximate the wrapped distribution by a wrapped normal distribution (Gaussians on a lattice parametrized by α), then calculating the first moment, i.e. finding $(\mu_\alpha, \sigma_\alpha)$, provides the parameters of the wrapped normal distribution approximation. In other words, the equation for effective squeezing is simply the average per-peak variance of the distribution taking into account the periodic nature of the lattice. Thus, when setting the variance of the Gaussian noise channel used for the scalable threshold simulations, we can use the variance found from computing the effective squeezing ($\sigma_\alpha^2 = 2\epsilon$, where ϵ is the variance used in the GRN model in Sec. VI). We will discuss shortly how to set the variance of the Gaussian noise channel for our state factories which produce distributions of Bell pairs, but first we will explore a useful property of stabilizer expectation values.

B. Stabilizer expectation values and Gaussian channels

Stabilizer expectation values update in a straightforward way under Gaussian channels, which is valuable for determining how much tolerance the architecture has to Gaussian noise sources like loss. A completely positive, trace-preserving (CPTP) Gaussian channel Φ is parameterized by matrices X and Y such that the covariance matrix and vector of means of a state in the q - p basis transform as $\Sigma \rightarrow X\Sigma X^T + Y$ and $\mu \rightarrow X\mu$ under the application of the channel, and such that $Y + i\Omega \geq iX\Omega X^T$, where

$$\Omega = \begin{pmatrix} 0 & \mathbb{I} \\ -\mathbb{I} & 0 \end{pmatrix}$$

is the symplectic form [11]. Suppose we want to find the expectation value of some complex displacement $\hat{D}(\gamma_f)$ in a single-mode state after undergoing a Gaussian channel $\Phi(\rho)$. As the expectation value of a displacement is just the characteristic function evaluated at that displacement, we use the result from Eq. (5.38) in Ref. [11] for how the characteristic function changes under a Gaussian channel:

$$\text{Tr}(\Phi(\rho)\hat{D}(\gamma_f)) = e^{-\frac{1}{2}\gamma_f^T \Omega^T Y \Omega \gamma_f} \text{Tr}\left(\rho \hat{D}\left((1, i)\Omega^T X^T \Omega \gamma_f\right)\right), \quad \gamma_f = \begin{pmatrix} \text{Re}(\gamma_f) \\ \text{Im}(\gamma_f) \end{pmatrix}. \quad (8)$$

This equation is written in terms of the stabilizer of the final state after applying the channel, γ_f , but rewriting in terms of the stabilizer of the initial state, γ_{in} , we have

$$\text{Tr}(\Phi(\rho)\hat{D}(\gamma_f)) = e^{-f(X, Y, \gamma_{\text{in}})} \text{Tr}(\rho \hat{D}(\gamma_{\text{in}})) \quad (9)$$

where

$$\gamma_f = \Omega^T (X^{-1})^T \Omega \gamma_{\text{in}}, \quad \gamma_{\text{in}} = \begin{pmatrix} \text{Re}(\gamma_{\text{in}}) \\ \text{Im}(\gamma_{\text{in}}) \end{pmatrix}$$

$$f(X, Y, \gamma_{\text{in}}) = \frac{1}{2} \gamma_{\text{in}}^T \Omega^T X^{-1} Y (X^{-1})^T \Omega \gamma_{\text{in}}.$$

Hence, if we want to find a stabilizer expectation value of a state after a Gaussian CPTP channel with respect to the displacement γ_f , then we only need to know its stabilizer expectation value before the channel with respect to a known displacement γ_{in} . In particular, it is not necessary to compute the final output state if we only want the stabilizer expectation values of that state after the channel. Moreover, as the matrix Y is symmetric, we can write it as $Y = ZZ^T$ for some Z and therefore $f(X, Y, \gamma_{\text{in}}) = \frac{1}{2} \|\gamma_{\text{in}}^T \Omega^T X^{-1} Z\|^2$ which is always non-negative. Therefore, the factor $e^{-f(X, Y, \gamma_{\text{in}})} \leq 1$ will only ever maintain or degrade the stabilizer expectation value, which means that Gaussian channels cannot increase stabilizer expectation values; the case of no reduction in the stabilizer expectation value, i.e. $e^{-f(X, Y, \gamma_{\text{in}})} = 1$, occurs if and only if the channel is a unitary ($Y = 0$).

This last point—that stabilizer expectation values remain invariant under Gaussian unitaries $\hat{U}_G$ applied to the state as long as we apply the corresponding lattice transformation $\hat{U}_G^\dagger$ to the stabilizer—is valuable when we recall that Gaussian unitaries applied to GKP states simply change the lattice. This means that if we compute the stabilizer expectation values of a state with respect to the original lattice, then transform the state to a new lattice via a Gaussian unitary and recompute the stabilizer expectation values with respect to the new lattice, the quality of the state has not changed. This allows us to compare two approximate GKP states on different lattices by considering a fictitious alignment of the GKP states to a common lattice, such as the square-grid quanaught state lattice with stabilizers $\hat{D}(\sqrt{\pi})$ and $\hat{D}(i\sqrt{\pi})$ [6, 12]. Then, we can either directly compare their stabilizer expectation values, or we can compute the effective squeezing relative to the common lattice. Comparing approximate GKP states on different lattices is useful for our architecture since, as will be seen in Sec. IV, the Gaussian Boson Sampling (GBS) devices we consider generate different GKP lattices for different photon number patterns and we would like a way to rank these states in order to determine which states to choose when multiplexing.

Using the way the effective squeezing changes under a Gaussian unitary, it will be convenient to define a single quantity to use as a figure of merit for the quality of approximate GKP states (as opposed to having to use two stabilizer expectation values). We define the

symmetric effective squeezing to be the arithmetic mean of the effective squeezing in the q quadrature and the effective squeezing in the p quadrature after the state has been aligned to the square quanaught state lattice (where the separation between the peaks for both q and p is $\sqrt{2\pi}$), i.e.,

$$\sigma_{\text{sym}}^2 \equiv \frac{\sigma_{q,\emptyset}^2 + \sigma_{p,\emptyset}^2}{2}, \quad (10)$$

where $\sigma_{q,\emptyset}^2$ and $\sigma_{p,\emptyset}^2$ are the effective squeezing values in both quadratures after aligning the state to the square-grid quanaught state lattice.

Finally, when we know our states have been transformed by a noisy Gaussian channel, this result provides guidance for how to choose approximate GKP lattices which may not necessarily satisfy $\hat{D}(\alpha)\hat{D}(\beta) = e^{2\pi i n}\hat{D}(\beta)\hat{D}(\alpha), n \in \mathbb{Z}$. In particular, were we to start with a state whose stabilizers satisfy $\hat{D}(\alpha)\hat{D}(\beta) = e^{2\pi i n}\hat{D}(\beta)\hat{D}(\alpha), n \in \mathbb{Z}$, then pass that state through a loss channel with transmission η , i.e. $X = \sqrt{\eta}\mathbb{I}, Y = (1 - \eta)\mathbb{I}/2$, we would find that the new stabilizers are given by $\hat{D}(\alpha/\sqrt{\eta}), \hat{D}(\beta/\sqrt{\eta})$, with $\hat{D}(\alpha/\sqrt{\eta})\hat{D}(\beta/\sqrt{\eta}) = e^{2\pi i n/\eta}\hat{D}(\beta/\sqrt{\eta})\hat{D}(\alpha/\sqrt{\eta}), n \in \mathbb{Z}$. Although the stabilizers for the state no longer commute, we already know that this state cannot exactly satisfy the GKP stabilizers since it underwent loss; since we are working with an approximate GKP, we can also choose to work with approximate GKP stabilizers. Intuitively, these stabilizers correspond to a rescaling of phase space variables by $\sqrt{\eta}$, which makes sense because we know how the loss channel rescales phase space (the noise is due to the Gaussian random noise on top of this deterministic rescaling). This means that when we measure the stabilizers of the state by measuring the homodyne distribution and binning the outcomes, we simply need to rescale the width of the bins by $\sqrt{\eta}$. This adaptation of the homodyne binning was performed in Ref. [1] to mitigate the effect of loss in the homodyne measurements of the cluster state, and led to better thresholds. We can also use such approximate stabilizers when choosing how to program our state factory candidates; namely, we can account for states only approximately satisfying approximate GKP stabilizers e.g. when combining two GBS output states in the breeding protocol.

C. Effective squeezing for multipeak states

A common approximation to GKP states is a superposition of an infinite number of evenly spaced finite-width Gaussian peaks, under a Gaussian envelope. As the outer peaks become negligibly small, even a state with a finite number of Gaussian peaks can be a good approximation to GKP states. We define an ideal multipeak state, with N peaks, as

$$|\psi\rangle = \sum_{n=-(N-1)/2}^{(N-1)/2} c_n \hat{D}(n\alpha) \hat{S}(r) |0\rangle \quad (11)$$

where $\hat{S}(r) = \exp[\frac{1}{2}r(\hat{a}^2 - (\hat{a}^\dagger)^2)]$ is the squeezing operator, the sum is over integers for odd N and the half integers for even N , $\alpha \in \mathbb{R}$ determines the spacing between the peaks, r is the per-peak squeezing and $c_{-n} = c_n$ give the relative differences in height of the peaks. The per-peak squeezing can be written in dB units as

$$\sigma^2[\text{dB}] = \frac{20r}{\ln(10)}. \quad (12)$$

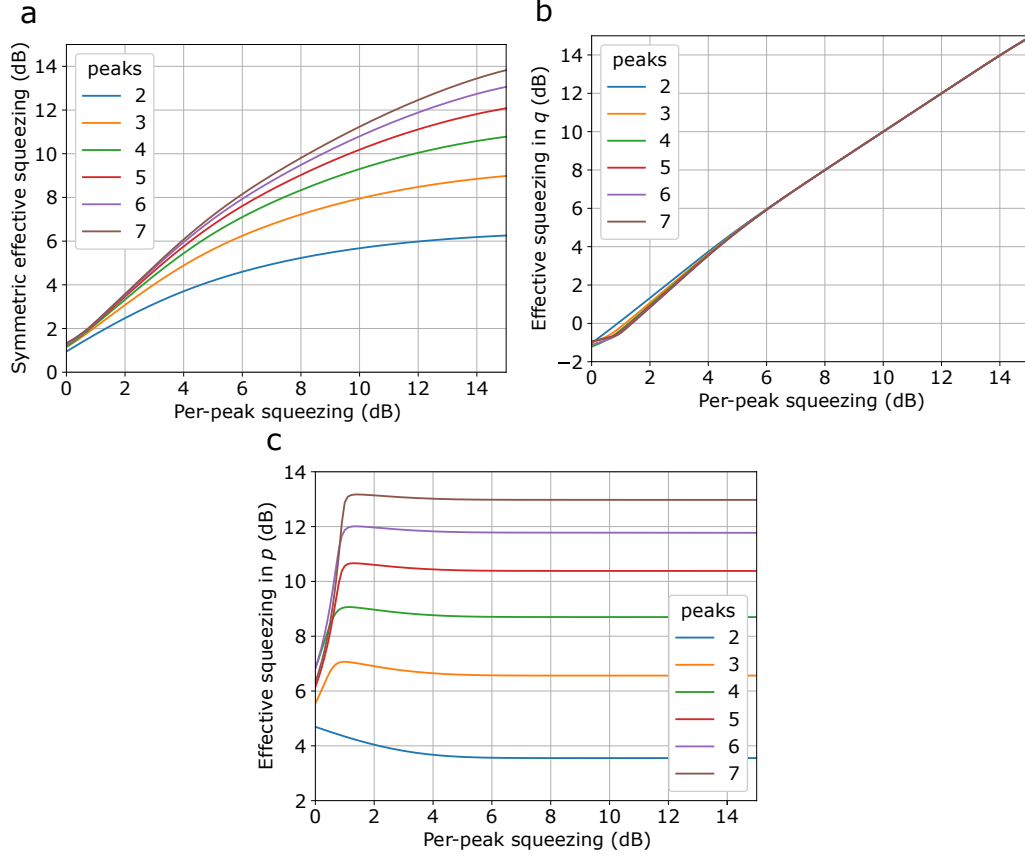


Figure S1. **a. Symmetric effective squeezing, b. q effective squeezing, c. p effective squeezing** of multi-peak states as a function of the per-peak squeezing (σ^2 [dB] in Eq. (12)). The effective squeezing in q is approximately proportional to the per-peak squeezing of the state, independent of the number of peaks. The effective squeezing in p increases as the number of peaks increases, but is approximately independent of the per-peak squeezing. As the symmetric effective squeezing is a function of the other two, it saturates at some value too, but a higher value for a higher number of peaks.

As the number of peaks N increases, the approximation to a GKP state improves.

Fig. S1 shows the symmetric, q (with respect to an imaginary-valued stabilizer) and p (with respect to a real-valued stabilizer) effective squeezing for multi-peak states, as a function of the per-peak squeezing, σ^2 [dB]. The multi-peak states have 2 to 7 peaks that are aligned with the square-grid qunaught state lattice spacing, and the heights of the peaks have been optimized to maximize the symmetric effective squeezing. As expected, the q effective squeezing is approximately equal to the per-peak squeezing σ^2 [dB], and independent of the number of peaks when the squeezing is high enough (> 4 dB), and increasing the per-peak squeezing will indefinitely improve the q effective squeezing. However, the p effective squeezing saturates at some value that depends on the number of peaks. Consider the effective squeezing of a two-peak state. For a pure state, the stabilizer expectation value

is the overlap of the state with itself displaced by the lattice spacing. If a state has two peaks, and the peaks are infinitely squeezed, only one of the two peaks can overlap with the displaced state and so the maximum p stabilizer expectation value is 0.5. This corresponds to an effective squeezing in p of 3.55 dB, which is met for infinitely squeezed peaks. Note that for smaller per-peak squeezing, this value actually increases slightly because there is some overlap with all the peaks as they have finite width; however, this improvement is very small compared to the loss in the q quadrature. Since the effective squeezing in p saturates as the per-peak squeezing increases, the symmetric effective squeezing also saturates and the only way to increase it significantly once the per-peak squeezing is reasonably high is to increase the number of peaks in the multippeak state.

The two-peak state cannot have a symmetric effective squeezing greater than 6.56 dB and, from numerically optimizing the peak heights, we find the maximum symmetric effective squeezing for a three-peak state is 9.57 dB. Therefore, it is not possible to have a state above 9.75 dB, the target threshold from Sec. VI, with only two or three peaks. Instead, we must target states that have at least four peaks, where we need more peaks if the quality of the state (per-peak squeezing) is not high enough. In practice, although we are targeting ideal multippeak states, the states we generate may be non-ideal, with sub-optimal peak heights, slightly unequal spacing, non-Gaussian peaks with differing squeezing, and small distortions. We consider these as valid approximate GKP states as long as they have high enough symmetric effective squeezing.

D. Gaussian random noise approximation for approximate GKP states

Here we describe how we will use the symmetric effective squeezing as a figure of merit for candidate GKP states produced by state factories, and map to the variance of the Gaussian random noise model used for threshold simulations. In our architecture, the state factories produce a distribution of output states across all the measurement outcomes in the factory (photon number resolving and homodyne). Thus, from shot to shot, the states can be different approximations of GKP states. We can consider the effective squeezing of a distribution of states by computing the average mixed state over these probabilistic events, then computing the stabilizer expectation values for this mixed state. Note, here we mean the mixed state *after* any feedforward displacements heralded by the PNR and homodyne measurements in the state factory have been applied to the state; in practice, we do not need to physically apply these displacements as they can be conjugated through the cluster state construction and applied as shifts in software to the homodyne measurements in the QPU.

We will be interested in investigating how much loss (along the principle P_1 , P_2 , P_3 , and P_4 loss paths of the architecture defined in Fig. 1 in the main text) different architecture variants can tolerate while still producing states that are compatible with fault-tolerance. We will characterize the loss levels across the architecture in terms of the symmetric effective squeezing of a single-mode quanaught state. The state used is a noise-degraded version of the average state (taking displacements into account as described above) immediately after the measurement-based squeezer but before the beamsplitter that creates a GKP Bell pair in the refinery. To capture loss that occurs after this point in the refinery, we degrade the average state by applying a loss channel equal in strength to the total losses from that point to the homodyne detectors in the QPU. Note that the MUX in the QPU selects or discards states at a later point in the circuit than when the averaging occurs. We restrict the averaging to be only over the set of states that are selected by the MUX in the QPU. If we order the states in the distribution by decreasing symmetric effective squeezing, then we can define

the stabilizer expectation values, effective squeezing and symmetric effective squeezing of the average of the best $x\%$ of states in the distribution, where x is set by how much post-selection or multiplexing is possible following the state factory. The position and momentum effective squeezings may not be equal, resulting in a biased noise model. These biases can be taken into account using a modification to the cluster state construction process described in Ref. [13]. Instead, here we simplify the noise model by assuming each single-mode state undergoes a Fourier transform with 50% probability after the measurement-based squeezer, which has the effect of symmetrizing the position and momentum effective squeezing while preserving the properties of the ideal GKP state, since qunaught states are eigenstates of the Fourier transform. (Note, this Fourier transform is straightforward to implement in practice by including it in the measurement-based squeezer.) Thus, computing the effective squeezing in position or momentum of the average state while including probabilistic Fourier transform results in both position and momentum versions being equal to the symmetric effective squeezing (i.e., the biased noise has been symmetrized). The criterion we use to decide if a given architecture (with losses) is fault tolerant is whether or not the symmetric effective squeezing of this average state is greater than or equal to 9.75 dB. This value comes from threshold simulations (described below in Sec. VI) that make use of the Gaussian random noise model.

III. THEORY: REFINERY PROTOCOLS

In our architecture, the refinery has the role of converting the non-Gaussian states probabilistically output by the GBS devices (discussed in Sec. IV) into GKP Bell pairs. Since the GBS devices have a low success probability when used to produce high-quality GKP states directly [14–16], one role of the refinery is to multiplex, filtering away states that are not useful, and boosting the probability of states that are useful. The second function of the refinery is to convert the useful states, typically squeezed cat states or low quality multipeak states, into higher quality GKP states using the breeding protocol. In Sec. III A, we will review the breeding protocol, and in Sec. III B, we show that by combining both multiplexing and breeding stages in one circuit, we can improve overheads and circuit depth. Shot-to-shot, the lattice of the GKP states change depending on the photon count patterns of the GBS devices, so the refinery also includes measurement-based (MB) squeezing to align all the GKP states to the square-grid qunaught state lattice, as discussed in Sec. III C. This is followed by a beamsplitter to entangle two GKP states into a GKP Bell pair.

A. Cat state breeding

Cat state breeding is a general protocol that distills multipeak states into approximate GKP states through a circuit consisting of beamsplitters and homodyne detectors [8, 17, 18]. The basics of the protocol can be understood by the “unit cell” circuit shown in Fig. S2a that combines two squeezed multipeak states on a beamsplitter and then performs a p-homodyne measurement on mode 2.

The utility of the protocol can be illustrated using an example of a pair of 2-peak input states (i.e., squeezed cat states) as shown in Fig. S2c. The squeezed cat state with amplitude $\alpha/2$ and squeezing r is defined as:

$$|\text{sq. cat}\rangle \propto \left(\hat{D}(\alpha/2) + \hat{D}(-\alpha/2) \right) \hat{S}(r) |\text{vac}\rangle, \quad (13)$$

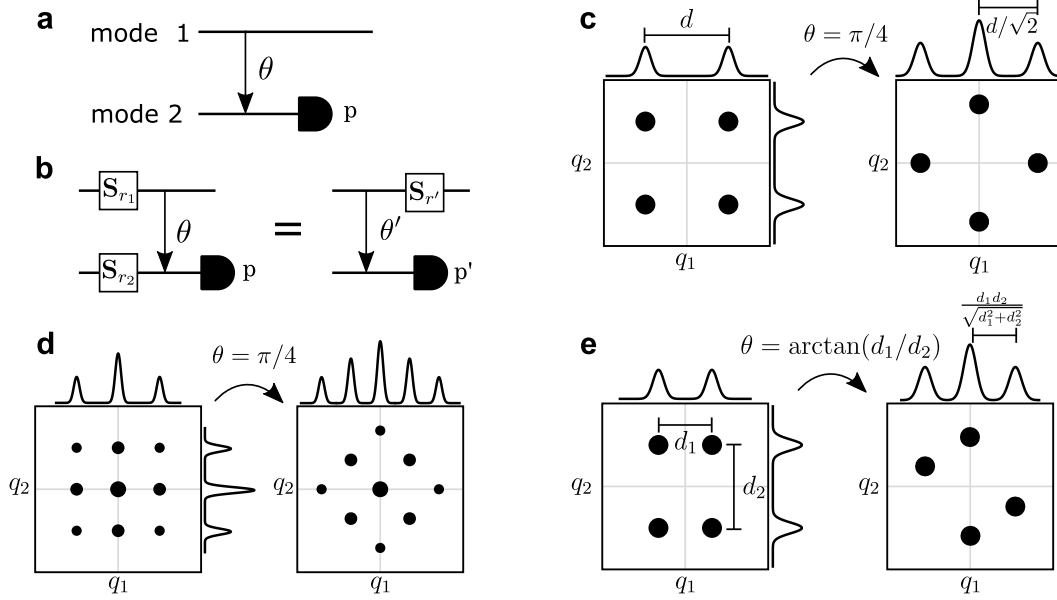


Figure S2. **(a)** Breeding circuit diagram. **(b)** Circuit identity for breeding states with non-equal lattices, where $\tan \theta' = e^{r_1 - r_2} \tan \theta$, $e^{-2r'} = e^{-2r_1} \cos^2 \theta + e^{-2r_2} \sin^2 \theta$, $p' = p e^{r' - r_1 - r_2}$ and the displacement is by an amount $\alpha = -i\sqrt{2}p'e^{r'} \csc 2\theta'(\sin^2 \theta - \sin^2 \theta')$. Instead of squeezing all the input states to match a common lattice and using a static beamsplitter (l.h.s.), one can instead use an adaptive beamsplitter and postpone the squeezing operation to the output state (r.h.s.). **(c)** Two-mode q -wavefunction picture of the breeding operation. The figure shows a contour of the two-mode wavefunction, $\Psi(q_1, q_2) = \langle q_1, q_2 | \Psi \rangle$, for two input cat states that undergoes a rotational evolution under the action of the beamsplitter. Finally, a measurement along p_2 projects the state in mode 1 to have the three-peak wavefunction above the second plot. Note the number of peaks in q has increased (which can boost the effective squeezing in the p quadrature), while the spacing of the peaks in the q quadrature has decreased (which reduces the effective squeezing in the q quadrature), so we can understand the protocol as trading effective squeezing from one quadrature to another. **(d)** Evolution of the two-mode q -wavefunction for the breeding of two 3-peak states into one 5-peak state. **(e)** Breeding of two cat states with non equal lattices using an adaptive beamsplitter angle.

with corresponding q -quadrature wavefunction given by:

$$\psi_{\text{sq. cat}}(q) = \langle q | \text{sq. cat} \rangle \propto \exp \left(-\frac{\left(q - \frac{\alpha}{\sqrt{2}} \right)^2}{2e^{-2r}} \right) + \exp \left(-\frac{\left(q + \frac{\alpha}{\sqrt{2}} \right)^2}{2e^{-2r}} \right), \quad (14)$$

The distance between the peaks in phase-space (also known as lattice spacing) is thus $d = \sqrt{2}\alpha$. The beamsplitter transforms the q quadrature as

$$\hat{B}S(\theta) |q_1, q_2\rangle = |q_1 \cos \theta + q_2 \sin \theta, -q_1 \sin \theta + q_2 \cos \theta\rangle. \quad (15)$$

That is, the beamsplitter performs a rotation of the two-mode q -quadrature wavefunction, as illustrated in Fig. S2c. The two-mode wavefunction after a 50:50 beamsplitter ($\theta = \pi/4$)

is then

$$\begin{aligned} \Psi(q_1, q_2) &= \psi_{\text{sq. cat}}\left(\frac{q_1 + q_2}{\sqrt{2}}\right) \psi_{\text{sq. cat}}\left(\frac{q_1 - q_2}{\sqrt{2}}\right) \\ &\propto e^{-\frac{q_1^2}{2e^{-2r}}} \left(e^{-\frac{(q_2 - \alpha)^2}{2e^{-2r}}} + e^{-\frac{(q_2 + \alpha)^2}{2e^{-2r}}} \right) + \left(e^{-\frac{(q_1 - \alpha)^2}{2e^{-2r}}} + e^{-\frac{(q_1 + \alpha)^2}{2e^{-2r}}} \right) e^{-\frac{q_2^2}{2e^{-2r}}}. \end{aligned} \quad (16)$$

Finally since $\langle p| = \int dq e^{-ipq} \langle q|$, a p -quadrature measurement on mode 2 with homodyne outcome p_m corresponds to an integration of the wavefunction along q_2 with a phase $e^{-ip_m q_2}$:

$$\int dq_2 e^{-ip_m q_2} \Psi(q_1, q_2) \propto e^{-\frac{(q_1 + \alpha)^2}{2e^{-2r}}} + 2 \cos(p_m \alpha) e^{-\frac{q_1^2}{2e^{-2r}}} + e^{-\frac{(q_1 - \alpha)^2}{2e^{-2r}}} \quad (17)$$

Thus we get a 3-peak state with relative peak heights $\{1, 2 \cos(p_m \alpha), 1\}$ and peak spacing $\alpha = d/\sqrt{2}$. Note that the quality of the state as an approximate GKP state depends on the distribution of the peaks and thus on the homodyne outcome of the breeder, p_m . However, as shown in Fig. S4, the homodyne outcomes yielding large effective squeezing values coincide with those occurring with high probability, and so the protocol succeeds with high probability. Furthermore, as shown in Ref. [8], iterating the protocol further reduces the probability of producing poor output states, assuming one starts with high-enough quality squeezed cat states.

Iterating the breeding protocol can allow us to further increase the number of peaks in the q quadrature. An example is shown in Fig. S2d where by applying a balanced beamsplitter to a pair of states with three peaks and measuring the p quadrature of mode 2 we end up with a state with five peaks (again with a lattice spacing shrunk by a factor $\sqrt{2}$). In general, breeding a pair of n -peak states yields a $(2n - 1)$ -peak output state with a lattice spacing reduced by $\sqrt{2}$. As we will discuss in Sec. IV, this means we can initialize the breeding protocol directly with states that already have more than two peaks, which allows us to do fewer rounds of breeding overall, and to start the breeding protocol with states that are more likely to succeed at producing a GKP state.

We can further generalize this protocol to what we call adaptive breeding. This allows us to breed any two multipeak states independent of the spacing of the lattices and the number of peaks by choosing a beamsplitter angle that depends on the lattices of the input states ($\theta = \arctan(d_1/d_2)$ with d_i the lattice spacing of the mode i input) as shown in Fig. S2e. Additional intuition is provided in Figs. S2b-d: from Figs. S2c-d it is clear that if we were to squeeze the input states to a common lattice, then a static 50:50 beamsplitter will always work, and in Fig. S2b we show that squeezing the states before the beamsplitter is equivalent to changing the beamsplitter angle and squeezing the states after the beamsplitter. Given that the spacing of the output states of the GBS depends on the photon count outcome, which is probabilistic, adaptive breeding allows us to breed many GBS outputs from shot to shot directly without needing to squeeze all the input states to a common lattice (which is a costly and noisy operation), as has previously been proposed [9]; rather, we can apply one final squeezing operation to the output of the breeding circuit.

In order to distill a high quality GKP state from several multipeak input states we repeatedly apply the unit cell (Fig. S2a) with the output state being fed into the next unit cell. The way we concatenate those unit cells will give us different breeding networks. Two examples of those networks are shown in Fig. S3. In Sec. IIIB, we will see that all these configurations produce equivalent distributions of output states.

However, one cannot keep performing this protocol indefinitely in order to increase the symmetric effective squeezing arbitrarily. While each round of breeding adds peaks in the q

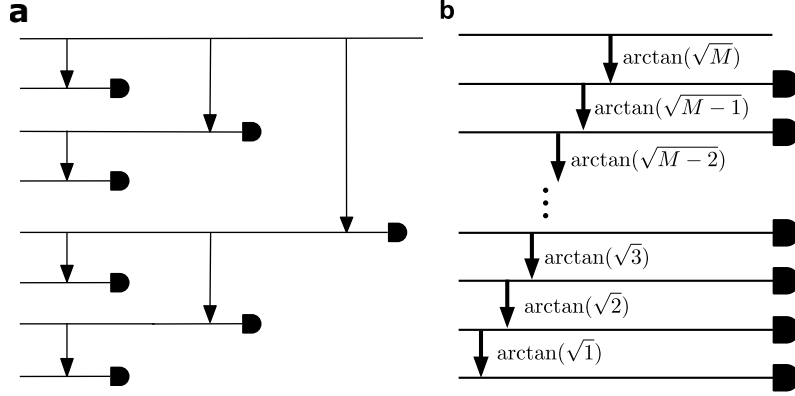


Figure S3. **a. Binary tree breeding network of 8 modes. b. Linear cascade breeding network of $M + 1$ modes.** Arrows without labels are 50:50 beamsplitters. Here the beamsplitter angles were set assuming equal spacing lattices of the input states.

quadrature, the spacing between the peaks is also reduced. The increased number of peaks improves the effective squeezing along the p quadrature (see Sec. II C), while the reduced peak spacing decreases peak resolvability, and thus degrades the effective squeezing along the q -quadrature. As a result, the symmetric effective squeezing is optimal after a finite number of rounds, which generally occurs once the effective squeezing in both quadratures of the states are roughly equal.

As a rule of thumb, the effect of the breeding protocol on multi-peak input states of the form in Eq. 11 is to transfer ~ 3 dB (on average) of effective squeezing from the q quadrature to the p quadrature. This can be explained intuitively by the $\sqrt{2}$ contraction of the spacing of the peaks in the q -quadrature lattice in each layer of breeding in the binary tree network, which translates to a half factor reduction (or -3 dB) of effective squeezing in q quadrature (see Eq. 4). In the p -quadrature representation, breeding can be viewed as the multiplication of the momentum wavefunctions (see Sec. V B 2). Squaring a peaked wave function results in a half factor reduction of the peak variance and therefore in a factor of 2 (or +3 dB) increase of the p -effective squeezing. This 3 dB transfer heuristic, also supported by numerical evidence, is another argument for why we cannot perform arbitrarily many rounds of breeding, since at some point the effective squeezing in the p quadrature will begin to exceed the q quadrature. Moreover, from Sec. II C, we know that squeezed cats only begin with ~ 3.5 dB of effective squeezing in p , so we need on average three rounds of breeding in the binary tree (or 8 squeezed cats) to ensure the effective squeezing in the p quadrature is greater than 9.75 dB.

Finally, we note that the homodyne measurements in the breeding protocol can in general provide the added benefit of purifying the output state. If the input states to the breeding protocol are a probabilistic mixture, then each combination of states in the mixture can in general lead to distinct homodyne distributions. Thus, the homodyne outcomes in one shot of the protocol can provide partial information about which subset of states were bred in that shot, since the outcomes will in general be more likely to have come from one of the homodyne distributions over another. As an illustrative example, consider initializing the protocol with a mixture of either two even squeezed cats, or one even and one odd squeezed cat. In Fig. S4(a), we see that the homodyne distributions associated with these two scenarios are very different—where the even+even distribution is peaked, the even+odd

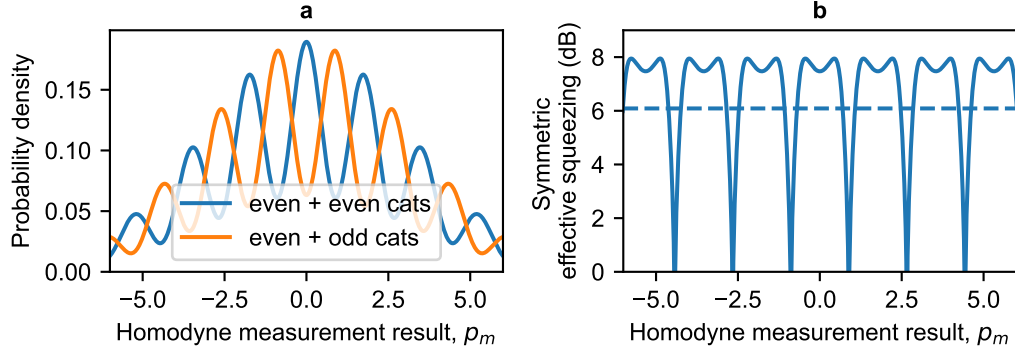


Figure S4. **a. Homodyne outcome probability density** of the breeding circuit of Fig. S2a when inputting two squeezed cat states with $\alpha = \sqrt{\pi/2}$ and $r = 1.15$ (10 dB). Even cats are defined in Eq. (13), and odd cats are defined similarly, but with a relative phase of $-\pi$ between the two displacements. **b. Symmetric effective squeezing as a function of homodyne outcomes** for the breeding circuit, inputting the even cats of (a). The dashed line indicates the symmetric effective squeezing of the input even cat state before breeding. The regions of good symmetric effective squeezing coincide with the high-probability homodyne outcomes, ensuring the high success probability of the protocol.

is not, and vice versa. Thus, were we to find a homodyne outcome e.g. at $p = 0$, then it is more likely that we bred the even squeezed cats, which will purify the output state. Note that it is especially relevant that parity differences in the input states lead to distinct homodyne distributions in the breeding protocol, since losing odd numbers of photons in the GBS devices leads to parity flips on the output states; therefore, breeding, and considering the joint information provided by both the photon counting and the homodyne detections, allows us to identify strings of outcomes which probabilistically lead to less noisy output states, which in turn can be selected by multiplexing in the QPU chip, and allows us to tolerate more loss in the GBS devices.

