Low-Space Quantum Discrete Logarithms on Genus-Two Jacobians

Yan Huang[1], Yuling Chen[1], and Fangguo Zhang[2]

[1] School of Mathematics and Statistics, Hunan University of Science and Technology, Xiangtan 411201, China.
[2] School of Computer Science and Engineering, Sun Yat-sen University, Guangzhou 510006, China.

Abstract. Quantum discrete logarithms on genus-two Jacobians require reversible arithmetic with controlled workspace. Generic divisor-addition formulas apply only outside an exceptional set, which must be accounted for in quantum phase computation.

We give a randomized reduction using two fresh independent uniform public seeds in a finite abelian group of odd order to randomize a complete balanced addition tree. For inputs independent of the seeds, exceptional-pair densities bound the averaged phase-channel error and sampling failure probability. We apply this reduction to an odd-prime-order subgroup of a genus-two Jacobian over $\mathbb{F}_q$, where q is an odd prime and the supplied curve model is a square-free monic quintic with zero quartic coefficient. Integer lifting, an independently shifted Legendre phase, and streaming Chinese remainder reconstruction yield a low-space sampler.

For subgroup order $r = \Theta(q^2)$, the useful-sample probability is $1 - \mathcal{O}(L/q)$, averaged over fresh seeds, for an exact uniform input state and ideal Fourier readout, where L is the number of tree leaves. With window width $w = \Theta(\log_2 n)$, the sampler uses $10n + o(n)$ logical qubits, including both scalar registers, with $n = \lceil \log_2 q \rceil$, and allows intermediate measurements, reset, and classical feedforward. For the Gaudry–Schost instance over $\mathbb{F}_{2^{127}-1}$, a 16-bit window gives an analytic allocation of 1,971 qubits and a capped-run Toffoli bound below $2^{60.561}$. A matched schedule derived from Chen et al.'s addition module uses 3,063 qubits; both schedules retain the same 500 scalar-input qubits. These comparisons are schedule-specific, and rotation synthesis is counted separately.

Keywords: quantum discrete logarithms; genus-two Jacobians; reversible divisor arithmetic; Legendre symbol; output compression.

1 Introduction

Shor's algorithm factors integers and solves discrete logarithms in finite cyclic groups in polynomial time [40]. Its reversible group arithmetic accounts for much of the gate count and logical qubit allocation. Estimating these costs is therefore central to assessing quantum attacks on public-key cryptography.

For RSA factoring, Beauregard's circuit uses $2B + 3$ logical qubits, where B is the modulus bit length [3]. Häner, Roetteler, and Svore reduced this to $2B + 2$ qubits using Toffoli-based modular multiplication [22]. Chevignard, Fouque, and Schrottenloher obtained $B/2 + o(B)$ qubits for balanced RSA moduli [9]. They combine residue arithmetic and output compression with the Ekerå–Håstad short-discrete-logarithm algorithm [18], computing only a small part of the modular-exponentiation output with sublinear workspace. Repeated quantum samples are then processed classically. Gidney further optimized this approach and reported 2,043 logical qubits for a 3,072-bit RSA modulus [20].

Space reductions for prime-field elliptic-curve discrete logarithms have followed both arithmetic and output-compression approaches. Writing n for the base-field bit length, Proos and Zalka proposed register sharing with an asymptotic allocation of $5n + o(n)$ qubits [35]. Roetteler et al. [37] and Häner et al. [23] supplied explicit arithmetic circuits and detailed estimates. The output-compression construction of Chevignard, Fouque, and Schrottenloher uses $3.12n + o(n)$ qubits at the cost of more gates [11]. Luo et al. [31] obtained $3n+6\lfloor \log_2 n \rfloor + O(1)$ qubits through compact reversible inversion, register sharing, intermediate measurements, and classical feedforward. Thus quantum space depends both on the arithmetic circuits and on how much output is retained for period finding.

For genus-two Jacobians, a reduced divisor of degree two has four field coefficients, and addition and normalization require further temporary registers. Huang et al. [25] estimated these resources over prime

fields; Chen et al. [8] later gave explicit circuits over Mersenne prime fields. These arithmetic baselines lead to the question considered here: how much quantum space can be saved by computing only the information needed for Fourier sampling?

Barbulescu and Bisson [2] extend Regev-type discrete-logarithm algorithms to hyperelliptic curves. Their genus-two construction is heuristic and concerns splitting the computation among parallel runs; their unconditional result applies to growing genus. Our construction instead gives an explicit logical-space bound for a genus-two phase sampler.

Shor's algorithm and output compression. Let C be a genus-two curve over an odd prime field $\mathbb{F}_q$. Let $G = \langle P \rangle \subseteq \mathrm{Jac}(C)(\mathbb{F}_q)$ have known odd prime order r. We write $\mathbb{Z}_r = \mathbb{Z}/r\mathbb{Z}$ and use $0, \ldots, r-1$ as register representatives. The notation $[a]D$ denotes scalar multiplication of a divisor class D. Given $Q = [k]P$, we seek $k \in \mathbb{Z}_r$. The group-valued function

$$F(a, b) = [a]P + [b]Q, \qquad (a, b) \in \mathbb{Z}_r^2,$$

satisfies $F(a + k, b - 1) = F(a, b)$. Fourier sampling recovers k from this periodicity. A standard evaluation circuit stores a canonical representation of $F(a, b)$ before the Fourier stage.

Output compression replaces the group representation with a smaller value that retains enough information about the period. May and Schlieper [33] studied this idea for period finding. Chevignard, Fouque, and Schrottenloher (CFS) [11] developed a prime-field elliptic-curve construction: they lift projective formulas to the integers, evaluate them through a residue number system, and compress the output to one bit using a Legendre symbol. This avoids field inversion and reduces storage at the cost of more gates. The compressed value must be computed directly to obtain the space saving; computing a full group element first still incurs that evaluator's peak width.

The genus-two case. The genus-two extension requires control of exceptional inputs and integer growth, together with a proof that the chosen phase gives useful samples. Cantor's algorithm provides complete divisor arithmetic [6], and explicit weighted coordinates and inversion-free addition laws are available in [29,14,24,34]. We use a generic weighted Mumford formula, which may fail when a denominator vanishes or a reduced divisor has lower degree. These inputs must be included in the quantum error analysis.

Integer lifting also increases the intermediate values, whose sizes determine the number of auxiliary residue moduli. Moreover, invariance under coordinate scaling alone does not ensure useful Fourier samples. Both the height bounds and the sampling property therefore need separate arguments.

Contribution. We construct a compressed discrete-logarithm sampler for curves with a square-free monic quintic model $C : y^2 = f(x)$ and zero quartic coefficient; the assumptions are stated in Section 2. Write $n = \lceil \log_2 q \rceil$ and $m_r = \lceil \log_2 r \rceil$. For $r = \Theta(q^2)$ and window width $w = \Theta(\log_2 n)$, the sampler uses at most $10n + o(n)$ logical qubits, including both m_r-bit scalar registers. We count peak live qubits and allow intermediate measurements, reset, and classical feedforward. Under this window choice, classical preprocessing and quantum arithmetic both have polynomial cost. Theorem 6.1 gives the full statement.

The construction has two main components within the CFS output-compression and residue-arithmetic framework [11]. Binary Jacobi evaluation [39] supplies the phase computation, and measured tree evaluation [28] supplies the cleanup method. First, two independent uniform public group seeds randomize the leaves of a complete balanced addition tree. For every fixed scalar input, the true child sums at each internal node are independent and uniform. The density of exceptional pairs therefore gives a failure bound uniform over inputs. For a genus-two tree with L leaves, this bound is $(L - 1)(14q + 3)/r$. We also bound the error of the quantum channel averaged over fresh seeds, retaining their classical records. The input and any reference system are assumed independent of the seeds. The masking argument applies to every finite abelian group of odd order, independently of the genus-two coordinates. It uses affine randomization [7] and the random translations appearing in earlier quantum discrete-logarithm analyses [35,37].

Second, we lift the weighted Mumford formulas to the integers and give a reversible schedule for the compressed phase. An independent uniform $\gamma \in \mathbb{F}_q$ defines the shifted Legendre phase. On generic inputs, a terminal polynomial of degree at most 19 evaluates this phase without field inversion. We compute its residues modulo small auxiliary primes and reconstruct its value modulo q by streaming Chinese remainder reconstruction. Explicit height bounds determine the required prime product, while a shared evaluation

schedule and its inverse clear the temporary registers. The integer lift and reversible implementation, beyond the existing coordinate formulas, provide the arithmetic basis of our space estimate.

For an exact uniform input state on $\mathbb{Z}_r^2$ and ideal Fourier readout, the useful-sample probability, averaged over fresh independent tree seeds and a Legendre shift, is $1 - O(L/q)$ when $r = \Theta(q^2)$. The finite analysis also includes preparation, Fourier, and rotation-synthesis errors. Every candidate k' is checked classically by $[k']P = Q$.

Resource estimates. On the Gaudry–Schost instance over $\mathbb{F}_{2^{127}-1}$, a 16-bit window gives an upper bound of 1,971 logical qubits and a capped-run bound below $2^{60.561}$ Toffoli gates. The classical leaf tables contain at most 2^{21} entries. A matched schedule derived from Chen et al.'s Mersenne-field addition module uses 3,063 qubits. Both schedules retain the same 500 scalar-input qubits. The comparison concerns these specified module allocations and makes no optimality claim over other circuit designs. Section 6.2 also analyzes recycled-control sampling and separates its complete translation requirement from the published generic arithmetic widths.

With a 32-bit window, the allocation falls to 1,661 qubits, while the Toffoli bound rises to about $2^{71.36}$ and the classical tables have at most 2^{36} entries. The gate estimates include reversible table selection, reconstruction, cleanup, preparation, and Fourier readout; rotation synthesis is counted separately. The smaller quantum allocation thus requires substantially more gates and classical storage. Detailed comparisons appear in Section 6.2.

Organization of the paper. Section 2 collects notation and introduces the curve model and reversible arithmetic. Section 3 sets out compressed Fourier sampling on the exact input domain and its resource conventions. The shifted Legendre phase and terminal polynomial are constructed in Section 4. Section 5 develops the residue tree, height bounds, reconstruction, and masking theorem. Section 6 combines these results and evaluates the sampler's resources. Section 7 discusses the remaining questions. Appendix A gives complete arithmetic and terminal identities. Appendix B proves the general masked reduction. Reversible implementations and sampling interfaces are collected in Appendix C, and verification certificates and resource comparisons in Appendix D. The computational supplement contains the verification programs, exact certificates, and recorded results; its contens and execution entry point are specified in the data availability statement.

2 Hyperelliptic Curves and Arithmetic

We first collect notation, then describe reversible base-field arithmetic, the curve model, and canonical divisor arithmetic.

2.1 Notation

We write $\mathbb{F}_q$ for the field with q elements, where q is an odd prime, and use $\mathbb{Z}_m = \mathbb{Z}/m\mathbb{Z}$ for residues modulo m. The binary alphabet is $\{0, 1\}$. Field elements and scalar inputs use canonical representatives. The symbol $\oplus$ denotes bitwise XOR, and binary inner products are taken modulo two. We use $\mathbf{1}_E$ for the indicator of an event E, and Pr, $\mathbb{E}$, and Var for probability, expectation, and variance. Logarithms with subscript 2 are binary; ln is natural.

For a nonzero integer polynomial $P(\boldsymbol{z}) = \sum_\alpha c_\alpha \boldsymbol{z}^\alpha$, we define

$$\deg P = \max_{c_\alpha \neq 0} |\alpha|, \qquad \|P\|_1 = \sum_\alpha |c_\alpha|. \tag{1}$$

Here α is the multi-index of variable exponents and $|\alpha|$ is their sum. Unless stated otherwise, classical parameters are fixed coefficients when forming this norm.

For a subgroup of known odd prime order r, the size parameters are

$$n = \lceil \log_2 q \rceil, \qquad m_r = \lceil \log_2 r \rceil, \qquad N_{\mathrm{in}} = 2m_r \quad \text{for the exact domain } \mathbb{Z}_r^2. \tag{2}$$

Table 1 summarizes the main notation.

Table 1: Frequently used notation.

Symbols	Meaning and defining location
n, m_r, N_{in}	Field and subgroup bit lengths and the two live scalar registers' combined width; (2).
Enc, ℓ	Injective canonical Mumford encoding and its length, with $\ell = 2 + 4n$ for the stated degree-tag encoding; Section 2.3.
$(u, v), u_j^{(i)}, v_j^{(i)}$	Reduced Mumford polynomials and their coefficients; i indexes divisors and j coefficients; (5).
$\omega_r, \mathcal{F}_r$	Root of unity $\omega_r = e^{2\pi i/r}$ and the Fourier transform on $\mathbb{Z}_r$; Section 3.
$g, \widehat{g}, \mu_g$	Binary sign phase, its normalized Fourier coefficients, and its mean; (10).
χ_q, s_q	Quadratic character with $\chi_q(0) = 0$ and its binary completion with $s_q(0) = 1$; (19).
γ, g_γ	Uniform classical field shift and the shifted Legendre phase on divisor classes; (17).
W, C_{Tof}	Peak live logical qubits and ordinary three-qubit Toffoli count; Section 6.
$\varepsilon_{\mathrm{F}}, \varepsilon_{\mathrm{syn}}$	Distribution-error budgets for Fourier readout and rotation synthesis; Sections 3 and 5.3.
p_{run}	Probability of a classically verified logarithm in one run; (53).

2.2 Arithmetic over the base field

Multiplication, squaring, and inversion account for much of the cost of quantum finite-field arithmetic. We represent each element of $\mathbb{F}_q$ by its canonical residue in $\{0, \dots, q-1\}$ in an n-qubit register. The bounds below count peak live logical qubits, including source, target, and workspace registers, and all Toffoli gates needed for computation and cleanup.

Multiplication. We implement multiplication by the clean accumulation map

$$|x, y, z, 0\rangle \longmapsto |x, y, z + xy \bmod q, 0\rangle, \qquad 0 \le x, y, z < q.$$

The target is distinct from both sources and may contain any canonical residue. The sources are preserved and the workspace is cleared. A Horner scan reads the bits of y from most to least significant, interleaving controlled additions of x with modular doublings. Luo et al. [31, Appendix B] use $11n + O(1)$ Toffolis for each controlled addition and $6n + O(1)$ for each doubling. Their zero-target multiplier has n additions and $n - 1$ doublings, hence $17n^2 + O(n)$ Toffolis and $3n + O(1)$ logical qubits. For an arbitrary initial target z, we first replace it by $2^{-(n-1)}z \bmod q$; the Horner scan then returns

$$2^{n-1}\big(2^{-(n-1)}z \bmod q\big) + x\sum_{j=0}^{n-1} y_j 2^j \equiv z + xy \pmod q.$$

The $n - 1$ initial halvings cost the same as modular doublings. The complete accumulation uses $3n + O(1)$ logical qubits and

$$n\big(11n + O(1)\big) + 2(n-1)\big(6n + O(1)\big) = 23n^2 + O(n)$$

Toffoli gates. Fixed-modulus arithmetic uses Gidney's measurement-assisted adder [19]. Each block restores its borrowed source qubits and applies the prescribed measurement-dependent phase corrections.