B. Combining multiplexing and breeding

In our architecture, we use GBS devices as probabilistic sources of a set of non-Gaussian states. The different states in this set are heralded by the photon count outcomes, as will be discussed in more detail in Sec. IV. Some of these non-Gaussian output states will be refined via breeding (as explained in III A) into high quality GKP states with good enough symmetric effective squeezing for fault-tolerant quantum computation (see Sec. VI). However, not all GBS outputs are useful for the breeding protocol, so in this sense, the GBS devices are probabilistic sources of “refinable” states. A way to deal with the probabilistic nature of the sources is to use multiplexing.

The multiplexer (MUX) is a device that accepts N states (the outputs of copies of the N GBS devices in our case), and associated information labeling these states (in our case, the photon number detection outcomes) and then—according to a ranking function—selects and outputs along a subset of size M of those states, discarding the rest, (see left of Fig. S5). The probability of success at the output of the multiplexer depends on the number of

inputs as well as the input success probability. This multiplexed probability can be set as close to unity as desired by increasing the number of inputs. If the probability of success of one GBS device is denoted by P_{GBS} then, given N independent and identical inputs at the multiplexer, the probability of success after an N -to- M multiplexer, P_{MUX} , is given by the binomial distribution

$$P_{\text{MUX}}^{N\text{-to-}M} = \sum_{k=M}^N \underbrace{\binom{N}{k} P_{\text{GBS}}^k (1 - P_{\text{GBS}})^{N-k}}_{\text{probability that exactly } k \text{ GBSs succeed}}, \quad (18)$$

where the factor $\binom{N}{k}$ takes into account all the different ways (permutations) that k GBS devices can succeed within a total of N devices. In the particular case of an N -to-1 MUX, the equation above can be simplified to

$$P_{\text{MUX}}^{N\text{-to-}1} = 1 - (1 - P_{\text{GBS}})^N. \quad (19)$$

The binary tree circuit with adaptive beamsplitters acting as switches (see Fig. S5b) is a simple way to implement an N -to-1 MUX given its $\log_2 N$ circuit depth. Other optical circuits that implement an N -to- M MUX may result in impractical overheads when increasing the number of inputs. This is either because of the need of additional modes and/or optical depths of beamsplitter switches or crossings scaling as $O(N)$ or as $O(N^2)$ depending on the particular circuit (see Ref. [19] for MUX based on switch fabrics and Ref. [20] for MUX based on generalized Mach-Zehnder interferometers).

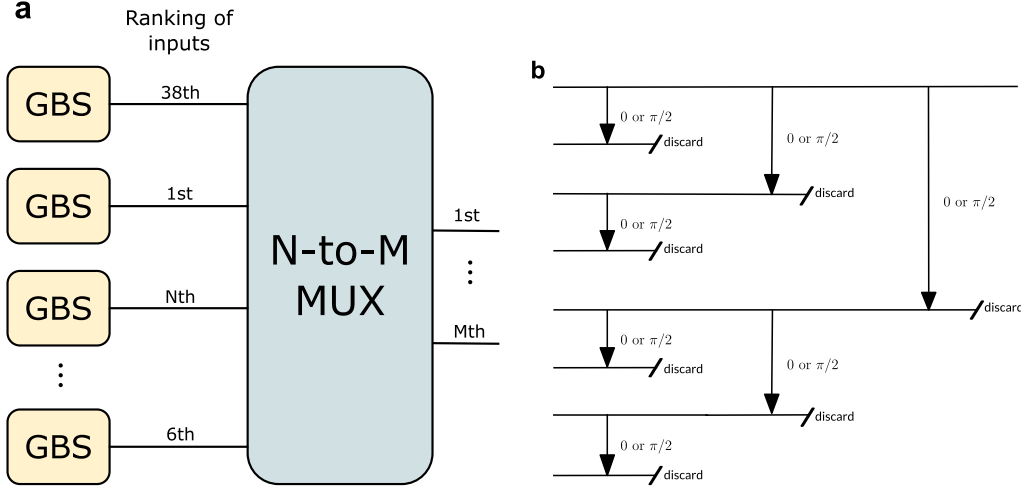


Figure S5. **a. Conceptual illustration of an N -to- M Multiplexer. b. Binary tree configuration of adaptive beamsplitters implementing an N -to-1 multiplexer.**

The conceptually simplest option for the implementation of the refinery would be to employ M $\frac{N}{M}$ -to-1 multiplexers, with each of the outputs acting as an input for an M -to-1 breeding circuit. Such a configuration could be constructed out of an N -to-1 binary tree of adaptive beamsplitters, where the last layers of the tree perform breeding. However, if we

revisit the unit cell of the binary tree MUX, and compare it with the unit cell of the adaptive breeding protocol, from Fig. S6 we see that the only difference between the two is that in the unit cell for breeding the adaptive beamsplitter angle is between 0 and $\pi/2$ such that the states are interfered, and that one of the outputs is measured with a homodyne detector. Thus, if one built the entire binary tree out of the unit cell of the adaptive breeding protocol, then at any beamsplitter one has the option to multiplex or to breed, with the beamsplitter either acting as a switch or to generate interference between the states. Note that if a unit cell implements MUX (a switch), then the outcome from the homodyne detector is simply discarded.

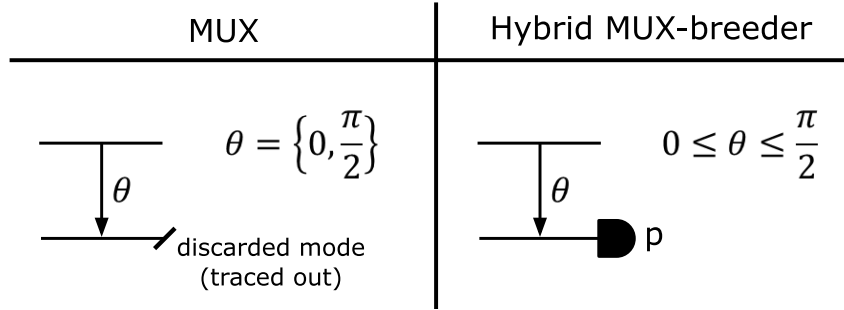


Figure S6. **Modification of the unit cell of the binary tree MUX to a hybrid MUX-breeder configuration.** In the MUX configuration the beamsplitter angle only takes two discrete values, 0 and $\pi/2$, acting as a switch. One of the output ports is routed to the next layer and the other port is discarded. In the hybrid MUX-breeder configuration, the beamsplitter angle takes a value between $0 \leq \theta \leq \pi/2$. It can still function as a MUX (i.e. $\theta = 0$ or $\pi/2$) if only one of the states in the input port is selected or can behave as a breeder by interfering the input states and performing adaptive breeding assisted by the homodyne detector placed in the port of the formerly discarded mode.

This hybrid MUX-breeder is formally equivalent to an N -to- M MUX followed by breeding because it allows us to select any M modes from the N inputs of the circuit and breed them together. This is more resource-efficient than using $M \frac{N}{M}$ -to-1 multiplexers, followed by an M -to-1 breeding circuit, because the probability of success of an N -to- M MUX is higher than $M \frac{N}{M}$ -to-1 multiplexers. An illustration of the functioning of this circuit is given in Fig. S7. The figure on the left shows a circuit with $N = 2^4 = 16$ inputs from which we have decided to breed 7 of them (marked with stars). The MUX beamsplitters (dashed arrows) act as switches set to reroute the bottom input states to pass the selected state to the next layer. When two selected states meet at the input port of a beamsplitter (thick arrows), the angle is configured to interfere the states according to the adaptive breeding protocol and breed them using the homodyne detector. The remainder of the beamsplitters in the binary tree (thin arrows) are set to transmit the top input state, since in some cases (θ_{13}) this corresponds to keeping the desired state from the two inputs, and in other cases that state will be discarded regardless, and so the beamsplitter angle is unimportant. This process can be done when selecting any number $M \leq N$ of states to breed, which can begin in any position at the input ports.

While the circuit on the left of Fig. S7 is what we would want to implement, it produces the same distribution of output states as the middle circuit. To see this, we first use the identity in Fig. S2b to set specific beamsplitter angles (absorbing the squeezing operations onto the input states so that they all share a common lattice). Assuming the input states now lie on a common lattice, the beamsplitter settings in the left circuit pictured ensure

the quadrature distribution of the output state of the breeding protocol has peaks that still lie on a lattice. Additionally, we relabel the modes such that the modes to breed are positioned in the first inputs of the circuit. The sub-circuit composed only of the modes that mix through the passive unitary $\hat{U}_B$ marked in a box is the effective breeding network that is implemented. Finally the last circuit in Fig. S7 shows that in fact, the binary tree hybrid MUX-breeder circuit can simulate any breeding network on the selected states. This is because the passive unitaries that describe all the possible breeding networks (see examples in Fig. S3) have in common that, in the Heisenberg picture, the output mode is an equal linear combination of all the input modes being bred in the circuit, $\hat{a}_{\text{out}} = \frac{1}{\sqrt{N}} \sum \hat{a}_k$. This means that, for instance, the linear cascade network interferometer $\hat{U}_{LC}$ and the interferometer $\hat{U}_B$ are related by $\hat{V}\hat{U}_{LC} = \hat{U}_B$, where $\hat{V}$ leaves the first mode unchanged. The $\hat{V}$ unitary is a real interferometer that does not mix the $\hat{p}$ operators with the $\hat{q}$ operators and since all the modes are measured in the p quadrature right after applying $\hat{V}$, we can absorb the $\hat{V}$ unitary into the measurement (by taking linear combinations of the outcomes) making $\hat{U}_{LC}$ and $\hat{U}_B$, together with the measurements, equivalent breeding circuits.

We also note that this hybrid MUX-breeder configuration is compatible with other protocols within the binary tree, such as cat state amplification [21], which could be useful for probabilistically converting two low quality cat states generated by some photon number outcomes of the GBS device into higher quality cat states that are useful for breeding.

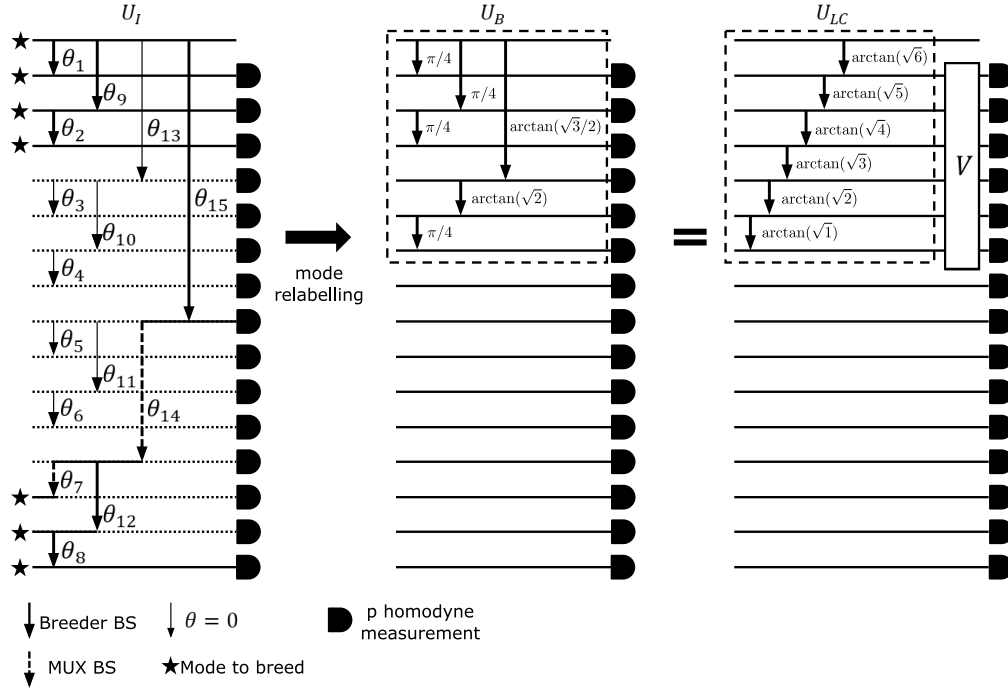


Figure S7. **Circuit identities for Hybrid MUX-breeding.** The first circuit shows how the Hybrid MUX-breeding binary tree would be programmed in practice. Conceptually, it is equivalent to an N -to- M MUX, followed by an M -to-1 breeder. The next two circuits are equivalent as discussed in the text, which demonstrates that all possible permutations of the input states lead to the same distribution of output states, simplifying the analysis of the protocol.

Finally we need to specify the selection rule of the MUX, i.e. what is going to be the ranking function that determines which states the MUX selects. In our case, the ranking function R is the sum of the effective squeezing in q and p and the symmetric effective squeezing, all in dB units (see Sec. II):

$$R = \sigma_q^2[\text{dB}] + \sigma_p^2[\text{dB}] + \sigma_{\text{symm}}^2[\text{dB}]. \quad (20)$$

The MUX will select the state with the highest value of the ranking function. This heuristic was chosen because we know that breeding transfers effective squeezing from the q quadrature to the p quadrature, so we may want to favour states that start with high amounts in both, and because we know that our GBS devices generally produce approximate GKP states with more effective squeezing in the q quadrature compared to the p quadrature. The symmetric effective squeezing is included in the ranking function to bias towards states that have coherence between the peaks, as opposed to just squeezed states or mixtures of squeezed states that could still have very high effective squeezing in one quadrature.

C. Measurement-based squeezing and Bell pair generation

The heralded generation of non-Gaussian states in the GBS devices, together with the adaptive MUX-breeder in the refinery, produce approximate GKP states which have different phase space lattices from shot to shot. These lattices are determined by the photon number measurement history of the state. In order to generate GKP Bell states by interfering two GKP states at a balanced beamsplitter, the two states need to be on the square-grid quanaught state lattice, up to a heralded displacement. To align the output states to this lattice, we need to apply inline squeezing to them. In our architecture, any deviation from the lattice by a heralded displacement can always be handled in post-processing, simply by conjugating the displacement through the cluster state stitching network and updating the final homodyne measurements on the cluster state accordingly.

In hardware, while we have direct access to squeezing operations on vacuum states via non-linear ring resonators, it is more challenging to squeeze an arbitrary state using a ring resonator to apply a squeezing operation given the difficulties of mode matching the involved fields. Rather, a viable option for inline squeezing involves interfering the arbitrary state with an ancillary squeezed state prepared offline and performing a homodyne measurement on one of the outputs (see Fig. S8). This applies a noisy inline squeezing operation to the arbitrary state, where the noise depends on how much squeezing one wishes to apply, the level of squeezing in the ancillary state, and loss in the circuit including the homodyne measurement [22]. This process is commonly referred to as measurement-based (MB) squeezing.

The MB squeezing process, when averaged over the homodyne outcomes and their associated feedforward displacement (applied in post-processing of the cluster state), can be described by a Gaussian CPTP channel (see Sec. II) with X and Y matrices (for details see appendix H in [23]) given by

$$\begin{aligned} &\text{for } q\text{-squeezing} \\ X &= \begin{pmatrix} \cos \theta & 0 \\ 0 & \sec \theta \end{pmatrix} \quad Y = \begin{pmatrix} V_{\text{sqs}} \sin^2 \theta & 0 \\ 0 & \frac{1}{2} \frac{1-\eta_h}{\eta_h} \tan^2 \theta \end{pmatrix} \end{aligned} \quad (21)$$

$$\begin{aligned} &\text{for } p\text{-squeezing} \\ X &= \begin{pmatrix} \sec \theta & 0 \\ 0 & \cos \theta \end{pmatrix} \quad Y = \begin{pmatrix} \frac{1}{2} \frac{1-\eta_h}{\eta_h} \tan^2 \theta & 0 \\ 0 & V_{\text{sqs}} \sin^2 \theta \end{pmatrix} \end{aligned} \quad (22)$$

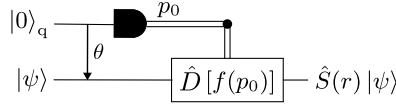


Figure S8. **Circuit that implements MB squeezing.** An ancillary $|q=0\rangle$ eigenstate interferes with the target state $|\psi\rangle$ at a beamsplitter, followed by a p -homodyne measurement and a feedforward displacement. This implements a perfect squeezing operation that amplifies the p quadrature and squeezes the q quadrature (in the Heisenberg picture) as $\hat{q} \rightarrow \cos(\theta)\hat{q}$ and $\hat{p} \rightarrow \sec(\theta)\hat{p}$ where θ is the beamsplitter angle. Losses and finite squeezing in the ancillary state introduce noise to the target state. Anti-squeezing of the q quadrature (and p -squeezing) can be achieved with an ancillary $|p=0\rangle$ and q -homodyne. The feedforward displacement operator is defined in Eq. 23.

where $\cos \theta = e^{-r}$, $r \geq 0$ being the parameter of the squeezing applied, η_h the efficiency of the homodyne detector, and V_{sqz} the variance of the squeezed quadrature of the ancilla. To derive the map above, we have chosen the feedforward displacement as

$$\hat{D}[f(p_0)] = \exp\{-ip_0\hat{q}\tan(\theta)/\sqrt{\eta_h}\} \quad (23)$$

in order to eliminate the anti-squeezed fluctuations of the ancillary mode.

The introduction of a squeezing operation means that, given that squeezing and losses do not commute with each other, the effect of the loss is broken in two, before (what we have called P_2 path) and after the MB squeezing (defined as the P_3 path in the main text). The assumption of homogeneous losses allows us to commute the losses through the linear optical components of the architecture, but the MB squeezing operation is a channel in the state factory which does not commute with loss, and hence marks a point where it is natural to divide into two independent paths.

IV. THEORY: GAUSSIAN BOSON SAMPLING STATE PREPARATION DEVICES

We use Gaussian boson sampling (GBS) devices to produce the non-Gaussian states that can be refined into GKP states. In general, this consists of m bosonic modes, each with a single-mode Gaussian state input, a universal linear interferometer and finally photon number measurement on $m-1$ of the modes. Here we derive a family of m -mode circuits that prepare multipeak states that can be refined into approximate GKP states. We also explore how to optimize the parameters of these circuits and the advantages and disadvantages of more modes in the GBS. We begin with the simplest device, the two-mode GBS, which is used as a building block for this family of circuits.

A. Two-mode GBS devices to make squeezed cat states

There are multiple proposals and experimental demonstrations for preparing approximate squeezed cat states using specific configurations of two-mode GBS [10, 24–27], depicted in Fig. S9. One such configuration, photon subtraction [24, 25], comprises sending a squeezed vacuum state into one port of a beamsplitter with high transmissivity and measuring the photon number in the reflected mode. Conditional on any non-zero measurement, the output state is non-Gaussian and an approximation to a squeezed cat state, with the parity of the photon number measurement setting the parity of the cat, and higher photon number

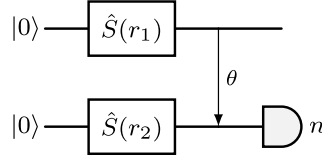


Figure S9. **Two mode GBS for generating squeezed cat states.** The photon number outcome n sets the parity of the squeezed cat, and higher n yield squeezed cats with larger amplitudes (spacing between the peaks).

outcomes yielding larger amplitude squeezed cats. However, due to the low reflectivity required for the beamsplitter, the heralding probability is impractically low for use in quantum computing. Generalized photon subtraction [27] was proposed to improve the success probability. In this protocol, a squeezed state is input into both arms of the beamsplitter that is chosen to satisfy $e^{2r_1} \sin^2 \theta + e^{2r_2} \cos^2 \theta = 1$ with $r_1 \neq r_2$ and $\theta \neq \frac{1}{2}n\pi$ (as parametrized in Fig. S9). We refer to this condition as the ‘monomial condition’ as it is chosen so that the q -quadrature wavefunction of the state can be expressed as a monomial in q , multiplied by a Gaussian in q (the general case is a polynomial times a Gaussian). The parametrization at the monomial condition ensures that the fidelity to a targeted squeezed cat state approaches 1 as the number of detected photons goes to infinity. However, since we would like to use the output states of our GBS devices as input states for the breeding protocol, tuning the GBS circuit parameters to produce as high quality squeezed cat states as possible is not necessarily optimal. Rather, we want to set the GBS parameters to prepare the best states relative to how useful they are for the breeding protocol.

We are interested in preparing a distribution of states that results in an average state with the highest symmetric effective squeezing at the output of the state factory, which is not the same as maximizing the symmetric effective squeezing for the output states of the two-mode GBS devices before the breeding protocol. We find that we can increase the beamsplitter angle so that it is greater than the one set by the monomial condition and consequently increase the separation between the peaks in the q quadrature and hence the resolvability of the peaks in the p quadrature. This comes at the expense of introducing small distortions to the Wigner functions; i.e. the states are worse approximations to squeezed cats. In Fig. S10a-c we show the output states from three parametrizations of the two-mode GBS device. The circuit at the monomial condition with $\theta = 0.3$ prepares states with no distortion but fewer peaks in the p quadrature (see Fig. S10a); $\theta = 0.45$ produces states with a higher symmetric effective squeezing due to the increased spacing of the peaks in the q quadrature/increased number of peaks in the p quadrature and only minimal distortions (see Fig. S10b); however $\theta = 0.5$ is too high and here the symmetric effective squeezing decreases again due to the distortions becoming more dominant (see Fig. S10c). So if we restrict ourselves to a state with only two peaks in the q quadrature, $\theta = 0.45$ would be the best of these three options. However, if we consider breeding two copies of these states together and post-selecting on a zero outcome in the homodyne detector, more distortions can be tolerated. As seen in Fig. S10d-f, breeding states from the circuit with $\theta = 0.5$ results in a state with the highest symmetric effective squeezing. We also find that the optimal beamsplitter angle depends on how many photons are detected. As we cannot know in advance what the detected photon number will be, the beamsplitter angle must be independent of the photon number resolving (PNR) measurement. Therefore, it is not possible to fix the optimal angle simultaneously for all photon number outcomes and we choose the beamsplitter angle based on the most likely and useful outcomes. Therefore the

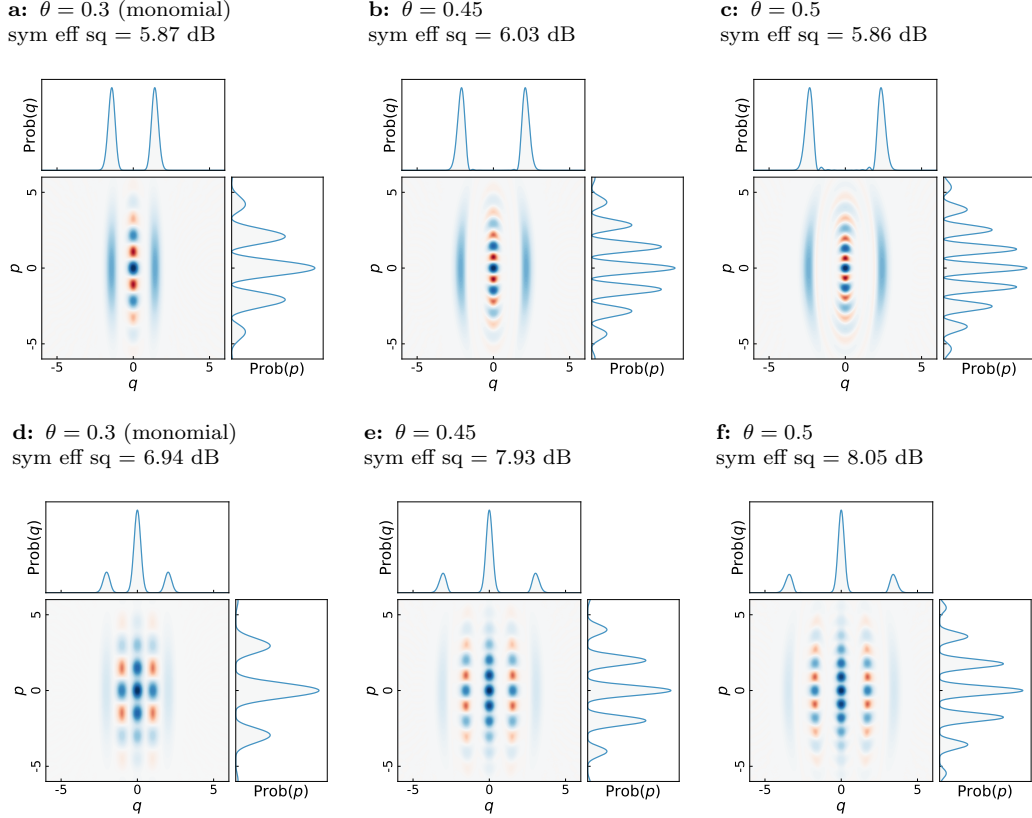


Figure S10. The Wigner functions, marginal q and p homodyne distributions of states created by two-mode GBS with and without breeding. The top row (**a**, **b** and **c**) shows the states from two-mode GBS (see Fig. S9 for notation) detecting 10 photons with $r_1 = -r_2 = 1.15$ (10 dB of squeezing) and beamsplitter $\theta = 0.3, 0.45$, and 0.5 respectively. The bottom row (**d**, **e** and **f**) shows the state after one round of breeding two copies of the state above, post-selecting on measuring zero in the homodyne detector, which is equivalent to a 3-mode GBS as shown in Sec. IV B. The symmetric effective squeezing (sym eff sq) of the states are shown above each plot. We see from **c** being converted to **f** that the best state for breeding is not necessarily the best approximation to a squeezed cat.

optimal GBS parameters depend on how many rounds of breeding the state will undergo as well as how much multiplexing there is, as this effectively changes the probability distribution of the GBS device outcomes. Furthermore, we find that there is a line in r_1 - r_2 - θ space along which the output state differs only by a squeezing operation and this can be used to target a specific q -quadrature spacing; we use this to ensure that for a given factory, the distribution of lattices (over the PNR patterns) right before the MB squeezing operation is already close to the square-grid quanaught state lattice, leading to less squeezing and noise being introduced by the MB squeezer.

B. A family of m -mode GBS devices for making m -peak approximate GKP states

The only GKP-like states (with high effective squeezing) that two-mode devices can produce are states with two peaks. To create states with more peaks we could use breeding as described in Sec. III A, which is a probabilistic process involving a Gaussian measurement. Alternatively, we can consider using a GBS device with more modes to directly generate approximate GKP states with more peaks. Here, we derive such a GBS device starting from breeding the outputs of two-mode GBS devices.

Consider the breeding circuit described in Sec. III A, but instead of the usual scenario, we can consider applying the Gaussian measurements before the non-Gaussian photon number measurements. By applying the Gaussian measurement (post-selecting on a particular outcome), we reduce the multimode Gaussian state to another Gaussian state across fewer modes followed by photon number measurements, which is precisely a GBS device. This results in a family of GBS devices, shown in Fig. S11g, that produce the same output states as though we had bred together the outputs of two-mode GBS devices (though with different heralding probabilities). As we will see, this family has many nice properties: the construction of the circuit is clear for any number of modes, with an extra mode corresponding to the addition of a peak in the post-selected output states. Note that to continue producing higher quality states with more modes, we require detecting more photons, otherwise the output state will be equivalent to a breeding circuit with too many rounds of breeding (as described in Sec. III A). In theory, this gives us a recipe for creating an arbitrarily good GKP state (under lossless conditions) by adding more modes and post-selecting on higher photon number outcomes; however, this comes at the cost of a diminishing probability. The interferometer is non-universal with the number of beamsplitters scaling linearly with the number of modes. Finally, there are many different multi-mode photon number outcome combinations that generate useful states that are either already good enough quality GKP states or that can be refined into one via breeding. Compared to circuits with the same number of modes, but which require post-selection on only one photon number outcome [14], the members of this family of circuits have a greater probability of generating useful states, reducing the burden on MUX. We do not claim optimality of these GBS circuits; however, they are very convenient, both for understanding in theory and for numerical optimization (they have fewer parameters than general GBS circuits).

The derivation of this family of circuits is pictured in Fig. S11 and outlined here:

1. Apply the breeding circuit for two input states iteratively with the output state of each round being the input state of the bottom mode in the next round. The circuit is shown in Fig. S11a. This involves alternating squeezed vacuum input states with parameters r_1 and r_2 , followed by pairwise beamsplitters and PNR detectors to make the M squeezed cat states. Then there is a cascade of beamsplitters which are parametrized depending on the photon number outcomes and finally some Gaussian measurement on all but the top mode. We could have chosen to breed the states in any order rather than always adding the new state in the top mode, but this ordering results in a simple iterative rule for mapping another round of breeding to another mode in the GBS device.
2. We can introduce resolutions of the identity operation by a series of beamsplitters and their inverses, highlighted by the coloured arrows in Fig. S11b, starting with the orange arrows and ending with the blue.
3. Find every pair of beamsplitters that act the same on the bottom and top modes of two adjacent two-mode devices. These are colour coordinated in Fig. S11c.

4. Starting with the left-most pair (orange), use the fact that identical mean-preserving channels on both modes commute with a beamsplitter to commute both highlighted beamsplitters with the earlier ones to be applied directly on the squeezed vacuum states. The circuit after commuting the first pair is shown in Fig. S11d. Repeat for each pair in order, finishing with the blue ones.
5. Use the property that a beamsplitter acting on two identical squeezed vacuum states does nothing, so we can remove these beamsplitters to end with the circuit shown in Fig. S11e.
6. We can now pick out a series of two-mode circuits with a Gaussian measurement on the top mode, shown in the blue boxes in Fig. S11f. Here we can use the rule for partial Gaussian measurements to find a new single-mode Gaussian state after measuring some outcome [28]. In the event that we choose the outcome to be zero displaced, the single mode state will be some squeezed state.
7. The final circuit is shown in Fig. S11g. It consists of $M + 1$ modes with up to three different squeezing strengths, M beamsplitters and M photon number detectors. Due to the stepwise arrangement of beamsplitters, we refer to this family as the staircase GBS. The same structure of circuit was presented in Ref [29]; here, we have shown how to relate such a structure to the breeding protocol such that we can explicitly relate high quality output states from breeding to GBS circuits capable of producing the same state.

The above derivation holds for the general circuit shown with any Gaussian measurement with a zero mean outcome, any beamsplitter angles and any r_1 and r_2 . In practice, these parameters are chosen in the same manner as for the breeding circuit with r_3 being a function of r_1 , r_2 , θ and the Gaussian measurement. Note that a measurement onto a non-zero mean Gaussian state would result in the input states being displaced squeezed states. In breeding, the beamsplitter parameters ϕ are usually chosen conditional on the photon number outcomes; however, as the order of the measurements is reversed here, we must instead decide which photon number outcomes to post-select on to determine the breeding circuit. We pick the breeding circuit to target outcomes with n photons in every measured mode. This is because the circuit is the same for every n (the beamsplitters depend on only the ratio of the lattice spacing of the input squeezed cat states, so if all the input states came from the same n , then they will all have the same lattice). Therefore, there are many possible outcomes that are useful, which boosts the probability of an acceptable state. Although the circuit is targeted towards these outcomes, there are many other outcomes that also yield states with high symmetric effective squeezing such as $(n, n + 2)$, $(n + 2, n)$ and $(0, n)$ for three modes. We note that by allowing multiple photon number combinations, the lattice spacing of the output state changes depending on the photon number outcomes, meaning MB squeezing is needed to align all the states to the square-grid quanaught state lattice.