For a Mersenne prime $q = 2^n - 1$, the relation $2^n \equiv 1 \pmod q$ implements doubling and halving by cyclic bit rotations. Zero is encoded by the all-zero word, and the all-one word q is excluded. Rotations preserve this encoding and require no Toffolis. With Luo's controlled adder, complete multiplication then uses $11n^2 + O(n)$ Toffolis and $3n + O(1)$ qubits. A coherent rotation implementation uses $4n + 4$ qubits and $9n^2 + n$ Toffolis; see (62). Cyclic shifts are also used in [12,8].

Alternatively, compute the $2n$-bit integer product and fold its high half into the low half. Writing $xy = c_0 + 2^n c_1$, reduce by

$$s = c_0 + c_1, \qquad xy \bmod q = s - q\mathbf{1}_{s \ge q}, \qquad 0 \le s < 2q,$$

where $\mathbf{1}_E$ is the indicator of E. One conditional subtraction suffices, including when $s = q$. After adding the reduced product to the target, reverse the fold and integer multiplication. This gives $7n + 3$ logical qubits and $8n^2 + 26n - 4$ Toffolis (Appendix C.1), using the Mersenne reduction of [26, Appendix A]. For $n \geq 25$, the gate bound is smaller than for coherent rotation multiplication, with $3n - 1$ additional qubits.

Squaring. The corresponding clean accumulation map for squaring is

$$|x, z, 0\rangle \longmapsto |x, z + x^2 \bmod q, 0\rangle, \qquad 0 \leq x, z < q.$$

The source is preserved, the target is a distinct canonical-residue register, and the workspace is cleared. One may use multiplication with equal operands. In Luo's Horner circuit, CNOTs first copy the source bits to a separate word and clear that copy after accumulation. The source, copy, and target occupy three n-qubit words. The resulting bounds are $3n + O(1)$ logical qubits and $23n^2 + O(n)$ Toffolis over a general prime field, or $11n^2 + O(n)$ Toffolis over a Mersenne prime field. Copying and clearing use $2n$ CNOTs and no Toffolis.

Dedicated squaring also exploits the repeated partial products in

$$x^2 = \sum_{j=0}^{n-1} x_j 2^{2j} + \sum_{0 \leq i < j < n} x_i x_j 2^{i+j+1}, \qquad x = \sum_{j=0}^{n-1} x_j 2^j.$$

The diagonal terms need no nonlinear bit products, and each off-diagonal pair occurs only once. This identity is used by the integer squarers in [13,16]. For coherent Mersenne rotations, a separate copied control allows the cited adder to be used without a second source word. The complete square-accumulate map costs $3n + 4$ logical qubits and $9n^2 + n$ Toffolis. Product-and-fold squaring costs $6n + 3$ qubits and $8n^2 + 22n - 4$ Toffolis, including reduction, accumulation, and reversal of both temporary computations. For $n \geq 25$, this has a smaller gate bound and requires $3n - 1$ more qubits. Section 5.1 and Appendix C.1 give the constructions and cleanup.

Inversion. Define $\mathrm{inv}_0(0) = 0$ and $\mathrm{inv}_0(x) = x^{-1}$ for $x \neq 0$. Normalization uses the clean inverse map

$$|x\rangle|z\rangle|0\rangle \longmapsto |x\rangle|z \oplus \mathrm{inv}_0(x)\rangle|0\rangle, \qquad 0 \leq x < q, \quad z \in \{0,1\}^n.$$

Here $\oplus$ is bitwise XOR, so the target may contain any n-bit word. The extended Euclidean algorithm updates the remainders of (x, q) and their Bézout coefficients. For $x \neq 0$, the terminal gcd is one, and the coefficient of x gives $x^{-1} \bmod q$. Luo et al. [31, Sections 3, 6.1–6.2] share registers between shrinking remainders and growing coefficients, with length registers locating the active data. Their forward circuit replaces x by x^{-1} and keeps the Euclidean state for reversal, using at most $2n + 6\lfloor \log_2 n \rfloor + 19$ logical qubits and $229n^2 + O(n \log_2 n)$ Toffolis.

To obtain the clean interface, add an n-qubit target and a flag $e = [x = 0]$. Replace a zero source by one, run the forward Euclidean circuit, and XOR the inverse into z when $e = 0$. The inverse logical circuit restores the source and clears the Euclidean state. Undoing the zero-input substitution then clears the flag. Forward and reverse calls have equal cost; the zero tests and controlled copy add $5n - 6$ Toffolis. The complete bounds are

$$3n + 6\lfloor \log_2 n \rfloor + 20 \quad \text{logical qubits}, \qquad 458n^2 + O(n \log_2 n) \quad \text{Toffoli gates}.$$

These bounds include source restoration, the arbitrary XOR target, and workspace cleanup. All measurement-assisted blocks, including those in the inverse logical circuit, retain their phase corrections.

Resource summary. Table 2 collects the complete interfaces. Multiplication and squaring add into a canonical modular target; inversion XORs into a binary target. Every row preserves the sources and clears the workspace. The Horner constructions and the Luo inverse adapter allow intermediate measurements, reset, and classical feedforward. The remaining constructions are coherent.

All three operations use linearly many qubits and quadratically many Toffolis, with different leading constants. The larger gate coefficient for clean inversion motivates the inversion-free polynomial map in Section 5. Mersenne bounds require the active modulus itself to be Mersenne; this property of the base field does not pass to auxiliary CRT primes. The finite sampler estimates in Section 6 use the explicit coherent constants from Section 5.1, with outer controls counted separately.

Table 2. Resource upper bounds for clean base-field arithmetic, including all input and target registers and workspace cleanup. The Horner bounds include compensation for a nonzero initial target. Luo squaring uses a source-word copy; coherent rotations use a copied control.

Operation	Implementation	Logical qubits	Toffoli gates
General odd prime q, with $n = \lceil \log_2 q \rceil$			
Multiplication	Luo Horner	$3n + O(1)$	$23n^2 + O(n)$
Squaring	Luo Horner	$3n + O(1)$	$23n^2 + O(n)$
Inversion	Luo adapter	$3n + 6\lfloor \log_2 n \rfloor + 20$	$458n^2 + O(n \log_2 n)$
Mersenne prime $q = 2^n - 1$			
Multiplication	Luo Horner	$3n + O(1)$	$11n^2 + O(n)$
Squaring	Luo Horner	$3n + O(1)$	$11n^2 + O(n)$
Multiplication	Coherent rotations	$4n + 4$	$9n^2 + n$
Squaring	Coherent rotations	$3n + 4$	$9n^2 + n$
Multiplication	Product and fold	$7n + 3$	$8n^2 + 26n - 4$
Squaring	Product and fold	$6n + 3$	$8n^2 + 22n - 4$

2.3 Curve model, encodings, and reference arithmetic

The curve C is the smooth projective model of

$$C: \quad y^2 = f(x),$$

where the supplied model has

$$f(x) = x^5 + c_3x^3 + c_2x^2 + c_1x + c_0 \in \mathbb{F}_q[x] \quad \text{square-free.} \tag{3}$$

The supplied model is a monic quintic with zero quartic coefficient; this assumption does not cover every genus-two curve over $\mathbb{F}_q$. For $q \neq 5$, a translation removes the quartic term from an existing monic quintic. In characteristic five, it must already vanish. Let ∞ be the unique point at infinity, and set

$$J = \mathrm{Jac}(C)(\mathbb{F}_q), \qquad G = \langle P \rangle \subseteq J, \qquad |G| = r, \tag{4}$$

where r is a known odd prime. Given $Q = [k]P$, we seek $k \in \mathbb{Z}_r$.

Every class has a unique reduced Mumford representation (u, v), with u monic,

$$\deg v < \deg u \leq 2, \qquad u \mid v^2 - f.$$

The identity is $(1, 0)$ [6]. For degree two, we write

$$u(x) = x^2 + u_1x + u_0, \qquad v(x) = v_1x + v_0. \tag{5}$$

Thus u_1, u_0, v_1, v_0 are field elements, whereas u, v are polynomials. For several divisors we write $u_j^{(i)}, v_j^{(i)}$: i identifies the divisor and j identifies the coefficient. The Weil bounds give $(\sqrt{q} - 1)^4 \leq |J| \leq (\sqrt{q} + 1)^4$. In particular, $m_r = 2n + O(1)$ when the cofactor $|J|/r$ is bounded. For smaller subgroups, we retain m_r as a separate size parameter.

Cantor's algorithm gives complete reference addition over $\mathbb{F}_q$ [6]. For reduced inputs $(u^{(1)}, v^{(1)})$ and $(u^{(2)}, v^{(2)})$, with superscripts indexing the divisors, compute the monic gcd

$$d = \gcd(u^{(1)}, u^{(2)}, v^{(1)} + v^{(2)})$$

and Bézout coefficients a, b, c such that

$$au^{(1)} + bu^{(2)} + c(v^{(1)} + v^{(2)}) = d.$$

The composition step is

$$\widetilde{u} = u^{(1)}u^{(2)}/d^2, \tag{6}$$

$$\widetilde{v} = \frac{au^{(1)}v^{(2)} + bu^{(2)}v^{(1)} + c(v^{(1)}v^{(2)} + f)}{d} \pmod{\widetilde{u}}. \tag{7}$$

All divisions are exact polynomial divisions. Reduce $\widetilde{v}$ modulo $\widetilde{u}$, then, while $\deg \widetilde{u} > 2$, replace the pair by

$$\left(\text{monic}\left(\frac{f - \widetilde{v}^2}{\widetilde{u}}\right), -\widetilde{v} \bmod \text{monic}\left(\frac{f - \widetilde{v}^2}{\widetilde{u}}\right)\right).$$

using the old pair on the right-hand side. The result is the unique reduced representative of the sum. The gcd step handles shared support, inverse pairs, and doubling, and the algorithm includes the identity $(1, 0)$. We use it as a correctness reference, with the clean inversion circuit of Section 2.2 supplying normalization inverses.

Gcds, degree decisions, and normalization over $\mathbb{F}_q$ do not directly yield a fixed polynomial circuit over $\mathbb{Z}$ of bounded degree. This is the distinction from the complete elliptic-curve polynomial formulas in [10, Section 2.2]. The residue interface in Section 5 must account for all reachable cases.

Canonical Mumford encoding. Fix an injective encoding $\text{Enc} : G \to \{0,1\}^\ell$. An explicit Mumford encoding uses $\ell = 2 + 4n$ bits: a two-bit degree tag d followed by four n-bit slots in the order below. Every coefficient is represented canonically in $\{0, \ldots, q-1\}$.

d	coefficient slots (u_1, u_0, v_1, v_0)	represented pair
2	(u_1, u_0, v_1, v_0)	$(x^2 + u_1x + u_0,\ v_1x + v_0)$
1	$(0, u_0, 0, v_0)$	$(x + u_0,\ v_0)$
0	$(0, 0, 0, 0)$	$(1, 0)$

The tag d determines the implicit monic coefficient; tag 3 is unused. The convention encodes every class of J, and Enc is its restriction to G. It fixes the representation used for complete divisor arithmetic below. Weighted coordinates are introduced in Section 4.

The compute–copy–uncompute construction [4] makes Cantor's algorithm [6] a clean reversible reference routine. For fixed genus two, fixed coefficient arrays and degree tags bound the number of field operations independently of n. The base-field primitives therefore give $O(n)$ qubits and $O(n^2)$ Toffoli gates. For a classically specified divisor B, a compute–swap–uncompute schedule also implements

$$|c\rangle|D\rangle|0\rangle \longmapsto |c\rangle|D + [c]B\rangle|0\rangle, \qquad c \in \{0,1\}. \tag{8}$$

Appendix A.1 describes the fixed-length Cantor implementation and cleanup. The finite sampler estimates instead use the weighted polynomial evaluator below, with exceptional inputs controlled by the masking argument in Section 5.3.

3 Fourier Sampling with a Compressed Phase

We use Shor's Fourier-sampling framework [40] with the output-compression methods of May and Schlieper [33] and CFS [11]. We count peak logical qubits, including both scalar registers, and allow intermediate measurements, reset, and classical feedforward. Classical records are excluded from this width. Arithmetic subroutines must realize their stated clean coherent maps even when their inputs are entangled with other registers. Single-qubit rotation synthesis has a separate distribution-error budget ε_{syn}. All gate bounds are analytical.

Our exact-domain analysis uses

$$F(a, b) = [a]P + [b]Q = [a + kb]P, \qquad (a, b) \in \mathbb{Z}_r^2. \tag{9}$$

Here (a, b) are scalar inputs and $(u, v) \in \mathbb{Z}_r^2$ are Fourier labels. The latter are distinct from the Mumford polynomials $u(x), v(x)$ in (5). Set $\omega_r = e^{2\pi i/r}$ and adopt the convention

$$\mathcal{F}_r^{-1}|a\rangle = \frac{1}{\sqrt{r}} \sum_{u \in \mathbb{Z}_r} \omega_r^{-au}|u\rangle.$$

Each m_r-qubit register has valid basis states $0, \ldots, r-1$. Extend the transform from this subspace to a unitary on the full binary register. Uniform preparation on the valid subspace and Fourier readout are counted separately: when r is not a power of two, Hadamards alone do not prepare the required state.

For $h : G \to \{0,1\}$, we define $g(D) = (-1)^{h(D)}$. A clean evaluation of $h \circ F$ is a reversible circuit satisfying

$$|x\rangle|z\rangle|0\rangle \longmapsto |x\rangle|z \oplus h(F(x))\rangle|0\rangle.$$

With target $|-\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$, the circuit induces the phase $g(F(x))$. All input-dependent workspace must be cleared before the Fourier stage, since retained workspace changes the off-diagonal entries of the input density operator.

We identify $g([t]P)$ with $g(t)$. We define its normalized Fourier coefficient and mean by

$$\widehat{g}(u) = \frac{1}{r} \sum_{t \in \mathbb{Z}_r} g(t) \omega_r^{-ut}, \qquad \mu_g = \frac{1}{r} \sum_{t \in \mathbb{Z}_r} g(t). \tag{10}$$

Parseval's identity gives $\sum_u |\widehat{g}(u)|^2 = 1$.

The standard Fourier-sampling and output-compression framework [40,33,11] gives the following exact-domain identity for a clean sign phase.

Theorem 3.1 (Exact-domain phase compression)**.** *Let r be an odd prime, let $G = \langle P \rangle$ have order r, and let $Q = [k]P$ for $k \in \mathbb{Z}_r$. Let $g : G \to \{-1, 1\}$, with $g([t]P)$ identified with $g(t)$ and with $\widehat{g}$ and μ_g as in* (10)*. Assume that the phase $g(a + kb)$ is evaluated cleanly, with all workspace reset before the Fourier stage. Starting from*

$$|\psi_0\rangle = \frac{1}{r} \sum_{a,b \in \mathbb{Z}_r} |a, b\rangle, \tag{11}$$

apply this phase and then measure after $\mathcal{F}_r^{-1} \otimes \mathcal{F}_r^{-1}$. The resulting distribution is

$$p_g(u, v) = \begin{cases} |\widehat{g}(u)|^2, & v = ku \pmod r, \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

The event $u \neq 0$ has probability

$$p_{\text{use}} = 1 - \mu_g^2 = \mathrm{Var}_{t \leftarrow \mathbb{Z}_r}(g(t)). \tag{13}$$

Every sample in this event yields $k = vu^{-1} \pmod r$.

Proof. After clean phase evaluation, the scalar state is $r^{-1} \sum_{a,b} g(a + kb)|a, b\rangle$. With our inverse-Fourier convention, its amplitude at (u, v) is

$$\begin{aligned} A(u, v) &= \frac{1}{r^2} \sum_{a,b \in \mathbb{Z}_r} g(a + kb) \omega_r^{-ua - vb} \\ &= \frac{1}{r^2} \left(\sum_{t \in \mathbb{Z}_r} g(t) \omega_r^{-ut} \right) \left(\sum_{b \in \mathbb{Z}_r} \omega_r^{(uk - v)b} \right). \end{aligned}$$

For each fixed b, the substitution $t = a + kb$ is a bijection. Character orthogonality makes the second factor r if $v = ku \pmod r$ and zero otherwise. Hence $A(u, ku) = \widehat{g}(u)$, proving (12); Parseval's identity gives total probability one.

The only supported label with $u = 0$ is $(0, 0)$, of probability $|\widehat{g}(0)|^2 = \mu_g^2$. The useful probability is therefore $1 - \mu_g^2$. Since $g(t)^2 = 1$, $\mathbb{E}_t g(t)^2 - (\mathbb{E}_t g(t))^2 = 1 - \mu_g^2$, which proves (13). Finally, r is prime, so every nonzero $u \in \mathbb{Z}_r$ is invertible and $v = ku$ yields $k = vu^{-1} \pmod r$. □

Corollary 3.2 (Variance criterion)**.** *Under the assumptions of Theorem 3.1, suppose each sign occurs on at least an η fraction of G, where $0 < \eta \leq 1/2$. Then $p_{\text{use}} \geq 4\eta(1 - \eta)$, and independent executions have expected waiting time at most $1/(4\eta(1 - \eta))$.*

Proof. The sign-frequency assumption gives $|\mu_g| \leq 1 - 2\eta$. Equation (13) yields the probability bound, and the waiting-time bound follows from independent repetition. □

Uniform preparation and terminal Fourier readout use standard methods [27,21,15]. Appendix C.4 gives the schedule underlying the following bounds.

Theorem 3.3 (Space and error of exact-domain sampling)**.** *Let $r \geq 3$ be an integer, write $m_r = \lceil \log_2 r \rceil$, and let $0 < \varepsilon_{\mathrm{F}} < 1$ and $J_{\mathrm{prep}} \geq 1$, with J_{prep} an integer. Set*

$$t = m_r + \lceil \log_2(4 + 2/\varepsilon_{\mathrm{F}}) \rceil . \tag{14}$$

Allow intermediate measurements, reset, and classical feedforward. The schedule in Appendix C.4 prepares uniform inputs on $\mathbb{Z}_r^2$ and performs terminal Fourier readout with the following bounds, excluding the phase oracle and rotation synthesis.

The peak allocation for preparation and readout is at most $3m_r + 5$ logical qubits, including both m_r-qubit scalar registers and all quantum workspace; classical records are excluded. With at most J_{prep} trials per register, successful preparation gives exactly the state in (11), and the abort probability is at most $2^{1-J_{\mathrm{prep}}}$. Preparation costs fewer than $16m_r - 8$ Toffolis in expectation and at most $4J_{\mathrm{prep}}(2m_r - 1)$ under the trial cap.

Fourier readout uses at most $2(9m_r + 1)t$ Toffolis and $t(t-1)$ individual classically controlled dyadic rotations across both registers, including all semiclassical feedback. Combining the angles in each round reduces the rotation count to at most $2(t-1)$. For any joint state supported on $\mathbb{Z}_r^2$, the decoded labels differ in total variation distance by at most ε_{F} from ideal measurement after $(\mathcal{F}_r^{-1})^{\otimes 2}$, before rotation-synthesis errors. Conditional on successful preparation and clean phase evaluation, this also bounds the sampler's error relative to (12).

Measuring only the predicate $x < r$ leaves an exact uniform state on acceptance, with success probability $r/2^{m_r} > 1/2$ per trial. Readout estimates the cyclic-shift eigenphase with one recycled control and classical feedback, then rounds it to a Fourier label [27,21]. The chosen t bounds each incorrect-label probability by $\varepsilon_{\mathrm{F}}/2$, hence the joint error by ε_{F}. Standard modular shifts [32] give the gate count. Appendix C.4 records the allocation and error estimate. Rotation synthesis contributes the separate error budget $\varepsilon_{\mathrm{syn}}$.

4 Output Compression Using the Legendre Symbol

We evaluate the shifted Legendre phase in Lange's weighted Mumford coordinates [29], fixing the second scaling parameter to one. The nonleading coefficients of u have weight two and those of v have weight three. With scale coordinate Z of weight one, the degree-two chart is $Z \neq 0$ in $\mathbb{P}(2,2,3,3,1)$, with coordinates $[U_1 : U_0 : V_1 : V_0 : Z]$ and affine decoding

$$u_1 = \frac{U_1}{Z^2}, \qquad u_0 = \frac{U_0}{Z^2}, \qquad v_1 = \frac{V_1}{Z^3}, \qquad v_0 = \frac{V_0}{Z^3}. \tag{15}$$

Thus the equivalence relation is

$$(U_1, U_0, V_1, V_0, Z) \sim (\lambda^2 U_1, \lambda^2 U_0, \lambda^3 V_1, \lambda^3 V_0, \lambda Z), \qquad \lambda \in \mathbb{F}_q^{\times}. \tag{16}$$

Every affine degree-two pair has representative $[u_1 : u_0 : v_1 : v_0 : 1]$. Conversely, rescaling by Z^{-1} gives this unique representative. Thus (15) identifies the valid part of the chart with the degree-two Mumford chart and allows the shifted quadratic character to be evaluated without division by Z. The chart requires $Z \neq 0$ and does not define a complete projective group model; polynomial evaluation at $Z = 0$ need not represent a group operation.

4.1 Field-shifted Legendre phases

The symbol χ_q denotes the quadratic character of $\mathbb{F}_q$, with $\chi_q(0) = 0$. We sample a classical seed $\gamma \in \mathbb{F}_q$ uniformly and define the phase by

$$g_\gamma(D) = \begin{cases} -1, & \deg u = 2 \text{ and } \chi_q(u_0 + \gamma) = -1, \\ +1, & \text{otherwise.} \end{cases} \tag{17}$$

On the weighted chart (15), the character satisfies

$$\chi_q(u_0 + \gamma) = \chi_q(U_0 + \gamma Z^2). \tag{18}$$

Indeed, $U_0 + \gamma Z^2 = (u_0 + \gamma)Z^2$, and the rescaling in (16) multiplies the argument by λ^2 while leaving the classical seed γ fixed. The shift therefore requires no division by Z. It acts on a field coefficient rather than translating the group element. The zero convention makes g_γ a binary phase on every class. To bound its useful-sample probability, we combine shifted-character correlations with the sizes of the u_0 fibers.

Lemma 4.1 (Shifted character correlations)**.** *Let q be an odd prime and let χ_q be the quadratic character with $\chi_q(0) = 0$. Define the completed binary character by*

$$s_q(x) = \chi_q(x) + \mathbf{1}_{x=0}. \tag{19}$$

Then s_q takes values in $\{-1, 1\}$, with $s_q(0) = 1$, and $s_q(x) = -1$ precisely for nonsquares. If γ is uniform in $\mathbb{F}_q$, then for every $a \in \mathbb{F}_q$,

$$\mathbb{E}_\gamma s_q(a + \gamma) = \frac{1}{q}.$$

For distinct $a, b \in \mathbb{F}_q$,

$$\mathbb{E}_\gamma[s_q(a + \gamma)s_q(b + \gamma)] = \frac{-1 + \chi_q(a - b) + \chi_q(b - a)}{q} \le \frac{1}{q}. \tag{20}$$

For $a = b$ the correlation equals one.

Proof. The value statements follow from the definition. Since $\sum_x \chi_q(x) = 0$, we have $\sum_x s_q(x) = 1$, giving the mean. For $a \neq b$, the standard identity is $\sum_\gamma \chi_q((\gamma + a)(\gamma + b)) = -1$ [30, Chapter 5]. For a direct proof, set $t = (2\gamma + a + b)/(a - b)$. Then $(\gamma + a)(\gamma + b) = (a - b)^2(t^2 - 1)/4$ with a nonzero square prefactor, so the sum becomes $\sum_t \chi_q(t^2 - 1)$. The equation $z^2 = t^2 - 1$ has $q + \sum_t \chi_q(t^2 - 1)$ solutions. It also has $q - 1$ solutions, since $(t - z)(t + z) = 1$ gives one for each nonzero $t - z$ when q is odd. On expanding (19), the cross terms give $\chi_q(a - b) + \chi_q(b - a)$ and the product of the zero indicators vanishes. Dividing by q proves (20). Each character value is at most one, giving the inequality. For $a = b$, use $s_q(x)^2 = 1$. □

For a class of degree two, (17) therefore gives $g_\gamma(D) = s_q(u_0 + \gamma)$. Classes of degree below two retain phase $+1$.

Lemma 4.2 (Counts of reduced Mumford classes)**.** *Let $G \subseteq J$ be a subgroup of order r. Let e denote the number of classes in G with $\deg u < 2$, and define, for $a \in \mathbb{F}_q$,*

$$m_a = \#\{D \in G : \deg u = 2,\ u_0 = a\}.$$

Then

$$\sum_a m_a = r - e, \qquad e \le 2q + 1, \qquad m_a \le 4q \quad (a \in \mathbb{F}_q). \tag{21}$$

Proof. The reduced Mumford representation [6] partitions G into lower-degree classes and degree-two fibers indexed by u_0, so $\sum_a m_a = r - e$. Lower-degree classes consist of the identity and at most two affine points for each $x \in \mathbb{F}_q$, giving $e \le 2q + 1$.

For $u_0 = a$, there are q monic polynomials $u = x^2 + u_1 x + a$. If u is square-free, $\mathbb{F}_q[x]/(u)$ is a quadratic field or a product of two copies of $\mathbb{F}_q$; in either case $v^2 = f \pmod u$ has at most four solutions. If $u = (x - t)^2$, a solution must satisfy $v(t)^2 = f(t)$ and $2v(t)v'(t) = f'(t)$. When $f(t) = 0$, square-freeness of f gives $f'(t) \neq 0$, so no solution exists. Otherwise $v(t)$ has at most two choices, each determining $v'(t)$. Thus every u has at most four reduced pairs, and $m_a \le 4q$. □

Theorem 4.3 (Shift-averaged useful probability)**.** *We assume that G has odd prime order r. The exact-domain sampling algorithm uses (17) with a fresh uniform γ in each run. For the counts e, m_a in Lemma 4.2, the mean useful-sample probability satisfies*

$$\mathbb{E}_\gamma\big[1 - \mu^2_{g_\gamma}\big] \ge \left(1 - \frac{1}{q}\right)\left(1 - \frac{e^2 + \sum_a m_a^2}{r^2}\right) =: V_G. \tag{22}$$

In particular, if $r = \Theta(q^2)$, then the mean useful-sample probability is $1 - O(q^{-1})$. Under this condition, the expected number of independent runs is $1 + O(q^{-1})$.

Proof. The phase mean is

$$\mu_{g_\gamma} = \frac{e + \sum_a m_a s_q(a+\gamma)}{r}.$$

Expanding the square and applying Lemma 4.1, with $\sum_a m_a = r - e$, gives

$$\begin{aligned}\mathbb{E}_\gamma \mu_{g_\gamma}^2 &\le \frac{e^2 + \sum_a m_a^2}{r^2} + \frac{r^2 - e^2 - \sum_a m_a^2}{qr^2} \\ &= \frac{1}{q} + \left(1 - \frac{1}{q}\right) \frac{e^2 + \sum_a m_a^2}{r^2}.\end{aligned}$$

By Theorem 3.1, the useful probability is the variance in (13), which proves (22). Also, $e^2 + \sum_a m_a^2 \le (e + \sum_a m_a)^2 = r^2$, so $V_G \ge 0$. Lemma 4.2 gives $e = O(q)$ and $\sum_a m_a^2 = O(qr)$. For $r = \Theta(q^2)$, these imply the stated asymptotic probability. Independent resampling then gives the geometric waiting-time bound. □

By Lemma 4.2, $\sum_a m_a^2 \le 4q(r-e)$, so (22) also gives

$$\mathbb{E}_\gamma\left[1 - \mu_{g_\gamma}^2\right] \ge \max\left\{0, \left(1 - \frac{1}{q}\right)\left(1 - \frac{e^2 + 4q(r-e)}{r^2}\right)\right\}. \tag{23}$$

This estimate depends only on q, r, e. Substituting $e \le 2q + 1$ gives a bound without subgroup enumeration:

$$\mathbb{E}_\gamma\left[1 - \mu_{g_\gamma}^2\right] \ge \max\left\{0, \left(1 - \frac{1}{q}\right)\left(1 - \frac{(2q+1)^2}{r^2} - \frac{4q}{r}\right)\right\}.$$

For $q \equiv 3 \pmod 4$, the exact shift-averaged useful-sample probability is

$$\mathbb{E}_\gamma\left[1 - \mu_{g_\gamma}^2\right] = 1 - \frac{\left(e + \frac{r-e}{q}\right)^2 + (1 + 1/q)\sum_a \left(m_a - \frac{r-e}{q}\right)^2}{r^2}. \tag{24}$$

For $a \ne b$, $\chi_q(a-b) + \chi_q(b-a) = 0$, so (20) gives off-diagonal correlation $-1/q$. Consequently,

$$r^2 \mathbb{E}_\gamma[\mu_{g_\gamma}^2] = e^2 + \frac{2e(r-e)}{q} + \left(1 + \frac{1}{q}\right)\sum_a m_a^2 - \frac{(r-e)^2}{q}.$$

Combining $\sum_a (m_a - (r-e)/q)^2 = \sum_a m_a^2 - (r-e)^2/q$ with (13) proves the identity. It applies in particular to $q = 2^{127} - 1$ and separates mean phase bias from variation in fiber sizes. Evaluating it requires the actual fiber counts; (23) needs only q, r, e.

An exact character implementation may compute an integer X_γ satisfying

$$X_\gamma \bmod q = \begin{cases} U_0 + \gamma Z^2, & \deg u = 2 \text{ in a valid chart}, \\ 0, & \deg u < 2. \end{cases} \tag{25}$$

The seed is classical, with a canonical integer lift of at most n bits. The arithmetic interface must include this terminal polynomial and exceptional-case masking. Section 5 constructs a total polynomial and bounds its disagreement with the ideal phase.