The chosen values for parameters r_1 , r_2 and θ affect the quality of the squeezed cat state that would have been bred in the equivalent circuit from Fig. S11a. As shown above, these parameters can be optimized depending on how many rounds of breeding the states will undergo, i.e. depending on the number of modes in the equivalent GBS, as well as the targeted lattice spacing of the output state. The Gaussian measurement in the circuit mapping is usually either p -homodyne or more generally generaldyne (projection onto a squeezed coherent state basis), and the choice of the squeezing parameter of the generaldyne measurement determines the peak heights of the output state. We post-select on the outcome with zero displacement in either case as this results in a GBS device with no displacement

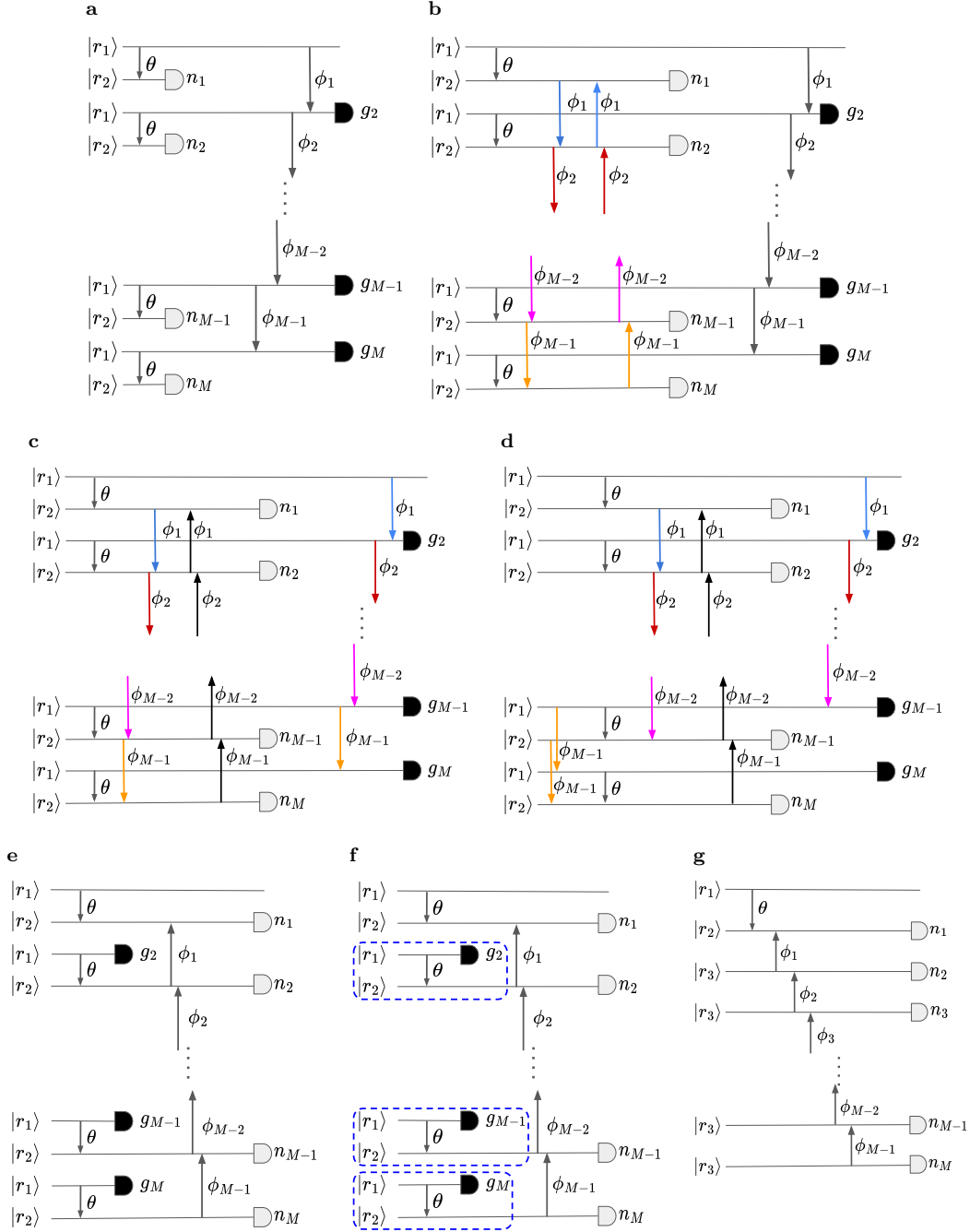


Figure S11. Steps to derive the staircase circuit from multiple rounds of breeding. From this derivation, we find a family of GBS devices which can produce the same output states as breeding the output states of two-mode GBS devices, meaning, for many photon number outcomes, the output states are approximate GKP states. The grey detectors are number-resolving measurements and the black detectors are any general Gaussian measurement. All arrows are beamsplitters and the colours only identify pairs useful in the derivation as explained in Sec. IV B

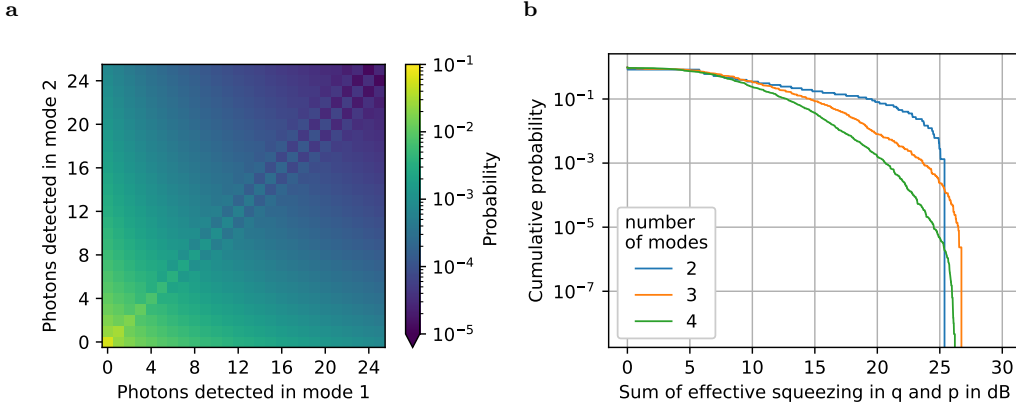


Figure S12. (a) The photon number measurement probabilities of a three-mode staircase GBS parametrized with input squeezing 15 dB, -15 dB and -13.7 dB in each mode and beamsplitters $\theta = 0.4833$ and $\theta = \frac{\pi}{4}$. Near the region of interest, where $x = y$, the probability of an odd number of photons is significantly smaller than the probability of the same number of photons in each measured mode. This property explains why the staircase GBS are quite robust to parity errors due to photon loss before the photon number detectors. (b) The distribution of the quality of states output from 2- to 4-mode GBS. This plot shows the probability of outputting a state of at least the quality given on the x -axis. We are only interested in the potential of outputs to make approximate GKP states because they can be bred further in the refinery. Therefore, we use the sum of the effective squeezing in q and p , i.e. $\sigma_q^2[\text{dB}] + \sigma_p^2[\text{dB}]$, as the quality of a state before refinement. As breeding approximately moves 3 dB of squeezing from the q to p effective squeezing and the final state needs at least 9.75 dB in each quadrature, states with $\sigma_q^2[\text{dB}] + \sigma_p^2[\text{dB}] \gtrsim 20$ dB are likely to be useful states.

on the inputs. The measurement parameters only affect the value of r_3 in the reshuffled GBS circuit and we numerically optimize this value independently.

C. Loss tolerance and success probabilities

Now that we have a family of m -mode GBS devices, we would like to find the best number of modes to use for the GBS in the state factory in conjunction with the hybrid MUX-breeder. In particular, we want the factory to be as tolerant to loss as possible. Conditional on certain measurement outcomes, an m -mode staircase GBS is equivalent to breeding together $m - 1$ outputs from two-mode devices. Therefore, roughly speaking, we can trade one round of breeding for an additional mode in the GBS to end up with a state of the same quality for a given photon number outcome. However, due to the post-selected Gaussian measurement in the GBS derivation and the adaptivity in breeding, we shall see below that there are advantages and disadvantages of increasing the number of modes in the GBS or the number of rounds of breeding.

As explained in Sec. (III A), certain homodyne outcomes in breeding result in output states that are higher quality, effectively partially reducing the effects of loss in the path to the photon number measurements from the GBS devices. When employing a breeding circuit in the refinery, we utilize MUX in the QPU to post-select on “good” homodyne outcomes in the refinery, restoring higher effective squeezing in the output states. However, as

the staircase GBS family is derived from reshuffling the breeding circuit with post-selected Gaussian measurements, we can choose to derive the circuit equivalent to post-selecting on the best outcome. Thus, on average, the state output from the GBS is more robust to loss before the photon number measurements than from circuits made from both GBS and breeding, as long as the chosen Gaussian measurement outcome is selected carefully. We find that post-selecting on $p = 0$ or $(q, p) = (0, 0)$ for homodyne and generaldyne measurements respectively does lead to better tolerance to loss before the photon number measurements. This can be seen by observing the probability distribution of the photon number measurements. Let $\rho_{d_1, d_2}^{(\eta)}$ denote the output state of a 3-mode GBS d_i photons detected in mode i and loss with transmissivity η acting before the photon number measurements. This can be written as

$$\rho_{d_1, d_2}^{(\eta)} = \sum_{n_i \geq d_i} \Pr(n_1, n_2 | \eta = 1) \Pr(d_1, d_2 | n_1, n_2, \eta) \rho_{n_1, n_2}^{(\eta=1)} \quad (24)$$

where the first factor is the probability of detecting (n_1, n_2) photons in the absence of loss, and the second term is the probability of losing $n_1 - d_1$ and $n_2 - d_2$ photons due to the loss. To reduce the effects of loss, we want to reduce the coefficients of the terms with $n_i > d_i$. For small enough losses, $\Pr(d_1, d_2 | n_1, n_2, \eta)$ is very small for differences of more than one photon. Therefore, we want $\Pr(n_1, n_2 | \eta = 1)$ to be as small as possible for $(n_1, n_2) = (d_1 - 1, d_2)$ and $(d_1, d_2 - 1)$. Fig. S12a shows the probability of detecting n_1 and n_2 photons in modes 1 and 2 of a three-mode staircase GBS device. Around the photon number patterns of interest—which are around $n_1 = n_2$ —there is a clear checkerboard pattern where the probability of $(n_1, n_2) = (n, n)$ is much higher than the probability of $(n + 1, n)$ or $(n, n + 1)$. Therefore, if we want to maximize tolerance to loss in the photon detection path, more modes in the GBS device is generally better; however, this does not account for loss in the output mode of the GBS device.

It is very important to take into account the probability of producing a useful state, as the refinery that turns the inputs into a fault-tolerant state needs to have a practical circuit depth. Here, a useful state is considered to be one that can be bred (in the refinery) into a state with high enough quality for fault tolerance. To determine whether a state is useful, we consider the rule-of-thumb that breeding transfers ~ 3 dB of effective squeezing from the q quadrature to the p quadrature, and therefore conserves the sum of the effective squeezings, along with the fact that most multi-peaked states generated by the staircase family of circuits have a higher effective squeezing in q than p . Consider the sum of the effective squeezing in q and p in dB, $\sigma_p^2[\text{dB}] + \sigma_q^2[\text{dB}]$. Given that the fault-tolerant threshold of a symmetric state is ~ 9.75 dB of effective squeezing in each quadrature, useful states are those such that $\sigma_p^2[\text{dB}] + \sigma_q^2[\text{dB}] \geq 19.5$. Fig. S12b shows the distribution of states from 2- to 4-mode staircase GBS devices, where the probability of useful states drops by approximately an order of magnitude for an additional mode. Therefore, the MUX requirements significantly increase for higher numbers of modes. Of course, to get the same quality of state, we should take into consideration the probability to get two copies of the state and successfully breed them. However, this has a much higher probability than the equivalent GBS device for two reasons. Firstly, we can put MUX after the GBS and before the breeding. Because the success probabilities of the smaller GBS circuits are much higher, the MUX requirements to boost them to near-deterministic are much smaller. Secondly, the GBS parameters are fixed to target equal numbers of photons in each photon number detector, which means outcomes far from this do not result in useful states. However, due to the adaptive beamsplitters in adaptive breeding, there are many more useful combinations of photon number outcomes, improving the success probability. Although breeding is a probabilistic process, the decrease in success probability is negligible in comparison.

To summarize, there is a trade-off between adding more modes to the GBS devices to increase the tolerance to loss before the photon number measurements and removing modes to boost the probabilities of useful outputs. It is not obvious where the optimum balance is without running full simulations including the refinery and QPU. Therefore, we considered both 2- and 3-mode staircase GBS in the simulations described in the next section, performing numerical sweeps. As the probabilities of producing useful states in the 4-mode GBS are so low, we do not consider such circuits here. The best option that we found from these simulations depends on the relative losses in each path of the hardware. If there is very little loss in path P_1 , the two-mode circuit is preferable as it can withstand higher losses in the other paths, as well as having lower depth refineries. However, the three-mode device allows for a higher loss tolerance in path P_1 whilst simultaneously having similar losses in the other paths. There is also potential for further improvement by investigating the performance of 4-mode GBS circuits, despite their low probabilities. We leave this as a possible candidate to consider in the future.

V. THEORY: STATE FACTORY SIMULATION TECHNIQUES

A. Loss tolerance simulations

We fix the structure of the GKP Bell pair state factory as GBS devices, followed by hybrid MUX-breeding, then MB squeezing, and finally MUX in the QPU (see Fig. 1 in the main text). Nevertheless, there are many parameters that can be optimized to improve the output state quality. They are the:

- Input squeezing levels in GBS
- GBS interferometer
- Number of modes in GBS
- Depth of N -to- M MUX in the refinery (i.e. N)
- Number of states to breed (i.e. M)
- Depth of MUX in the QPU
- Ranking functions of the multiplexers.

The aim is to find a combination of these parameters that results in an average output state that has symmetric effective squeezing above 9.75 dB (the level of squeezing required by the Gaussian random noise model to reach the fault-tolerant threshold; see Secs. II A and VI), with hardware demands that are as realistic as possible. In particular, we want to have the highest possible tolerance to loss while restricting the required squeezing, the depth of MUX and breeding, and the number of photons needed to be detected in a single detector. The GBS interferometer is chosen to be the staircase derived in Sec. IV B as this was the best option found from small scale optimizations on only the GBS. We fixed $r_2 = -1.727$ (i.e. 15 dB, the maximum allowed squeezing), and for the other parameters, r_1 , r_3 and θ , we determined good values from simplified simulations and reasoning, and swept over each parameter for values in this region of interest in simulations of the full state factory. For each value of loss considered in all paths, we selected the combination of these parameters that outputted an average state with the highest symmetric effective squeezing. We also performed a sweep over the depth of the multiplexers, as increasing this with fixed total

loss cannot decrease the quality of the state, but we wish to minimize the depth as much as possible. In our simulations, we considered squeezing levels in the GBS devices up to the free-space record of 15 dB [30], detecting up to 40 (24) photons per mode for two-mode (three-mode) GBS, which is close to the record of 37 detected by a single transition edge sensor (TES) detector [31], up to a 1024-to-1 binary tree and 32-to-1 MUX in the QPU, and 2 or 3 modes in the GBS.

We consider four loss paths as depicted in Fig. 1 in the main text: P_1 from squeezer to photon number measurement, P_2 from squeezer to homodyne in the refinery (everything before the MB squeezing), P_3 from the MB squeezing to the QPU (except the homodyne detector in the QPU which is assumed to have the same efficiency as those in the refinery and included in P_2), and finally P_4 which is from the ancillary squeezer to the homodyne measurement in the MB squeezing. We want to find a state factory that produces pairs with quality better or equal to what is needed for fault tolerance, with as much loss as possible in all of these paths. There is a trade-off where more loss can be tolerated in one path if we remove some loss in another path and the share of the losses across the paths may change as hardware capabilities improve. As such, there is a four-dimensional surface that describes the loss threshold for fault-tolerance as a function of the losses in P_1 , P_2 , P_3 and P_4 and hardware performance targets can be set within the volume bounded by that surface. As described in the main text, we leave P_1 , P_2 , and P_3 as free parameters but fix the P_4 path by the following heuristic, which attempts to model the relative difficulty of the components required in the measurement-based squeezing operation relative to the rest of the refinery: refinery: $P_4 = (0.21P_2)/(0.3 + 0.02d_r)$, where d_r is the depth of the refinery. Since d_r runs from 6 to 10, the factor multiplying P_2 goes from 0.5 to 0.42. We ran large-scale simulations sweeping the two loss parameters in paths P_1 and P_2 as well as the GBS squeezing and beamsplitter angles and MUX depths on AWS EC2 C7a instances. Then we also swept the loss path P_3 in post-processing (see Sec. V B 4).

In Fig. S13, we provide a 3D plot showing for which values of these losses there exist factories that are fault-tolerant. In Fig. S13(a) we show the results for factories based on 3-mode GBS devices, with factory depths (the binary tree depth in the refinery plus the binary tree depth in the QPU) ranging from 9 to 15 (this is the same as in Fig. 4 in the main text). In Fig. S13(b) we show the same, but for factories based on 2-mode GBS devices, with factory depths ranging from 6 to 15. For the factories based on three-mode (two-mode) GBS devices, the refinery selects four (eight) GBS output states and breeds them together to make a GKP state. In all cases, the surfaces are approximately planar, roughly forming triangles between the three loss axes (P_1 , P_2 , P_3). Looking at the corners of the triangles, the two-mode (three-mode) GBS based factories can tolerate up to $\sim 1.1\%$ ($\sim 1.75\%$) loss in P_1 , up to $\sim 3.5\%$ ($\sim 4\%$) loss in P_2 , and up to $\sim 6\%$ ($\sim 4.5\%$) loss in P_3 . We see that in most cases, increasing the depth can make the tolerance to loss better. This makes sense since for equal levels of end-to-end loss, more depth means a greater possibility to multiplex and hence select the very best states across the distribution of photon number and homodyne outcomes. In practice, however, increasing the depth will incur more loss, so this just means that there will be a tradeoff point where the boost in performance gained by using deeper factories is not enough to overcome the extra loss that has been added.

To understand how loss tolerance scales with depth, for each subfigure Fig. S13(a) and (b), we approximate each surface as a plane and find its normal vector, then use those to compute a vector that is approximately normal to all the surfaces in that subplot. In the three-mode case, this vector is $(P_1, P_2, P_3) = (0.85, 0.41, 0.34)$, and in the two-mode case the approximate normal vector is $(P_1, P_2, P_3) = (0.95, 0.28, 0.16)$. We can understand these approximate normal vector as follows: were the surfaces exactly planar and parallel to each other, then as we move along the line defined by that vector, the projection of each factory

on the plane of factories onto that line will be the same. Moving away from the origin in the direction of the vector increases the depth, but also the loss tolerance while preserving the relative ratios of the tolerable losses of each path. In reality, the surfaces are not exactly planes and are not exactly parallel, but projecting the surface of factories onto this vector can tell us approximately how the factories scale with depth. In Fig. S13(c), we plot how the projections of the factory surfaces onto these two approximate normal vectors scale with depth. We see that for factory depths greater than 10, factories based on three-mode GBS devices are generally the better option, whereas for shallower factory depths, factories based on two-mode GBS devices are the better option.

Ultimately, it is sufficient to realize one state factory within the volume created by these surfaces. An example of a fault-tolerant state factory with a reasonable loss tolerance in each path employed three-mode staircase GBS devices with input squeezing of 13.8 dB, -15 dB and -13.2 dB in each mode and beamsplitter angle 0.5553, and photon number measurement outcomes of up to 24 photons in each mode. The refinery chip employed a binary tree of adaptive beamsplitters and homodyne detectors of depth 10, and an ancilla-assisted squeezing step employing a 15 dB squeezed state. The QPU chip employed a MUX with a depth of 5. The following losses were simultaneously tolerable in each of the optical paths while still producing an average state with more than 9.75 dB of symmetric effective squeezing: 1% for the P_1 path, 1.25% for the P_2 path, 1% for P_3 and 0.525% for P_4 .

B. Simulation details for one state factory

The process employed here to make a GKP state is non-deterministic with many probabilistic measurements. It is too resource intensive to compute the full distribution of output states, so instead of finding the probabilities and states of all possible outcomes, we use Monte Carlo sampling with a sample size of 1000 to estimate the average state output by the factory. Even though loss is accumulated throughout the circuit, since we assume uniform loss across each mode in the same path and each path only includes linear optics, the uniform loss can be commuted to specific points in the circuit where it is most convenient to simulate. Therefore we commute all loss in path P_1 to be directly before the photon number measurements, in P_2 to be directly before the homodyne detectors in the refinery, in P_4 to be directly after the ancillary squeezer in the MB squeezing, and in P_3 to be on the GKP state after breeding but before the Bell pair generation. To run end-to-end simulations of the state factory, we employed multiple representations of states as the most convenient representation is different for each stage of the state factory. Motivated by the simplicity of analyzing breeding and squeezing in the p -quadrature basis, we simulated everything after the output of the GBS devices in the p -quadrature representation. However, for the multimode Gaussian states before the photon number measurements in the GBS, we use the Bargmann representation even though the measurements are in the Fock basis. We avoid directly using the Fock basis because we would have to introduce a finite cutoff, which introduces an error depending on how squeezed or displaced the states are as they evolve through the circuit, despite such transformations being Gaussian and not affecting the quality of the state. We choose the Bargmann basis as this is a compact representation of Gaussian states that can be readily converted to the p -quadrature basis in the output mode and projected onto the Fock basis in the modes that are measured with photon number detection. Here we describe the process of how to simulate each element in turn: the lossy GBS devices, the hybrid MUX-breeder, the MB squeezer, loss path P_3 and MUX in the QPU.

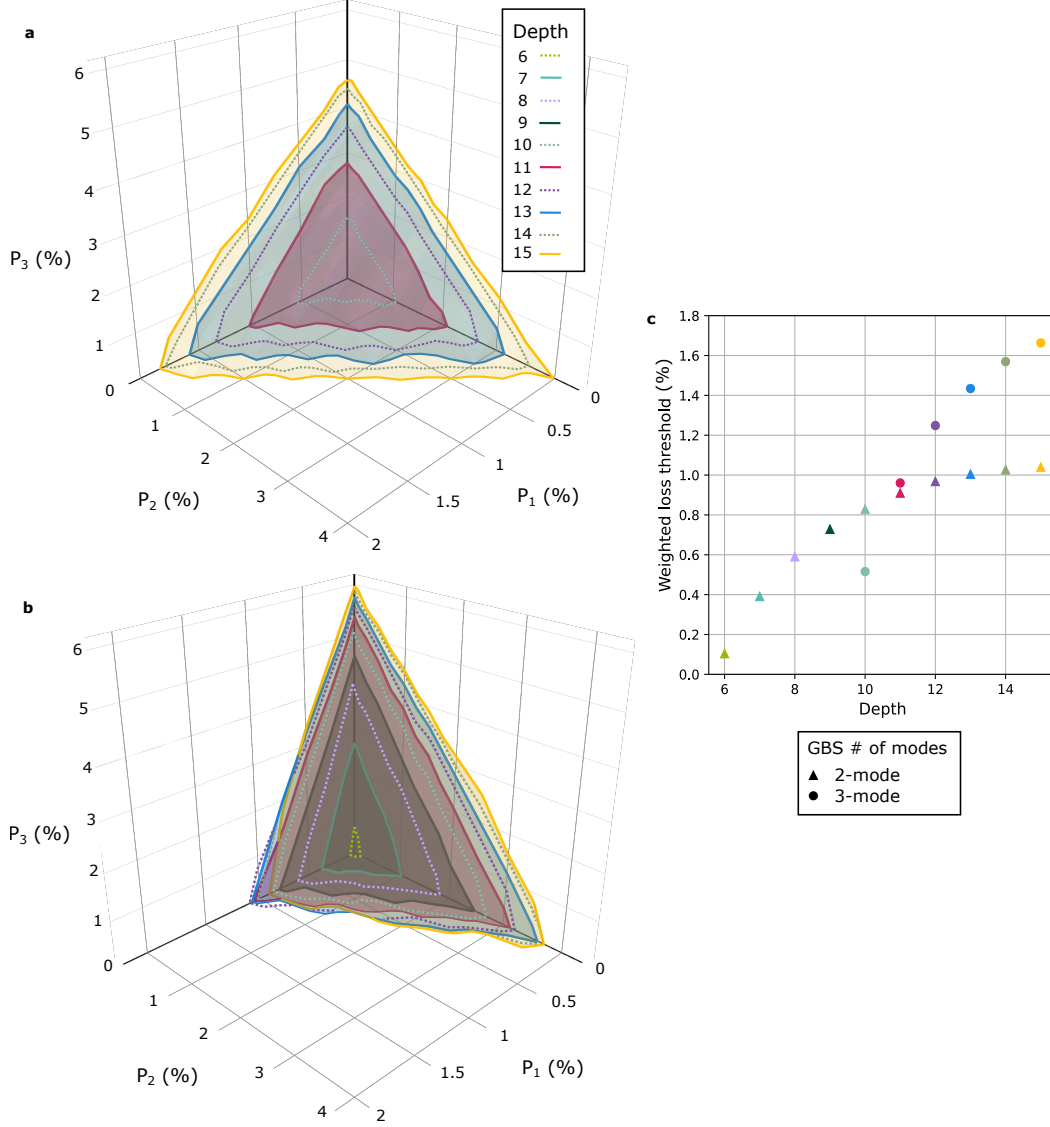


Figure S13. **a. Fault-tolerance surfaces vs. loss (three-mode GBS).** For different factory depths, we plot the surfaces that determine whether factories of that depth lead to GKP states with high enough quality for fault-tolerance, as a function of loss in the three major optical paths of the architecture. Along each surface, factory parameters (e.g. GBS parameters) are being varied, with the most loss-tolerant factories being plotted. **b. Fault-tolerance surfaces vs. loss (two-mode GBS).** Same style of plot as **a.** for factories based on two-mode GBS devices. In this case, we see greater loss tolerance in P_3 , but less in P_1 , and we also see the possibility of attaining shallower overall depths when the loss is low. **c. Loss tolerance vs. depth.** For the plots in **a.** and **b.**, we find the vectors that in each case are approximately normal to all the surfaces, then see how the projection of the surfaces onto those vectors scales with depth. From this, we see that factories based on two-mode GBS devices can generally tolerate more loss for depths 10 or smaller, while factories based on three-mode GBS devices are better for depths 11 or greater.

1. Simulating GBS devices

We want to simulate the output state of the GBS in the p -quadrature basis. Prior to the photon number measurements we have a multimode Gaussian state given by the squeezing, beamsplitters and loss. An m -mode Gaussian state can be uniquely described by the Bargmann triple $(A, \mathbf{b}, c)$ such that the state in the Bargmann basis is

$$\langle \alpha || \rho || \beta \rangle = c \exp \left(\frac{1}{2} \gamma^T A \gamma + \mathbf{b}^T \gamma \right), \quad (25)$$

where $\gamma^T = (\beta_1, \alpha_1^*, \dots, \beta_m, \alpha_m^*)$ and $||\alpha\rangle = e^{|\alpha|^2/2} |\alpha\rangle$ is the rescaled coherent basis (see [32] where the Bargmann basis is applied to Gaussian unitaries, and [33] where it is applied to any Gaussian object including density matrices and channels). All matrices and vectors below are ordered in the same way as γ , with the two variables for each mode grouped together. We will measure modes 2 to m in the photon number basis and output the resulting state in mode 1. Therefore the matrix A and vector $\mathbf{b}$ can be divided into blocks describing the output mode and measured modes as

$$A = \left(\begin{array}{c|c} A_{\text{out}} & A_{\text{corr}} \\ \hline A_{\text{corr}}^T & A_{\text{meas}} \end{array} \right) \quad \mathbf{b} = \left(\begin{array}{c} \mathbf{b}_{\text{out}} \\ \mathbf{b}_{\text{meas}} \end{array} \right) \quad (26)$$

where A_{out} and $\mathbf{b}_{\text{out}}$ are the 2×2 matrix and length 2 vector corresponding to the unmeasured output mode, A_{meas} and $\mathbf{b}_{\text{meas}}$ are the $2(m-1) \times 2(m-1)$ matrix and length $2(m-1)$ vector corresponding to the $m-1$ measured modes and A_{corr} is the $2 \times 2(m-1)$ matrix describing the correlations between the output mode and the measured modes. Since the Bargmann basis is convenient for simulating photon number measurement of a Gaussian state and the p -quadrature basis is convenient for simulating breeding, we change only the output mode into the p -quadrature basis and leave the rest in the Bargmann basis for now. Denoting $\langle \alpha, \alpha_2, \dots, \alpha_m || = \langle \alpha, \boldsymbol{\alpha}_{\text{meas}} ||$ and $|| \beta, \alpha_2, \dots, \alpha_m \rangle = || \beta, \boldsymbol{\alpha}_{\text{meas}} \rangle$ (dropping the subscript in mode 1) and $\gamma^T = (\beta, \alpha^*, \alpha_2, \alpha_2^*, \dots, \alpha_m, \alpha_m^*) = (\beta, \alpha^*, \gamma_{\text{meas}}^T)$ (i.e. $\alpha_i = \beta_i$ for $i > 1$), we have:

$$\begin{aligned} & (\langle p | \langle \boldsymbol{\alpha}_{\text{meas}} ||) \rho(|p'\rangle || \boldsymbol{\alpha}_{\text{meas}} \rangle) \\ &= \frac{1}{\pi^2} \iint d^2 \alpha d^2 \beta \langle p | \alpha \rangle \langle \beta | p' \rangle (\langle \alpha | \langle \boldsymbol{\alpha}_{\text{meas}} ||) \rho(|\beta\rangle || \boldsymbol{\alpha}_{\text{meas}} \rangle) \\ &= \frac{1}{\pi^2} \iint d^2 \alpha d^2 \beta e^{-(|\alpha|^2 + |\beta|^2)/2} \langle p | \alpha \rangle \langle \beta | p' \rangle \langle \alpha, \boldsymbol{\alpha}_{\text{meas}} || \rho || \beta, \boldsymbol{\alpha}_{\text{meas}} \rangle \end{aligned} \quad (27)$$

where $|p\rangle$ is an eigenstate of the momentum basis. By using the following formula for the overlap of a coherent state and p eigenstate [34]

$$\langle p | \alpha \rangle = \frac{e^{-|\alpha|^2/2}}{(\pi)^{1/4}} \exp \left(-\frac{1}{2} p^2 - \sqrt{2} i \alpha p + \frac{1}{2} \alpha^2 \right), \quad (28)$$

and gathering the terms with the same powers of α and β we arrive at

$$\begin{aligned} (\langle p | \langle \boldsymbol{\alpha}_{\text{meas}} ||) \rho(|p'\rangle || \boldsymbol{\alpha}_{\text{meas}} \rangle) &= \frac{c e^{-\frac{1}{2}(p_1^2 + p_2^2)}}{\pi^2 \sqrt{\pi}} \exp \left[\left(\frac{1}{2} \gamma_{\text{meas}}^T A_{\text{meas}} + \mathbf{b}_{\text{meas}}^T \right) \gamma_{\text{meas}} \right] \times \\ &\quad \iint d^2 \alpha d^2 \beta \exp \left[\frac{1}{2} \tilde{\gamma}_{\text{out}}^T \tilde{A}_{\text{out}} \tilde{\gamma}_{\text{out}} + (\gamma_{\text{meas}}^T \tilde{A}_{\text{corr}} + \tilde{\mathbf{b}}_{\text{out}}^T) \tilde{\gamma}_{\text{out}} \right] \end{aligned} \quad (29)$$

where

$$\tilde{A}_{\text{out}} = \begin{pmatrix} I & -X \\ -X & A_{\text{out}} \end{pmatrix} \quad \tilde{A}_{\text{corr}} = \begin{pmatrix} \mathbf{0} \\ A_{\text{corr}} \end{pmatrix} \quad \tilde{\gamma}_{\text{out}} = \begin{pmatrix} \alpha \\ \beta^* \\ \beta \\ \alpha^* \end{pmatrix} \quad \tilde{\mathbf{b}}_{\text{out}} = \begin{pmatrix} \sqrt{2}ip' \\ -\sqrt{2}ip \\ \mathbf{b}_{\text{out}} \end{pmatrix}$$

and

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

This is a Gaussian integral and evaluates to

$$(\langle p | \langle \boldsymbol{\alpha}_{\text{meas}} || \rangle \rho(|p'\rangle || \boldsymbol{\alpha}_{\text{meas}} \rangle) = \tilde{c}(p, p') \exp(\frac{1}{2} \gamma_{\text{meas}}^T \tilde{A} \gamma_{\text{meas}} + \tilde{\mathbf{b}}^T \gamma_{\text{meas}}) \quad (30)$$

with

$$\tilde{A} = A_{\text{meas}} - \tilde{A}_{\text{corr}}^T \tilde{A}_{\text{out}}^{-1} \tilde{A}_{\text{corr}} \quad \text{and} \quad \tilde{\mathbf{b}} = \mathbf{b}_{\text{meas}} - \tilde{A}_{\text{corr}}^T \tilde{A}_{\text{out}}^{-1} \tilde{\mathbf{b}}_{\text{out}}$$

and

$$\tilde{c}(p, p') = \frac{c \exp(-\frac{1}{2}(p^2 + p'^2 + \tilde{\mathbf{b}}_{\text{out}}^T \tilde{A}_{\text{out}}^{-1} \tilde{\mathbf{b}}_{\text{out}}))}{\sqrt{\det(\tilde{A}_{\text{out}}) \pi}}.$$

Now that we have the first mode in the p -quadrature basis, we can simulate the photon number measurements on modes 2 to m . This corresponds to taking derivatives of the Bargmann function, $F_{|\psi\rangle}(\alpha) = \langle \alpha^* || \psi \rangle$ for states described by kets or $F_{\rho}(\alpha) = \langle \alpha^* || \hat{\rho} || \alpha \rangle$ for states described by density matrices, with respect to the coherent basis parameters, evaluated at zero [33, 35]. This correspondence arises because the Bargmann function of the photon number state $|n\rangle$ is $F_{|n\rangle}(\alpha) = \langle \alpha^* || n \rangle = \frac{\alpha^n}{\sqrt{n!}}$, and the complex integral (inner product) against such a monomial picks out the n -th Taylor coefficient of F . In other words, the Taylor coefficients of the Bargmann function are the Fock coefficients of the object represented by the Bargmann function:

$$\begin{aligned} \langle p, \mathbf{n} | \rho | p', \mathbf{n} \rangle &= \left[\prod_{k=2}^m \frac{1}{n_k!} \left(\frac{\partial^2}{\partial \alpha_k \partial \alpha_k^*} \right)^{n_k} \right] (\langle p | \langle \boldsymbol{\alpha}_{\text{meas}} || \rangle \rho(|p'\rangle || \boldsymbol{\alpha}_{\text{meas}} \rangle) \Big|_{\boldsymbol{\alpha}=\mathbf{0}} \\ &= \tilde{c}(p, p') \left[\prod_{k=2}^m \frac{1}{n_k!} \left(\frac{\partial^2}{\partial \alpha_k \partial \alpha_k^*} \right)^{n_k} \right] \exp(\frac{1}{2} \gamma_{\text{meas}}^T \tilde{A} \gamma_{\text{meas}} + \tilde{\mathbf{b}}^T \gamma_{\text{meas}}) \Big|_{\boldsymbol{\alpha}=\mathbf{0}} \end{aligned} \quad (31)$$

This is mathematically exactly the same as measuring the photon numbers of modes 2 to m on a state described by Bargmann triple $\tilde{A}$, $\tilde{\mathbf{b}}$ and $\tilde{c}(p, p')$. Therefore we can follow the same process as we would for finding the state after photon number measurements. There are many ways to evaluate the derivatives, but in our case we use the recursion relations of hermite polynomials to calculate the output states for all photon number outcome patterns up to the chosen cutoff number of photons detected in each mode [32, 33, 36]. This is numerically implemented in MrMustard [37] in the function `hermite_renormalized`. This gives the unnormalized state after measuring some photon number outcome pattern as a function of p and p' . To find the probability, we build the diagonal of the p -quadrature density matrix where $p = p'$ with discretized p for each photon number pattern and perform a numerical integral over p , which is equivalent to computing the norm of the state.