4.2 A phase-only terminal node

The terminal node is used only for the shifted Legendre phase, so it need not compute all five coordinates of the sum. Internal nodes still require the full output. We retain the monic degree-five model with zero x^4

coefficient from Section 2. Write the inputs as $\boldsymbol{U}^{(i)} = (U_1^{(i)}, U_0^{(i)}, V_1^{(i)}, V_0^{(i)}, Z_i)$, $i = 1, 2$, and, for $j = 0, 1$, define the scaled differences and sum by

$$\begin{aligned} \Delta U_j &= U_j^{(2)} Z_1^2 - U_j^{(1)} Z_2^2, \\ \Delta V_j &= V_j^{(2)} Z_1^3 - V_j^{(1)} Z_2^3, \\ \Sigma U_1 &= U_1^{(2)} Z_1^2 + U_1^{(1)} Z_2^2. \end{aligned} \tag{26}$$

Here Δ and Σ denote two-input differences and sums. For $Z_1 = Z_2 = 1$, they become $\Delta U_j = u_j^{(2)} - u_j^{(1)}$ and $\Sigma U_1 = u_1^{(2)} + u_1^{(1)}$. The arithmetic listings use `dU`, `dV`, and `sU` for these quantities. Define also

$$\begin{aligned} \mathcal{C}_* &= U_1^{(1)} U_0^{(2)} - U_0^{(1)} U_1^{(2)}, \\ \mathcal{R}_* &= (\Delta U_0)^2 - \mathcal{C}_* \Delta U_1, \\ \mathcal{E}_* &= \Delta V_1 \Delta U_0 - \Delta V_0 \Delta U_1, \\ \mathcal{L}_* &= \Delta U_0 \Delta V_0 - \mathcal{C}_* \Delta V_1. \end{aligned} \tag{27}$$

Set $\mathcal{K} = Z_1 Z_2$ and $\mathsf{A}_{\mathrm{sc}} = \mathcal{K}^2$. The subscript distinguishes this scale factor from the addition map $\mathcal{A}$ in (36). The terminal polynomials are

$$\begin{aligned} N_* &= \mathsf{A}_{\mathrm{sc}} \mathcal{L}_*^2 + 2U_1^{(1)} Z_2^2 \mathcal{L}_* \mathcal{E}_* + (U_1^{(1)} U_1^{(2)} - \Delta U_0) \mathcal{E}_*^2 \\ &\quad - 2V_1^{(1)} Z_2^3 \mathcal{R}_* \mathcal{E}_* + \Sigma U_1 \mathsf{A}_{\mathrm{sc}} \mathcal{R}_*^2, \\ D_* &= \mathsf{A}_{\mathrm{sc}} \mathcal{E}_*^2, \qquad \Psi_\gamma = N_* + \gamma D_*. \end{aligned} \tag{28}$$

On the generic branch, N_* and D_* are the numerator and denominator of $u_0^{(3)}$, and Ψ_γ is the shifted phase argument. The group-valued map F in (9) has a separate role. All expressions are integer polynomials, so the terminal program uses no division. Their interpretation as a group operation requires the nonvanishing condition below.

Proposition 4.4 (Square-factor elimination at a generic terminal node)**.** *We assume that the degree-two inputs are valid and satisfy $Z_1 Z_2 \mathcal{R}_* \mathcal{E}_* \neq 0$. The symbol $u_0^{(3)}$ denotes the constant coefficient of the monic quadratic representing their sum. The following identities hold:*

$$u_0^{(3)} = N_*/D_*, \qquad s_q(\Psi_\gamma) = s_q(u_0^{(3)} + \gamma) \quad (\gamma \in \mathbb{F}_q). \tag{29}$$

The identity includes $u_0^{(3)} + \gamma = 0$. In particular, the phase is unchanged by independently rescaling either weighted input.

Proof. The interpolation argument also proves the compact addition map (36). Begin with normalized inputs

$$(u^{(1)}, v^{(1)}) = (x^2 + ax + c, sx + u), \qquad (u^{(2)}, v^{(2)}) = (x^2 + bx + d, tx + v).$$

The letters in this display denote field elements. We define

$$\begin{aligned} R &= (d - c)^2 - (ad - cb)(b - a), \\ E &= (t - s)(d - c) - (v - u)(b - a), \\ L &= (d - c)(v - u) - (ad - cb)(t - s), \\ B_1 &= sR - aL - (ab + c)E, \qquad B_0 = uR - cL - bcE. \end{aligned}$$

Here $R = \mathrm{Res}_x(u^{(1)}, u^{(2)})$. For $R \neq 0$, the quadratics are coprime, so a unique polynomial of degree at most three has remainders $v^{(i)}$ modulo $u^{(i)}$. It is

$$\ell(x) = \frac{-Ex^3 - [L + (a + b)E]x^2 + B_1 x + B_0}{R}.$$

Its numerator reduces to $Rv^{(i)}$ modulo each input quadratic. This verifies interpolation without choosing roots or extending the base field. The assumption $E \neq 0$ makes the interpolant cubic.

Validity of the inputs implies $u^{(1)}u^{(2)} \mid \ell^2 - f$. The quotient is quadratic with leading coefficient E^2/R^2; write its monic form as $x^2 + \alpha x + \beta$. Since f is monic with zero x^4 coefficient, comparing the x^5 and x^4 coefficients in

$$R^2(\ell^2 - f) = E^2 u^{(1)} u^{(2)} (x^2 + \alpha x + \beta)$$

gives

$$\begin{aligned} A &= 2LE + (a+b)E^2 - R^2, \qquad T = LE - R^2, \\ N &= L^2 + 2aLE + (ab - d + c)E^2 - 2sRE + (a+b)R^2, \\ \alpha &= A/E^2, \qquad \beta = N/E^2. \end{aligned}$$

For normalized inputs, (28) and (35) give these polynomials. Cantor reduction yields $v^{(3)} = -\ell \bmod (x^2 + \alpha x + \beta)$. Using $x^2 \equiv -\alpha x - \beta$ and $x^3 \equiv (\alpha^2 - \beta)x + \alpha\beta$, we obtain

$$v_1^{(3)} = \frac{AT - E^2(B_1 E + N)}{RE^3}, \qquad v_0^{(3)} = \frac{NT - B_0 E^3}{RE^3}.$$

Choosing output scale RE and clearing the decoding denominators gives the normalized compact five-coordinate output. Thus the formulas perform composition and reduction on the generic branch, including when the input supports do not split over the base field.

For arbitrary nonzero scales, normalize the inputs by Z_1^{-1} and Z_2^{-1}. Every monomial of N_* and D_* has weighted bidegree $(12, 12)$, so their quotient is unchanged. The five output polynomials have bidegrees $(20, 20), (20, 20), (30, 30), (30, 30), (10, 10)$, respectively. Their common output rescaling cancels in (15), proving the compact addition formula and $u_0^{(3)} = N_*/D_*$ for the original weighted inputs. Only the assumed nonzero scales are needed; no further generic condition is introduced.

Finally, $D_* = (Z_1 Z_2 \mathcal{E}_*)^2$ is a nonzero square. Hence $\Psi_\gamma = D_*(u_0^{(3)} + \gamma)$ has the same completed quadratic character as $u_0^{(3)} + \gamma$. If the latter vanishes, both completed characters are one. This proves the phase identity, including its zero case and invariance under independent rescaling of either input. □

Using (1), and treating γ as a fixed coefficient, propagation through the unexpanded program gives $\deg N_* \leq 19$, $\deg D_* \leq 18$, $\|N_*\|_1 \leq 640$, and $\|D_*\|_1 \leq 64$. Exact sparse expansion yields $\|N_*\|_1 = 232$ and $\deg N_* = 19$. The supplied certificate has 115 numerator monomials and 34 denominator monomials and verifies (59) over $\mathbb{Z}$, rather than only at sampled field points. Consequently a classical integer lift of γ gives

$$\deg \Psi_\gamma \leq 19, \qquad \|\Psi_\gamma\|_1 \leq 232 + 64|\gamma|. \tag{30}$$

If γ is instead treated as an additional input variable, the polynomial has degree 19 and coefficient norm 296. The terminal polynomial can therefore be reduced at auxiliary moduli. This statement does not apply to the generic-branch test, and the polynomial alone gives no formulas for nongeneric inputs.

We convert the reconstructed value to a phase with the binary Jacobi algorithm of Shallit and Sorenson [39], used for quantum character evaluation by CFS [11, Sections 4.2–4.3]. Fix $2n+1$ rounds and use the completion $s_q(0) = 1$ from (19). The argument Y is one working operand. Recording the branches, applying the sign phase, and reversing the computation gives

$$|Y\rangle|0\rangle \longmapsto s_q(Y)|Y\rangle|0\rangle \qquad (0 \leq Y < q). \tag{31}$$

The input and workspace are restored. The ripple and dirty-ancilla decompositions in Appendix C.4 give total logical width, including the n-qubit input,

$$W_\chi(n) \leq 6n + 12. \tag{32}$$

Each forward macro costs at most $36n$ Toffolis. The $2n + 1$ forward macros and their full reversal therefore give

$$C_\chi(n) \leq 72n(2n + 1). \tag{33}$$

This bound counts one clean character phase, including cleanup. At $n = 127$, it is 2,331,720 Toffolis. The sampler uses these allocations; Appendix C.4 gives the fixed schedule and handles the zero input.

5 Addition of Divisor Classes Using Residues in a Binary Tree

Following CFS [10, Section 5], we compute the final character argument by evaluating the tree modulo small auxiliary primes and reconstructing the result with a guarded CRT calculation. The internal node is a compact weighted genus-two addition formula. Public masks control its exceptional inputs on the exact uniform sampling domain, giving a sampler with bounded error. Appendix C.3 gives the clean residue-tree implementation.

The two scalar inputs occupy $N_{\mathrm{in}} = 2m_r$ qubits, as in (2). Choose a positive integer window width w and put $s_w = \lceil m_r/w \rceil$. Write $a = \sum_{j=0}^{s_w-1} a^{(j)} 2^{jw}$ and $b = \sum_{j=0}^{s_w-1} b^{(j)} 2^{jw}$, with $a^{(j)} = \lfloor a/2^{jw} \rfloor \bmod 2^w$ and similarly for $b^{(j)}$. The top window may have fewer than w bits. Order the P-windows before the Q-windows and pad the $L_0 = 2s_w$ summands to $L = 2^{\lceil \log_2 L_0 \rceil} = 2^h$ leaves. For $x = (a, b)$, define

$$A_i(x) = \begin{cases} [a^{(i)} 2^{iw}]P, & 0 \le i < s_w, \\ [b^{(i-s_w)} 2^{(i-s_w)w}]Q, & s_w \le i < 2s_w, \\ 0_G, & 2s_w \le i < L. \end{cases} \tag{34}$$

Here 0_G is the identity, and $\sum_i A_i(x) = [a]P + [b]Q = F(x)$. Each nonpadding leaf is selected from a public table by one window; padding leaves are fixed. The tables have at most $L2^w$ entries in total. Masks are applied also to zero windows and padding leaves. All tables and masks are prepared classically. Both m_r-bit scalar registers remain live throughout the quantum arithmetic.

5.1 Integer height and tree evaluation

We bound the integer height of the compact weighted addition map and the workspace needed to evaluate its residue tree. The height bound applies to every integer tuple produced by the tree, including exceptional tuples, and fixes the CRT modulus product in Section 5.2. A measurement-assisted schedule then bounds the total workspace and number of local moves. The local reversible implementation and its operation counts are given in Appendix C.2.

Weighted addition map. The recovery rule (15) allows us to reduce the degree of a denominator-cleared internal node. Set $\mathcal{K} = Z_1 Z_2$ and retain $\mathcal{R}_*, \mathcal{E}_*, \mathcal{L}_*$ from (27) and N_* from (28). In particular, $\Sigma U_1 = U_1^{(2)} Z_1^2 + U_1^{(1)} Z_2^2$ is the scaled sum in (26). Define the local polynomials

$$\begin{aligned} \mathcal{J}_* &= \mathcal{K}^2 \mathcal{L}_* + \Sigma U_1 \mathcal{E}_*, \\ A_* &= 2\mathcal{K}^2 \mathcal{L}_* \mathcal{E}_* + \Sigma U_1 \mathcal{E}_*^2 - \mathcal{K}^4 \mathcal{R}_*^2, \\ T_* &= \mathcal{L}_* \mathcal{E}_* - \mathcal{K}^2 \mathcal{R}_*^2, \\ B_1 &= V_1^{(1)} Z_2^3 \mathcal{R}_* - U_1^{(1)} Z_2^2 \mathcal{L}_* - (U_1^{(1)} U_1^{(2)} + U_0^{(1)} Z_2^2) \mathcal{E}_*, \\ B_0 &= V_0^{(1)} Z_2^3 \mathcal{R}_* - U_0^{(1)} Z_2^2 \mathcal{L}_* - U_0^{(1)} U_1^{(2)} \mathcal{E}_*. \end{aligned} \tag{35}$$

The interpolation polynomial $\mathcal{J}_*$ is distinct from the Jacobian $J = \mathrm{Jac}(C)(\mathbb{F}_q)$. Likewise, the numerator N_* is distinct from the scalar-input width N_{in}. The compact polynomial map $\mathcal{A} : \mathbb{Z}^5 \times \mathbb{Z}^5 \to \mathbb{Z}^5$ has output

$$\begin{aligned} U_1' &= A_* \mathcal{R}_*^2, & U_0' &= N_* \mathcal{R}_*^2, \\ V_1' &= \mathcal{R}_*^2 \big(A_* T_* - \mathcal{E}_*^2 (B_1 \mathcal{E}_* + N_*) \big), & & \\ V_0' &= \mathcal{R}_*^2 \big(N_* T_* - B_0 \mathcal{E}_*^3 \big), & Z' &= \mathcal{K} \mathcal{R}_* \mathcal{E}_*. \end{aligned} \tag{36}$$

The formula computes the five integer polynomials in (36), using the same polynomial map at every auxiliary modulus for CRT reconstruction.

Integer height bounds. The five outputs in (36) have ordinary total degrees bounded by $(32, 31, 48, 47, 15)$. For repeated additions, weighted bihomogeneity gives a sharper estimate. Under separate input rescalings with weights $(2, 2, 3, 3, 1)$, the bidegrees are

$$(20, 20),\ (20, 20),\ (30, 30),\ (30, 30),\ (10, 10).$$

The output scale is therefore multiplied by $\lambda_1^{10}\lambda_2^{10}$, consistently with the recovery weights. A direct coefficient-norm certificate gives

$$(\|U_1'\|_1, \|U_0'\|_1, \|V_1'\|_1, \|V_0'\|_1, \|Z'\|_1) \le (20480, 40960, 6291456, 6029312, 64). \tag{37}$$

For an integer tuple $\boldsymbol{U} = (U_1, U_0, V_1, V_0, Z)$, we define

$$\mathcal{H}(\boldsymbol{U}) = \max\{1, |U_1|^{1/2}, |U_0|^{1/2}, |V_1|^{1/3}, |V_0|^{1/3}, |Z|\}. \tag{38}$$

This height measures the chosen integer tuple and need not be projectively invariant. Writing $H_U = \mathcal{H}(\boldsymbol{U})$, we have $|U_j| \le H_U^2$, $|V_j| \le H_U^3$, and $|Z| \le H_U$. We use $\mathcal{H}$ for this weighted height and H in (41) for an integer bit bound. Bihomogeneity and (37) imply the following estimate, also for exceptional tuples that represent no group element:

$$\mathcal{H}(\mathcal{A}(\boldsymbol{U}, \boldsymbol{V})) \le 2^8 \mathcal{H}(\boldsymbol{U})^{10} \mathcal{H}(\boldsymbol{V})^{10}. \tag{39}$$

To check the constant, put $H_U = \mathcal{H}(\boldsymbol{U})$ and $H_V = \mathcal{H}(\boldsymbol{V})$. A coordinate of weight $\omega \in \{2, 3, 1\}$ has bidegree $(10\omega, 10\omega)$. With coefficient norm c_ω, its absolute value is at most $c_\omega H_U^{10\omega} H_V^{10\omega}$; taking the ω-th root gives $c_\omega^{1/\omega} H_U^{10} H_V^{10}$. The five constants in (37) satisfy

$$\max\{20480^{1/2}, 40960^{1/2}, 6291456^{1/3}, 6029312^{1/3}, 64\} < 2^8.$$

Taking the maximum of these bounds and 1 proves (39). Each child height has exponent 10, whose sum gives the balanced-tree growth factor 20 after taking logarithms.

Normalized leaves, including every fixed dummy leaf, have $Z = 1$ and four coordinates in $[0, q)$. Since $q < 2^n$, their logarithmic weighted height is at most $n/2$. Number tree heights from $j = 0$ at the leaves and bound twice the logarithmic weighted height by

$$t_0 = n, \qquad t_j = 20t_{j-1} + 16 = n20^j + 16\frac{20^j - 1}{19}. \tag{40}$$

If both children satisfy $\log_2 \mathcal{H} \le t_{j-1}/2$, then (39) gives $\log_2 \mathcal{H}_{\text{parent}} \le 8 + 10t_{j-1} = t_j/2$, proving the bound inductively. The terminal $\Psi_\gamma = N_* + \gamma D_*$ has weighted bidegree $(12, 12)$ and coefficient norm at most $232 + 64|\gamma| < 2^{n+9}$ for $0 \le \gamma < q$. Thus a tree with 2^h leaves and a direct terminal phase polynomial has the signed-integer bound

$$|\Psi_\gamma| < 2^H, \qquad H = n + 10 + 12t_{h-1} \quad (h \ge 1). \tag{41}$$

In particular, $H = O(nL^{\log_2 20})$ for $L = 2^h$. The index $h - 1$ refers to the two subtrees below the root. The terminal bidegree $(12, 12)$ gives $\log_2 |\Psi_\gamma| < n + 9 + 12t_{h-1}$ for nonzero output; the extra bit in H provides a reserve. The smaller terminal does not change the internal growth exponent $\log_2 20$.

The height bound determines the required CRT modulus product. Let b bound every auxiliary residue-word length. Write $C_M(b), C_S(b), C_\gamma(b)$ for the Toffoli costs of clean modular multiply-accumulate, square-accumulate, and classical-constant multiply-accumulate. Each map preserves its sources, accepts an arbitrary canonical target, and clears its scratch. The scratch allowance $A_{\text{prim}}(b)$ excludes source and target words; Appendix C.1 gives $A_{\text{prim}}(b) \le b + 4$.

Proposition C.2 gives the local costs in Table 3. All three operations use at most eight local words, excluding inputs, outputs, and primitive scratch; ghost correction also reserves a zero scalar word. Write G_* for a common Toffoli bound on leaf lookup, local evaluation, erasure, and phase correction. Its finite value is given in Section 6.1.

Table 3. Clean local evaluation costs, including computation and cleanup. The two five-coordinate inputs are preserved.

Operation	Toffoli upper bound
Five-coordinate node	$274C_M + 262C_S$
Scalar terminal	$130C_M + 138C_S + C_\gamma$
Scalar correction of a five-coordinate ghost	$638C_M + 776C_S$

Measurement-assisted tree evaluation. Measured erasure uses the phase-corrected uncomputation of CFS and Kornerup et al. [11,28]. Suppose a clean computation produces a d-bit word $f(x)$ while retaining its logical input x. Hadamard-basis measurement gives a uniform classical outcome $\mu \in \{0,1\}^d$ and applies phase $(-1)^{\langle \mu, f(x)\rangle}$ to the retained state. Correcting that phase coherently allows the measured word to be reset while preserving the logical state and its entanglement with other registers.

We write the state before measurement as

$$\sum_x |x\rangle|\eta_x\rangle|f(x)\rangle, \qquad \sum_x \|\eta_x\|^2 = 1,$$

where the possibly unnormalized vectors η_x include all other registers. The unnormalized state conditioned on μ is

$$2^{-d/2} \sum_x (-1)^{\langle \mu, f(x)\rangle} |x\rangle|\eta_x\rangle. \tag{42}$$

Orthogonality of the x register gives probability 2^{-d} for each μ. Recompute $f(x)$ in clean workspace, apply Z at the positions selected by μ, and uncompute. This supplies the same sign, cancelling the measurement phase for every x and restoring the workspace. The argument holds for arbitrary logical states.

The correction inputs must be available, and their recomputation is included in the pebbling cost. Outcomes are uniform even conditional on earlier measurements, so the full classical transcript reveals no logical input. After correction, the logical channel is the required unitary. A fresh invocation on an arbitrary XOR target erases the residue without reversing a measurement.

For a five-coordinate ghost, Proposition C.2 provides phase correction from the ten input words, using eight local words and one reserved scalar correction word. Reconstructing the inputs is charged separately by the tree schedule.

The measurement-assisted schedule in Appendix C.3, following [11,28], uses at most $8(h+1)L$ moves and $h+2$ vector pebbles. Each vector pebble stores five b-bit residue words; the root uses a scalar terminal. To realize a clean XOR oracle on an arbitrary external b-bit target, copy the computed scalar root by CNOTs and reverse its local evaluation. This fused terminal action costs twice the terminal evaluation. The schedule then removes every ghost and corrects the recorded phases. Uniformity of the Hadamard-basis outcomes makes the transcript independent of the input, yielding the desired XOR unitary. A fresh invocation clears its output without inverting a measurement. Including the external target, local evaluator, and scalar correction word, the allocation is

$$[5(h+2)+10]b + A_{\mathrm{prim}}(b) \tag{43}$$

qubits beyond the two scalar registers. This conservatively reserves $h+2$ full vector pebbles throughout, and the move count includes terminal evaluation and reversal.

5.2 Streaming reconstruction for the residue sampler

The residue-tree oracle evaluates one integer polynomial at successive auxiliary primes. We recover its value modulo q by explicit Chinese remaindering [5], as in CFS [11, Section 2.3], using a short fixed-point sum and an n-bit modular accumulator Y. Assume $X \in \mathbb{Z}$ has certified bound $|X| \le B$, with $B \ge 1$. For the masked sampler, X is the value of Ψ_γ in (28); take $B = 2^H$ from (41). Choose distinct auxiliary primes $p_1, \dots, p_K$, $K \ge 1$, none equal to q, and set

$$M = \prod_{i=1}^{K} p_i > 4B, \quad M_i = M/p_i, \quad a_i = M_i^{-1} \pmod{p_i}.$$

All residues below are canonical. Given $x_i = X \bmod p_i$, we define

$$t_i = a_i x_i \bmod p_i, \qquad 0 \le t_i < p_i.$$

Since $\sum_i t_i M_i \equiv X \pmod{M}$, there is an integer η such that

$$\sum_i t_i M_i = X + \eta M, \qquad \sum_i \frac{t_i}{p_i} = \eta + \frac{X}{M}. \tag{44}$$

The bounds $|X/M| < 1/4$ and $0 \le \sum_i t_i/p_i < K$ imply $0 \le \eta \le K$.

For the centered representative $|X| < M/4$, rounding recovers the integer quotient with the following guard precision. Define

$$\tau = \lceil \log_2(8K) \rceil, \qquad A_{\mathrm{fp}} = \sum_{i=1}^{K} \left\lfloor \frac{2^\tau t_i}{p_i} \right\rfloor.$$

The quotient satisfies

$$\eta = \left\lfloor \frac{A_{\mathrm{fp}}}{2^\tau} + \frac{1}{2} \right\rfloor, \tag{45}$$

and the reconstructed residue therefore satisfies

$$X \bmod q = \left(\sum_i t_i (M_i \bmod q) - \eta (M \bmod q) \right) \bmod q. \tag{46}$$

An accumulator of

$$a_{\mathrm{fp}} = \tau + \lceil \log_2 K \rceil + 1 \tag{47}$$

bits holds the nonnegative sum and the rounding addition $2^{\tau-1}$ without overflow. The quotient uses at most $\lceil \log_2(K+1) \rceil$ bits; both lengths are $O(\log_2(K+1))$.

For each summand, the definition of the floor function gives

$$0 \le \frac{t_i}{p_i} - 2^{-\tau} \left\lfloor \frac{2^\tau t_i}{p_i} \right\rfloor < 2^{-\tau}.$$

The total error therefore satisfies $0 \le \sum_i t_i/p_i - 2^{-\tau} A_{\mathrm{fp}} < K 2^{-\tau} \le 1/8$. Together with (44), this gives

$$\eta + \frac{1}{8} < \frac{A_{\mathrm{fp}}}{2^\tau} + \frac{1}{2} < \eta + \frac{3}{4}.$$

The quantity between the bounds lies strictly in $(\eta, \eta+1)$, so taking its floor proves (45). Reducing $X = \sum_i t_i M_i - \eta M$ modulo q gives (46). Each fixed-point summand is below 2^τ, hence $A_{\mathrm{fp}} + 2^{\tau-1} < (K+1/2)2^\tau$, so (47) prevents overflow. The quotient length follows from $0 \le \eta \le K$.

For reversible reconstruction, use odd auxiliary primes and initialize an n-bit accumulator $Y \in \mathbb{Z}_q$ and an a_{fp}-bit accumulator A_{fp} to zero. For each p_i, the clean residue-tree oracle first writes x_i into an external b-bit word and clears its internal workspace. A separate zero word receives $t_i = a_i x_i \bmod p_i$ through a constant-multiplication accumulator. Both words remain live during the accumulator updates. The fixed-point update includes inverse doublings that restore t_i before its modular contribution is added to Y. The constant-multiplication accumulator is then reversed to clear the t_i word. A fresh phase-corrected residue-XOR invocation then clears x_i. Thus each modulus uses the sequence

$$(0,0) \longmapsto (x_i, 0) \longmapsto (x_i, t_i) \longmapsto (x_i, 0) \longmapsto (0,0),$$

with the accumulator updates performed while (x_i, t_i) are live. The two words account for the $2b$ term in the large-modulus stage of (55). During either tree invocation, the t_i word is zero and its space is available to the tree.

We describe the fixed-point contribution without the index i. We set $y_0 = t$, $y_j = 2y_{j-1} \bmod p$ in canonical residues. We define $d_j = \lfloor 2y_{j-1}/p \rfloor \in \{0,1\}$. Since p is odd,

$$y_j = 2y_{j-1} - p d_j, \qquad d_j = y_j \bmod 2.$$

Iteration of this recurrence gives

$$2^\tau t = p\sum_{j=1}^{\tau} d_j 2^{\tau-j} + y_\tau, \qquad 0 \le y_\tau < p,$$

That is,

$$\lfloor 2^\tau t/p \rfloor = \sum_{j=1}^{\tau} d_j 2^{\tau-j},$$

After doubling at step j, the low bit of y_j coherently controls addition of $2^{\tau-j}$ to A_{fp}. Finally, apply the τ inverse doublings to the residue word alone, restoring t while retaining its fixed-point contribution.

For a general odd modulus, extend the b-bit word by a zero high bit and use an initially zero underflow flag in addition to the adder workspace. One left rotation of the $b+1$ bits gives $2t$. Subtract p, copy the high bit to a flag e, and add p back when $e = 1$. The result is the canonical $y = 2t \bmod p$, with zero extension bit. Since $e = 1 - \lfloor 2t/p \rfloor$ and $y \bmod 2 = \lfloor 2t/p \rfloor$, the update $e \oplus= 1 \oplus (y \bmod 2)$ clears the flag. The inverse is modular halving, $y \mapsto \big(y + p(y \bmod 2)\big)/2$, on canonical residues. For $p = 2^d - 1$, use the rotations in (62); the 2τ rotations cost no Toffolis, while the controlled additions to A_{fp} keep their cost.

For the modular contribution, add $t_i(M_i \bmod q)$ to Y modulo q. After processing all moduli, (45) gives the quotient correction without a separate η register. Temporarily add $2^{\tau-1}$ to A_{fp}. Its bits from position τ upward encode η, with positions numbered from zero. The bit at $\tau + k$ controls subtraction of $2^k(M \bmod q) \bmod q$ from Y, implementing $Y \leftarrow Y - \eta(M \bmod q) \pmod q$. Undo the addition to A_{fp}. By (46), Y now holds $X \bmod q$.

Clear the residue-tree workspace before arithmetic on Y, allowing the $n+1$-bit constant-adder operand to reuse it. Retain A_{fp} through the character phase, then undo the quotient correction and per-modulus updates to clear both accumulators. Subtraction reverses the modular additions. For residue oracles with measured erasure, cleanup uses fresh phase-corrected XOR invocations as described in Section 5.1. Appendix C.3 gives the clean interface and the tree schedule supplying its residues.

5.3 Public masks and sampling error bounds

The window summands in (34) are structured functions of the scalar inputs and may vanish, so their child sums need not be generic with high probability. Public masks make each true child pair independent and uniform for every fixed input. Counting exceptional pairs then gives an input-uniform arithmetic error bound. The induced translation of F preserves Fourier probabilities on $\mathbb{Z}_r^2$.

Proposition 5.1 (Two-seed masks for balanced trees)**.** *We consider any finite abelian group G of odd order. For a complete balanced tree with $L = 2^h \ge 2$ ordered leaves, we choose independent uniform $U, V \in G$. We replace each leaf $A_i(x)$ by $A_i(x) + R_i$, where*

$$R_i = U + [i]V, \qquad T_R = [L]U + \left[\frac{L(L-1)}{2}\right] V. \tag{48}$$

For every fixed x, the two true child sums at every internal node are independent uniform elements of G. The true group sum at the root is $F(x) + T_R$. The seeds attain the minimum entropy $2\log_2|G|$ among schemes that require an exact independent uniform root pair for every fixed input.

Proof. A subtree occupies a consecutive leaf interval. For adjacent child intervals of sizes s_1, s_2 starting at j, their mask sums are the image of (U, V) under

$$M_v = \begin{pmatrix} s_1 & s_1 j + s_1(s_1-1)/2 \\ s_2 & s_2(j+s_1) + s_2(s_2-1)/2 \end{pmatrix}, \qquad \det M_v = s_1 s_2 (s_1+s_2)/2.$$

For a balanced tree, $s_1 = s_2 = 2^k$ and the determinant is 2^{3k}, whose multiplication is invertible on an odd-order group. The adjugate formula makes M_v bijective on G^2. Adding the fixed unmasked sums preserves uniformity, and the root mask sum is T_R. For fixed x, an independent uniform root pair has entropy $2\log_2|G|$ and is a deterministic function of the seed. This is therefore a lower bound on seed entropy, attained by the independent uniform U, V with $H(U, V) = 2\log_2|G|$. □

The same argument works for an ordered binary tree if $\gcd(\exp G, s_1 s_2 (s_1 + s_2)/2) = 1$ at every node; prime order $|G| > L$ suffices. This use of affine randomization [7] requires invertibility for child-interval sums. The corresponding channel and sampling error bounds are proved in Appendix B.

Write $R = (U, V)$ for the public seeds. In the genus-two sampler, sample them classically using the known generator and prepare the masked window tables. Degree-two entries have $Z = 1$; lower-degree entries are replaced by $(0, 0, 0, 0, 1)$. Every nonterminal node evaluates (36), and the root evaluates $\Psi_\gamma = N_* + \gamma D_*$ with an independent uniform field seed γ. Denote the completed character by $\widetilde{g}_{R,\gamma}(x)$. This integer program is defined on all tuples and is reduced unchanged at each auxiliary modulus. It needs no quantum mask, degree-tag, or exceptional-branch register.