As the multiplexer after the GBS will significantly reduce the probability of photon number outcomes that do not yield useful states, it would be wasteful to calculate every single state. Instead, we select outcomes that we know produce good states and calculate their probabilities and the corresponding output state. All other outcomes are grouped together into a so-called “failure state”, which occurs with the remaining probability and has a mixed output state given by the thermal state from tracing out the measured modes and subtracting the successful states’ density matrices. In practice, the failure state never needed to be used in the cluster state and was filtered out by the MUX.

2. Simulating the MUX-breeder

The sampled GBS outputs will be input into an adaptive breeder where the choice for the angle of the beamsplitter is determined by the ratio of the lattice q -spacing of the input states. Recall from Sec. II A that a displacement $\hat{D}(\alpha)$, $\alpha \in \mathbb{C}$, stabilizes a lattice of periodicity $\frac{\sqrt{2}\pi}{\alpha}$ in the quadrature orthogonal to that which diagonalizes $\hat{D}(\alpha)$. Therefore, the ratio of the lattice spacing of two states is given by the inverse ratio of their corresponding stabilizer parameters. Hence, it is necessary to find the stabilizers of the states output by the GBS, prior to simulating the MUX-breeder. Although these states have a peaked structure, they are not perfect lattices (aperiodic, finite width, different heights) and so we perform an optimization to find the two single-mode displacements that maximize the product of the stabilizer expectation values, and equivalently the symmetric effective squeezing. As explained in Sec. II B, the area of the stabilizers of a state reduces by a factor of $1/\eta$ when a loss channel of transmissivity η is applied to a state. Therefore we restrict the displacements such that their area in phase space is $\frac{2\pi}{\eta}$ where η is the transmissivity of the loss on the output mode P_2 . From observation, the states output from GBS devices are rectangular and so we also restrict to real-valued and imaginary-valued displacements, which simplifies the optimization of the stabilizers. As we know the states in the p -quadrature basis, we can use Eq. (6) with $\phi = \frac{\pi}{2}$ such that $\hat{q}_\phi = \hat{p}$ to find the expectation values $\langle \hat{D}(\alpha) \rangle$ for real α . To find the expectation values $\langle \hat{D}(i\beta) \rangle$ for imaginary $i\beta$, we employ the q -quadrature basis representation of the state. This is equivalent to the p -quadrature representation of the GBS output state rotated by $\frac{\pi}{2}$. Therefore we recompute the output states for the GBS using the process described above but with an updated Bargmann triple $(A, \mathbf{b}, c)$ to include a rotation on the output mode. This is more computationally efficient than finding the Fourier transform of the p -quadrature distributions directly.

As shown in Sec. III B, the hybrid MUX-breeder can be decomposed into an N -to- M MUX followed by a linear cascade breeder of M states, where the order of the states to breed does not matter (see Sec. III B). So to simulate sampling a state from each of N identical GBS devices, we use the `multinomial` function from NumPy [38] to sample from the probability distribution of the acceptable photon number outcomes or the “failure” state. This gives us the frequency of each photon number outcome. Then to simulate the MUX, we simply discard all but the best M states according to the ranking function (see Sec. III B). The remaining states will be carried forward into the breeder.

To simulate the breeder, we must sample from the homodyne measurements and then find the state conditional on those outcomes. As the beamsplitter network can be understood as a linear transformation on the quadratures that does not mix $\hat{q}$ operators with $\hat{p}$ operators, we can find the homodyne measurement probability density of the state after the beamsplitters as a function of the input states. For some real valued unitary $\hat{U}$ described by the linear transformation T such that $\hat{U}^\dagger \hat{p}_i \hat{U} = (T\mathbf{p})_i$, the p -quadrature density matrix elements of a

multimode state $\hat{\rho}$ transform as

$$\langle \mathbf{p} | \hat{U} \hat{\rho} \hat{U}^\dagger | \mathbf{p} \rangle = \langle T^{-1} \mathbf{p} | \hat{\rho} | T^{-1} \mathbf{p} \rangle. \quad (32)$$

The probability of measuring $\bar{\mathbf{p}}$ from the input state is the same as measuring $T\bar{\mathbf{p}}$ from the transformed state. Therefore to sample from the output state, $\hat{U} \hat{\rho} \hat{U}^\dagger$, we can sample $\bar{\mathbf{p}}$ from the input state, $\hat{\rho}$, and then transform that sample as $T\bar{\mathbf{p}}$. As the input state to the breeder is separable and already in the p -quadrature basis, the probability distributions of the homodyne outcomes before the unitary is applied are given by the diagonals of the input states' density matrices. After accounting for the unitary in the circuit, we are able to infer homodyne data on all modes, including the output mode (which we do not want to measure). However, simply discarding the outcome on the output mode is the same as not drawing a sample from that mode. This method is more efficient than finding the multimode state after the beamsplitters and then calculating the joint probability distribution.

Now that we have the sampled homodyne measurement outcomes, we need to find the output state, which is convenient to compute in the p -quadrature basis. We consider the unnormalized density matrix element conditional on measuring $\bar{\mathbf{p}}$ in modes 2 to m :

$$\begin{aligned} \langle p_1, \bar{\mathbf{p}} | \hat{\rho}_{out} | p'_1, \bar{\mathbf{p}} \rangle &= \langle p_1, \bar{\mathbf{p}} | \hat{U} \bigotimes_{i=1}^m \hat{\rho}_i \hat{U}^\dagger | p'_1, \bar{\mathbf{p}} \rangle \\ &= \langle T^{-1}(p_1, \bar{\mathbf{p}}) | \bigotimes_{i=1}^m \hat{\rho}_i | T^{-1}(p'_1, \bar{\mathbf{p}}) \rangle \\ &= \prod_{i=1}^m \langle [T^{-1}(p_1, \bar{\mathbf{p}})]_i | \hat{\rho}_i | [T^{-1}(p'_1, \bar{\mathbf{p}})]_i \rangle \end{aligned} \quad (33)$$

From this, we see that the output state is just a product of the density matrix elements of the input states but with the function arguments (p, p') all rescaled and displaced by T^{-1} and $\bar{\mathbf{p}}$. As we have the p -quadrature functions of the states output from the GBS, we can find the p -quadrature functions of the state output by breeding using Eq. 33.

3. Calculating the stabilizer expectation values

The rest of the state factory (MB squeezing, loss and MUX) can be simulated by updating the stabilizer expectation values and changing the statistics of the sampled states. Therefore we calculate the stabilizer expectation values at this point and can discard the full state. As shown in Sec. III A, the lattice of the state output from a breeder is a known function of the lattices of the input states. Given the previously calculated stabilizers of the GBS output states, and using the rules for how the lattice changes under breeding (see Fig. S2), we can determine the displacements that stabilize the states after breeding, which are used to calculate the expectation values. Like before, for expectation values of real-valued displacements, we can use Eq. (6) as we know the p -quadrature basis of the output states. However, for expectation values of imaginary displacements we cannot use this equation as we do not know what the q -quadrature basis is and cannot simply apply a rotation unlike in the GBS. Rather than computing the Fourier transform of the p -quadrature distribution—which would be much more computationally expensive—we use another formula for the

expectation value that is in terms of the p -quadrature basis [39]:

$$\begin{aligned}
\text{Tr}(\hat{\rho}\hat{D}(\alpha)) &= \iint dpdp' \langle p|\hat{\rho}|p'\rangle \langle p'|\hat{D}(\alpha)|p\rangle \\
&= e^{-i\text{Re}(\alpha)\text{Im}(\alpha)} \iint dpdp' \langle p|\hat{\rho}|p'\rangle \langle p'|e^{i\sqrt{2}\text{Im}(\alpha)\hat{q}}e^{-i\sqrt{2}\text{Re}(\alpha)\hat{p}}|p\rangle \\
&= e^{-i\text{Re}(\alpha)\text{Im}(\alpha)} \iint dpdp' e^{-i\sqrt{2}\text{Re}(\alpha)p} \langle p|\hat{\rho}|p'\rangle \langle p' - \sqrt{2}\text{Im}(\alpha)|p\rangle \\
&= e^{-i\text{Re}(\alpha)\text{Im}(\alpha)} \int dp e^{-i\sqrt{2}\text{Re}(\alpha)p} \langle p|\hat{\rho}|p + \sqrt{2}\text{Im}(\alpha)\rangle.
\end{aligned} \tag{34}$$

As we are interested in the absolute value of the expectation value, we can ignore the phase before the integral. For real α this reduces to Eq. (6), but for imaginary α the exponential term inside the integral evaluates to 1 and we need to integrate over an off-diagonal of the p -quadrature density matrix.

4. Simulating measurement-based squeezing, and loss and MUX in the QPU

As the final step of the refinery, the MB squeezer aligns the states to the square-grid qunaught state lattice so that the cluster state can be stitched together using beamsplitters; this is followed by loss that is commuted from everywhere in the P_3 path accumulated after the MB squeezing, and MUX in the QPU. These effects can all be applied in post-processing to the stabilizer expectation values rather than needing to keep the full state. As explained in Sec. IIIC, we average over the homodyne outcomes in the MB squeezer and so this and the loss are both deterministic Gaussian channels. As such we can use the degradation of stabilizer expectation values under noisy Gaussian channels described in Sec. IIB to find the stabilizer expectation values after applying the channels. The X and Y matrices for MB squeezing are given in Eq. (21) for squeezing in the q direction and Eq. (22) for squeezing in p . The variance of the squeezed state is $V_{\text{sqz}} = \frac{1}{2}(\eta e^{-2|r_{\text{anc}}|} + 1 - \eta)$ where r_{anc} is the squeezing parameter of the ancillary squeezed state and η is the loss it undergoes before the beamsplitter. We fix the ancillary squeezing to be the maximum of the squeezing used in the GBS as the hardware limitations are the same for squeezing in both places. For uniform loss, the channel is $X = \sqrt{\eta}\mathbb{I}$ and $Y = \frac{1}{2}(1 - \eta)\mathbb{I}$. Putting these matrices into Eq. (8) gives the stabilizer expectation values of the final state prior to the MUX in the QPU. Since squeezing the state more with MB squeezing introduces more noise, less noise is introduced if the GBS devices are configured such that the states are approximately already close to the square-grid qunaught lattice by the time they reach the MB squeezing. However, in order to target this spacing, we find that, while bounding the squeezing, the corresponding GBS has a lower probability of high photon number counts than a GBS with both $r_1 = -r_2$ fixed as 15 dB, and so outputs lower quality states on average before the MB squeezing. Therefore, when setting the GBS parameters, there is a trade-off between reducing the noise from the MB squeezing and improving the quality of states that reach the MB squeezer.

The final improvement in quality comes from the MUX in the QPU. After the MB squeezing and final loss, two GKP states from the same chip are paired together to create a Bell pair. If we have R layers of MUX in the QPU, 2^R of these Bell pairs are input into the QPU and only the best survives. Therefore to update the statistics of the GKP states to take this MUX into account, we sample 2^{R+1} from the 1000 samples and randomly pair them up. Then according to a ranking function, we keep only the best pair and repeat this 5000 times to get 10000 states that are now sampled according to the probability of being included in

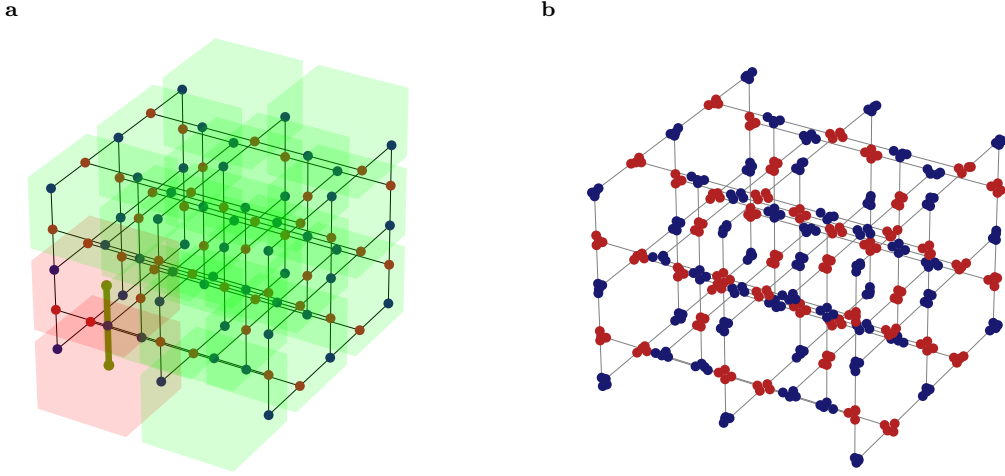


Figure S14. (a) A distance-3 foliated surface code (i.e. RHG lattice), with time going into the page. In bulk, the stabilizers are six-body X operators associated with the displayed cubes (boundary stabilizers are incomplete). Red (blue) modes correspond to the primal (dual) error complex. The task for a logical memory is to correct Pauli Z errors on these qubits. An example syndrome is displayed: green cubes are associated with even-parity stabilizers, whereas red cubes indicate some physical qubit errors have occurred that have flipped the parity of the associated stabilizers. Each stabilizer parity is computed by combining the single-qubit measurement outcomes of the qubits in its support. A minimum-weight matching algorithm is able to predict the most likely chain of physical errors that has led to this observed syndrome, shown as a line between the red cubes. (b) An implementation of (a) with static linear optics as in [1, 13]. Bell pairs and macronodes (multiple modes meeting at a lattice site) are shown. Modes in each macronode are stitched with an additional beamsplitter network (not shown).

the cluster state. We tested two ranking functions, selecting the Bell pair that comprised of the two states with the highest mean symmetric effective squeezing, and selecting the Bell pair comprising of the two states with the highest minimum symmetric effective squeezing. The former resulted in the best symmetric effective squeezing of the output average state. Given the final set of states sampled, we find their average state and compute its symmetric effective squeezing to get a single value as explained in Sec. II. Finally we compare this value to the threshold to determine whether the state factory in question is fault-tolerant, with anything greater than 9.75 dB considered fault-tolerant.

VI. THEORY: SQUEEZING THRESHOLD FOR FAULT TOLERANCE

Here we describe the process for obtaining a fault-tolerance threshold for the architecture we present in this paper: the point in physical noise parameter space beyond which logical errors can be suppressed arbitrarily. As described in Sec. IID, we judge whether a given architecture configuration is fault tolerant by relating the effective squeezing of the GKP Bell pairs produced in the presence of loss to the required squeezing level for fault tolerance.

We begin by selecting a qubit-level error-correcting code and decoder. Our choice is the surface code [40–43] and a minimum-weight perfect matching (MWPM) decoder [44], since

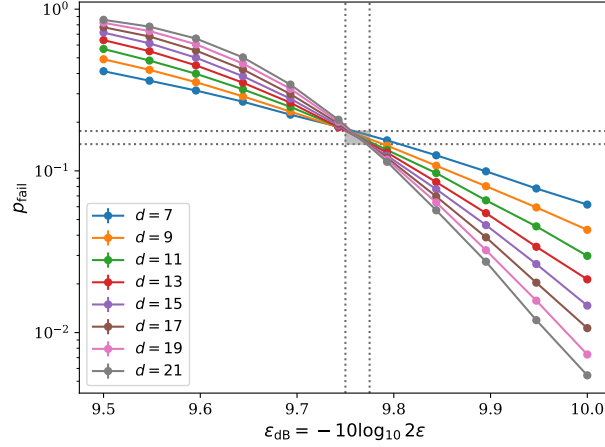


Figure S15. Logical error rate over a physical error parameter quantifying the quality of GKP states in the presented architecture. The simulated experiment is a fault-tolerant memory implemented with the foliated surface code with periodic boundaries (i.e. the toric code). Each data point comes from 50,000 trials of a Monte Carlo simulation with a fixed code distance and the parameter ϵ , the variance associated to the Gaussian random noise channel modelling the non-ideal GKP states. The simulation is repeated over various odd code distances. Improvement to the 10.1 dB threshold value in Ref. [1] is obtained through an updated correlation-aware inner decoder; a minimum-weight perfect matching outer decoder is used, weighted through soft information from simulated GKP circuits. Threshold behaviour is observed; there is a small drift in the threshold due to finite size effects, but the expected region is highlighted. We take a value of 9.75 dB for state preparation targets in this paper. See Sec. VI and VIA for more details on methods.

these give some of the best thresholds for a host of noise models, including those arising from GKP-based architectures [1, 3, 44–53]. In practice, a different QEC code may be chosen for the architecture, such as a finite-rate LDPC code [54–64], which may result in better overheads. For a measurement-based implementation, one foliates [65, 66] the surface code to obtain a cluster or graph state [67, 68]—a multipartite entangled state—also known as the RHG lattice [69]. The RHG lattice is a simple cubic lattice; in bulk, each node has a valence of four (the number of incident edges or neighbours), and combinations of six X operators centered at the faces of cubic unit cells comprise the foliated stabilizers of the code. We interpret two dimensions of this graph spatially and one temporally; see Fig. S14a.

Starting with GKP Bell pairs, whose production is detailed in the previous sections, one can implement the RHG lattice using a static linear-optical network [1]. Each edge in the lattice is associated to a Bell pair. For Bell pairs that form the time-like connections, half of the modes are sent through delay lines. All the Bell pairs are then routed to multimode sites called *macronodes*. Each macronode corresponds to one node in the RHG lattice, with the valence of the node equal to the number of modes within the macronode (the number of Bell pair halves meeting at that site); see Fig. S14b. At each macronode, Fourier transforms are applied to one mode in each pair to transform it into a two-qubit GKP cluster state (as in Eq. 7 in Ref. [1]), and then a network of beamsplitters is used to generate entanglement locally. In bulk, one can use a balanced four-splitter to stitch the macronodes [1, 70], whereas the boundary sites with a valence of less than four can be entangled with ancillae. For the discussion and simulations here, we choose to stitch the lattice together with a linear

cascade of beamsplitters—introduced in [13]—a more general construction that precludes the need for ancillae. Finally, one performs a sequence of homodyne measurements, followed by post-processing rules, to reduce this continuous-variable object to a qubit graph state and then to effect some qubit-level operation. The process of obtaining bit values from the homodyne outcomes, wherein the GKP error correction can be said to be performed, is the *inner decoding* step. Details of the reduction can be found in [1, 13], with an improved correlation-aware inner decoding method introduced in Sec. VIA.

In our fault-tolerance analysis, we consider a logical memory: teleporting a $|\bar{+}\rangle$ state encoded into the surface code onto $|\bar{+}\rangle$ fault-tolerantly by performing d complete rounds of error correction on this state, for code distance d . This requires us to combine the homodyne measurement outcomes into canonical p -homodyne measurements on all the modes of the lattice, which effect X measurements at the qubit level. Parities of combinations of X -measurement outcomes corresponding to the foliated code stabilizers are computed and used to form decoding graphs. As the surface code is a CSS code, we are able to construct two independent primal and dual decoding graphs whose edges are weighted by log likelihoods of estimated phase error probabilities. These probabilities are obtained from GKP soft information, including the modelled state quality and obtained homodyne outcomes, as in [1, 46]. The procedure is modified for the correlation-aware inner decoder, as discussed in Sec. VIA.

To model the noise in our architecture, we imagine we begin with ideal square-grid qunaught states and apply a classical Gaussian random noise (GRN) channel acting uniformly (the same for all modes) and isotropically with strength ϵ (noise along the p and q quadratures is equal). This results in a mixture of infinitely-many equally-separated Gaussians in phase space whose variance ϵ we associate with the symmetric effective squeezing relative to the square-grid qunaught state lattice (in particular, the value we cite in dB is equal to $-10 \log_{10}(2\epsilon)$). Although this model does not capture, for example, the finite-energy envelope effects of realistic GKP states, it is tractable for threshold estimations involving hundreds to thousands of modes. These noise channels can be commuted through the linear-optical stitching network to act before the measurements. Thus, we emulate noisy homodyne outcomes by sampling from uncorrelated Gaussians with variance ϵ . For more details, see the methods in the appendix of [1].

With the details above and the updated inner decoding method in Sec. VIA, we obtain a threshold plot for the architecture in Fig. S15. Simulations were performed using a private version of *FlamingPy* [71]. We observe a fault-tolerant threshold in the shaded region, $\gtrsim 9.75$ dB, a notable improvement over the $\gtrsim 10.1$ dB figure in [1], attributable to the improved inner decoding. Note a few differences between the simulation parameters: in the cited work, only primal error-correction is considered, the lattice is periodic in all three directions, and only balanced four-splitters are used to entangle modes within the macronodes.

A. Details on inner decoding

We begin with general considerations, where the system is initialized as a tensor product of square-grid qunaught states afflicted by Gaussian random noise. The modes then interact via an optical stitching circuit, followed by homodyne measurements, as described in the previous section. Following the measurements, conditional displacements are applied based on specific linear combinations of measured outcomes. These displacements need not be physically applied; they can be kept tracked of in software [1].

For this work, we consider the Gaussian random noise channels, described by probability distribution functions of the form $G_{\vec{\mu}, \Sigma}$, with mean $\vec{\mu}$ and covariance matrix Σ . When $\vec{\mu} = 0$,

we simply write G_Σ . The input noise, before the application of linear optimal elements is described by the input covariance matrix Σ_{in} , of size $2n \times 2n$, with n being the number of modes in the system, and with $\vec{0}$ mean. The optical circuit applied to the input GKP states can be described by a symplectic matrix S of the same dimensions. The covariance matrix after applying the optical circuits, but before performing any homodyne measurements, is given by

$$\Sigma_{\text{out}} = S\Sigma_{\text{in}}S^T. \quad (35)$$

After the homodyne measurements, the noise is described by the covariance matrix given by

$$\Sigma^{\text{meas}} = P\Sigma_{\text{out}}P^T, \quad (36)$$

where P is the $n \times 2n$ matrix indicating which rows of Σ_{out} corresponds to the measured quadrature outcomes. This can also be seen as taking the marginal of the probability distribution describing the noise on the measured quadratures. Additionally, linear transformations of the measured outcomes must be taken to return the system to the GKP qubit subspace [1]. Writing these transformations as an $n \times n$ matrix T , we get

$$\Sigma_Q^{\text{meas}} = T\Sigma^{\text{meas}}T^T. \quad (37)$$

In the absence of noise, applying the transformation T brings the system back into the GKP qubit subspace.

If the noise is uniform, *i.e.* $\Sigma_{\text{in}} = \epsilon I_{2n}$, we have that $\Sigma^{\text{meas}} = \epsilon I_n$. Furthermore, as long as the system coordinates are chosen carefully, Σ_Q^{meas} can be written in a particularly simple form given by

$$\Sigma_Q^{\text{meas}} = \bigoplus_{m \in \mathcal{M}} \begin{pmatrix} \epsilon|m| & \\ & \Sigma_m \end{pmatrix}, \quad (38)$$

where $\mathcal{M}$ is the set of macronodes, $|m|$ denotes the number of modes in macronode m , and Σ_m is a $|m-1| \times |m-1|$ positive-definite matrix. This allows a factorisation of $G_{\Sigma_Q^{\text{meas}}}$, the Gaussian probability distribution function of covariance Σ_Q^{meas} as

$$G_{\Sigma_Q^{\text{meas}}} = \prod_{m \in \mathcal{M}} G_{\epsilon|m|}(p_1^m) G_{\Sigma_m}(q_2^m, \dots, q_{|m|}^m), \quad (39)$$

with p_1^m the first mode in each macronode (measured in the p quadrature), and q_i^m the i th mode in macronode m (measured in the q quadrature).

The structure of Eq. 39 suggest a natural binning strategy to perform in parallel for the various macronodes, described in algorithm 1, that takes into account the correlations present. Given the factorisation of the distribution in Eq. 39, the homodyne outcomes of the modes measured in the p quadrature can be treated separately, using algorithm 1, while those modes that are part of a given macronode m and measured in the q quadrature are used to find a close valid point on the GKP lattice together via algorithm 1. Note that the input vector of algorithm 1 has been processed using the linear transformation T , which in practice also contains a factor of $\sqrt{\pi}$ in order to work with integer lattices. Given that binning strategy, weights can be computed by using algorithm 2 and used in the outer decoder, in this case MWPM. With this decoding algorithm, we find a threshold of ~ 9.75 dB as shown in Fig. S15, outperforming the standard GKP binning strategy used in [1].

The measurements obtained in the various macronodes are binned in parallel, given the structure of Eq. 39, and furthermore the mode measured in the p quadrature can be binned separately. Additionally, in practice one needs only consider a few terms in the sums present in algorithm 2.

Algorithm1 Correlation-aware inner decoder

Input: $\vec{x} = (x_1, \dots, x_n) \in \mathbb{R}^n$, positive definite $n \times n$ matrix Σ

Output: $\vec{q}_{min} \in \mathbb{Z}^n$ is returned

$$Q \leftarrow \bigcup_{\substack{q_i \in \{\lfloor x_i \rfloor, \lceil x_i \rceil\} \\ 1 \leq i \leq n}} \{(q_1, \dots, q_n)\}$$

return $\vec{q}_{min} = \arg \min_{\vec{q} \in Q} (\vec{q} - \vec{x}) \Sigma^{-1} (\vec{q} - \vec{x})^T$

Algorithm2 Weight calculation algorithm

Input: $\vec{x} \in \mathbb{R}^n$, positive definite $n \times n$ matrix Σ , $\vec{q} \in \mathbb{Z}$ and i , an integer labeling a qubit.

Output: weight w , the log-likelihood of the estimate of qubit i being in error

$\vec{u}_k \leftarrow \vec{q} + k\hat{e}_i$, with $k \in \mathbb{Z}$, and $\hat{e}_i$ the standard basis vector along i th dimension

$$a \leftarrow \sum_k G_{\vec{u}_{2k+1}, \Sigma}(\vec{x})$$

$$b \leftarrow \sum_k G_{\vec{u}_k, \Sigma}(\vec{x})$$

return $w \leftarrow \log(\frac{b-a}{a})$

VII. EXPERIMENT: THEORETICAL DESCRIPTION

The fault-tolerant universal architecture presented in this paper consists of GBS sources of approximate GKP states, refinery chips with MUX and breeding networks, and QPU chips, in addition to real-time decoding and active feedforward. Here we provide the mathematical details for a prototypical version of this architecture as well as experiments that capture its salient features. As described in the main text, the prototype consists of:

- *State sources.* Two-mode GBS sources capable of making squeezed cat states, which are precursors to approximate GKP states. Cat states with a particular peak spacing can be made probabilistically, heralded on a particular photon number. Otherwise, single-mode squeezed states can be produced on-demand.
- *Refineries.* Chips capable of boosting the probability of obtaining the squeezed cat states through a 4-to-1 MUX (without breeding), conditioned on photon number measurements. On these chips, states from independent sources are also entangled into pairs, the basic unit of a broader static linear-optical entangling circuit.
- *Fiber interconnects.* Fibers that connect the refinery chips to the QPU chips and include time delays so that the desired foliated cluster state is produced. In this case, the connectivity corresponds to the distance 2 foliated repetition code cluster state.
- *QPUs.* Chips for interacting the independent two-mode entangled states and measuring them out with homodyne detectors. These circuits can construct a qubit-level (GKP-encoded) or CV cluster state, depending on the inputs, whose nodes have alternating valence two and three. They are capable of performing two- and three-body GHZ measurements.
- *Classical computation and feedforward.* An FPGA to perform real-time inner and outer decoding, as well as capability for active updates to homodyne detection angles conditioned on the outcomes of the decoding.

These properties of the prototype allow us to implement a foliated distance-2 repetition code concatenated with (alternatively, whose physical qubits are) approximate GKP states.

For independent and identically distributed (IID) Pauli noise, the distance-2 code can only be used to detect errors. In our setting, the noise model and the decoder—informed about probabilities of qubit errors via soft information from GKP states—promotes the demonstration to error correction rather than merely detection. Note that the high degree of photon loss precludes the demonstration from being performant. This setup is a baseline for several important extensions. Additional state sources and QPU circuits capable of performing four-body GHZ measurements will increase the distance of the demonstrated code. Furthermore, although not a fully quantum code, the foliated repetition code is degenerate, in that multiple errors can result in the same syndrome, a property not found in classical codes [72]. One can also leverage the codes’ ability to suppress one kind of error (primal or dual); we know from [73], for example, that concatenating biased GKP states with repetition codes can be a promising encoding at a low level of larger concatenations.

A. Two-mode GBS devices for squeezed cat generation

As reviewed in Sec. IV A, two-mode GBS devices are excellent candidates for generating approximate squeezed cats, which can be thought of as low quality GKP states that consist of a superposition of only two squeezed states in one quadrature, as opposed to a superposition of a full comb of squeezed states. Moreover, squeezed cat states are “precursors” to GKP states, since they can be bred into GKP states using the breeding protocol as discussed in Sec. III A.

In this experiment, we employed two-mode GBS devices as sources of non-Gaussian states. Since the refinery chip did not include integrated homodyne detection, we did not consider breeding these states, nor did we consider using MB squeezing (Sec. III C) to be able to use the squeezed cats generated by multiple photon number patterns but on different lattices (different spacing between the two peaks). Rather, we targeted the squeezed cat generated by the $n = 2$ outcome. For the events in which no GBS device attached to a binary tree MUX successfully detected $n = 2$, a deterministic squeezed state source is always included as one of the inputs to the MUX to swap in to the cluster state.

In Fig. S16, we present the theoretical lossless two-mode GBS output state for 6 dB of input squeezing, and a beamsplitter transmissivity of 75%, heralding on photon number outcome $n=2$, which occurs with probability 7.35% for the idealized, lossless circuit. As discussed in Sec. IV A, we can improve the symmetric effective squeezing by setting the beamsplitter away from the “monomial condition”, as was done here. As was discussed in Sec. II B, we can choose approximate stabilizers for approximate GKP states, corresponding to a rescaling of phase space, to maximize the effective squeezing. In this case, we find that a peak spacing of 2.48 (2.69) in q (p), corresponding to a phase space unit cell area of $1.06 \times (2\pi)$ optimizes the symmetric effective squeezing to be 4.3 dB. Note that to match the square-grid quanaught state aspect ratio and peak at the origin of phase space, we need to consider applying a post-processing correction to the homodyne outcomes on the cluster state constructed from these states, consisting of a rescaling and a displacement corresponding to half the peak spacing in q .

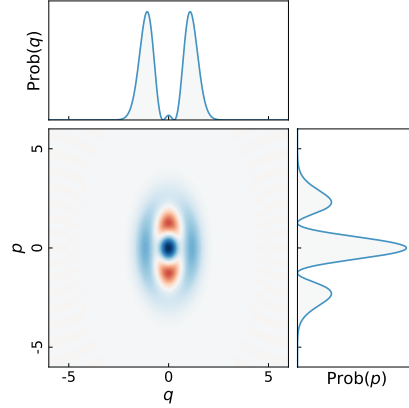


Figure S16. Two-mode GBS output state for 6 dB of input squeezing, a beamsplitter transmissivity of 75%, photon number outcome $n=2$, and no losses. See Fig. S9 for the circuit.

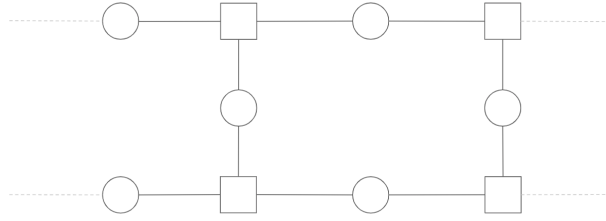


Figure S17. Graph state of a distance-2 foliated repetition code. The circular modes, which we call primal, form the parity checks or stabilizers. Phase errors on dual modes can be detected, but not corrected, by this code. We may also interpret the top and bottom rows to comprise the data qubits, with the middle nodes responsible for measuring two-body Z checks.

B. Cluster state for implementing the distance-2 foliated repetition code

1. Foliated repetition codes

The repetition codes are the simplest classical error-correcting codes. In a classical repetition code of length n , the bit value is repeated n times, i.e.,

$$i \mapsto \underbrace{ii \dots i}_n. \quad (40)$$

This is a distance- n code; if noise corrupts the message, flipping $0 \leftrightarrow 1$, one can correctly decode up to $(n-1)/2$ bit flip errors (for odd n), yielding a threshold for obtaining the correct codeword of 50% error probability in the limit of large n . This can be seen by applying simple majority voting to decode the information.