The genericity conditions are

$$\deg u^{(1)} = \deg u^{(2)} = 2, \qquad Z_1 Z_2 \mathcal{R}_* \mathcal{E}_* \neq 0. \tag{49}$$

Under these conditions, the map gives group addition for every reached valid weighted representative. Bihomogeneity preserves the conditions under nonzero rescaling. The polynomial program remains defined when they fail.

Lemma 5.2 (Frequency of an exceptional generic addition)**.** *We define $e = \#\{D \in G : \deg u_D < 2\}$. For independent uniform $D, E \in G$, the probability that their canonical degree-two representatives fail the generic condition* (49) *is at most*

$$\min\left\{1, \frac{3e + 8q}{r}\right\} \leq \min\left\{1, \frac{14q + 3}{r}\right\}. \tag{50}$$

Here an input of degree less than two is counted as a failure.

Proof. Input degree failures contribute at most $2e/r$. For a fixed monic quadratic u_D, at most $2q$ monic quadratics u_E share a factor with it: if u_D splits, each of its at most two roots leaves at most q choices for the other root; if u_D is irreducible, only $u_E = u_D$ is possible. Each such u_E has at most four valid v_E by the count in Section 4.1. Common support therefore contributes at most $8q/r$, also within a subgroup.

For degree-two inputs with coprime u-polynomials, the interpolant has degree at most three. Its cubic coefficient is nonzero exactly when Cantor reduction has degree-two output. If it vanishes, the monic quintic equation gives degree-one output. The remaining denominator failure is therefore contained in $\{\deg u_{D+E} < 2\}$. Since $D + E$ is uniform, this has probability e/r. The union bound and $e \leq 2q + 1$ prove the claim. □

Theorem 5.3 (Input-uniform masked arithmetic)**.** *We use e from Lemma 5.2 and define*

$$\varepsilon_{\mathrm{tree}} = \min\left\{1, \frac{(L-1)(3e + 8q)}{r}\right\}.$$

For every fixed input x, the probability over the two fresh group seeds that at least one true child pair is exceptional is at most $\varepsilon_{\mathrm{tree}}$. Consequently, for every seed-independent input distribution, its bad-input mass ε_R satisfies

$$\mathbb{E}_R \varepsilon_R \leq \varepsilon_{\mathrm{tree}}. \tag{51}$$

Outside the bad set, $\widetilde{g}_{R,\gamma}(x) = g_\gamma(F(x) + T_R)$ for every γ. The randomized clean phase channel differs from this translated ideal channel by at most $\min\{1, 2\sqrt{\varepsilon_{\mathrm{tree}}}\}$ in half-diamond norm, even with the classical seeds retained. The half-diamond distance between channels $\mathcal{E}$ and $\mathcal{F}$ is $\frac{1}{2}\|\mathcal{E} - \mathcal{F}\|_\diamond$. This distance is the maximum output trace distance over input states with an arbitrary reference system.

Proof. Define bad events using true group sums, even when a lower polynomial node has failed. Proposition 5.1 and Lemma 5.2 bound the event at each node. A union bound over $L - 1$ nodes gives the pointwise estimate, without independence between node events. Off the bad set, generic correctness applies inductively, and the terminal square-factor identity gives the required phase.

Applying Theorem B.1 with $\delta_v = (3e + 8q)/r$ at each internal node gives the stated half-diamond distance, including when the classical seeds are retained. □

The channel estimate averages over fresh masks independent of the input and its reference. The Legendre analysis uses the exact uniform domain $\mathbb{Z}_r^2$. Masks change only classical lookup constants; the canonical leaf and dummy bounds, degrees, and arithmetic schedule are unchanged.

Let V_G be the shift-averaged ideal lower bound in (22). For each mask, translation by T_R changes Fourier amplitudes only by unit phases, preserving their probabilities. Write $\widetilde{p}_{\mathrm{use}}$ for the success probability of masked arithmetic with exact preparation and readout, averaged over R, γ. By Theorem 3.1, the translated ideal failure probability is $\mu^2_{g_\gamma}$. The failure projector complements the outcomes $u \neq 0$, $v = ku \bmod r$, where $Q = [k]P$. Applying the failure-probability bound in Theorem B.1 with $p_0 = \mathbb{E}_\gamma \mu^2_{g_\gamma}$ gives

$$\begin{aligned} \sqrt{1 - \widetilde{p}_{\mathrm{use}}} &\leq \sqrt{\mathbb{E}_\gamma \mu^2_{g_\gamma}} + 2\sqrt{\varepsilon_{\mathrm{tree}}} \\ &\leq \sqrt{1 - V_G} + 2\sqrt{\varepsilon_{\mathrm{tree}}}. \end{aligned} \tag{52}$$

This allows correlation between T_R and the bad set. With J_{prep} trials per register, preparation aborts with probability at most $2^{1-J_{\mathrm{prep}}}$ by Theorem 3.3. Let ε_{F} and $\varepsilon_{\mathrm{syn}}$ be the distribution-error budgets for Fourier readout and rotation synthesis. Squaring (52) and including these losses yields

$$p_{\mathrm{run}} \geq \max\left\{0, 1 - \left(\sqrt{1 - V_G} + 2\sqrt{\varepsilon_{\mathrm{tree}}}\right)^2 - 2^{1-J_{\mathrm{prep}}} - \varepsilon_{\mathrm{F}} - \varepsilon_{\mathrm{syn}}\right\}. \tag{53}$$

For $u \neq 0$, form $k' = vu^{-1} \bmod r$ and accept only if the classical check $[k']P = Q$ succeeds. Each run draws fresh independent masks and a fresh field shift. Any positive lower bound p_* in (53) therefore gives expected waiting time at most $1/p_*$ and failure probability at most $(1 - p_*)^N$ after N independent runs.

For $r = \Theta(q^2)$ and $w = \Theta(\log_2 n)$, the bounds are $1 - V_G = O(q^{-1})$ and $\varepsilon_{\mathrm{tree}} = O(L/q)$. The square of (52) therefore gives $1 - \widetilde{p}_{\mathrm{use}} = O(L/q)$. Equation (53) yields

$$p_{\mathrm{run}} \geq 1 - O(L/q) - \varepsilon_{\mathrm{F}} - \varepsilon_{\mathrm{syn}} - 2^{1-J_{\mathrm{prep}}}. \tag{54}$$

Appendix D.2 gives the asymptotic details. The complete width allocation and finite resource estimates are collected in Section 6.1.

6 Detailed Estimates

Use the field and subgroup bit lengths in (2). We use W for peak logical width and C_{Tof} for ordinary three-qubit Toffoli gates. Clifford gates, measurements, and classical feedforward are counted separately. Here T denotes the Clifford+T gate. Earlier G bounds become Toffoli bounds once every primitive, control, and cleanup has been decomposed in that basis.

Unless stated otherwise, a bound covers one run, including its live inputs, arithmetic, preparation, and Fourier processing. For independent identically distributed runs of success probability $p > 0$ and fixed cost C_{Tof}, the expected cost of a verified logarithm is C_{Tof}/p. A worst-case run cost and conditional success lower bound give the corresponding upper bound. Serial runs reuse the qubits.

6.1 Sampler bounds and finite resource estimates

Sampler bounds. We assume the curve and subgroup hypotheses of Section 2 and the computational model of Section 3, which allows intermediate measurements, reset, and classical feedforward. Both scalar registers remain live during arithmetic, with $N_{\mathrm{in}} = 2m_r$. Classical tables and measurement records are excluded from the logical width.

The window decomposition in (34) gives a complete balanced tree with $L = 2^h \geq 2$ leaves. We choose distinct odd primes $p_i < 2^b$, none equal to q, such that $M = \prod_{i=1}^{K} p_i > 4 \cdot 2^H$, where H is given by (41). Residue words have width b, with $w - 1 \leq 8b$, and guarded reconstruction uses $a_{\mathrm{fp}} = 2\lceil \log_2 K \rceil + 4$ fixed-point bits.

Each run draws fresh independent uniform $U, V \in G$ and $\gamma \in \mathbb{F}_q$. The group seeds define the masks in (48). Masked leaf tables are prepared classically by complete divisor arithmetic, with lower-degree entries

replaced by the fixed dummy tuple from Section 5.3. The circuit evaluates the same integer polynomials on every input and uses no quantum mask register or exceptional-branch record.

Let V_G be the ideal useful-sample bound in (22), and let $\varepsilon_{\mathrm{tree}}$ be the exception bound in Theorem 5.3. Preparation on the exact domain $\mathbb{Z}_r^2$ is capped at $J_{\mathrm{prep}} \geq 1$ trials per register. The distribution-error budgets for Fourier readout and rotation synthesis are ε_{F} and $\varepsilon_{\mathrm{syn}}$, respectively. Success probabilities include preparation and measurement outcomes and are averaged over the classical seeds. Every candidate k' is accepted only after the classical check $[k']P = Q$.

Theorem 6.1 (Masked residue sampler)**.** *For the sampler and parameters specified above, the following statements hold.*

(i) The probability of obtaining a classically verified discrete logarithm in one run satisfies

$$p_{\mathrm{run}} \geq \max\left\{0,\ 1 - \left(\sqrt{1 - V_G} + 2\sqrt{\varepsilon_{\mathrm{tree}}}\right)^2 - 2^{1-J_{\mathrm{prep}}} - \varepsilon_{\mathrm{F}} - \varepsilon_{\mathrm{syn}}\right\}.$$

(ii) The peak logical width, including both scalar inputs, arithmetic, preparation, and Fourier readout, satisfies

$$W_{\mathrm{sample}} \leq \max\begin{Bmatrix} N_{\mathrm{in}} + n + a_{\mathrm{fp}} + [5(h+2)+11]b + 4, \\ N_{\mathrm{in}} + 2n + a_{\mathrm{fp}} + 2b + 20, \\ N_{\mathrm{in}} + a_{\mathrm{fp}} + 6n + 12, \\ 3m_r + 5 \end{Bmatrix}. \tag{55}$$

(iii) If $r = \Theta(q^2)$ and $w = \Theta(\log_2 n)$, the auxiliary moduli can be chosen so that

$$W_{\mathrm{sample}} \leq 10n + o(n). \tag{56}$$

Classical preprocessing and arithmetic then have polynomial cost in n. For the exact uniform input state and ideal Fourier readout, the useful-sample probability is $1 - O(L/q)$. For $w = \lceil \log_2 n \rceil$, arithmetic, including reversible table selection and cleanup, uses

$$O\left(\frac{n^{3+\log_2 20}}{(\log n)^{\log_2 20}}\right) \quad \textit{Toffoli gates.}$$

Fixed or inverse-polynomial preparation and Fourier error budgets do not change this exponent; rotation synthesis is counted separately.

Proof. For each fixed scalar input, Proposition 5.1 makes the two true child values at every internal node independent and uniform. Theorem 5.3 therefore bounds the probability of any exceptional addition by $\varepsilon_{\mathrm{tree}}$. Outside this event, the polynomial phase agrees with $g_\gamma(F(a,b) + T_R)$. Translation by T_R preserves Fourier probabilities, so the mean ideal useful-sample probability is at least V_G. The failure-amplitude estimate (52) gives

$$1 - \widetilde{p}_{\mathrm{use}} \leq \left(\sqrt{1 - V_G} + 2\sqrt{\varepsilon_{\mathrm{tree}}}\right)^2.$$

By Theorem 3.3, capped preparation aborts with probability at most $2^{1-J_{\mathrm{prep}}}$ and gives the exact uniform input state on acceptance. The two distribution-error budgets reduce the probability of the useful event by at most $\varepsilon_{\mathrm{F}} + \varepsilon_{\mathrm{syn}}$. Every useful outcome satisfies $u \neq 0$ and $v = ku \bmod r$; hence $k' = vu^{-1} \bmod r$ passes the classical check. Subtracting these losses proves (i).

For (ii), (43) and $A_{\mathrm{prim}}(b) \leq b + 4$ bound the residue-tree workspace by $[5(h+2)+11]b+4$. Reconstruction also retains the two scalar inputs, the n-bit output, and the a_{fp}-bit fixed-point sum, giving the first entry of (55). The tree workspace is cleared before each large-modulus update. The two b-bit words holding x_i and t_i remain live during that update, as described in Section 5.2, giving the second entry. During the character phase, the fixed-point sum remains live, and $W_\chi(n) \leq 6n + 12$ already includes the n-bit character argument. This gives the third entry. Preparation and Fourier readout use at most $3m_r + 5$ qubits by Theorem 3.3. Clean workspace is reused between these stages, so their maximum proves the claimed peak width.

Now suppose $r = \Theta(q^2)$ and $w = \Theta(\log_2 n)$. The window decomposition gives $L = O(n/\log n)$ and $h = O(\log n)$, while (41) gives $H = O(nL^{\log_2 20})$. The prime-product estimates in Appendix D.2 allow $b = O(\log n)$, polynomially many auxiliary primes, and $a_{\mathrm{fp}} = O(\log n)$, with the required reconstruction margin and $w - 1 \le 8b$. Since $N_{\mathrm{in}} = 4n + O(1)$, the four entries of (55) become

$$5n + O(\log^2 n), \qquad 6n + O(\log n), \qquad 10n + O(\log n), \qquad 6n + O(1).$$

Their maximum proves (56). The $O(L2^w)$ table entries, the prime list, and the integer constants have polynomial size. The tree schedule makes $O((h+1)L)$ local moves per modulus, and each local operation and CRT update has polynomial cost. This proves the preprocessing and arithmetic claims. The explicit arithmetic gate bound follows from the modulus count and per-modulus costs in Appendix D.2.

Finally, the bounds $e \le 2q + 1$ and $\sum_c m_c^2 \le 4q(r - e)$ imply

$$1 - V_G \le \frac{1}{q} + \frac{e^2}{r^2} + \frac{4q}{r} = O(q^{-1}), \qquad \varepsilon_{\mathrm{tree}} = O(L/q).$$

Using $(\sqrt{A} + 2\sqrt{B})^2 \le 2A + 8B$ for $A, B \ge 0$ in the failure bound above yields $1 - \widetilde{p}_{\mathrm{use}} = O(L/q)$. Conditional on successful preparation, the input is exactly uniform. With ideal Fourier readout, this is the useful-sample probability claimed in (iii). □

A 127-bit field instance. Apply Theorem 6.1 to the Gaudry–Schost curve used by μKummer [36], whose parameters satisfy

$$\begin{aligned} &q = 2^{127} - 1, \qquad \#J(\mathbb{F}_q) = 16r, \\ &r = 2^{250} - \texttt{0x334D69820C75294D2C27FC9F9A154FF47730B4B840C05BD}. \end{aligned} \tag{57}$$

The cited construction establishes primality of r and supplies a nonzero divisor of order r. Substituting $X = x + a$ with $a = -f_4/5$ in its monic model removes the quartic term. The resulting curve and generator are

$$C' : y^2 = x^5 + c_3x^3 + c_2x^2 + c_1x + c_0, \qquad P = (x^2 + u_1x + u_0,\, v_1x + v_0),$$

with the hexadecimal constants in Table 4.

Table 4. An isomorphic zero-quartic Gaudry–Schost model. Constants are canonical residues modulo q, written in hexadecimal with a `0x` prefix.