One can also use the repetition code to correct qubit errors; that is, one can use the code to correct either for Pauli X or Pauli Z errors but not both at once. The code can be defined

on n physical qubits, in which case the stabilizer generators are

$$\{Z_1Z_2, Z_2Z_3, \dots, Z_{n-1}Z_n\}. \quad (41)$$

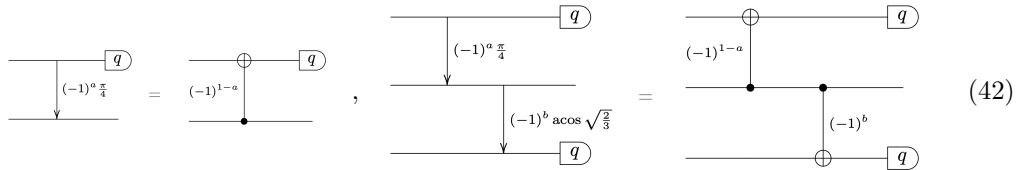
The two-qubit version of the above repetition code reduces to the single stabilizer $\{Z_1Z_2\}$ and the logical codeword $|\bar{0}\rangle \equiv \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$, i.e., a Bell pair. This is a distance-2 code: it is capable of detecting a (bit-flip) error on a single qubit, but incapable of correcting it.

Since measurements are faulty as well—that is, ancilla qubits are also imperfect—one is required to perform multiple rounds of error correction. In the measurement-based framework, this means one is required to foliate the (1D) cluster state of the repetition code to create a 2D cluster, where one can interpret the even rows as repetition code checks and the odd rows as teleportation steps. This is shown in Fig. S17. In the foliated repetition code—an example of a degenerate code, making it a truly quantum code—the notion of qubit Z and X errors instead maps to primal Z and dual Z errors, *i.e.* phase errors on primal (circular) and dual (square) modes. The repetition code is able to correct primal but not dual errors. The relevant foliated stabilizers are four-qubit X checks.

2. Distance-2 repetition code with GKP states and linear-optical entanglement

To implement the distance-2 repetition code above with GKP states and static linear optics, we follow the theory described briefly above and more comprehensively in [1, 13]. The result is in Fig. S18. We associate edges with Bell pairs, creating macronodes with 2 and 3 modes. The macronodes are entangled with a modified linear beamsplitter cascade [13]. For two modes this is just a single beamsplitter with transmissivity $1/2$; for three modes, one applies an additional beamsplitter with transmissivity $2/3$, as shown in Fig. S18. Indices 3, 4, 7, 9, and 10 correspond to the central modes of each macronode, i.e. they govern the desired homodyne measurement basis on the reduced (qubit GKP) graph. The rest of the modes are measured out in q . Measuring p , $p \pm q$, and q on the central mode then effects a Pauli X , Y , and Z measurement, respectively, on the reduced graph state. Homodyne outcomes are processed and binned, as described in the next section, to obtain the corresponding bit values.

Equivalence of the passive linear-optical network in Fig S18(a) to the qubit graph for the distance-2 repetition code relies on the creation of a GHZ state among the modes of each macronode. Reductions such as this are examined in [13]; we can reduce the 2- and 3-mode macronodes in the experiment via an LDU decomposition of the beamsplitter [12] as follows:



$$\begin{array}{c} \text{2-mode macronode} \\ \downarrow (-1)^a \frac{\pi}{4} \end{array} = \begin{array}{c} \text{qubit graph} \\ \oplus \text{ on top line, } \bullet \text{ on bottom line} \end{array}, \quad \begin{array}{c} \text{3-mode macronode} \\ \downarrow (-1)^a \frac{\pi}{4} \\ \downarrow (-1)^b \arccos \sqrt{\frac{2}{3}} \end{array} = \begin{array}{c} \text{qubit graph} \\ \oplus \text{ on top line, } \bullet \text{ on middle line, } \oplus \text{ on bottom line} \end{array} \quad (42)$$

where $a, b \in \{0, 1\}$ and are determined by a π -phase on the beam splitters (which varies between macronodes). The repetition code cluster state emerges by commuting the $\hat{C}_X$ gates to the input states, a process which copies each Bell pair's entanglement across the $\hat{C}_X$ gates to the central modes. The $\hat{C}_X$ gates then trivially act on the (ideal) input states.

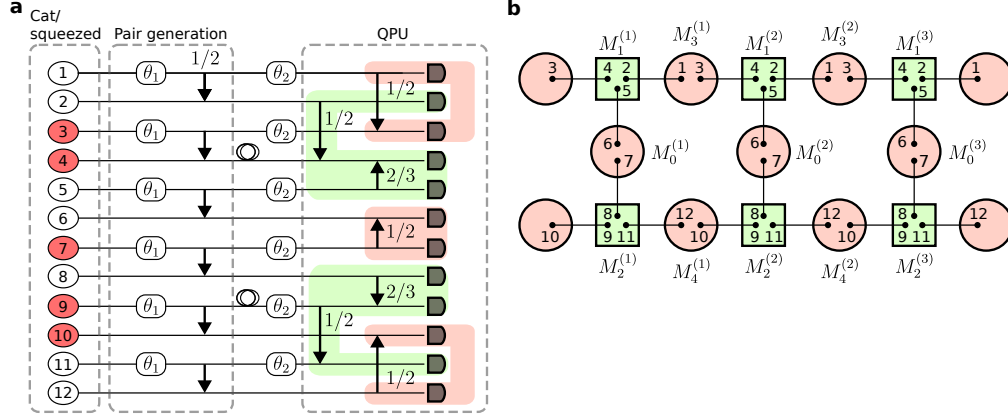


Figure S18. **a. Circuit diagram for synthesizing the system's multimode entangled state.** Input modes, containing squeezed cats or squeezed states, are labeled from 1 to 12. They are entangled into two-mode Bell pairs (pair generation) with a single-mode phase shift labelled θ_1 (Fourier transform, $\hat{R}(\pi/2)$) followed by a 50:50 (1/2) beamsplitter. Bell pairs are routed towards the QPU that entangles pairs into a spatio-temporal cluster state and implement gates with balanced-homodyne detectors. Two of the modes, 4 and 9, are temporally delayed by one clock cycle to interfere with modes from an adjacent time bin. Additional phase shifts labelled θ_2 (Fourier transforms) are applied to convert Bell pairs into two-mode cluster states, or in the case where squeezed states only are used, converting two-mode squeezed states into two-mode continuous-variable cluster states. The QPUs implement a network of beamsplitters and homodyne measurements, equivalent to implementing 2- or 3-mode GHZ basis measurements. Input modes highlighted in red correspond to the central modes of each macronode, i.e. they govern the desired homodyne measurement basis on the reduced (qubit GKP) graph. **b. Graph representation of the 12-mode cluster state.** For each time window ($t \pm \tau$), Bell pairs (numbered dots joined by a solid line) form the macronodes, clustered in doublets (colored circles) or triplets (colored squares). Two time stitches (between modes 3 and 4, and modes 9 and 10) in the circuit allow computation, and entanglement, to persist in time.

3. Inner and outer decoding

In the repetition code experiment, we use two levels of decoding: inner decoding, to translate homodyne outcomes into bit values and estimate the GKP-qubit-level error probabilities, followed by the outer (qubit) decoding. In the experiment, the inner decoder used is based on the GKP binning strategy described in Sec. VIA, considering the inner decoding problem over a single mode without taking the correlations between modes into account. This involves considering the covariance matrix Σ_Q^{meas} to be matrix in Eq. 38 with non-diagonal elements set to 0. We will use the same notation as in Sec. VIA.

The input state, which is a two-peaked state modeled as a GKP state, contains two peaks in the q quadrature with a spacing of 2.48 (see Sec. VII A). As the input to the circuit should ideally be a quanaught state [1] which has a peak at the origin, we first undo the effect of the displacement of 1.24, which is half of the space between the peaks of the input state, on the homodyne outcomes. This involves processing the homodyne outcomes to remove the relevant displacements on the measured quadratures of the optical state prior to homodyne measurement obtained by propagating the initial displacements through the passive linear-optical network. Here we will use x_i to denote the homodyne outcome of mode i and $x_i^{(p)}$

for the processed homodyne outcome.

To bring the system back to the GKP qubit subspace, a linear transformation T based on the equivalence of the quantum optical circuit to the GKP qubit foliated quantum code is applied on the processed homodyne outcomes within a macronode [1]. Let $\mathbf{x}_i$ be the transformed homodyne outcome. Note that the cluster state shown in Fig. S18 has two types of macronodes—some containing two modes and others containing three modes—and that the 12 modes are arranged into 5 macronodes labeled by M_0 , M_1 , M_2 , M_3 , and M_4 , as described in table SI.

Node	M_0	M_1	M_2	M_3	M_4
Modes in macronode	6, 7	2, 4, 5	8, 9, 11	1, 3	10, 12

Table SI. Nodes in the distance-2 foliated repetition code and the modes in their corresponding macronodes.

Within each macronode, we label the mode measured in p as the first mode. When the macronode has two modes, the other mode is the second mode. In macronodes M_1 and M_2 , modes 5 and 2 and modes 8 and 11 are the second and third modes, respectively. Let j_l be the index of the l^{th} mode of macronode M_j , where $l \in \{1, \dots, |M_j|\}$ and $|M_j|$ is the number of modes in macronode M_j . Let $\bar{x}_j^{(p)} = [x_{j_l}^{(p)}]_{l=1}^{|M_j|}$ be a vector of processed homodyne outcomes of modes in macronode M_j .

The homodyne transformation matrices $T^{(2)}$, $T^{(3, M_1)}$ and $T^{(3, M_2)}$ used for modes in the two-mode macronodes, three-mode macronode M_1 , and three-mode macronode M_2 in Fig. S18 are the following:

$$T^{(2)} = \begin{bmatrix} \sqrt{2} & 0 \\ 0 & \sqrt{2} \end{bmatrix}, \quad T^{(3, M_1)} = \begin{bmatrix} \sqrt{3} & 0 & 0 \\ 0 & \sqrt{\frac{3}{2}} & \frac{1}{\sqrt{2}} \\ 0 & 0 & \sqrt{2} \end{bmatrix} \quad \text{and} \quad T^{(3, M_2)} = \begin{bmatrix} \sqrt{3} & 0 & 0 \\ 0 & \sqrt{\frac{3}{2}} & -\frac{1}{\sqrt{2}} \\ 0 & 0 & \sqrt{2} \end{bmatrix} \quad (43)$$

We note that these homodyne transformation matrices correspond to the transformation T in Sec. VIA. The homodyne transformation T corresponding to the macronode M_j is applied to the processed homodyne outcomes vector $\bar{x}_j^{(p)}$ to obtain the transformed homodyne outcomes vector $\bar{\mathbf{x}}_j^{\text{T}} := T\bar{x}_j^{(p)\text{T}}$, where $\bar{\mathbf{x}}_j = [\mathbf{x}_{j_l}]_{l=1}^{|M_j|}$ used in the inner decoder.

The inner decoding involves using these transformed homodyne outcomes $\mathbf{x}_i$ to obtain estimates for bit values and probability of qubit errors of each macronode. We use the single mode binning function for inner decoding that involves performing the procedure in algorithm 1 and algorithm 2 over a single mode, and using $\Sigma = \sigma^2 \mathbf{I}$ to obtain the bit value and probability of qubit error as follows:

$$b^{(i)} = b(\mathbf{x}_i) = \left\lfloor \frac{\mathbf{x}_i}{\sqrt{\pi}} \right\rfloor \bmod 2,$$

$$p^{(i)} = p(\mathbf{x}_i, \sigma^2) = \frac{\sum_{i=-l}^l \exp(-(\tilde{\mathbf{x}}_i - (2n+1)\sqrt{\pi})^2/2\sigma^2)}{\sum_{i=-2l}^{2l+1} \exp(-(\tilde{\mathbf{x}}_i - n\sqrt{\pi})^2/2\sigma^2)}, \quad (44)$$

where $\tilde{\mathbf{x}}_i = \mathbf{x}_i - \left\lfloor \frac{\mathbf{x}_i}{\sqrt{\pi}} \right\rfloor \sqrt{\pi}$ and $l = 1$.

Assuming no squeezed states are swapped in, the processed homodyne outcomes on the second and third modes (sometimes referred to as satellite modes [1]) are binned individually, since if there is no noise, the measurement probabilities are equivalent to that of ideal GKP states [1]. When the satellite modes are initialized with squeezed states rather than GKP, they are binned together with the central mode of the neighboring macronode containing the mode that forms a dumbbell with the satellite mode. The central modes are binned together with the satellite modes of the neighboring macronodes that are initially in p-squeezed states and form dumbbells with the modes in the central mode's macronode. The bit value and the probability of error are obtained by combining the results from the binning problems corresponding to the central mode of the qubit's macronode and the satellite modes from neighboring macronodes that form a dumbbell with the modes in the macronode using logical XOR operations and the union bound [1].

We next provide explicit expressions for obtaining the bit value and the probability of error for each macronode in our experiment. Let gkp_i be a Boolean variable that is true when mode i in Fig. S18 is a GKP state. The expressions used to obtain the bit value and the qubit error probability of the nodes in the inner decoder are provided in table SII to table SVI, where ϵ is the variance of the uncorrelated Gaussian noise considered initially over each mode.

gkp_5	gkp_8	b_0	p_0
true	true	$b(\mathbf{x}_7) \oplus b(\mathbf{x}_8) \oplus b(-\mathbf{x}_5)$	$p(\mathbf{x}_7, 2\epsilon) + p(\mathbf{x}_8, 2\epsilon) + p(-\mathbf{x}_5, 2\epsilon)$
true	false	$b(\mathbf{x}_7 - \mathbf{x}_8) \oplus b(-\mathbf{x}_5)$	$p(\mathbf{x}_7 - \mathbf{x}_8, 4\epsilon) + p(-\mathbf{x}_5, 2\epsilon)$
false	true	$b(\mathbf{x}_7 + \mathbf{x}_5) \oplus b(\mathbf{x}_8)$	$p(\mathbf{x}_7 + \mathbf{x}_5, 4\epsilon) + p(\mathbf{x}_8, 2\epsilon)$
false	false	$b(\mathbf{x}_7 + \mathbf{x}_5 - \mathbf{x}_8)$	$p(\mathbf{x}_7 + \mathbf{x}_5 - \mathbf{x}_8, 6\epsilon)$

Table SII. Computation of bit value and probability of error for node M_0 .

gkp_1	gkp_6	b_1	p_1
true	true	$b(\mathbf{x}_4) \oplus b(\mathbf{x}_1) \oplus b(\mathbf{x}_6)$	$p(\mathbf{x}_4, 3\epsilon) + p(\mathbf{x}_1, 2\epsilon) + p(\mathbf{x}_6, 2\epsilon)$
true	false	$b(\mathbf{x}_4 - \mathbf{x}_6) \oplus b(\mathbf{x}_1)$	$p(\mathbf{x}_4 - \mathbf{x}_6, 5\epsilon) + p(\mathbf{x}_1, 2\epsilon)$
false	true	$b(\mathbf{x}_4 - \mathbf{x}_1) \oplus b(\mathbf{x}_6)$	$p(\mathbf{x}_4 - \mathbf{x}_1, 5\epsilon) + p(\mathbf{x}_6, 2\epsilon)$
false	false	$b(\mathbf{x}_4 - \mathbf{x}_1 - \mathbf{x}_6)$	$p(\mathbf{x}_4 - \mathbf{x}_1 - \mathbf{x}_6, 7\epsilon)$

Table SIII. Computation of bit value and probability of error for node M_1 .

gkp_12	b_2	p_2
true	$b(\mathbf{x}_9) \oplus b(-\mathbf{x}_{12})$	$p(\mathbf{x}_9, 3\epsilon) + p(-\mathbf{x}_{12}, 2\epsilon)$
false	$b(\mathbf{x}_9 + \mathbf{x}_{12})$	$p(\mathbf{x}_9 + \mathbf{x}_{12}, 5\epsilon)$

Table SIV. Computation of bit value and probability of error for node M_2 .

The bit values and the probability of qubit errors obtained from the inner decoder are further used for outer decoding using the foliated distance-2 repetition code. We perform decoding using the macronodes shown in Fig. S19, where each macronode yields a single qubit after the processing and inner decoding described above. The decoding algorithm uses a bipartite graph, called the Tanner graph, comprising nodes for every qubit and every stabilizer, with edges between a qubit node and a stabilizer node if a qubit is in the support

gkp_2	b_3	p_3
true	$b(\mathbf{x}_3) \oplus b(\mathbf{x}_2)$	$p(\mathbf{x}_3, 2\epsilon) + p(\mathbf{x}_2, 2\epsilon)$
false	$b(\mathbf{x}_3 - \mathbf{x}_2)$	$p(\mathbf{x}_3 - \mathbf{x}_2, 4\epsilon)$

Table SV. Computation of bit value and probability of error for node M_3 .

gkp_11	b_4	p_4
true	$b(\mathbf{x}_{10}) \oplus b(\mathbf{x}_{11})$	$p(\mathbf{x}_{10}, 2\epsilon) + p(\mathbf{x}_{11}, 2\epsilon)$
false	$b(\mathbf{x}_{10} - \mathbf{x}_{11})$	$p(\mathbf{x}_{10} - \mathbf{x}_{11}, 4\epsilon)$

Table SVI. Computation of bit value and probability of error for node M_4

of that stabilizer. The qubit node and the stabilizer node are also called the variable node and the check node, respectively.

The Tanner graph for our decoding problem provided in Fig. S20 is a tree and does not contain any cycle. The belief propagation (BP) decoder is known to be optimal for trees [74] [75]; hence, it is our decoder of choice. We note that the BP decoder might not converge for some values of the input probabilities as the code used is an error detecting code for an independent and identically distributed channel. When the decoder converges, it provides the most likely error based on the syndrome, which is sub-optimal for decoding of a quantum code.

The inputs to the min-sum belief propagation decoder are the probability of the qubit errors and the syndromes of the two stabilizers obtained from the bit values as $s_1 = b_0^{(1)} \oplus b_3^{(1)} \oplus b_4^{(1)} \oplus b_0^{(2)}$ and $s_2 = b_0^{(2)} \oplus b_3^{(2)} \oplus b_4^{(2)} \oplus b_0^{(3)}$. We refer to log-likelihood ratios as *beliefs*; they have a higher value when the probability of error is lower. The initial beliefs of the variable nodes are set to the log-likelihood ratios $\log((1 - p_i)/p_i)$ based on the probability of qubit errors p_i . The decoding algorithm consists of alternating cycles of sending messages from the check nodes to the variable nodes, and from the variable nodes to the check nodes. These messages are the beliefs of the corresponding variable nodes updated based on the syndrome and other variable node's beliefs. The first set of messages from the variable nodes

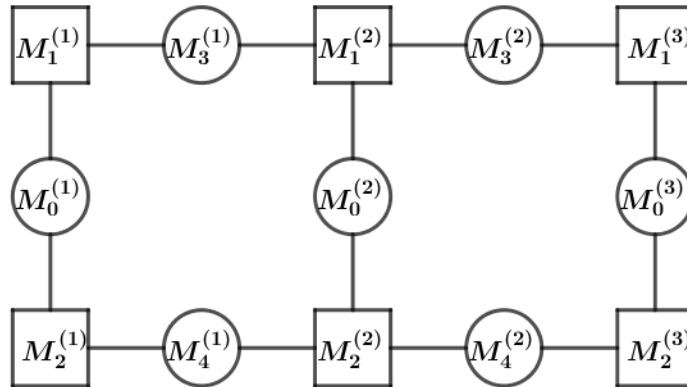


Figure S19. The subgraph of the graph of the distance-2 foliated repetition code considered for decoding with the nodes considered for decoding represented by circles.

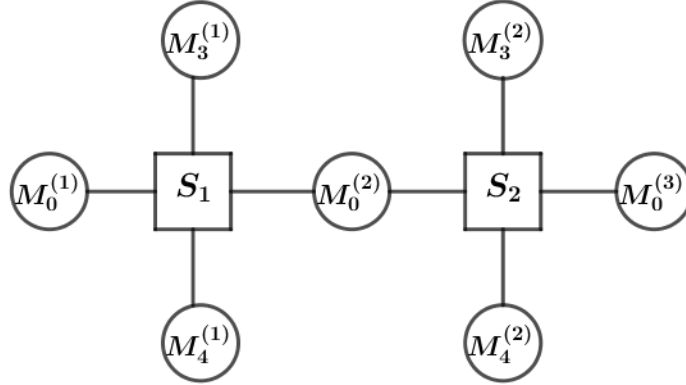


Figure S20. Tanner graph of the distance-2 foliated repetition code

to the check nodes are the initial beliefs of the corresponding variable nodes. The message from a check node i to a variable node is the minimum value of the messages received at that check node from other variable nodes multiplied with the signs of all those received messages and $(-1)^{s_i}$. The message from a variable node to a check node is the sum of all other messages received at that variable node and the initial belief of the variable node. At each stage, a provisional recovery is computed and checked. The recovery computation involves computing a belief for each variable node to be the sum of all incoming messages and the initial belief, and mapping the positive beliefs to bit value 0 and negative beliefs to bit value 1. Thus, the recovery consists of one bit at every variable node. If the logical XOR of the recovery bits of variable nodes adjacent to a check node is equal to its syndrome, the decoding stops as the recovery operator would take the state back to the codespace. In addition to outputting the recovery operator, the decoder outputs the *soft decision*, which is the updated post-recovery probability of qubit error based on the belief obtained at the end of BP decoding.

C. Entanglement verification

In our measurement-based architecture, the prepared cluster state is the entanglement resource on which projective measurements are performed to implement gates and circuits like the distance-2 repetition code in Sec. VII B. Thus, in validating the performance of our system, we validate the entanglement in the prepared cluster state with the measurement apparatus used for projective measurements.

The refinery chips, along with the connected GBS devices, comprise the fundamental entanglement resources of our architecture: Two-mode entangled states being a combination of probabilistically generated approximate GKP states (a squeezed cat state in the experiment), and deterministically generated squeezed states, interfered in the refinery. As shown in Fig. S18, this fundamental entanglement resource is then spatially and temporally distributed among macronodes in the QPU such that macronodes are connected by two-mode entanglement.

Each macronode consists of a beamsplitter unitary followed by quadrature measurements by homodyne detection—shown in Fig. S18a. As in Sec. VI, we may consider the QPU uni-

tary as transforming the fundamental two-mode entanglement, shared among macronodes, into a larger more complex entanglement structure with now multiple cluster state edges between all modes of macronodes sharing two-mode entanglement—a multilayered version of the simpler macronode graph similar to the quad-rail lattice [76, 77]. Measurement-based gates, or entanglement verification, can then be performed by the homodyne detection of this cluster state of complex entanglement structure.

In the presence of high photon loss, it is difficult to validate full genuine inseparability between all modes of the complex entanglement structure after the QPU unitary. Instead, we will validate the pairwise inseparability between neighboring macronodes, as the entanglement across a link connecting macronodes is what carries the encoded information in the architecture when implementing gates [78]. After the QPU unitary, information is encoded in *distributed modes* within modes of each macronode, and the entanglement seen in these distributed mode bases is the initial two-mode entanglement shared by macronodes [78, 79].

Since the distributed mode bases exactly correspond to the QPU unitary applied to the physical modes, another equivalent perspective is to consider the QPU unitary together with the homodyne detection of each macronode as a multimode quadrature measurement, each multimode measurement corresponding to a colored region of Fig. S18a. From this perspective, the entanglement verification relaxes to verification of the fundamental two-mode entanglement output from the refinery chips. For simplicity, the refinery MUX is set to pass squeezed states for the entanglement generation, generating two-mode squeezing with nullifiers $n_q = (q_A - q_B)/\sqrt{2}$ and $n_p = (p_A + p_B)/\sqrt{2}$ where mode A and B are two output modes of a refinery. By the Duan inseparability criterion [80] a two-mode squeezed state is inseparable in quadrature when $\langle \Delta n_q^2 \rangle + \langle \Delta n_p^2 \rangle$ is measured to be less than 2 vacuum units, corresponding to measuring variance of both n_q and n_p below shot noise. Note, imperfections as loss in the QPU unitary are now included in the detection and still affect the entanglement verification.

To measure q_A and q_B of n_q (or p_A and p_B of n_p) each homodyne detector of the macronode to which mode A and B are input are set to measure in q (or p) quadrature. The QPU unitary within a macronode can then be undone by taking linear combinations of the homodyne measurement outcomes [78]. As example, consider a two-mode macronode with input mode α and β , and a QPU unitary (in this case a single beamsplitter) with symplectic matrix $\mathbf{S}$ transforming the input quadratures before homodyne detection as

$$\begin{pmatrix} q'_\alpha \\ q'_\beta \\ p'_\alpha \\ p'_\beta \end{pmatrix} = \mathbf{S} \begin{pmatrix} q_\alpha \\ q_\beta \\ p_\alpha \\ p_\beta \end{pmatrix} = \begin{pmatrix} \mathbf{S}_{qq} & \mathbf{S}_{qp} \\ \mathbf{S}_{pq} & \mathbf{S}_{pp} \end{pmatrix} \begin{pmatrix} q_\alpha \\ q_\beta \\ p_\alpha \\ p_\beta \end{pmatrix}. \quad (45)$$

As long as the QPU unitary does not mix quadratures such that $\mathbf{S}_{qp} = \mathbf{S}_{pq} = \mathbf{0}$ (as is the case for our architecture), measured quadratures input to the QPU unitary are extracted as

$$\begin{pmatrix} q_\alpha \\ q_\beta \end{pmatrix} = \mathbf{S}_{qq}^{-1} \begin{pmatrix} q'_\alpha \\ q'_\beta \end{pmatrix} \quad , \quad \begin{pmatrix} p_\alpha \\ p_\beta \end{pmatrix} = \mathbf{S}_{pp}^{-1} \begin{pmatrix} p'_\alpha \\ p'_\beta \end{pmatrix}. \quad (46)$$

and similarly for the 3-mode macronodes in our system.

Changing perspective, considering the QPU unitaries as part of the entangled resource generation—rather than as a part of implementing multimode measurements on inseparable pairs—inseparability between two macronodes (but not full inseparability between all modes of the two macronodes) can be verified using the van Loock-Furusawa criterion [81]: Modes of two macronodes are distributed in two sets, one set corresponding to one macronode, and nullifiers of the complex entanglement structure are measured. In this case, n_q and

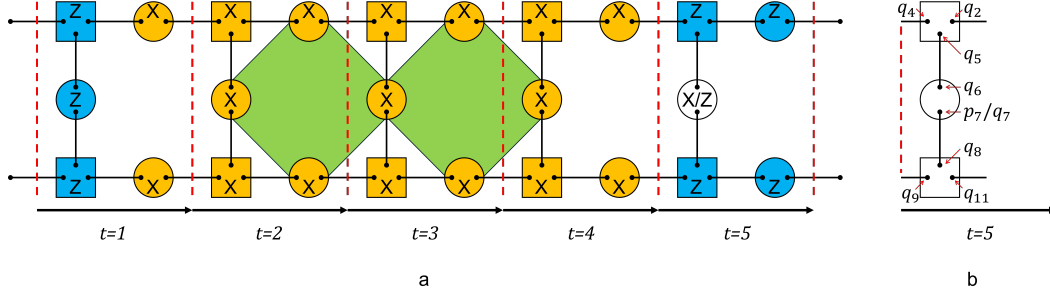


Figure S21. **a. Measurement pattern for decoding and adaptive feedforward measurement.** This measurement-based protocol uses five time steps of our experiment. X and Z labels indicate choice of measurement basis, and the X/Z label is used to indicate that either can be chosen. Decoding is implemented based on processing homodyne data from time steps 1 to 4. The green diamonds indicate four-body X stabilizers used for syndrome extraction in the outer decoder. After decoding, a decision is made to measure the left-center macronode in the last time step in either the X or the Z basis. See text for details. **b. Quadrature basis labels for left-most macronodes at the last time step.**

n_p can be propagated through the QPU unitary as nullifiers. Carrying out this exercise, one again finds an inseparability criterion corresponding to the Duan inseparability criterion with said-inseparability verified when the new nullifiers are measured to have variance below shot noise.

D. Adaptivity demo

Fault-tolerant quantum computation requires real-time decoding and active feedforward. Pauli corrections in Clifford circuits remain Pauli, meaning that one can perform decoding and recovery passively by (efficiently) updating the Pauli frame “in software”. On the other hand, Pauli operators in non-Clifford circuits may propagate outside of the Pauli frame. For example, T gates map Pauli operators to Cliffords—keeping track of the Clifford frame is no longer scalable, requiring one to know the Pauli frame prior to the application of the T gate to know which subsequent Clifford correction to apply. This necessitates real-time decoding and fast feedforward [82].

In our setup we demonstrate the machinery involved in this critical piece of a fault-tolerant computation: real-time error-correction and active feedforward in the next time step. The steps of the protocol, illustrated in Fig. S21, are as follows (with the mapping between qubit-level Pauli measurements and mode-level homodyne measurements in VII B 2):

1. A region of the cluster state is carved out for a fresh experiment via decoupling qubit Z in macronodes M_0 , M_1 , and M_2 (blue modes in $t = 1$ in the figure above). The remaining two qubits are measured in X (macronodes M_3 and M_4 at $t = 1$).
2. Two complete foliated repetition code checks are evaluated—that is, two overlapping four-body stabilizers are measured with a sequence of seven X measurements (macronodes M_0 , M_3 , and M_4 across $t = 2$ and $t = 4$); see also Fig. S19 and Fig. S20. These checks diagnose Pauli Z errors on the primal qubits of the code. Additionally, six teleporting X measurements are performed on modes M_1 and M_2 across these same timesteps, which would be interpreted as the dual checks in a fully quantum code.

3. To compute the bit values and error probabilities to be fed into the outer decoder, we use the true obtained homodyne measurement outcomes and the expressions in Table SII to Table SVI. With these outcomes, and with soft information from homodyne outcomes and prior characterization of the system, belief-propagation decoding is performed, as discussed in Sec. VII B 3. A number of variables are taken as inputs for the noise model. We use the lattice spacing, peak variances, and dependence on the error probabilities commensurate with a lossless version of the experiment; in the low-loss regime, the expressions become a faithful representation of the underlying physics.
4. After a few iterations, the BP algorithm outputs a set of estimates of error probabilities known as soft decisions, as well as the recovery. We can take this distribution of soft decisions and, based on its properties, assess our confidence in the recovery. When the probability of each logical error is much smaller than the probability of no error, we can be confident in the recovery. The exact expressions of the probability of each logical error can be computed and compared to the probability of no error. We note that when the probability of each qubit is below a particular value, we can be confident that the probability of any logical error is small. Thus, the heuristic we choose, as a function of the post-recovery probabilities P , is a simple $T(P) = \max P$. Additional details of the FPGA implementation are presented in Sec. VIII E 1.
5. We use the thresholding function in the previous step to make a decision about which homodyne measurement to perform on a mode in the following clock cycle ($t = 5$). The scenarios are as follows: If $T(P)$ is smaller than some pre-determined threshold, ξ , we are confident in the recovery and therefore maintain entanglement by performing an additional repetition code check (mode M_0 at $t = 5$)—we call this the “keep” scenario. If $T(P)$ is greater than ξ , we instead decouple the check mode M_0 from the state by performing a Z measurement—we call this the “cut” scenario. We also discard the state in the rare cases that BP does not converge, such as for an input probability distribution uniformly at 50%. For this experiment, we choose a threshold of $\xi = 0.2689$ as, through simulations, we observed that the probability of obtaining a logical error was low when each qubit had a probability of error below this threshold.
6. The “keep” and “cut” scenarios each have different implications on the topology of the cluster state. In the “keep” case we are measuring a cluster state stabilizer $Z_2 X_0 Z_1$, and so (ideally) expect a deterministic outcome when combining the measurements. For this investigation we postselect the modes 5, 6, 7, and 8 to be in squeezed states (i.e. free of squeezed cats). The expected behaviour is shown in the simulated data plots in Fig. 3c in the main text.

VIII. EXPERIMENT: FULL SYSTEM BLUEPRINT

The highly modular system presented in the main text, Aurora, consists of several subsystems, labeled in the top band of Fig. S22, interconnected with each other via fiber optical cables. Each subsystem is distributed among different assemblies containing one or more modules and mounted in a total of four server racks. The laser system, Fig. S24, is fully contained in a single server rack and coupled to all three sources assemblies via bundles of single-mode fibers. On the blueprint (Fig. S22), dashed lines represent trains of pulses (classical, red and quantum, black.) delayed by one clock cycle relative to those with solid lines, for reasons described in Sec. VIII B. Eight Gaussian Boson Sampling (GBS) modules are

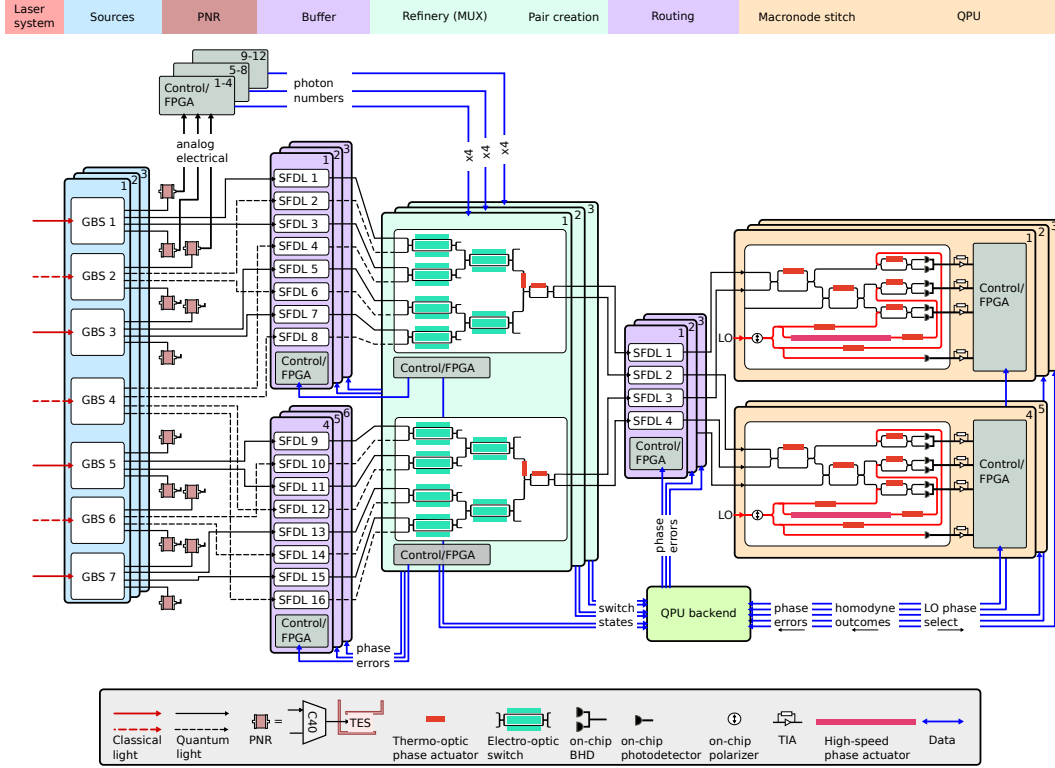


Figure S22. Full system hardware blueprint. The full system, named Aurora, is furnished with all the important stages and networking infrastructure of our FTQC architecture. Gaussian states and heralded non-Gaussian states are prepared at the sources subsystem, comprised of Gaussian Boson Sampling (GBS) cells, coupled to the Laser subsystem at its input and transition-edge sensors, for PNR detection, at a subset of its output. In total, the system hosts 84 squeezers, distributed across an array of 21 GBS modules (in addition to three extras for redundancy, not shown), providing 12 squeezed states and 36 heralded non-Gaussian states using 36 PNR detectors. The 48 outputs are fed into the refinery (MUX), hosting adaptive interferometer (switch) trees and a beamsplitter for pair creation. There are 12 binary switch trees across six MUX modules, each yielding one entangled pair. To compensate for the latency associated with specific photon number heralding events, arrays of stabilized fiber delay lines (SFDL) are inserted in each MUX input (buffer). More SFDLs route Bell pairs towards the QPUs, responsible for entangling all pairs into a spatio-temporal cluster state and implement measurements with on-chip balanced-homodyne detectors (BHD) across all five QPUs. Continuous and autonomous operations of the system is enabled by persistent optical phase locks, fast processing and data streaming to and from a control computer (not shown).

hosted per assembly, one being a spare, for a total of 21 GBS modules active simultaneously in the system. Each GBS circuit (Fig. S26) receives modulated light from the laser system and contains four squeezers and two interferometers to generate either two-mode Gaussian states or single-mode squeezed states.

To generate squeezed cat states, one spatial mode per pair is directed towards an array of PNR detectors, each consisting of a fiber-based wavelength-division multiplexing filter (C40) to filter out residual pump photons and a TES. There is a total of 36 PNR detectors

in the system, measuring photon numbers and heralding squeezed cat states (as described in Sec. VII A) in the event of a successful detection outcome (photon number equal to 2). Information about photon numbers from the PNR system is fed forward to the refinery (MUX) for selecting the switch pattern to boost the probability of cat states for pair generation. Note that one of the GBS modules per assembly is configured to deterministically output four single-mode squeezed states, coupled onto the dedicated input ports of each refinery, as depicted in Fig. S22 (GBS 4). Each refinery module has eight quantum inputs and two layers of adaptive beamsplitters (fast switches), routing any one of the four top inputs (three potential squeezed cats plus one deterministic squeezed state) and any one of the four bottom inputs (with the same input states as the top) towards the on-chip beamsplitter, at every experiment clock cycle (1 MHz repetition rate). One group of three PNR detectors control the top and another group of three PNR detectors control the bottom three switches at each refinery module, with the help of field programmable gate arrays (FPGAs) and data optical links. Two modules are integrated into an assembly, with a total of three assemblies in the full system.

To bridge the gap between the latency arising from photon-number extraction, transmissions and switching overhead, (stabilized) fiber delay lines (SFDLs) are inserted between the sources and refinery arrays. Assembled groups of eight identical SFDLs form the *buffer* subsystem and are responsible for delaying quantum pulses for over a microsecond, while stabilizing optical phases and polarization states of light, with the help of interleaved reference pulses (see Sec. IX B). There is a total of 48 buffer SFDLs in the system, one for each input to the refinery.

All six pairs of the refinery array are coupled out to the routing subsystem, responsible for distributing, in an actively phase- and polarization- stabilized fashion, entangled pairs to different QPUs. Two out of the 12 modes have an extra delay (not shown), each used to interfere two modes of adjacent time bins at their respective QPU. To synthesise the spatio-temporal cluster state, two or three spatial channels are combined together on each QPU chip, depending on the required connectivity (as described in Sec. VII B) among the five QPU units. For each module, the beamsplitter combining the top two inputs is set to 50:50, while the second BS is set to 25:75 or 100:0 (bar) in the three-mode and two-mode case, respectively. Each QPU also has one input dedicated to the local oscillator (LO) used for phase-sensitive quadrature measurements. The LO input is equally split on-chip into four spatial modes: one for monitoring purposes, e.g. set the state of polarization, while the other three are guided towards three different balanced homodyne detectors (BHD). On-chip thermo-optic (slow) phase actuators are responsible for setting static phases of the interferometers, variable beamsplitters for homodyne measurements and LO phases, with the exception of one phase actuator for the middle homodyne detector. The latter is much faster than the thermo-optic heaters and is used for feedforward operations in that mode. Therefore, we only have the ability to change the homodyne basis quickly (per clock cycle) on one mode per macronode. This is sufficient for full programmability of the cluster state; all other modes in the macronode can be measured in the same fixed quadrature basis [1, 13].

The QPU backend orchestrates data flow from refinery and QPU (frontend) and towards the routing SFDL modules for phase locking QPU inputs and back to QPU for fast feedforward adaptive measurements. The QPU backend is also responsible for implementing decoding protocols (described in Sec. VII B 3). Each subsystem and assembly house their own dedicated FPGAs and control boards (control-FPGA) for configuring and operating the modules and hence, initializing the full system to synthesize a resource state for measurement-based quantum computing. Complete operation is orchestrated by a single server computer (not shown).

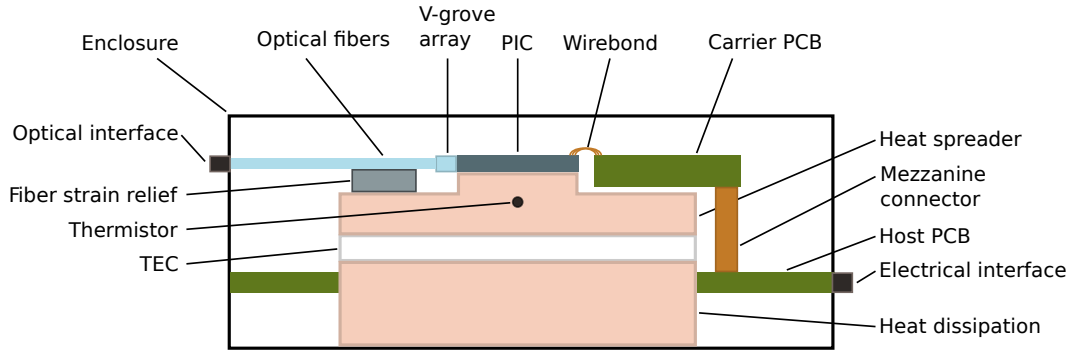


Figure S23. **Packaged PIC module** From bare chips to final modules, sources, refinery and QPU share a common packaging process with their optical and electrical interfaces exposed outside their enclosure.

A. Module Packaging

Each subsystem module, composed of photonic integrated circuits (PIC), control electronics, thermal/vibration management structures, electronic/optical interfaces, and an enclosure follow similar assembly procedures. Each PIC requires thermal stabilization and an electrical interface to its host printed circuit board (PCB). Both of these restrictions can be met by affixing the PIC on a submount/heat spreader which itself sits on a thermo-electric cooler. A thermistor is embedded in the heat-spreader or contained in the PIC to allow temperature sensing and feedback. A “carrier PCB” is also affixed to the top side of the heat-spreader, next to the PIC, and an array of pads is wirebonded directly to the corresponding pads on the PIC. This assembly sits on a heat sink to dissipate the heat generated by the thermo-electric cooler (TEC). Finally, an underside mezzanine connector forms the connection between carrier PCB and host PCB and allows for control signals between the two.

Each PIC is die bonded to its heat spreader using thermally conductive adhesive. Next, the carrier PCB is affixed and wirebonded. After electrical testing, the PIC assembly is optically packaged. A V-groove fiber array is affixed to the edge of the PIC using optical-grade epoxy. During and after packaging, optical transmission is measured (and optimized) using pairs of input and output waveguides connected by a short loop of fiber. Fiber strain relief can take several forms in the different modules, but in general a rigid structure affixed to the heat spreader serves as a support for the fibers glued to the chip and protects the fragile fiber-PIC interface.

The design of the 19-inch rack-mounted enclosures for the full system also follows a standardized template to accommodate the common packaging requirements of most modules. This enclosure prioritizes efficient cooling, vibration management, and accessibility while maintaining flexibility for module-specific requirements. The distinguishing features of this design compared to typical 19-inch enclosures are the inclusion of a mid-plane divider, optical breadboard, and connector plate. The divider increases overall stiffness and separates most electronics from the optical components, allowing airflow to cool the electronics while minimizing turbulent air affecting the fibers. An optical breadboard further minimizes vibration transmission from the chassis to the fibers using viscoelastic polymer standoffs, and the connector plate extends this mechanical isolation technique to fibers passing through

the front panel.

B. Laser system

The laser system is responsible for generating the two pump lasers (P1&P2), the local oscillator (LO) and probe lasers (probe and Ref). P1&P2 serve as pumps for generating degenerate single-mode squeezed light pulses via dual-pump spontaneous four-wave mixing, and the LO is used to perform homodyne detection of generated quantum light at the QPU array. The probe laser is used to stabilize the on-chip resonators to a common frequency reference. The laser system also delivers all of the reference lasers (Ref1 and Ref2, which are themselves derived from the Ref laser) for circuit calibration, initialisation, and stabilization between modules.

The four main stages of the laser system are the frequency stabilization stage, the modulation stage, the pulse interferometer lock stage, and the distribution stage. The frequency stabilization stage begins with five lasers: P1, P2, LO, probe and Ref. A small fraction of each laser is tapped off and interfered with a frequency reference. The frequency of each laser is thus stabilized using an optical phase-lock loop (OPLL), which removes fast phase noise between the lasers and the frequency references while stabilizing their frequencies to the offsets required by the microring resonators as depicted in Fig. S25. The main component at the heart of this stage is the *comb module*, which generates the frequency reference to stabilize all the lasers. The comb is derived from the LO laser. A small fraction of this laser is tapped off (10 mW) and passed through a strongly driven electro-optic phase modulator (10 W, 28.55 GHz tone). The RF tone is generated with a programmable synthesizer, synced to the main experimental clock, and amplified up to 40 dBm by a balanced Ka-Band power amplifier. The modulated light is then split six ways for the various OPLLs and interferometer locks resulting in ~ 0.8 mW of optical comb power per channel. The comb module is controlled by a custom electronic board containing the microcontroller in charge of communicating with the comb laser and the synthesizer via serial links, biasing the amplifier to its operating point and communicating with the main server computer.

Our custom phase/frequency detector (PFD) beatnote lock is optimized within a capture range from 50 MHz up to 2 GHz. The basic operation of the PFD is to provide an error voltage for control by comparing the feedback signal to a set reference signal. The RF tone in the frequency comb is chosen such that the beat notes of pumps (P1 & P2), Ref and probe all lie within the capture range of the PFD. Furthermore, the pumps, LO, and probe are required to line up with their respective microring resonances (see Fig. S25) while the Ref is required to partially transmit through the GBS chip input and output AMZI, whose free spectral range (FSR) FSR is 400 GHz, also depicted on top of Fig. S25. A 28.55 GHz comb spacing allows the probe, Ref and P1&P2 lasers to be locked at different beatnote offsets such as ~ 100 , ~ -100 MHz and $\sim \pm 500$ MHz, respectively. To ensure that P1&P2 are stabilized symmetrically about the LO, the beatnote of P2 is stabilized using the P1 beatnote as the reference signal.

The pulse generation stage independently modulates each CW laser output to obtain a desired temporal mode shape and amplifies them to appropriate optical powers. For P1 and P2, a fiber-based amplitude modulator is used and driven at 1 MHz with a train of 1 ns long square pulses. The LO is modulated using an IQ modulator to guarantee the LO temporal mode is optimally matched to the complex waveform of squeezed light exiting the squeezers. The probe laser is first phase modulated at 1.91 GHz to generate the sidebands necessary for Pound-Drever-Hall locking of the on-chip resonators [83] and then carved into a 50 ns square pulse using a fiber amplitude modulator. The Ref laser is first split into four paths,

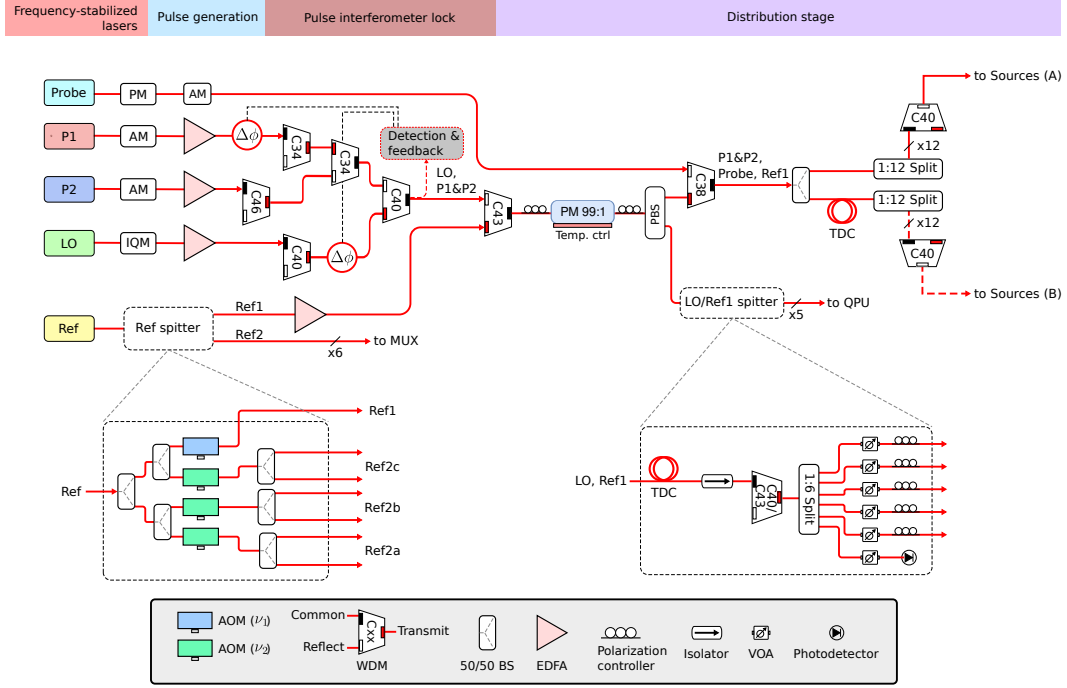


Figure S24. **Laser system.** The laser system consists of four main stages and contains five different laser modules: probe, P1, P2, LO and Ref. The frequency stabilization stage is responsible for stabilizing the frequency of each laser using an optical phase-lock loop. The pulse generation stage independently modulates each CW laser output, using an amplitude modulator (AM), phase modulator (PM) or IQ modulator (IQM) and amplifies them using erbium-doped fiber amplifiers (EDFAs). The third stage is the pulse interferometer lock, responsible for stabilizing the slow fiber phase drifts between the three arms of the pulse generation stage, allowing P1, P2 and LO to have a well defined phase reference on the output fiber. This stage contains a network of Wavelength Division Multiplexing (WDM) devices, optimized for different frequency channels (C34, C46 and C40) as well as a tap-off channel sent to a detection and feedback module for phase locking. The final stage is the distribution stage, responsible for splitting P1&P2 from the LO while simultaneously combining P1&P2 and probe with Ref1 (towards sources) and splitting Ref1 equally between the two output paths in a phase stable manner. Ref splitter is an independent module, containing beamsplitters and acousto-optic modulators (AOMs), responsible for generating the sequence of reference pulses for system-level phase stabilization. The LO/Ref1 splitter is the module responsible for dividing the LO and Ref1 power into five parts towards the QPU array, individually controlled using a variable optical attenuator (VOA) and polarization controller.

and passed through four different acousto-optic modulators (AOMs). The AOMs carve the four Ref beams (Ref1, Ref2a, Ref2b, Ref2c) into their respective 400 ns pulses and shift their frequencies to have a 6.3 MHz beatnote between Ref1 and the Ref2's, which is used later for lock-in detection of the optical phase. The P1&P2, LO, and Ref1 pulses are then amplified using erbium-doped fiber amplifiers (EDFAs). The probe laser is not amplified as it has sufficient optical power.

The third stage is the pulse interferometer lock, responsible for stabilizing the slow fiber phase drifts between the three arms of the pulse generation stage, allowing P1&P2 and LO to have a well defined phase reference once coupled into the output polarization-maintaining

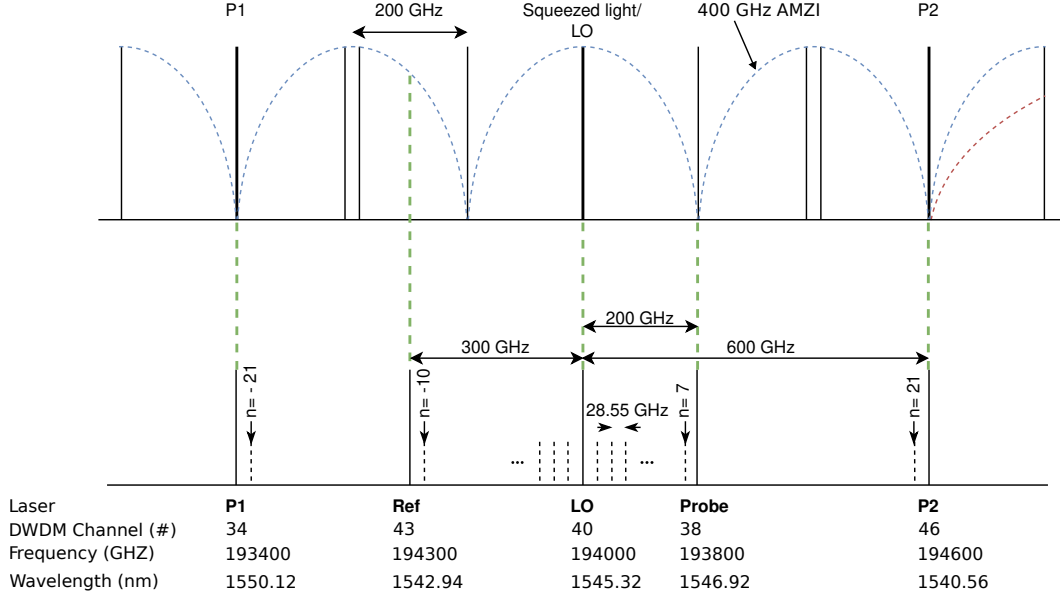


Figure S25. **top) Coupled microring resonances.** Squeezed light is generated at frequency f_0 by absorbing one photon from P1 (f_1) and one photon from P2 (f_2) such that $f_0 = (f_1 + f_2)/2$. The primary ring free-spectral range (FSR) is 200 GHz while the output AMZI filter has a 400 GHz FSR. **bot) Comb and laser lines.** The LO laser is required for phase sensitive measurement on the quantum states and is frequency and phase stabilized with respect to P1&P2. The LO laser is phase modulated such that a frequency comb with 28.55 GHz spacing is used to transfer stability between LO, P1&P2: n is the integer number of comb spacing from the carrier frequency. At the bottom of the spectrum is shown the ITU G.694.1 standard DWDM channel, frequency and wavelength for each of the laser lines.

(PM) fiber going to the next stage. This stabilization is achieved by combining the three beams onto a single fiber using a network of dense wavelength-division multiplexing modules (DWDMs), which also filter out amplified spontaneous emissions from the EDFAs, and tapping off a small amount of the optical power. This tap is interfered with the optical frequency comb and feedback on the phase of the three beams is performed using a combination of fiber stretcher and temperature controlled fiber spool as actuators. Together with the frequency stabilization stage, the robustness of this lock ensures the persistent synthesis of spatio-temporal cluster states in the experiment.

The final stage in the laser system is the distribution stage. It is necessary to split P1&P2 from the LO while simultaneously combining all three beams, including the probe laser, with the Ref laser and splitting that Ref laser equally between the two output paths in a phase stable manner. This frequency-dependent beamsplitting operation allows the optical phase information to be henceforth carried by the Ref1 beam, which will act as a phase reference for the rest of the system. This is all achieved with a custom built asymmetric polarization-dependent interferometer where the 1.2 THz FSR of the temperature stabilized interferometer is set by the length of the polarization-maintaining component (~ 60 cm of PANDA PM fiber). After exiting the frequency-dependent beamsplitter, the first channel containing the LO and Ref1 is directed towards another module responsible for splitting the LO&Ref1 in five equal parts (see Fig. S24) for phase sensitive measurements and inter-

module phase stabilization at the QPU array. The probe beam is injected into the second channel of the network containing P1&P2 and Ref1 using a DWDM. This channel is then split into 24 paths, 12 of which pass through a delay line matching precisely the inverse repetition rate of the experiment ($1\ \mu\text{s}$) according to the phase lock scheme explained in Sec. IX B. 21 of these fibers are sent onwards to the sources array, while three remain for monitoring purposes and as spares.

C. Sources

Probabilistic non-Gaussian and deterministic squeezed light states originate from the sources subsystem. At the heart of this subsystem is a set of Gaussian Boson Sampling (GBS) modules, hosting photonic integrated circuits (PIC) responsible for quantum state preparation. On each chip, a two-mode unit cell consists of a pair of squeezers in two separate spatial modes combined using a variable beamsplitter (parameters described in Sec. VII A). There are two such cells on each chip. The input to a GBS module is the output channel (A or B) from the laser system, described in the previous section, containing P1&P2, probe and Ref1 (without or with extra delay).

1. GBS modules

The GBS chips, depicted in Fig. S26a, were fabricated on 200mm wafers using a photolithographic process offered by Ligentec SA on a production line at X-Fab Silicon Foundries SE, and are based on silicon nitride waveguides and thermo-optic phase shifters. Each PIC contains components for programmable attenuation, filtering and coherent distribution of pump light in four spatial modes towards four separate squeezers. The input attenuator is an integrated MZI with a tunability range of approximately 20 dB. Its reject port contains an integrated polarizer, aligned to the TE mode of the optical waveguides, that is used to align the input polarization state of the pump. Between the attenuator and squeezers, the input filter is an asymmetric MZI with an FSR of 400 GHz and extinction of the light at squeezing frequency of more than 20 dB. This filter is used to reject any incoming light at the squeezing frequency that could have been erroneously generated at a previous stage. The pump is then distributed on-chip using a balanced 1x4 splitter.

Each GBS chip contains four squeezers responsible for generating quantum light in a nearly single temporal mode squeezed state. The squeezer geometry is based on two coupled resonators [84]: a primary resonator of FSR 200 GHz, which is strongly coupled to an auxiliary resonator that has an FSR of $3/4 \times 200$ GHz. The two pump lasers (P1&P2) are aligned to two resonances separated by 1.2 THz and through the process of strongly pumped spontaneous four-wave mixing, a squeezed state is generated in the central resonance (see Fig. S25). The auxiliary resonator is used to suppress spurious single-pump nonlinear processes that lead to contamination of the squeezed state [84]. The waveguide cross-section of the rings is 2000 nm x 800 nm (width x height).

Following the squeezers are asymmetric Mach-Zehnder interferometer (AMZI) filters to separate pump light from generated squeezed light. The 400 GHz FSR of each AMZI is designed such that pump light is rejected out of the chip while squeezed light is transmitted towards programmable linear-optical interferometers. These two-mode unitary transformations implement the interaction between two single-mode squeezed states. The interferometer can be programmed such that the output of the chip is either a Gaussian state

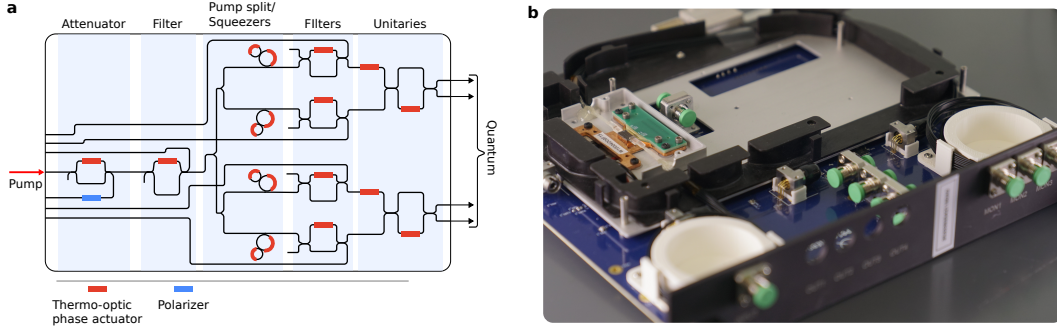


Figure S26. **a. Simplified schematic of the GBS PIC layout.** The on-chip submodules consist of a variable attenuator for the pump lasers, a filter to reject any light in the squeezed light frequency band and a 1x4 pump distribution network (pump split) before the squeezers. After the array of squeezers is pump filtering followed by programmable linear optical transformation to generate and output two-mode or single-mode squeezed states, fiber-coupled off the chip. The fiber optical input and output arrays are not shown. **b. Photograph of an assembled GBS module.** The PIC is uncovered and visible atop the copper mount in the left side of the image.

optimized for heralding a squeezed cat state (provided appropriate post-selection based on photon number outcomes), or a pair of independent squeezed states.

Light is edge coupled into and out of the chip using V-groove arrays of ultra-high numerical aperture (UHNA) single mode fibers. From measurements using loop-back waveguides placed on both sides of the chip, the average coupling loss is 0.91 dB (81% coupling efficiency). On the path towards PNR detectors, a low-loss fiber-based DWDM filter is inserted with more than 100 dB of pump and residual light rejection around CH40 (1545.32 nm), and an average insertion loss of 0.34dB (92% transmittance).

From 520 nominally identical sources PIC dies fabricated, the linear optical properties of each chip were extracted using an automatic PIC measurement system. From these measurements, the performance of each component on the chip can be determined. Chips were chosen based on several key parameters: the splitting ratio of the directional couplers (which are designed to be 50:50) and the FSR, loaded quality factor and escape efficiency of the primary ring of the squeezers. From the chips measured, 50 of the best were chosen for optical and electrical packaging. From these 50, 24 were chosen to furnish the sources subsystem. Finally, from these 24, a “master chip” was chosen as being most representative of the average, for the purposes of laser system tuning.

Each selected chip is installed in a *GBS module*: a custom PCB that contains all the necessary electronics and optics for characterisation, operation, and monitoring of the chip. A sources assembly contains up to eight such GBS modules, which are mounted in a eurocard-style chassis. The sources subsystem is then composed of three of these assemblies, for a total of 21 sources modules in the system. Each GBS module houses the packaged GBS PIC, the driver electronics for programming the on-chip thermo-optic phase shifters, several optical detectors used for locking on-chip resonators and monitoring the incoming polarization of the pump, an electrical fiber-based polarization controller and its associated driving electronics, a temperature controller for stabilizing the temperature of the packaged chip, and a micro-controller for orchestrating all of these tasks.

The sources assembly control system is composed of a host server class computer networked with each of sub-modules via a Controller Area Network (CAN) using the CANOpen protocol. The host computer performs high level control functions like calibration, persis-

tence and gathering/publishing of metrics using python to provide a user interface. Each module has a micro-controller using a real-time operating system to perform closed loop control of the squeezer primary ring heaters using feedback from the laser probe signal as well as interfaces to the TEC, polarization control, and board health monitoring peripherals. While operational the host computer sets, from persistent calibration values, chip heaters to the desired state and enables locking of the ring resonances to the pump lasers. It then monitors the performance, offloading the real-time feedback to each module's micro-controller.

Characterization procedure of each chip

Each GBS chip is fully characterized prior to assembly and integration into the final system. A replica of the laser system is used with lasers P1, P2, probe and a Ref injected into the input port of the chip under test. The temporal width of each pump pulse is 1 ns, while the Ref and probe laser pulse widths are 20 ns and 50 ns, respectively. The Ref and probe lasers are temporally multiplexed with the two pump lasers (P1&P2), to avoid unwanted nonlinear effects in fibers and waveguides.

The first step is to calibrate the input attenuator (refer to Fig. S26). Using the probe light, the heater voltage is scanned and the signal monitored downstream at the “probe detector” (not shown), measuring light in the reject port of the on-chip pump-drop filters. This measurement results in a curve of attenuation level (in dB) vs heater voltage (V). The input filter is calibrated in a similar fashion by scanning the heater voltage of the input filter and monitoring the probe detector using the probe laser. Note that this calibration can be fine tuned in the final system with the use of PNR detectors. Finally, using the same probe laser and detector, the output filter is calibrated by scanning its heater voltage. The results are logged into a database for later use.

The rings and interferometers are calibrated iteratively due to a small amount of thermal cross-talk. First, the primary resonator is calibrated by finding the resonance peak while scanning its heater voltage and monitoring the signal at the probe detector. Prior to aligning the auxiliary ring, the internal phase of the interferometer is calibrated. While the primary resonator is in a locked state at the resonance condition, the internal phase heater voltage is scanned and the mean photon number is monitored using a photon-number resolving detector. These calibration steps expose the conditions for the interferometer to be set to *bar* (max), *cross* (min) and 50:50 and logged into our database. With the internal phase set to 50:50, the heater voltage of the auxiliary ring is scanned and optimized by monitoring the mean photon number with PNR detectors.

Next, the FSR of the primary ring is measured again in this high-precision system and the external phase of the interferometer calibrated. The FSR of each squeezer depends on the auxiliary ring position and is set to its optimal position previously extracted. To determine the FSR, both pump lasers are operated without amplitude modulation enabled and their frequencies are independently swept. The ring transmission is detected while sweeping the pump frequency. Each trace corresponding to the transmitted power (P1 and P2) versus frequency is fitted to a Lorentzian function and FSR is derived from the distance between the two Lorentzian peaks. The range of FSR across all chips is 199.69 ± 0.07 GHz.

Finally, two-mode squeezing is implemented while the squeezers are locked and the voltage of the external phase heater of the unitary is scanned. Using a pair of PNR detectors and probing light at the squeezing frequency (CH40), the degree of second order correlation ($g^{(2)}$) is monitored [85] as the external phase is scanned. Data corresponding to $g^{(2)}$ vs heater voltage (power) is logged into the database for programming the unitary once deployed into the final system. Finally, one proceeds to a second and final calibration routine of the input and output filters, internal and external phases of the interferometer and the auxiliary ring

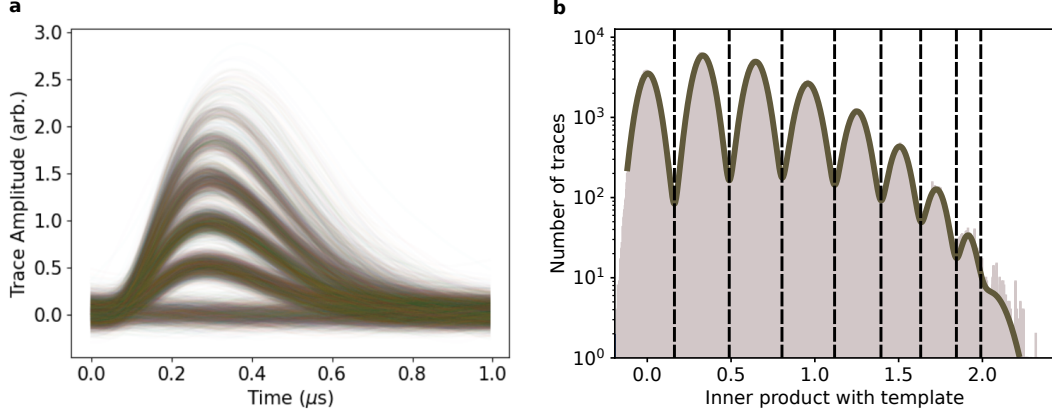


Figure S27. **a. Overlaid plot of 10,000 photon traces, measured with a typical PNR module.** Incident light is a pulsed coherent state with $\bar{n} = 1.71$, repetition rate of 1 MHz, and with an acquisition sampling rate of 125 MHz. Y-axis is the proportional voltage signal from a photon-detection event. **b. Histogram of the inner products of the traces with the principal component of the traces.** These form peaks which we identify as photon number events. A sum of Gaussian curves is fit to the histogram, and the intersections between adjacent Gaussians determine the boundaries for digitizing each trace/inner product as a distinct photon number.

position.