Parameter	Value
a	`0x5FAD7267DA7A005DC143AD1352526C82`
c_0	`0x3AF79604317A895BB28139B4DDD66D56`
c_1	`0x4761178EC206F6DB2E90B2BF4913BE47`
c_2	`0x432B709D2D5801982F927BB531F0F535`
c_3	`0x00D26A12F16AE9D00045BB92C5382A8E`
u_0	`0x384CB21DBD5E1F600F3FE0E336182F14`
u_1	`0x3CB88102BCDD5A7AAA402188B6783790`
v_0	`0x4F4AB1A434DFCBE8FD26C203D18D54F4`
v_1	`0x444569AF177A9C1C721736D8F288C942`

This instance has $n = 127$ and $m_r = \lceil \log_2 r \rceil = 250$. The live input width is $N_{\mathrm{in}} = 500$ for the exact domain $\mathbb{Z}_r^2$. For a tree with $L = 32$ leaves, the masking union bound evaluates to

$$\varepsilon_{\mathrm{tree}} \le \frac{31(14q + 3)}{r} < 4.082 \times 10^{-35} < 2^{-114}.$$

The corresponding $2\sqrt{\varepsilon_{\mathrm{tree}}}$ state-distance bound is less than 1.278×10^{-17}.

The program `check_gs_parameters.py` checks square-freeness, membership, $P \neq 0_G$, and $[r]P = 0_G$ using complete Cantor arithmetic. Lucas–Lehmer and a recursive Pocklington certificate prove primality

of q, r. Together with exact integer Weil bounds, these certify $\mathrm{ord}(P) = r$ and $\#\,\mathrm{Jac}(C')(\mathbb{F}_q) = 16r$; see Appendix D.1. The identification with the published model uses the cited source, and these are classical parameter certificates.

For $L = 32$, the coarse fiber bound in Section 4 and the square of (52) give $1 - \widetilde{p}_{\mathrm{use}}$ below $1.795 \times 10^{-34} < 2^{-112}$. We choose $J_{\mathrm{prep}} = 42$ and $\varepsilon_{\mathrm{F}} = \varepsilon_{\mathrm{syn}} = 2^{-40}$. The preparation, Fourier, and synthesis losses sum to $2^{-41} + 2 \cdot 2^{-40}$. Equation (53) therefore gives

$$1 - p_{\mathrm{run}} < 1.795 \times 10^{-34} + 2^{-41} + 2 \cdot 2^{-40} < 2.274 \times 10^{-12} < 2^{-38}.$$

Measured Fourier readout uses 292 classical precision bits. The finite estimates include preparation, Fourier readout, and the character circuit. Rotation synthesis is counted separately at the stated accuracy.

Finite widths. Use H from (41) and all odd primes at most 2^b, excluding q. Dusart's unconditional estimates [17] give the following bounds, with ln denoting the natural logarithm:

$$\frac{2^b}{\ln 2}\left(1 - \frac{0.2}{(b \ln 2)^2}\right) - 1 - n > H + 2 \quad \Longrightarrow \quad M > 4 \cdot 2^H$$

when $2^b \geq 3594641$. The number of primes satisfies

$$K \leq K_{\max} := \left\lceil \frac{2^b}{b \ln 2 - 1.1} \right\rceil.$$

Reserve fixed-point space using $K_{\max}$, with $\ell_K = \lceil \log_2 K_{\max} \rceil$, $\tau = \ell_K + 3$, and $a_{\mathrm{fp}} = 2\ell_K + 4$. Subtracting $1 + n$ permits removal of 2 and q. The numerical allocation is thereby certified without constructing a prime list.

We substitute $A_{\mathrm{prim}}(b) \leq b + 4$ and these register sizes into (55). The resulting allocations are

w	L	b	$K_{\max}$	a_{fp}	W_{recon}
8	64	32	203,739,217	60	2,323
16	32	28	14,662,098	52	1,971
24	32	28	14,662,098	52	1,971
32	16	24	1,079,926	46	1,661

Reconstruction has the largest width in every row. The four stage allocations are $1971, 882, 1326, 755$ for $w = 16$ and $1661, 868, 1320, 755$ for $w = 32$. Window 24 has the same width as window 16 and a larger lookup table.

Finite Toffoli count. Using Table 3 and the primitive bounds in Appendix C.1, set

$$A_b = 17b^2 + b - 8, \qquad Q_w = 4w2^w, \qquad G_* = 2\max\{Q_w, 707A_b\}.$$

The node and terminal costs are at most $707A_b$, while ghost correction costs at most $1414A_b$. A lookup equality ladder uses at most $w - 1 \leq 8b$ clean selector bits while the eight local words are free. Computing and clearing it costs less than $4w$ Toffolis per entry; CNOTs copy the selected output. Hence G_* covers local evaluation, erasure, and ghost correction.

One reconstruction direction has the bound

$$\begin{aligned} G_R \leq K_{\max}\big[&16(h+1)LG_* + 40b(b+1) + \tau\{8b + 8 + 20(a_{\mathrm{fp}} + 1)\} + 20b(n+1)\big] \\ &+ 40(a_{\mathrm{fp}} + 1) + 20(\ell_K + 1)(n+1). \end{aligned}$$

The $40b(b+1)$ term covers formation and clearing of t_i: the two constant-multiplication accumulators cost at most $2C_\gamma(b) \leq 18b^2 + 2b < 40b(b+1)$. Terms involving the n-bit target include large-modulus arithmetic. Reconstruction is performed before and after the character phase. With the stated preparation cap and Fourier precision, the full-run budget is

$$C_{\mathrm{Tof}} \leq 2G_R + C_\chi(n) + 4J_{\mathrm{prep}}(2m_r - 1) + 2 \cdot 292(9m_r + 1).$$

Here $C_\chi(n)$ includes character computation and reversal. For $w = 16$, both reconstruction directions cost at most 1,700,254,660,330,496,928 Toffolis. The character, capped preparation, and Fourier readout add at most 2,331,720, 83,832, and 1,314,584, respectively, giving

$$C_{\mathrm{Tof}} \le 1{,}700{,}254{,}660{,}334{,}227{,}064 < 2^{60.561}.$$

The same formula with $w = 32$ gives

$$C_{\mathrm{Tof}} \le 3{,}039{,}721{,}457{,}235{,}954{,}066{,}696 < 2^{71.365}.$$

Both bounds cover one capped run with the same preparation and readout budgets. The smaller $w = 32$ tree lowers the masking error, so $p_{\mathrm{run}} > 1 - 2^{-38}$ holds for both windows. Expected Toffoli costs per verified logarithm are therefore bounded by the respective run costs divided by $1 - 2^{-38}$. Rotation synthesis is counted separately at its accuracy budget, as is classical preprocessing. Every auxiliary prime uses the generic primitives in these estimates, including when the base field is Mersenne.

Classical storage and table preparation. The prime list and product M depend on the field and window parameters, independently of fresh masks. Preprocessing enumerates the primes, forms M, and records $M \bmod q$. There is no need to store every full M_i. For the active prime p_i, compute

$$a_i = \left(\frac{M \bmod p_i^2}{p_i}\right)^{-1} \bmod p_i, \qquad M_i \bmod q = (M \bmod q)(p_i^{-1} \bmod q) \bmod q.$$

The division is exact since $p_i \mid M$. Distinct primes guarantee the first inverse, and $p_i \neq q$ guarantees the second. Keep these reduced constants only for the active modulus and recompute them on the reverse pass. The prime list and M occupy $O(Kb)$ bits. Reducing the stored product to generate constants avoids keeping K integers of length $\log_2 M$.

Each run draws fresh group seeds and rebuilds its masked leaf tables. For each nonpadding leaf, begin at R_i and repeatedly add the fixed window base using complete Cantor arithmetic. Store four canonical coefficients for a degree-two result and four zeros otherwise, matching the dummy tuple in Section 5.3. Every entry has $Z = 1$. The coefficient tables therefore occupy at most $4nL2^w$ bits before metadata, and construction uses $O(L2^w)$ complete classical additions besides the window bases and masks. For $w = 16$, the bounds are 2^{21} entries and 133,169,152 packed bytes; for $w = 32$, they are 2^{36} entries and 4,363,686,772,736 bytes.

For residue lookup, reduce stored coefficients modulo the active p_i and emit the classically controlled X/CNOT pattern. Reuse the canonical tables for every modulus, phase correction, and cleanup. Patterns can be generated as needed, so neither per-modulus tables nor a stored list of elementary gates is required. Classical storage is $O(nL2^w + Kb)$ bits. The gate estimates include reversible selection; classical table preparation and constant generation are separate costs.

6.2 Matched comparison with Chen et al.'s circuits

We use Chen et al. [8] as the quantitative baseline because their genus-two circuits over Mersenne prime fields apply to the same field $\mathbb{F}_{2^{127}-1}$ as our finite instance. The field bit length $n = \lceil \log_2 q \rceil$ is distinguished from the subgroup bit length $m_r = \lceil \log_2 r \rceil$. For bounded-cofactor genus-two groups, $m_r = 2n + O(1)$; the two scalar inputs therefore occupy $2m_r$ qubits, and the unwindowed schedule makes $2m_r$ controlled additions. All finite counts below use the actual subgroup order. Huang et al. [25] treat general prime fields; their quoted qubit formula is a controlled-addition allocation, and their printed gate-count expansion differs from the stated operation sum. Using those estimates as a full-sampler baseline would require a separate reconciliation of the live scalar registers and gate count. The comparison below uses Chen et al.'s Mersenne-field inventory.

Chen et al. [8] report the controlled-addition allocation and modular-operation inventory

$$W_{\mathrm{add}} = 20n + 2\lceil \log_2 n \rceil + 10, \qquad C_{\mathrm{add}} = 836n^2 + 18n - 340.$$

Appendix D.3 reproduces the primitive inventory and distinguishes the printed polynomial from its expanded operation count, keeping the original circuit interfaces.

A controlled addition includes one control qubit in W_{add}. A sequential, unwindowed full-output schedule retains two m_r-bit scalar registers and performs $2m_r$ controlled additions. Its arithmetic count is $2m_r C_{\mathrm{add}}$, and its allocation is

$$W_{\mathrm{add}} + 2m_r - 1 \quad \text{logical qubits.} \tag{58}$$

Subtracting one shares the module control with a live scalar bit. Addition workspace is reused, and the maximum includes preparation and Fourier processing.

The width in [8, Sections 5.1–5.2] is that of the addition module; its gate total uses $2m_r$ module calls. Equation (58) adds the other live inputs. The rows in Table 5 retain both scalar registers. Section 6.2 gives a separate recycled-control analysis. The illustrative substitution $2m_r = 4n$ need not hold for every genus-two subgroup.

For the instance in Section 6.1, Chen's reported module uses $20 \cdot 127 + 2\lceil \log_2 127 \rceil + 10 = 2{,}564$ qubits, giving matched width $2564 + 500 - 1 = 3063$. Its inventory gives 6,742,895,000 Toffolis for 500 additions. Preparation and readout use at most 755 qubits and do not raise the peak. This row retains the reported module and its generic-addition domain.

The clean XOR adapter in Section 2.2 uses the later Luo et al. inversion circuit [31]. Replacing Chen's fourteen inversion calls would require controlled interfaces and a new live-register allocation; that replacement is not included in the reported Chen row. The product-and-fold alternative in Appendix C.1 gives a separate gate–space tradeoff; its larger workspace is not used in the finite sampler.

Table 5. Finite resources on the same 127-bit instance with 500 live input qubits. The Chen row uses the reported generic arithmetic allocation and operation inventory; its gate entry covers arithmetic only. The residue rows bound complete capped sampling runs and exclude rotation synthesis. The fourth column counts classical residue lookup entries. Chen's circuit uses fixed classical divisors. The fifth column states its published domain or the verified-logarithm probability proved here. All widths are schedule-specific bounds.

Construction	Qubits	Toffoli gates	Classical lookup entries	Domain / success probability
Chen et al. [8]	3063	$6.743 \cdot 10^9$	—	Published generic domain
Residue, $w = 16$	1971	$< 2^{60.561}$	$\leq 2^{21}$	$> 1 - 2^{-38}$
Residue, $w = 32$	1661	$< 2^{71.365}$	$\leq 2^{36}$	$> 1 - 2^{-38}$

The residue widths are 35.65% and 45.77% below the Chen-derived allocation. These percentages refer to the stated live-register schedules. Chen's arithmetic inventory does not give a full-run verified-logarithm probability under the budgets used here.

This space reduction has a large gate cost. At $w = 16$, the Toffoli upper bound is about $2.52 \cdot 10^8$ times Chen's reported arithmetic count; at $w = 32$, the ratio is about $4.51 \cdot 10^{11}$. These are ratios of analytical estimates, rather than measured running times, and classical computation is also substantial.

Recycled controls in the same measurement model. Retaining both scalar registers is not necessary for a sampler built from complete group translations. The following construction uses the same measurement, reset, and feedforward model as ours.

Proposition 6.2 (Recycled-control group sampling)**.** *Let $G = \langle P \rangle$ have known odd prime order r and let $Q = [k]P$. Suppose clean controlled translations $T_B|D\rangle = |D + B\rangle$ are available on every $D \in G$, for $B = [2^j]P, [2^j]Q$ and $0 \leq j < t$, with maximum total width W_{tr} including the divisor register, control, and workspace. Take t as in* (14)*. There is a sampler using $W_{\mathrm{tr}} + O(1)$ qubits and $2t$ controlled translations, whose probability of a classically verified logarithm is at least $1 - 1/r - \varepsilon_{\mathrm{F}} - \varepsilon_{\mathrm{syn}}$. Classical output records and rotation synthesis are counted separately.*

Proof. With $\omega_r = e^{2\pi i/r}$, the states

$$|\widehat{j}\rangle = r^{-1/2} \sum_{a \in \mathbb{Z}_r} \omega_r^{-ja} |[a]P\rangle$$

are simultaneous eigenstates of T_P, T_Q, with eigenvalues $\omega_r^j, \omega_r^{kj}$. Initialize the divisor register in $|0_G\rangle = r^{-1/2}\sum_j |\widehat{j}\rangle$. Run recycled-control phase estimation for T_P, followed by T_Q, on that same register. The powers are single translations by classically precomputed divisors. One control is reset and reused through all $2t$ rounds; the two readouts are stored classically. Tracing out the orthogonal eigenstates gives a uniform mixture over j. The rounding argument in Appendix C.4 bounds each incorrect-label probability by $\varepsilon_{\mathrm{F}}/2$. When both labels are correct and $j \neq 0$, their quotient modulo r is k. A union bound and the synthesis budget give the claim. □

Here $t = 292$, so the recycled schedule uses 584 translations, compared with 500 additions in the retained-register schedule. If a complete translation could be supplied at Chen's reported module width, the arithmetic peak would be $2564 + O(1)$ qubits. Reusing its per-call inventory would give approximately $7.876 \cdot 10^9$ arithmetic Toffolis. These are conditional accounting estimates: the published generic module does not by itself establish the complete clean interface assumed in the proposition. Exceptional cases require a complete implementation or a separate error analysis. The randomized-tree proof cannot be transferred to this sequential circuit without such an argument.

The percentages above compare explicitly retained-register schedules and do not establish the minimum width available in the measurement model. The independently proved $10n+o(n)$ bound remains the asymptotic space result of this construction.

Comparison across curve families. For bounded-cofactor genus-two groups, generic classical discrete logarithms cost $2^{n+O(1)}$ work. For bounded-cofactor elliptic-curve groups over an n_E-bit prime field, the cost is $2^{n_E/2+O(1)}$. Approximate comparison at equal generic security therefore uses $n_E \simeq 2n$. This aligns parameters without certifying every curve or subgroup. Mersenne circuits additionally require $q = 2^n - 1$ prime and are available only at bit lengths satisfying that condition.