2. Photon number resolving system

PNR detectors provide the non-Gaussian element necessary for heralding squeezed cat states in our present demonstration and GKP states in our FTQC architecture. To implement PNR photon counting, we employ TES detectors. These rely on the dynamic phase transition from the superconducting to the normal state of a thin film. For photon number resolving detection, the device includes $20 \times 20 \mu\text{m}$ square of tungsten (W) film, niobium leads and gold (Au) heat sinks [86, 87]. The detector is embedded in an optical stack, with anti-reflection coating on top and a dielectric spacer and distributed Bragg reflector (DBR) optimized for 1550 nm underneath [88]. The substrate is cooled well below the film's superconducting transition temperature to an approximate 12 mK base temperature. When the detector is voltage biased, the Joule power dissipated in the W provides a negative feedback effect to self regulate the temperature of the device. A shunt resistor is placed in parallel with the TES and the input coil of a DC-SQUID array. When photons are absorbed by the detector the W electron temperature increases with a corresponding increase in the resistance, changing the current flow at the SQUID array input coil. The transimpedance gain of the array boosts the TES signal which is then further amplified by ultra-low noise room temperature electronics.

For the measurements reported in the main text, the output of the second stage amplification is digitized at 125 MHz giving 125 points per pulse, of which only the first 60 were used due to latency requirements, from which the photon number is extracted. In particular, for each channel, photon number analysis is calibrated by first constructing a histogram of trace overlaps with an appropriate weighting function $w[t]$. A sum of Gaussians is fit to this histogram, and points of intersection between Gaussians S_n [89] are determined. Both

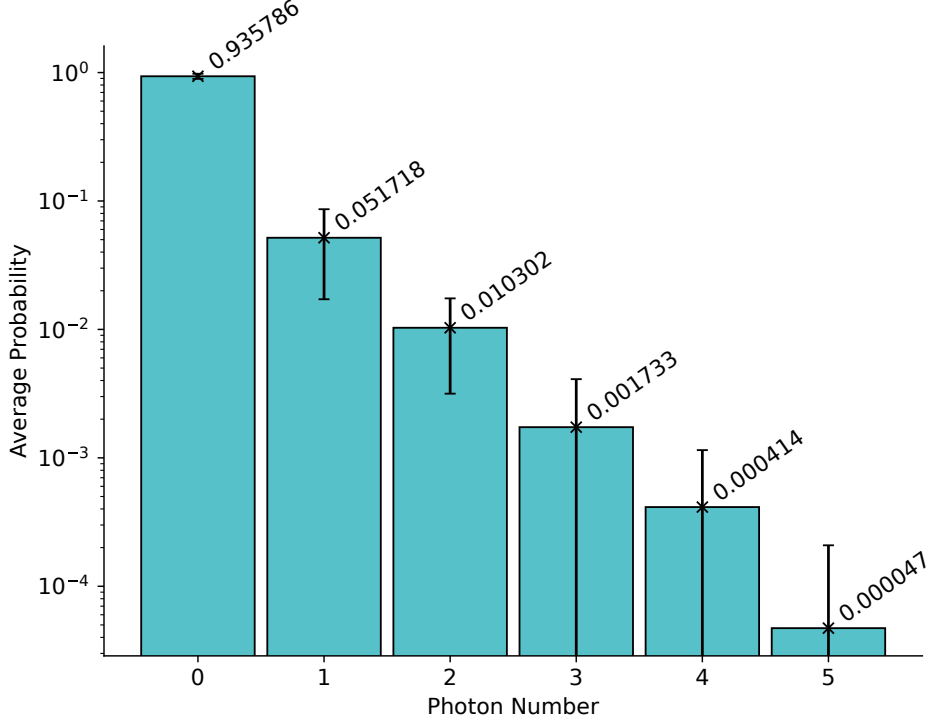


Figure S28. **Averaged photon number statistics from 36 two-mode GBS cells.** For this analysis, each of the 36 PNR detectors measures the output of their respective GBS cell. Histograms are constructed over five million clock cycles (5 seconds) and we show the average probability of events per photon number bin. Error bars correspond to the standard deviation over the 36 probabilities of each photon number bin.

$w[t]$ and the S_n are stored on an FPGA. Subsequent incoming traces are assigned a photon number n in real time by calculating $s_i = \sum_{t=0}^{59} v_i[t]w[t]$ as well as determining the points of intersection for which $S_n < s_i \leq S_{n+1}$ directly on an FPGA. Fig. S27 shows an example of typical photon traces and histogram.

3. Non-Gaussian state generation

In this section, we expand on the non-Gaussian state synthesis capabilities (low-quality GKP states) of Aurora, by performing additional measurements.

First, we characterized the photon counting statistics of our combined GBS-PNR system. We directly measured photon numbers at the output of the 36 two-mode GBS cells and plotted the average over all PNR detectors, for each photon number event, shown in Fig. S28. The total number of clock cycles for this measurement was five million. The heralding probability for photon number event $n=2$ is thus approximately 1% on average.

In order to characterize the non-Gaussian states generated from sources, we performed

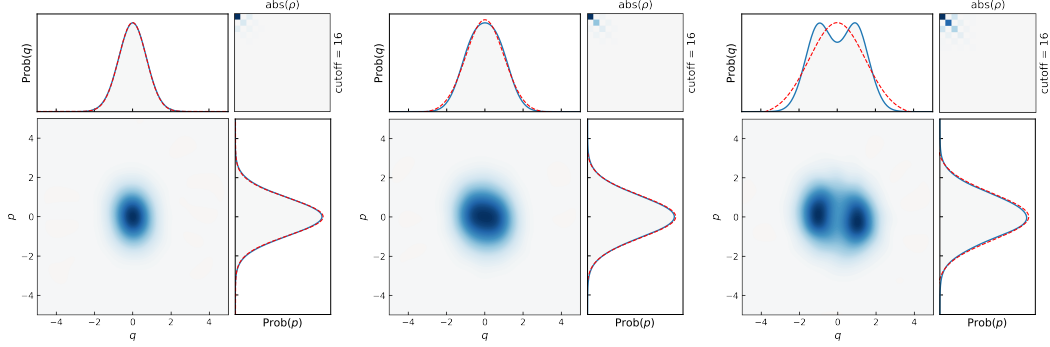


Figure S29. **Experimentally reconstructed Wigner functions from a GBS device.** Using quadrature measurements and photon numbers, Wigner functions for different photon number (0=left, 2=center and 5=right) heralding events are reconstructed using a maximum likelihood algorithm. To help visualize the non-Gaussianity of the experimental result, the Wigner function is fitted to a bivariate normal distribution. This distribution, integrated over the q - and p -quadratures (shown with red dashed lines) are compared with the non-Gaussian quadrature probability distributions. The upper right panel of each plot corresponds to the density matrix of the state, in the Fock basis, where the cutoff is the maximum photon number considered in the maximum-likelihood state reconstruction algorithm.

a set of homodyne measurements with a representative GBS chip and the characterization setup described above. The unitary of the GBS device was set to generate squeezed cat states. We were able to reconstruct Wigner functions with photon number heralding events from $n = 0$ up to $n = 5$. Quadrature and PNR acquisitions are synchronised using the appropriate triggering procedure.

Quantum states (density matrices and Wigner functions) can be reconstructed from quadrature values (homodyne measurements) at different angles. Here we follow a maximum-likelihood reconstruction method [90]. The measurement angle is swept from 0° to 180° , in steps of 5° . For each angle, synchronized quadrature values and photon numbers are saved, processed and fed into the maximum-likelihood state reconstruction algorithm. Based on the post-selected photon number, we get the Wigner functions as shown in Fig. S29, here showing the results for $n = 0$, $n = 2$ and $n = 5$. For each photon number and measurement angle, there are at least 20,000 data points acquired.

D. Refinery (MUX)

In our FTQC architecture, the role of the refinery is to convert raw output states from GBS devices into high-quality and high-probability GKP Bell pairs. For our experimental demonstration, the functionality of the refinery is limited to two important stages: adaptive multiplexing (MUX) and pair creation. The MUX submodule boosts the probability of generating pairs of squeezed cat states by adaptively rerouting, at each clock cycle, successful states from GBS devices heralded by PNR detectors. At the output of the MUX is the pair creation stage, responsible for the synthesis of two-mode entangled states.

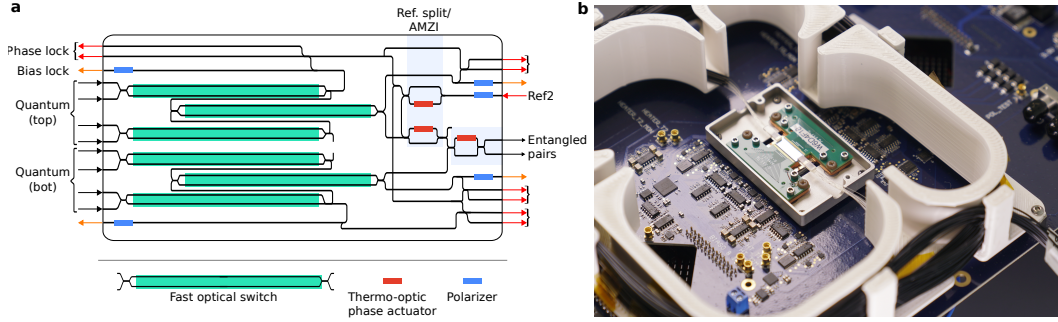


Figure S30. **a. Simplified schematic of the refinery (MUX) integrated circuit layout.** The refinery implements 8×2 fast switching for entangled pair generation, based on photon number outcomes. The chip also integrates a network of beamsplitters and thermo-optic phase actuators for active input phase and optical switch stabilization, using reference pulses interleaved with quantum inputs and Ref2. **b. Photograph of an assembled refinery module.** The packaged PIC is visible in the centre of the image.

1. MUX

In this section, we describe the MUX photonic chip and module as depicted in Fig. S30. Each MUX contains two $4:1$ binary trees of electro-optic switches. In total, eight outputs from GBS devices feed into a single MUX and are divided between the two sets of switches: *top* and *bottom* (*bot*) set. Each switch is an adaptive beamsplitter, $BS(\phi)$, with two possible settings: *cross* ($\phi = \pi$) or *bar* ($\phi = 0$). At each clock cycle, one mode from *top* and one mode from *bot* is dynamically routed to the pair creation sub-module, dependent on the outcomes recorded by the PNR detectors. The outputs of the two switch trees on each chip are sent to an MZI which, for the duration of the experiment, is tuned to a reflectivity of 50% to implement Bell pair synthesis.

As shown in Fig. S30, the MUX chip also has an input for light responsible for phase locks (Ref2). A BS (ref. split) controls the fraction of Ref2 that is used for phase locking. The remainder is combined with the output entangled pair using an AMZI (~ 600 GHz FSR). For simplicity here, details around active polarization and phase stabilization have been left out and will be treated in Sec. IX B. All inputs and outputs to the chip are enabled by edge couplers using arrays of single-mode fibers. On average per facet, the coupling loss is approximately 1.37 dB, across all MUX chips within Aurora.

The MUX also contains several components that are used for bias locking the electro-optic switches and phase locking the incoming light. One of the switches from the first layer of both *top* and *bot* has its unused output port routed to a static 50:50 beamsplitter. One of the outputs of this beamsplitter is sent off the chip to a photodiode, and the detected signal is used to lock the bias voltage of the switch and align the input polarization of the two optical modes associated with that switch. The other output of the beamsplitter is coherently combined with Ref2 on another 50:50 beamsplitter and the interference detected via an off-chip balanced detector. This signal is used in the phase locking procedure, as described in Sec. IX B. The second layer of both *top* and *bot* has similar structures for bias and phase locking, the only difference being that these structures are used to phase and bias lock a set of two cascaded switches and their respective modes. This is accomplished by interleaving the optical pulses used for feeding back on the bias, phase, and polarization of each switch and mode.

The MUX module is based on a thin-film lithium niobate (TFLN) on insulator platform offered by HyperLight Corp. and manufactured on a production semiconductor fabrication line. TFLN was selected for its large electro-optic response and low loss; it also enjoys large intrinsic bandwidth and exceptional signal integrity [91]. Due to the Pockels effect, it is possible to realize fast electro-optic modulation (EOM), which, when implemented inside an MZI, can lead to efficient amplitude modulations of light at the output of the MZI. Each MZI is designed as a balanced interferometer with two 50:50 fabrication-tolerant directional couplers and two optical paths. The EOM is implemented in a push-pull configuration, causing a phase delay on one arm and a phase advance on the other when an electric field is applied.

Given the sequence of pulses in the experiment, a fast rising time (in the range of tens of ns) is required. This translates into an analog bandwidth requirement of approximately 100-200 MHz, which was confirmed for all chips, including the response of any driving electronics used. The voltage required to completely switch the output between “bright” and “dark” is defined as the half-wave voltage (V_π). We measure V_π of ~ 6.35 V from a 5.2 mm long MZI modulator with more than 30 dB extinction ratio across all MUX chips.

The packaged chip is bonded to a custom “MUX driver board”, shown in Fig. S30b. The MUX driver board is responsible for several tasks: accepting and processing PNR detector data and programming the MUX switches accordingly, monitoring the bias state of each switch and the phase of all incoming light, programming the thermo-optic phase shifters on the chip, stabilizing the base temperature of the chip, and controlling the incoming polarization of the reference light used for phase locking. Two FPGAs are responsible for orchestrating all of these tasks on a variety of timescales. Commercially available small form-factor pluggable (SFP) optical fiber transceiver modules interface communications between MUX FPGAs and data from PNR detectors and towards both QPU backend and buffer assembly while ensuring the low latency required for these tasks. Other tasks like programming of the thermo-optic phase shifters and monitoring of the system’s status are performed over Ethernet.

E. Quantum processing unit

The array of five photonic quantum processing units (QPUs) in the system implements the final spatio-temporal stitch between entangled pairs generated in the previous stage, to form the cluster state (described in Sec. VII B 2), and also performs homodyne measurements on all 12 operating modes on every clock cycle. Equivalently, each QPU chip implements a multi-mode GHZ measurement.

Each QPU photonic chip is packaged according to the common process described in Sec. VIII A. Due to the sensitivity of the homodyne detectors to parasitic capacitance, their transimpedance amplifier (TIA) circuitry is placed on the carrier PCB, as close as possible to the bondpads. This carrier board comprises three high gain (475 k Ω) low noise 15 MHz TIAs for balanced homodyne detection, one low gain (5 k Ω) TIA for monitoring the input LO, filtering for photodiode bias voltage, heater drive current and TIA supply voltages. The board is enclosed by a Faraday shield to reduce electromagnetic interference.

Spatio-temporal entanglement in the cluster state is performed by two on-chip variable beamsplitters implemented with MZIs. These MZIs are tuned using thermo-optic phase shifters, which are also used for measurement basis tuning and balancing homodyne beamsplitters. Modulation of the LO phase of one homodyne detector (see Fig. S31) is performed with a silicon PN carrier depletion phase shifter for fast feedforward operations.

For the experiment described in the main text, the local oscillator pulse is in fact a pair of

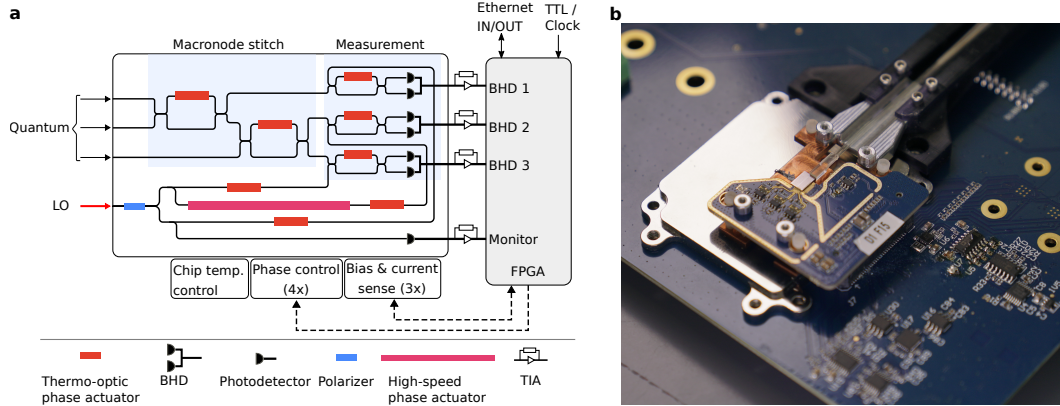


Figure S31. **a. Simplified schematic of the QPU integrated circuit layout and readout.** The chip has three quantum inputs, followed by an interferometer consisting of two adjustable MZIs with thermal phase control. Three balanced homodyne detectors are used for quadrature measurements. Each consists of an adjustable MZI, fixed to a 50:50 splitting ratio, and a pair of photodiodes followed by an off-chip transimpedance amplifier (TIA) and an analog-to-digital converter unit. The local oscillator input (LO) is first transmitted through an on-chip polarizer and split into four spatial modes. One mode is monitored using a single integrated photoreceiver for polarization management while each of the three other modes are directed towards the homodyne detectors. Bases for quadrature measurements are set by thermal phase shifters (slow) and one fast phase modulator for feedforward operations. The chip temperature is controlled and stabilized to within 1 mK/24hrs. **b. Photograph of the assembled QPU module.** Electro-mechanical assembly of the QPU chip showing the chip submount, carrier board and mechanical strain relief for the fiber array. The chip is wirebonded to the carrier board containing different electrical filtering and the TIA circuitry for quantum measurements and LO monitoring.

pulses, per clock cycle. The first pulse performs homodyne detection of the quantum input state, while the second serves as a reference. The reference allows for a continuous shot-noise normalisation of the quadrature values of the quantum state whilst also measuring and correcting drift in the splitting ratio of the homodyne beamsplitter. On each clock cycle, quadrature values are obtained from the TIA outputs with a weighted integration over the first LO pulse. This integration is performed locally, in real-time on the module FPGA controller, normalised and forwarded to the QPU backend.

A qualification and calibration routine is run on each packaged QPU PIC prior to assembly into modules and again before integration into the full system. During this routine, interference fringes for all phase shifters are collected and fit with calibration parameters. In the case of the thermo-optic phase shifters, fringes are fit with an equation of the form $a \sin(bx^2 + cx^4 + d) + e$, where x is the current applied to the phase shifter. For the fast LO modulator, fringes are fit with a cubic phase-voltage equation to account for non-linearity in the predominantly linear relationship.

The overall insertion loss of the quantum input channels—including the quantum efficiency of the photodiodes—is measured using the TIA output voltage for a known input power, with all MZIs set to 100% transmissivity. The overall transmittance of the QPU is detailed in the main manuscript and summarized in table SVIII, and amounts to 4.4dB of effective loss, which includes the finite shot noise clearance of the homodyne detectors. External to the PIC, there are two sets of fibre-optic components on each quantum input—a DWDM add-drop filter to filter residual pump light from the Sources modules, and an opti-

cal isolator to suppress LO back-reflections inside the PIC. The losses of these components are included in the total given in table SVIII.

1. QPU backend

The QPU backend subsystem receives data from each QPU frontend and refinery over bidirectional 900 Mbps 8b10b encoded fiber-optic serial links. These serial links connect Zynq 7010 or 7020 FPGAs inside the refinery and QPU frontend modules to one Zynq Ultrascale+ ZU19EG at the heart of the QPU backend.

From the QPU array, the backend receives homodyne measurement data (up to three 16-bit signed integers) and also interleaved error signals (up to three 16-bit signed integers) for optical phase stabilization. The QPU backend immediately forwards the error signals to the routing assemblies, housing the stabilized fiber delay lines (SFDL) at the QPU inputs.

From the refinery array, the QPU backend receives information on the type of pairs generated at the output of each MUX and input to the QPU array. This information is encoded in a single bit per pair input state, depending on if this input is a heralded squeezed cat state or a squeezed state.

Due to the modest speed of both Zynq 7010 and 7020 FPGAs, our implementation is limited to a 900 Mbps serial link using the LVDS 4x asynchronous sampling method documented in this application note [92]. The ZU19 FPGA uses standard GTY transceivers and the corresponding gtwizard ultrascale IP block. The link between FPGAs is verified by inspecting 8b10b codes and disparity errors, along with sending PRBS31 8b10b encoded data over the link.

Upon arrival of all QPU and MUX data from a single experimental clock cycle at the QPU backend, the latter performs a computation described in Sec. VIID, to select the next measurement basis on all QPU modules, i.e. one of two possible LO phase angles fed forward to and implemented by the QPU high-speed phase actuators. The measurement angle instructions are sent to each QPU module over the same bidirectional fiber-optic serial links that were used in the opposite direction to transmit homodyne measurement data. All operations are performed in time to actuate the LO phase angles for the next clock cycle of the experiment.

a. Decoding implementation As discussed in Sec. VIIB3, decoding occurs in two stages. In the first or inner decoding stage, the raw homodyne values and heralding information is used to compute error probabilities and bit values. Error probabilities are computed for M_0 , M_3 , and M_4 in Fig. S18; other nodes are not needed for the computation of the syndrome. To compute these values quickly on the FPGA, lookup tables are used for the exponential functions involved. In the second outer decoding stage, a min-sum Belief Propagation algorithm [75] is used to find the recovery which corrects the syndrome. The algorithm is constrained to only run up to a maximum number of 5 iterations. If the recovery is found, the soft information is used to determine the choice of X or Z measurement on the next clock cycle, else a Z measurement is performed in the next clock cycle (see Sec. VIID for more details). At an FPGA clock speed of 250 MHz, the entire decoder, from arrival of the homodyne and heralding data to the issuing of the LO phase selection for the next clock cycle, takes at most 64 ns to run (the remaining latency budget is consumed mostly by the round-trip latency of the serial links).

To provide evidence that the local oscillator phase angle is being adaptively chosen using feedforward from the decoder and correctly applied to the phase shifter, we show data in Fig. S32 overlaying the recorded experimental phase angle settings (X or Z) against the

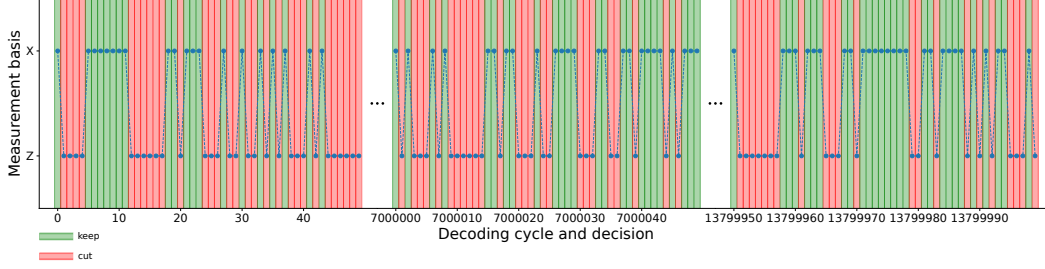


Figure S32. **Adaptivity of the local oscillator phase angle from the decoder's feedforward.** The recorded phase angle settings (measurement basis X or Z) are overlaid against data (green and red bands) from the Python-based decoding simulation. The first 50, mid 50 and last 50 decoding cycles of the full experiment containing a total of 13.8M adaptive measurements are shown.

keep-or-cut decisions from a Python-based decoding simulation. This simulation uses the same algorithm, raw homodyne values and state records as in the FPGA-based adaptive demonstration plotted in Fig. 3c. For simplicity, we only show the first 50, mid 50 and last 50 decoding cycles of the full experiment, which contains a total of 13.8M adaptive measurements. The occurrence for bit 0 (cut) and bit 1 (keep) is 51.206% and 48.794% respectively. The simulated decision bits and the LO phase settings read out during the experiment agree 100% of the time, across all 13.8M adaptive measurements.

F. Optical interconnects and feedback control

All modules are optically interconnected using low-loss single mode telecommunication fibres. While networking between the laser system and sources, between sources and PNRs, and between the laser system and QPU is relatively simple, interconnections between sources and refinery, and between refinery and QPU require fiber delay lines involving sophisticated stabilization techniques.

1. Stabilized fiber delay lines (SFDL)

An array of (stabilized) fiber delay lines (SFDLs) is inserted between sources and refinery and is referred to as the buffer subsystem. A similar array is present between the refinery and QPU subsystems, referred to as the routing subsystem. Each SFDL, sketched in Fig. S33a, contains three main submodules: i) a time-delay coil (TDC) with a slow thermo-optic phase control (TEC), ii) a piezo-electric fiber phase shifter (FPS) and iii) a three-axis electrical polarization controller (ePC). Those submodules are enclosed in a box as shown in Fig. S33b with their driving electronics. All control is performed via FPGAs sitting outside the SFDLs and communicated to the device via analog signals.

There are two types of SFDL, differentiated mainly by their total time delay, defined mainly by the length of the TDC. The first type, part of the buffer subsystem, is comprised of a longer fiber coil and has a total fiber length of (253.286 ± 0.009) m ($\sim 1.239 \mu\text{s}$), including all submodules. The second type, part of the routing subsystem, is (48.762 ± 0.004) m long ($\sim 0.239 \mu\text{s}$), also including all submodules. Note that 10 out of the 12 channels between refinery and QPU are of the short type, while two are of the long type, as depicted in

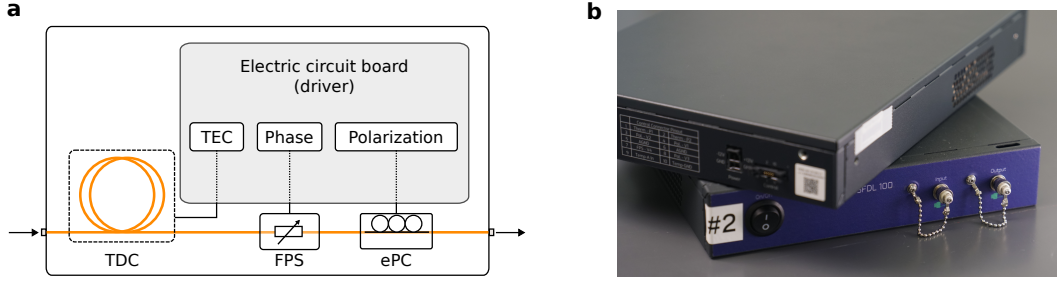


Figure S33. **a. Simplified schematic of the (stabilized) fiber delay line (SFDL).** Each SFDL contains a fiber time-delay coil (TDC), temperature controlled using a thermo-electric cooler (TEC), a fiber phase shifter (FPS), an electrical polarization controller (ePC) and a circuit board to drive the actuators. **b. Fully assembled SFDL.** SFDL modules are independently controlled via their electrical interface at the back and can be integrated into a chassis in groups of up to eight units.

Fig. S18a, such that the difference between the two interfering channels at the QPU is exactly $1\mu\text{s}$, the clock period of the experiment.

The combination of FPS (fast, but limited actuation range) and TEC (slow but large actuation range) allows for Aurora to be operated continuously, with a minimal amount of relocks. Traditional PID controllers are used for this architecture. A first PI-filter stage generates the feedback signal for the fast (10 kHz) phase actuation while a second PID applies slow feedback on the TDC to keep the FPS output near its mid-range to prevent relocks.

The individual SFDL modules are housed into a rack-mounted assembly that can take up to eight SFDLs. Each assembly has its own FPGA hardware, that runs all the 16 PIDs required to phase stabilize quantum channels. The phase error information is sent to the assembly via SFP+ transceiver modules from the subsystem arrays downstream. In the case of the refinery array, the information is sent directly from each MUX module to the corresponding 8-channel buffer, as there is a one to one correspondence between the eight MUX input channels and the eight SFDL units. The MUX's FPGAs compute signals for phase errors and send the latter digitally to the buffer which then computes the driving signals for the TEC and FPS actuators and implements the feedback to each SFDL. For polarization, the MUX's FPGAs compute the beatnote amplitudes or measure the raw reference laser amplitudes before sending data to the server computer where all the computations take place and are communicated to the buffer via Ethernet, to implement the feedback. Similarly for the QPUs, the backend module receives the phase error signal from each QPU and routes it to the correct routing module, housing the SFDLs.

Complete details of the phase calibration algorithm is treated in Sec. IX B. Polarization is set using the Nelder-Mead optimization algorithm. Polarization signals used for monitoring drifts are fed back through the server, also responsible for the computation. During the initial calibration of the full system, parameters are set to be at the middle range of operation of the electrical polarization controller. State of polarization is then optimized and indefinitely stabilized throughout the experimental run. This is done in parallel across all of the SFDLs.

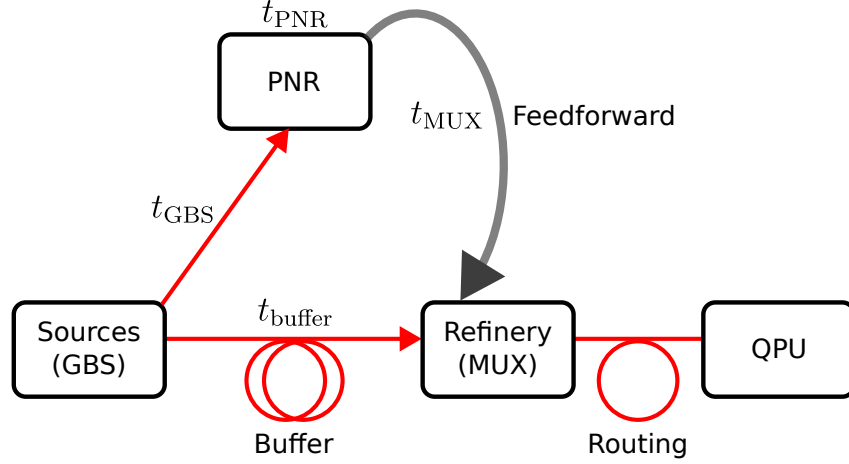


Figure S34. **Feedforward triangle.** The heralded generation of squeezed cat pairs involves three interconnected subsystems: sources (GBS), PNR detectors and refinery (MUX). Light propagation time from sources to PNR detector is t_{GBS} . Sampling time of a PNR detector analog signal and acquisition overhead is t_{PNR} while the processing time at both ends plus transmission via optical digital link towards MUX is t_{MUX} . The time delay of the buffer must be $t_{\text{buffer}} < (t_{\text{GBS}} + t_{\text{PNR}} + t_{\text{MUX}})$.

IX. EXPERIMENT: SYSTEM PERFORMANCE AND DEMONSTRATIONS

A. Photon number heralding latency

The heralded generation of squeezed cat pairs involves three interconnected subsystems as depicted in Fig. S34: sources (GBS), PNR detector and refinery (MUX). Each GBS device generates a two-mode Gaussian state at the experiment clock rate and transmits the heralding quantum pulses towards a PNR detector channel. The signal pulses (to be heralded) are guided towards the input channel of the refinery (MUX) via a low-loss optical fiber link (buffer). The PNR detector system communicates with the MUX through a classical data link.

At each experimental clock cycle (1 MHz rate), heralding pulses and signal pulses exit their respective GBS chip simultaneously. After transmission via a low-loss optical link, the array of PNR detectors detects heralding pulses upon arrival, measures the corresponding probabilistic photon number states, and communicates this information to the MUX array. At each MUX module, through a pre-defined look-up table, the photon number information is used to decide which GBS target pulse (among all of the MUX inputs) is ideal for down-stream entangled pairs synthesis. The MUX switch tree is adaptively re-configured accordingly, before the arrival of all signal pulses of the same experimental clock cycle.

Due to the feedforward nature of this system, the inter-chip fiber connection (buffer) between sources and refinery modules must compensate for the total latency, i.e. photon-number detection and all the overhead related to this task. Specifically, the buffer serves as a photonic memory delaying quantum pulses until the completion of the heralding pulse detection event and photon number extraction (t_{PNR}), data transfer, processing at the MUX and reconfiguration of switch trees (t_{MUX}), accounting for all the relevant optical (t_{GBS}), electrical and computational delays. This sets a lower bound on the optical path length, or

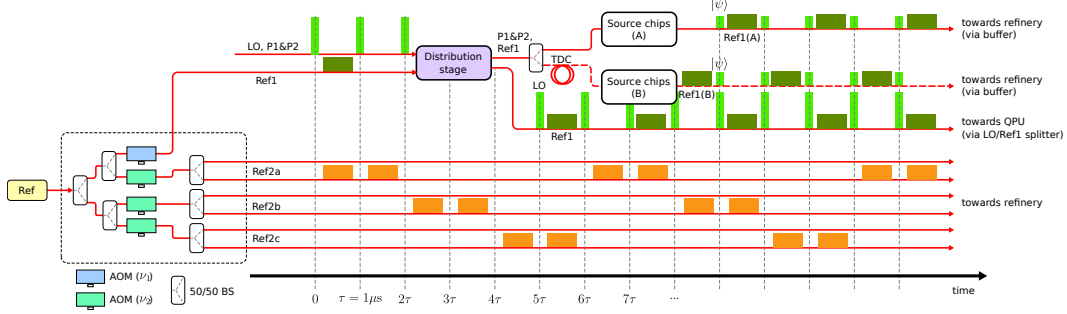


Figure S35. **Generation and distribution of reference pulses within the laser system.** The Ref splitter module contains beamsplitters and acousto-optic modulators (AOMs), responsible for generating and distributing the sequence of reference pulses for system-level phase stabilization. Ref1 is frequency detuned by $(\nu_2 - \nu_1) = 6.3$ MHz with respect to Ref2, modulated at a 500 kHz rate, and sent towards the laser system's distribution stage for interleaving with both the pulse trains of the pump (evolving into quantum states $|\psi\rangle$ on source chips) and local oscillator (LO). Ref2a, Ref2b and Ref2c pulses are generated by the other three AOMs and delayed in time by 2τ with respect to each other, with each a period of 6τ . Each Ref2 is sent towards a different refinery for phase locking its inputs.

buffering time (t_{buffer}), of this delay line. For our demonstration, $t_{\text{GBS}} = 60$ ns, $t_{\text{PNR}} = 680$ ns, $t_{\text{MUX}} = 279$ ns and thus we used $t_{\text{buffer}} \approx 1240$ ns for the buffer, corresponding to a fiber length of ~ 253 m.

B. Optical phase lock architecture

A photonic quantum computer is a complex interferometer requiring sophisticated phase stabilization techniques. The main criteria behind the ideal phase-lock scheme are i) scalability, ii) minimal crosstalk between different stages, iii) indefinite locking, and iv) resource efficiency, i.e. low physical component overheads and without adding otherwise unnecessary lossy elements to the quantum signal path.

The phase lock architecture in this work, responsible for inter-chip stability, relies on a time-multiplexed, heterodyne-based arbitrary phase lock technique. In the scheme, two reference pulses, one of them interleaved between quantum pulses, are frequency-detuned with respect to each other and interfere with one another to generate a beatnote at their difference frequency. Demodulating this signal generates the error signal fed back to phase actuators. Each PID closed-loop control system is responsible for locking the phase (and polarization) of their chaperoned channel.

Given the large number of channels within Aurora, which unavoidably cross one another across several submodules, a time multiplexed reference pulse scheme was chosen to minimize unwanted interference. To achieve such time-multiplexing capability, a reference laser (Ref, 1542.94 nm) and a network of four acousto-optic modulators (AOM) are employed, as shown in Fig. S35. One AOM generates a train of pulses repeated at half the clock cycle ($1/\tau$), i.e. at a rate of 500 kHz or a period of $2\tau = 2\mu\text{s}$, labeled as Ref1. The other three AOMs generate identical trains of pulses, but delayed in time by 2τ with respect to each other, with each a ~ 166.7 kHz rep. rate or a period of 6τ , and are labeled as Ref2a, Ref2b and Ref2c in Fig. S35. Ref1 and Ref2 pulses are frequency-shifted by 6.3 MHz with respect to

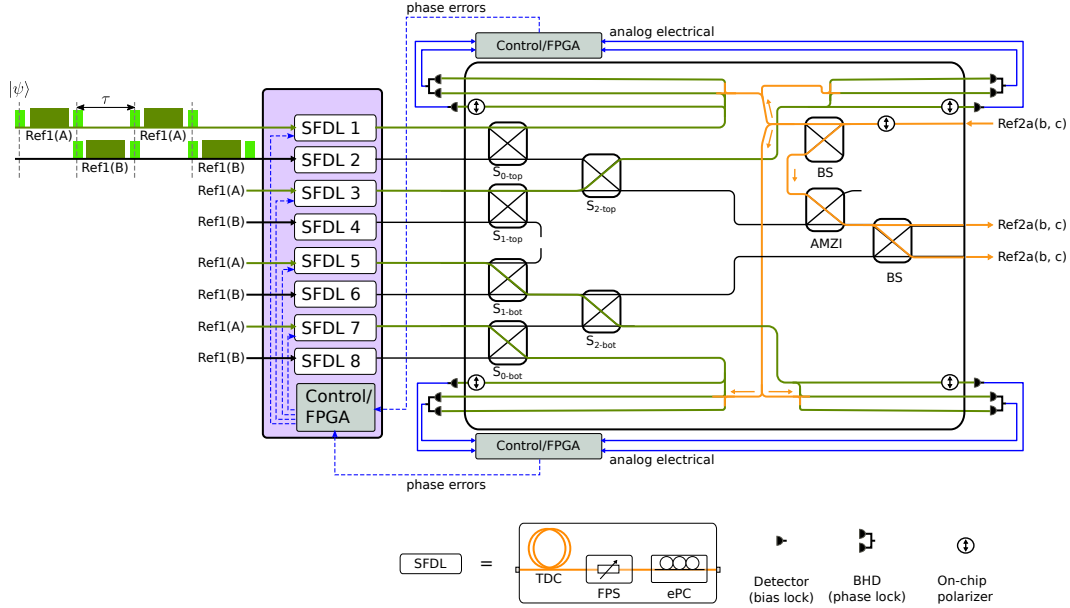


Figure S36. **Phase stabilization at the refinery.** See text for details.

each other.

Ref1 pulses are distributed between three spatial modes, two of which (labeled as Ref1(A) and Ref1(B)) will be traveling towards and through sources chips and used to stabilize the relative phase of each pair generated at the refinery (MUX). Ref1(B) is delayed by one clock cycle (τ) with respect to Ref1(A) using a time delay coil (TDC). The third channel containing Ref1 is also accompanied by the local oscillator (LO) pulse train, directed towards the QPU array for phase locking and phase sensitive quantum measurements. Each Ref2 pulse train (i.e. Ref2a, Ref2b and Ref2c) is split into two paths, which are each directed towards different MUX module.

Each MUX chip receives a single Ref2 pulse train that will interfere with both Ref1(A/B) at a different time windows and thus generate two temporally distinguishable beatnotes at the same balanced-homodyne detector (BHD) (see Fig. S36). Those signals result in error signals and are used to phase stabilize the respective input channels with the help of the SFDL array. In the *top* (input 1-4) or *bottom* (input 5-8) half of each MUX, four inputs are involved for which two beatnotes of ~ 6.3 MHz belonging to Ref1(A) and two beatnotes belonging to Ref2(B) are generated at the BHD.

Each switch of the MUX tree needs to operate around a set point (bias), for optimal extinction ratio. Therefore, each bias is locked at its optimal operating value using a dithering scheme on the input Ref1(A/B) pulses and a single photodetector. There are four such detectors per MUX chip, two for the top part and two for the bottom part as shown in Fig. S36. An on-chip polarizer in front of each bias lock detector allows input light to be actively aligned with the rest of the chip for optimal on-chip transmittance.

On each MUX chip, the input Ref2a(b, c) pulse train is split into two different paths with the help of an on-chip variable beamsplitter (BS). One output is used to interfere with all the Ref1(A/B) interleaved between quantum pulses and thus phase lock all the input modes with a common reference signal as previously discussed. The second output of the BS

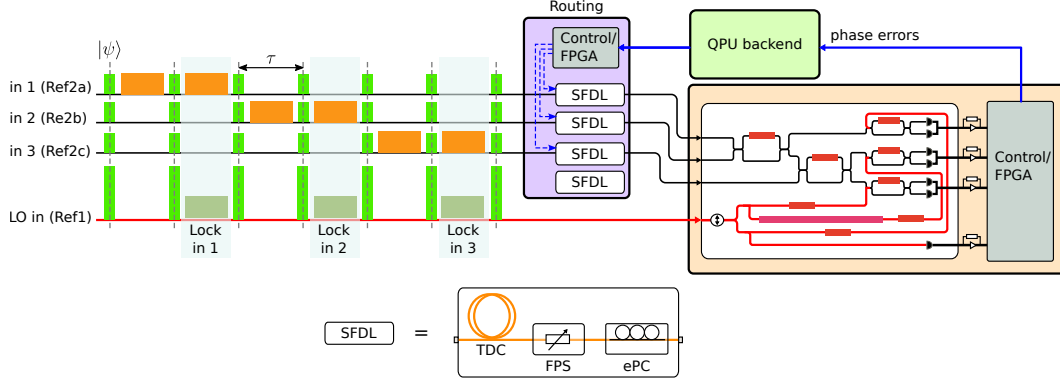


Figure S37. **Phase stabilization at the QPU array.** See text for details.

is sent out towards the QPU, after passing through an on-chip asymmetric Mach-Zehnder interferometer (AMZI), for interleaving the reference pulse train with the output quantum pairs. As a consequence of the previous step, at this point the relative phase between Ref2 and quantum pairs is fixed. The timing difference of Ref2 trains from different MUX modules simplifies phase locking at the QPU.

As shown in Fig. S37, Ref2a, Ref2b and Ref2c pulse trains are temporally arranged such that they do not interfere with each other at the QPU input interferometer. Note that each input to a QPU module comes from a different MUX and for each input, there is a Ref2 pulse to interfere with the incoming Ref1 pulse train interleaved with the LO pulse train. By phase locking each input mode with the common LO channel, one can ensure that LO pulses are phase locked with respect to the quantum pulses during quadrature measurements. RMS phase noise of phase locks, estimated from classical light pulses, is shown in Fig. S38.

In order to reconstruct the relative phase evolution of each optical beam that needs to be phase-stabilized at the MUX or QPU, we digitize and digitally process the heterodyne beatnote signals using fast 125 MSa/s ADCs and Zynq 7020 FPGAs. In a first step, we extract two quadratures for each beam by multiplying the digitized heterodyne signal with both a sine and cosine wave at the 6.3 MHz beatnote modulation frequency, and averaging both resulting quadrature signals for the duration of a single pulse with a rectangular weight function. The two values are then transformed into a wrapped phase angle (modulo 2π) using the CORDIC algorithm. The total phase drift since initialisation of the phase estimator module is continuously computed by unwrapping this wrapped phase angle, with a range that is only limited by the bit width of the phase accumulation register storing the result. For purposes such as measurements of open-loop phase evolution, this phase accumulation register represents phase as a 32-bit fixed point number with 14 bits representing the fractional part in units of 2π .

For closed-loop operation, we subtract a 32-bit phase setpoint from the phase accumulation register to obtain an error signal, which we represent as a 16-bit fixed-point number with 12 fractional bits. For large phase errors, overflows of the error signal are prevented by clamping this 16-bit signal at the most positive/negative value that it can assume, while the 32-bit phase accumulation register continues to keep track of the true phase evolution (overflows of the phase accumulation register would only occur for phase drifts by $2\pi \times 2^{17}$, well above any practical value). The error signal is sent over a serial digital optical link to

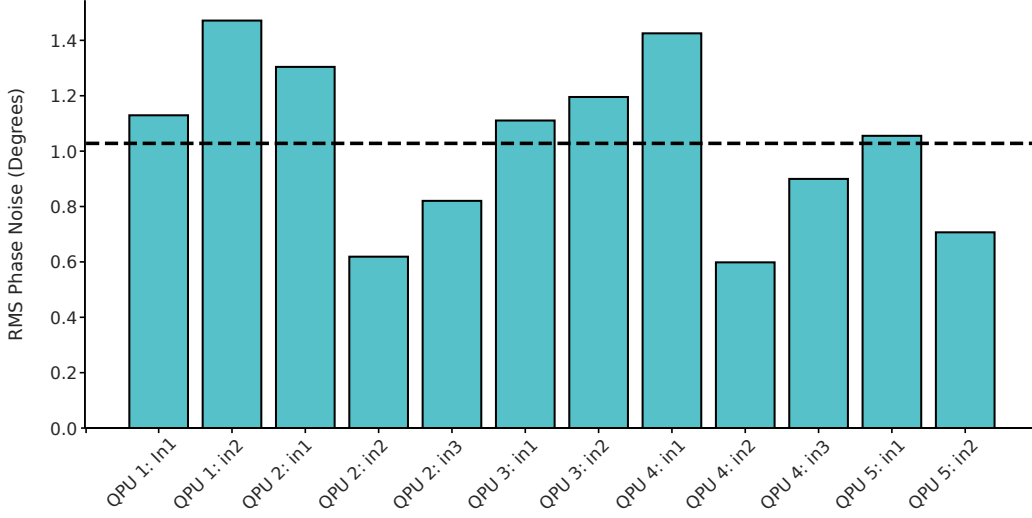


Figure S38. **RMS phase noise at the QPU.** This histogram represents the residual phase noise, while the phase stabilization loop is activated, measured with classical reference pulses. The mean value (black dashed line) is $(1.027 \pm 0.287)^\circ$.

a PID controller driving the actuator hardware. Our scheme guarantees that the actuator eventually brings the accumulated phase back to its setpoint, even if large temporary phase jumps are too fast for the actuator to keep up with.

1. Phase calibration and initialization

The reference (Ref) laser used for phase-locking has a frequency offset of 300 GHz with respect to the quantum and LO pulses and will, due to dispersion, experience a different phase rotation than the signal light it co-propagates with. This relative phase between reference and signal light, however, is temporally constant for each spatial mode and can be compensated for using the setpoint of the arbitrary-phase locks (achieved in practice by adjusting the phase offset of the mix signal used for demodulation of the heterodyne beatnote).

In order to align the signal phase in the presence of unknown relative offsets to the reference beams, we perform a system-level phase calibration and initialization procedure. Sweeping the phase setpoints of the SFDLS, between refinery and QPU, while tracking the variance of single-mode quadratures and entangled pair nullifiers, we obtain the necessary information to adjust all phase locks as well as align the twelve local-oscillator phase offsets to one another at the QPU.

C. System transmittance

Loss is inherent to all photonic systems. Transmittance through each subsystem is reported in the following tables. We refer to Fig. S39 for the different paths and subsystems.

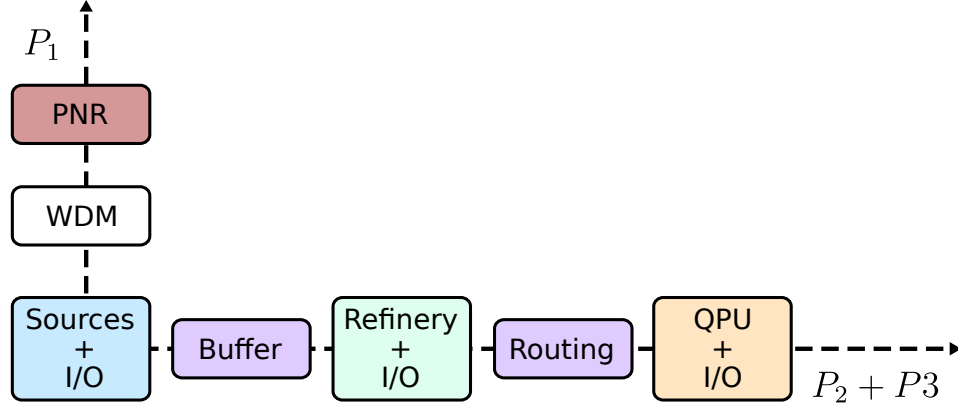


Figure S39. **Transmittance through the system.** The path going from sources to PNR detector is labeled as P_1 , while the path going from sources to QPU is labeled as $P_2 + P_3$.

Module	Loss (dB)	Transmittance (%)
Sources + I/O + WDM	2.2	60
PNR	0.76	84
Mating Sleeves	0.6	87
Total	3.6	44

Table SVII. **Heralding path (P_1)**

Module	Loss (dB)	Transmittance (%)
Sources + I/O	1.9	65
Refinery + I/O	4.2	38
QPU + I/O	4.4	36
Interconnects + Extras	0.95	80
Mating Sleeves	1.8	66
Total	13.25	4.7

Table SVIII. **Heralded path ($P_2 + P_3$).**

-
- [1] I. Tzitrin, T. Matsuura, R. N. Alexander, G. Dauphinais, J. E. Bourassa, K. K. Sabapathy, N. C. Menicucci, and I. Dhand, Fault-tolerant quantum computation with static linear optics, PRX Quantum **2**, 10.1103/prxquantum.2.040353 (2021).
 - [2] N. C. Menicucci, Fault-tolerant measurement-based quantum computing with continuous-variable cluster states, Phys. Rev. Lett. **112**, 120504 (2014).
 - [3] J. E. Bourassa, R. N. Alexander, M. Vasmer, A. Patil, I. Tzitrin, T. Matsuura, D. Su, B. Q. Baragiola, S. Guha, G. Dauphinais, K. K. Sabapathy, N. C. Menicucci, and I. Dhand, Blueprint for a scalable photonic fault-tolerant quantum computer, Quantum **5**, 392 (2021).
 - [4] K. Duivenvoorden, B. M. Terhal, and D. Weigand, Single-mode displacement sensor, Phys. Rev. A **95**, 012305 (2017).
 - [5] D. Gottesman, A. Kitaev, and J. Preskill, Encoding a qubit in an oscillator, Phys. Rev. A **64**, 012310 (2001).
-

- [6] K. Duivenvoorden, B. M. Terhal, and D. Weigand, Single-mode displacement sensor, *Phys. Rev. A* **95**, 10.1103/physrev.95.012305 (2017).
- [7] B. W. Walshe, B. Q. Baragiola, R. N. Alexander, and N. C. Menicucci, Continuous-variable gate teleportation and bosonic-code error correction, *Phys. Rev. A* **102**, 062411 (2020).
- [8] D. J. Weigand and B. M. Terhal, Generating grid states from Schrödinger ödinger-cat states without postselection, *Phys. Rev. A* **97**, 022341 (2018).
- [9] K. Takase, F. Hanamura, H. Nagayoshi, J. E. Bourassa, R. N. Alexander, A. Kawasaki, W. Asavanant, M. Endo, and A. Furusawa, Generation of flying logical qubits using generalized photon subtraction with adaptive Gaussian operations, *arXiv preprint arXiv:2401.07287* (2024).
- [10] S. Konno, W. Asavanant, F. Hanamura, H. Nagayoshi, K. Fukui, A. Sakaguchi, R. Ide, F. China, M. Yabuno, S. Miki, H. Terai, K. Takase, M. Endo, P. Marek, R. Filip, P. van Loock, and A. Furusawa, Logical states for fault-tolerant quantum computation with propagating light, *Science* **383**, 289–293 (2024).
- [11] A. Serafini, *Quantum continuous variables: a primer of theoretical methods* (CRC press, 2017).
- [12] B. W. Walshe, B. Q. Baragiola, R. N. Alexander, and N. C. Menicucci, Continuous-variable gate teleportation and bosonic-code error correction, *Phys. Rev. A* **102**, 062411 (2020).
- [13] B. W. Walshe, B. Q. Baragiola, H. Ferretti, J. Gefaell, M. Vasmer, R. Weil, T. Matsuura, T. Jaeken, G. Pantaleoni, Z. Han, *et al.*, Linear-optical quantum computation with arbitrary error-correcting codes, *arXiv preprint arXiv:2408.04126* (2024).
- [14] I. Tzitrin, J. E. Bourassa, N. C. Menicucci, and K. K. Sabapathy, Progress towards practical qubit computation using approximate Gottesman-Kitaev-Preskill codes, *Phys. Rev. A* **101**, 10.1103/physrev.101.032315 (2020).
- [15] K. Fukui, S. Takeda, M. Endo, W. Asavanant, J.-i. Yoshikawa, P. van Loock, and A. Furusawa, Efficient backcasting search for optical quantum state synthesis, *Phys. Rev. Lett.* **128**, 240503 (2022).
- [16] K. Takase, K. Fukui, A. Kawasaki, W. Asavanant, M. Endo, J.-i. Yoshikawa, P. van Loock, and A. Furusawa, Gottesman-Kitaev-Preskill qubit synthesizer for propagating light, *npj Quantum Information* **9**, 10.1038/s41534-023-00772-y (2023).
- [17] H. M. Vasconcelos, L. Sanz, and S. Glancy, All-optical generation of states for “encoding a qubit in an oscillator”, *Optics Letters* **35**, 3261 (2010).
- [18] J. Etesse, M. Bouillard, B. Kanseri, and R. Tualle-Broui, Experimental generation of squeezed cat states with an operation allowing iterative growth, *Phys. Rev. Lett.* **114**, 193602 (2015).
- [19] B. G. Lee and N. Dupuis, Silicon photonic switch fabrics: Technology and architecture, *J. Lightwave Technol.* **37**, 6 (2019).
- [20] S. Bartolucci, P. Birchall, D. Bonneau, H. Cable, M. Gimeno-Segovia, K. Kieling, N. Nickerson, T. Rudolph, and C. Sparrow, Switch networks for photonic fusion-based quantum computing, *arXiv preprint arXiv:2109.13760* (2021).
- [21] A. Laghaout, J. S. Neergaard-Nielsen, I. Rigas, C. Kragh, A. Tipsmark, and U. L. Andersen, Amplification of realistic Schrödinger ödinger-cat-state-like states by homodyne heralding, *Phys. Rev. A* **87**, 043826 (2013).
- [22] R. Filip, P. Marek, and U. L. Andersen, Measurement-induced continuous-variable quantum interactions, *Phys. Rev. A* **71**, 042308 (2005).
- [23] J. E. Bourassa, N. Quesada, I. Tzitrin, A. Száva, T. Isacson, J. Izaac, K. K. Sabapathy, G. Dauphinais, and I. Dhand, Fast simulation of bosonic qubits via Gaussian functions in phase space, *PRX Quantum* **2**, 040315 (2021).
- [24] M. Dakna, T. Anhut, T. Opatrny, L. Knoll, and D.-G. Welsch, Schrödinger ödinger cat-like states by conditional measurements on a beam-splitter, *arXiv preprint quant-ph/9612011* (1996).
- [25] J. S. Neergaard-Nielsen, B. M. Nielsen, C. Hettich, K. Mølmer, and E. S. Polzik, Generation of a superposition of odd photon number states for quantum information networks, *Phys. Rev. Lett.* **97**, 083604 (2006).
- [26] D. Su, C. R. Myers, and K. K. Sabapathy, Conversion of Gaussian states to non-Gaussian states using photon-number-resolving detectors, *Phys. Rev. A* **100**, 052301 (2019).
- [27] K. Takase, J.-i. Yoshikawa, W. Asavanant, M. Endo, and A. Furusawa, Generation of optical Schrödinger ödinger cat states by generalized photon subtraction, *Phys. Rev. A* **103**, 013710

- (2021).
- [28] C. Weedbrook, S. Pirandola, R. García-Patrón, N. J. Cerf, T. C. Ralph, J. H. Shapiro, and S. Lloyd, Gaussian quantum information, *Reviews of Modern Physics* **84**, 621 (2012).
 - [29] K. Takase, F. Hanamura, H. Nagayoshi, J. E. Bourassa, R. N. Alexander, A. Kawasaki, W. Asavanant, M. Endo, and A. Furusawa, Generation of flying logical qubits using generalized photon subtraction with adaptive Gaussian operations, arXiv preprint arXiv:2401.07287 (2024).
 - [30] H. Vahlbruch, M. Mehmet, K. Danzmann, and R. Schnabel, Detection of 15 db squeezed states of light and their application for the absolute calibration of photoelectric quantum efficiency, *Phys. Rev. Lett.* **117**, 10.1103/physrevlett.117.110801 (2016).
 - [31] M. Eaton, A. Hossameldin, R. J. Birrittella, P. M. Alsing, C. C. Gerry, H. Dong, C. Cuevas, and O. Pfister, Resolution of 100 photons and quantum generation of unbiased random numbers, *Nature Photonics* **17**, 106–111 (2022).
 - [32] F. M. Miatto and N. Quesada, Fast optimization of parametrized quantum optical circuits, *Quantum* **4**, 366 (2020).
 - [33] Y. Yao, F. Miatto, and N. Quesada, Riemannian optimization of photonic quantum circuits in phase and fock space, arXiv preprint arXiv:2209.06069 (2022).
 - [34] C. C. Gerry and P. L. Knight, *Introductory quantum optics* (Cambridge university press, 2023).
 - [35] R. Kruse, C. S. Hamilton, L. Sansoni, S. Barkhofen, C. Silberhorn, and I. Jex, Detailed study of Gaussian boson sampling, *Phys. Rev. A* **100**, 032326 (2019).
 - [36] R. De Prins, Y. Yao, A. Apte, and F. M. Miatto, A quadratic speedup in the optimization of noisy quantum optical circuits, *Quantum* **7**, 1097 (2023).
 - [37] XanaduAI, MrMustard, <https://github.com/XanaduAI/MrMustard> (2021).
 - [38] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, Array programming with NumPy, *Nature* **585**, 357 (2020).
 - [39] J. Hastrup, *Continuous-variable quantum codes for fault-tolerant quantum information processing*, Ph.D. thesis, Technical University of Denmark (2021).
 - [40] A. Kitaev, Fault-tolerant quantum computation by anyons, *Annals of Physics* **303**, 2 (2003).
 - [41] S. B. Bravyi and A. Y. Kitaev, Quantum codes on a lattice with boundary, arXiv preprint quant-ph/9811052 (1998).
 - [42] M. H. Freedman and D. A. Meyer, Projective plane and planar quantum codes, *Foundations of Computational Mathematics* **1**, 325 (2001).
 - [43] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, Surface codes: Towards practical large-scale quantum computation, *Phys. Rev. A* **86**, 032324 (2012).
 - [44] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, Topological quantum memory, *Journal of Mathematical Physics* **43**, 4452 (2002), <https://pubs.aip.org/aip/jmp/article-pdf/43/9/4452/19183135/4452.1.online.pdf>.
 - [45] C. Vuillot, H. Asasi, Y. Wang, L. P. Pryadko, and B. M. Terhal, Quantum error correction with the toric Gottesman-Kitaev-Preskill code, *Phys. Rev. A* **99**, 032344 (2019).
 - [46] K. Noh and C. Chamberland, Fault-tolerant bosonic quantum error correction with the surface-Gottesman-Kitaev-Preskill code, *Phys. Rev. A* **101**, 012316 (2020).
 - [47] C. Wang, J. Harrington, and J. Preskill, Confinement-higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory, *Annals of Physics* **303**, 31 (2003).
 - [48] R. Raussendorf and J. Harrington, Fault-tolerant quantum computation with high threshold in two dimensions, *Phys. Rev. Lett.* **98**, 190504 (2007).
 - [49] A. G. Fowler, A. M. Stephens, and P. Groszkowski, High-threshold universal quantum computation on the surface code, *Phys. Rev. A* **80**, 052312 (2009).
 - [50] S. D. Barrett and T. M. Stace, Fault tolerant quantum computation with very high threshold for loss errors, *Phys. Rev. Lett.* **105**, 200502 (2010).
 - [51] D. S. Wang, A. G. Fowler, and L. C. L. Hollenberg, Surface code quantum computing with error rates over 1%, *Phys. Rev. A* **83**, 020302 (2011).
 - [52] D. K. Tuckett, S. D. Bartlett, S. T. Flammia, and B. J. Brown, Fault-tolerant thresholds for the surface code in excess of 5% under biased noise, *Phys. Rev. Lett.* **124**, 130501 (2020).

- [53] S. Bartolucci, P. Birchall, H. Bombín, H. Cable, C. Dawson, M. Gimeno-Segovia, E. Johnston, K. Kielsing, N. Nickerson, M. Pant, F. Pastawski, T. Rudolph, and C. Sparrow, Fusion-based quantum computation, *Nature Communications* **14**, 912 (2023).
- [54] N. P. Breuckmann and J. N. Eberhardt, Quantum low-density parity-check codes, *PRX Quantum* **2**, 040101 (2021).
- [55] D. MacKay, G. Mitchison, and P. McFadden, Sparse-graph codes for quantum error correction, *IEEE Transactions on Information Theory* **50**, 2315 (2004).
- [56] J.-P. Tillich and G. Zémor, Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength, *IEEE Transactions on Information Theory* **60**, 1193 (2014).
- [57] S. Bravyi and M. B. Hastings, Homological product codes, in *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '14 (Association for Computing Machinery, New York, NY, USA, 2014) p. 273–282.
- [58] A. A. Kovalev and L. P. Pryadko, Quantum Kronecker sum-product low-density parity-check codes with finite rate, *Phys. Rev. A* **88**, 012311 (2013).
- [59] N. P. Breuckmann and B. M. Terhal, Constructions and noise threshold of hyperbolic surface codes, *IEEE Transactions on Information Theory* **62**, 3731 (2016).
- [60] N. P. Breuckmann and J. N. Eberhardt, Balanced product quantum codes, *IEEE Transactions on Information Theory* **67**, 6653 (2021).
- [61] P. Panteleev and G. Kalachev, Degenerate Quantum LDPC Codes With Good Finite Length Performance, *Quantum* **5**, 585 (2021).
- [62] P. Panteleev and G. Kalachev, Quantum LDPC codes with almost linear minimum distance, *IEEE Transactions on Information Theory* **68**, 213 (2022).
- [63] P. Panteleev and G. Kalachev, Asymptotically good quantum and locally testable classical LDPC codes, in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022 (Association for Computing Machinery, New York, NY, USA, 2022) p. 375–388.
- [64] A. Leverrier and G. Zémor, Quantum tanner codes, in *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)* (2022) pp. 872–883.
- [65] A. Bolt, G. Duclos-Cianci, D. Poulin, and T. Stace, Foliated quantum error-correcting codes, *Phys. Rev. Lett.* **117**, 10.1103/physrevlett.117.070501 (2016).
- [66] B. J. Brown and S. Roberts, Universal fault-tolerant measurement-based quantum computation, *Phys. Rev. Res.* **2**, 033305 (2020).
- [67] H. J. Briegel and R. Raussendorf, Persistent entanglement in arrays of interacting particles, *Phys. Rev. Lett.* **86**, 910–913 (2001).
- [68] M. Hein, J. Eisert, and H. J. Briegel, Multiparty entanglement in graph states, *Phys. Rev. A* **69**, 10.1103/physreva.69.062311 (2004).
- [69] R. Raussendorf, J. Harrington, and K. Goyal, A fault-tolerant one-way quantum computer, *Annals of Physics* **321**, 2242–2270 (2006).
- [70] R. N. Alexander and N. C. Menicucci, Flexible quantum circuits using scalable continuous-variable cluster states, *Phys. Rev. A* **93**, 10.1103/physreva.93.062326 (2016).
- [71] FlamingPy repository, <https://github.com/XanaduAI/flamingpy>.
- [72] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, 2010).
- [73] M. P. Stafford and N. C. Menicucci, Biased Gottesman-Kitaev-Preskill repetition code, *Phys. Rev. A* **108**, 10.1103/physreva.108.052428 (2023).
- [74] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann series in representation and reasoning (Elsevier Science, 1988).
- [75] T. Richardson and R. Urbanke, *Modern Coding Theory* (Cambridge University Press, 2008).
- [76] N. C. Menicucci, Temporal-mode continuous-variable cluster states using linear optics, *Phys. Rev. A* **83**, 062314 (2011).
- [77] P. Wang, M. Chen, N. C. Menicucci, and O. Pfister, Weaving quantum optical frequency combs into continuous-variable hypercubic cluster states, *Phys. Rev. A* **90**, 032325 (2014).
- [78] R. N. Alexander and N. C. Menicucci, Flexible quantum circuits using scalable continuous-variable cluster states, *Phys. Rev. A* **93**, 062326 (2016).

- [79] R. N. Alexander, S. C. Armstrong, R. Ukai, and N. C. Menicucci, Noise analysis of single-mode Gaussian operations using continuous-variable cluster states, *Phys. Rev. A* **90**, 062324 (2014).
- [80] L.-M. Duan, G. Giedke, J. I. Cirac, and P. Zoller, Inseparability criterion for continuous variable systems, *Phys. Rev. Lett.* **84**, 2722–2725 (2000).
- [81] P. Van Loock and A. Furusawa, Detecting genuine multipartite continuous-variable entanglement, *Phys. Rev. A* **67**, 052315 (2003).
- [82] B. M. Terhal, Quantum error correction for quantum memories, *Rev. Mod. Phys.* **87**, 307 (2015).
- [83] R. W. P. Drever, J. L. Hall, F. V. Kowalski, J. Hough, G. M. Ford, A. J. Munley, and H. Ward, Laser phase and frequency stabilization using an optical resonator, *Applied Physics B* **31**, 97 (1983).
- [84] Y. Zhang, M. Menotti, K. Tan, V. Vaidya, D. Mahler, L. Helt, L. Zatti, M. Liscidini, B. Morrison, and Z. Vernon, Squeezed light from a nanophotonic molecule, *Nature communications* **12**, 2233 (2021).
- [85] L. S. Madsen, F. Laudenbach, M. F. Askarani, F. Rortais, T. Vincent, J. F. Bulmer, F. M. Miatto, L. Neuhaus, L. G. Helt, M. J. Collins, *et al.*, Quantum computational advantage with a programmable photonic processor, *Nature* **606**, 75 (2022).
- [86] B. Calkins, A. E. Lita, A. E. Fox, and S. Woo Nam, Faster recovery time of a hot-electron transition-edge sensor by use of normal metal heat-sinks, *Applied Physics Letters* **99** (2011).
- [87] R. Hummatov, A. E. Lita, T. Farrahi, N. Otrooshi, S. Fayer, M. J. Collins, M. Durkin, D. Bennett, J. Ullom, R. P. Mirin, *et al.*, Fast transition-edge sensors suitable for photonic quantum computing, *Journal of Applied Physics* **133** (2023).
- [88] A. E. Lita, A. J. Miller, and S. W. Nam, Counting near-infrared single-photons with 95% efficiency, *Optics Express* **16**, 3032 (2008).
- [89] V. D. Vaidya, B. Morrison, L. G. Helt, R. Shahrokshahi, D. Mahler, M. Collins, K. Tan, J. Lavoie, A. Repington, M. Menotti, N. Quesada, R. C. Pooser, A. E. Lita, T. Gerrits, S. W. Nam, and Z. Vernon, Broadband quadrature-squeezed vacuum and nonclassical photon number correlations from a nanophotonic device, *Science Advances* **6**, eaba9186 (2020).
- [90] A. I. Lvovsky, Iterative maximum-likelihood reconstruction in quantum homodyne tomography, *J. Opt. B: Quantum Semiclassical Opt.* **6**, S556 (2004).
- [91] C. Wang, M. Zhang, B. Stern, M. Lipson, and M. Lončar, Nanophotonic lithium niobate electro-optic modulators, *Optics express* **26**, 1547 (2018).
- [92] M. Defossez, Lvs 4x asynchronous oversampling using 7 series fpgas and zynq-7000 ap socs, Xilinx Inc (2017).