Verification details and finite-field experiments are given in Appendix D.1.

7 Conclusion

Two public group seeds make the true children of each node independent and uniform in a complete balanced addition tree over an odd-order finite abelian group. This gives an input-uniform bound on exceptional arithmetic and a comparison of randomized quantum channels with the seeds retained. The determinant criterion identifies further tree shapes for which the argument applies.

For the stated genus-two model, we combine this reduction with an integer lift of weighted Mumford arithmetic and a shifted Legendre phase. For an exact uniform input state and ideal Fourier readout, the useful-sample probability, averaged over fresh independent tree seeds and a Legendre shift, is $1 - \mathcal{O}(L/q)$ for $r = \Theta(q^2)$. Height certificates, streaming reconstruction, shared evaluation, and clean scalar phase correction give $10n + o(n)$ logical qubits, including both scalar inputs in the measurement and reset model.

On the 127-bit instance, a 16-bit window uses at most 1,971 logical qubits. This is 35.65% below the matched retained-register schedule derived from Chen's module. Recycled-control group sampling provides a separate comparison: its width is that of a complete translation module, whose treatment of exceptional inputs must also be supplied. The capped-run bound is below $2^{60.561}$ Toffolis. A 32-bit window uses 1,661 qubits, about $2^{71.36}$ Toffolis, and at most 2^{36} table entries; rotation synthesis is counted separately. These estimates expose the time and classical-storage costs of reducing logical space. Polynomial identities, cited arithmetic circuits, register traces, and the small-field integration check support the construction. A complete cryptographic-size compilation remains open.

Reducing internal height growth and improving lookup and phase-correction schedules are the main routes to lower total cost. Special moduli offer a local improvement: Chen et al.'s Mersenne rotations [8] remove doubling and halving Toffolis on canonical residues, while our clean adapter makes the folding method of [26, Appendix A] usable in the same model. These alternatives require suitable CRT moduli; folding also requires space for the live integer product and residue.

Statements and declarations

Data availability. The verification programs, exact certificates, and recorded results are provided in the supplementary archive `Genus2_Verification_Supplement.zip`. Its `README.md` lists the dependencies and

execution commands, and `run_checks.py` is the entry point. The files in `expected/` contain reference outputs; the reports in `verification/` identify the checks that were executed and their outcomes.

References

1. A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, *Elementary gates for quantum computation*, Physical Review A **52** (1995), 3457–3467. doi:10.1103/PhysRevA.52.3457.
2. R. Barbulescu and G. Bisson, *Regev's attack on hyperelliptic cryptosystems*, Cryptography **10** (2026), no. 5, 62. doi:10.3390/cryptography10050062.
3. S. Beauregard, *Circuit for Shor's algorithm using* $2n + 3$ *qubits*, Quantum Information & Computation **3** (2003), no. 2, 175–185. doi:10.26421/QIC3.2-8. arXiv:quant-ph/0205095.
4. C. H. Bennett, *Time/space trade-offs for reversible computation*, SIAM Journal on Computing **18** (1989), no. 4, 766–776. doi:10.1137/0218053.
5. D. J. Bernstein and J. P. Sorenson, *Modular exponentiation via the explicit Chinese remainder theorem*, Mathematics of Computation **76** (2007), no. 257, 443–454. doi:10.1090/S0025-5718-06-01849-7.
6. D. G. Cantor, *Computing in the Jacobian of a hyperelliptic curve*, Mathematics of Computation **48** (1987), no. 177, 95–101.
7. J. L. Carter and M. N. Wegman, *Universal classes of hash functions*, Journal of Computer and System Sciences **18** (1979), no. 2, 143–154. doi:10.1016/0022-0000(79)90044-8.
8. C. Chen, P. Guan, Y. Huang, and F. Zhang, *Quantum circuits for hyperelliptic curve discrete logarithms over the Mersenne prime fields*, Quantum Information Processing **22** (2023), article 274. doi:10.1007/s11128-023-04017-x.
9. C. Chevignard, P.-A. Fouque, and A. Schrottenloher, *Reducing the number of qubits in quantum factoring*, in Advances in Cryptology—CRYPTO 2025, Part II, Lecture Notes in Computer Science, vol. 16001, Springer, 2025, 384–415. doi:10.1007/978-3-032-01878-6_13. Full version: ePrint 2024/222.
10. C. Chevignard, P.-A. Fouque, and A. Schrottenloher, *Reducing the number of qubits in quantum discrete logarithms on elliptic curves*, February 2026 full version, Cryptology ePrint Archive, paper 2026/280. ePrint version history.
11. C. Chevignard, P.-A. Fouque, and A. Schrottenloher, *Reducing the number of qubits in quantum discrete logarithms on elliptic curves*, in Advances in Cryptology—EUROCRYPT 2026, Lecture Notes in Computer Science, vol. 16541, Springer, 2026, 371–401. doi:10.1007/978-3-032-25291-3_13. Full version: ePrint 2026/280, last revised 4 June 2026.
12. S.-M. Cho, A. Kim, D. Choi, B.-S. Choi, and S.-H. Seo, *Quantum modular multiplication*, IEEE Access **8** (2020), 213244–213252. doi:10.1109/ACCESS.2020.3039167.
13. S.-M. Cho, C. Lee, and S.-H. Seo, *Quantum circuit designs of efficient squaring*, Quantum Information Processing **24** (2025), article 35. doi:10.1007/s11128-025-04647-3.
14. C. Costello and K. Lauter, *Group law computations on Jacobians of hyperelliptic curves*, in Selected Areas in Cryptography—SAC 2011, Lecture Notes in Computer Science, vol. 7118, Springer, 2012, 92–117. doi:10.1007/978-3-642-28496-0_6.
15. S. A. Cuccaro, T. G. Draper, S. A. Kutin, and D. P. Moulton, *A new quantum ripple-carry addition circuit*, 2004. arXiv:quant-ph/0410184.
16. L. M. Donaire, G. Ortega, E. M. Garzón, E. Filatovas, and F. Orts, *Low-qubit quantum circuits for efficient integer squaring*, The Journal of Supercomputing **82** (2026), article 164. doi:10.1007/s11227-026-08255-5.
17. P. Dusart, *Estimates of some functions over primes without R.H.*, arXiv:1002.0442 (2010). arXiv:1002.0442.
18. M. Ekerå and J. Håstad, *Quantum algorithms for computing short discrete logarithms and factoring RSA integers*, in: Post-Quantum Cryptography (PQCrypto 2017), Lecture Notes in Computer Science **10346**, 347–363 (2017). doi:10.1007/978-3-319-59879-6_20. arXiv:1702.00249.
19. C. Gidney, *A classical-quantum adder with constant workspace and linear gates*, arXiv:2507.23079 (2025). arXiv:2507.23079v1.
20. C. Gidney, *How to factor 2048 bit RSA integers with less than a million noisy qubits*, arXiv:2505.15917 (2025). arXiv:2505.15917.

21. R. B. Griffiths and C.-S. Niu, *Semiclassical Fourier transform for quantum computation*, Physical Review Letters **76** (1996), 3228–3231. doi:10.1103/PhysRevLett.76.3228.

22. T. Häner, M. Roetteler, and K. M. Svore, *Factoring using* $2n + 2$ *qubits with Toffoli based modular multiplication*, Quantum Information & Computation **17** (2017), nos. 7–8, 673–684. doi:10.26421/QIC17.7-8-7. arXiv:1611.07995.

23. T. Häner, S. Jaques, M. Naehrig, M. Roetteler, and M. Soeken, *Improved quantum circuits for elliptic curve discrete logarithms*, in Post-Quantum Cryptography—PQCrypto 2020, Lecture Notes in Computer Science, vol. 12100, Springer, 2020, 425–444. arXiv:2001.09580.

24. H. Hisil and C. Costello, *Jacobian coordinates on genus 2 curves*, Journal of Cryptology **30** (2017), 572–600. doi:10.1007/s00145-016-9227-7.

25. Y. Huang, Z. Su, F. Zhang, Y. Ding, and R. Cheng, *Quantum algorithm for solving hyperelliptic curve discrete logarithm problem*, Quantum Information Processing **19** (2020), article 62. doi:10.1007/s11128-019-2562-5.

26. Y. Huang, F. Zhang, Z. Hu, Z. Zhou, and L. Qu, *Improved quantum circuits for elliptic curve discrete logarithm problems on Ed25519*, Quantum Information Processing **24** (2025), article 311. doi:10.1007/s11128-025-04916-1.

27. A. Yu. Kitaev, *Quantum measurements and the Abelian stabilizer problem*, arXiv:quant-ph/9511026 (1995). arXiv:quant-ph/9511026.

28. N. Kornerup, J. Sadun, and D. Soloveichik, *Tight bounds on the spooky pebble game: Recycling qubits with measurements*, Quantum **9** (2025), 1636. arXiv:2110.08973.

29. T. Lange, *Weighted coordinates on genus 2 hyperelliptic curves*, Cryptology ePrint Archive, Report 2002/153, 2002. ePrint 2002/153.

30. R. Lidl and H. Niederreiter, *Finite Fields*, 2nd ed., Encyclopedia of Mathematics and its Applications, vol. 20, Cambridge University Press, 1996. doi:10.1017/CBO9780511525926.

31. H. Luo, Z. Yang, J. Luo, Z. Wang, Y. Su, X. Sun, L. Li, and T. Li, *Quantum algorithm for elliptic curve discrete logarithms with space-efficient point addition*, arXiv:2607.13816, version 3, 5 September 2026. arXiv:2607.13816v3.

32. A. Luongo, A. M. Miti, V. Narasimhachar, and A. Sireesh, *Measurement-based uncomputation of quantum circuits for modular arithmetic*, arXiv:2407.20167 (2024). arXiv:2407.20167v1. doi:10.48550/arXiv.2407.20167.

33. A. May and L. Schlieper, *Quantum period finding is compression robust*, IACR Transactions on Symmetric Cryptology **2022** (2022), no. 1, 183–211. Preprint: arXiv:1905.10074.

34. E. Ozbay Gurler and H. Hisil, *Complete group law for genus 2 Jacobians on Jacobian coordinates*, Journal of Cryptographic Engineering **14** (2024), 87–101. doi:10.1007/s13389-024-00351-7.

35. J. Proos and C. Zalka, *Shor's discrete logarithm quantum algorithm for elliptic curves*, Quantum Information & Computation **3** (2003), no. 4, 317–344. arXiv:quant-ph/0301141.

36. J. Renes, P. Schwabe, B. Smith, and L. Batina. μKummer: Efficient hyperelliptic signatures and key exchange on microcontrollers. In *Cryptographic Hardware and Embedded Systems—CHES 2016*, LNCS 9813, pages 301–320. Springer, 2016. Author version dated 26 January 2017, `https://cryptojedi.org/papers/mukummer-20170126.pdf`.

37. M. Roetteler, M. Naehrig, K. M. Svore, and K. Lauter, *Quantum resource estimates for computing elliptic curve discrete logarithms*, in Advances in Cryptology—ASIACRYPT 2017, Part II, Lecture Notes in Computer Science, vol. 10625, Springer, 2017, 241–270. arXiv:1706.06752.

38. P. Selinger, *Quantum circuits of* T*-depth one*, Physical Review A **87** (2013), 042302. arXiv:1210.0974.

39. J. Shallit and J. Sorenson, *A binary algorithm for the Jacobi symbol*, ACM SIGSAM Bulletin **27** (1993), no. 1, 4–11. doi:10.1145/152379.152384.

40. P. W. Shor, *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer*, SIAM Journal on Computing **26** (1997), no. 5, 1484–1509. arXiv:quant-ph/9508027.

A Complete Arithmetic and Terminal Identities

A.1 Complete clean Cantor evaluation

Cantor's formulas (6)–(7) cover every valid reduced input pair, including common support, inverse pairs, doubling, and the identity [6]. Two extended gcd computations give

$$g = e_1 u^{(1)} + e_2 u^{(2)} = \gcd(u^{(1)}, u^{(2)}), \qquad d = c_1 g + c_2(v^{(1)} + v^{(2)}) = \gcd(g, v^{(1)} + v^{(2)}),$$

with monic gcds. The coefficients $(a, b, c) = (c_1 e_1, c_1 e_2, c_2)$ supply the Bézout relation in the composition formulas, whose quotients are exact for valid inputs. The composed $\widetilde{u}$ has degree at most four, and $\widetilde{v}$ is reduced modulo it. If $\deg \widetilde{u} = 3$, then $(f - \widetilde{v}^2)/\widetilde{u}$ is quadratic; if $\deg \widetilde{u} = 4$, the quotient has degree at most two. The numerator is nonzero because f has odd degree. Thus at most one Cantor reduction is needed, including degree drops.

Fixed coefficient arrays, degree tags, and coherent activity flags allow both gcd computations and all divisions to be padded to their constant maximum lengths. Nonzero leading coefficients are normalized with the clean inverse of Section 2.2; inactive branches use $\mathrm{inv}_0(0) = 0$. After copying the canonical result, reversing all intermediate computations clears the workspace [4]. The number of field operations is bounded independently of n, giving the complete $O(n)$-qubit, $O(n^2)$-Toffoli reference used in Section 2.3.

A.2 An integer certificate for the terminal formula

Proposition 4.4 proves the terminal formula by interpolation and Cantor reduction. The following independent coefficient comparison uses a common interpolation denominator. Retain (26)–(27), set $\Sigma U_0 = U_0^{(2)} Z_1^2 + U_0^{(1)} Z_2^2$, and define

$$\begin{aligned}
j &= U_1^{(2)} Z_1^2 \mathcal{E}_* + Z_2^2 (Z_1^2 \mathcal{L}_* + U_1^{(1)} \mathcal{E}_*),\\
C_2 &= 2j, \qquad C_3 = 2\mathsf{A}_{\mathrm{sc}} \mathcal{E}_*, \qquad \mathcal{W} = -2\mathcal{K}\mathsf{A}_{\mathrm{sc}} \mathcal{R}_*,\\
\Lambda &= \mathcal{W} Z_1^4, \qquad \mathcal{B}_2 = C_2 Z_1^4, \qquad \mathcal{B}_3 = C_3 Z_1^4,\\
\mathcal{B}_1 &= V_1^{(1)} \mathcal{W} Z_1 + C_2 U_1^{(1)} Z_1^2 - C_3((U_1^{(1)})^2 - U_0^{(1)} Z_1^2),\\
a_5 &= 2\mathcal{B}_2 \mathcal{B}_3 - \Lambda^2, \qquad a_4 = \mathcal{B}_2^2 + 2\mathcal{B}_1 \mathcal{B}_3,\\
N_\alpha &= \mathsf{A}_{\mathrm{sc}} a_5 - \mathcal{B}_3^2 \Sigma U_1, \qquad D_\beta = \mathsf{A}_{\mathrm{sc}}^2 \mathcal{B}_3^2,\\
N_\beta &= \mathsf{A}_{\mathrm{sc}}^2 a_4 - \mathsf{A}_{\mathrm{sc}} \mathcal{B}_3^2 (\Sigma U_0 + U_1^{(1)} U_1^{(2)}) - \Sigma U_1 N_\alpha.
\end{aligned}$$

On the generic branch, $\mathcal{B}_2/\Lambda$ and $\mathcal{B}_3/\Lambda$ are the two highest interpolation coefficients. Comparing the x^5, x^4 coefficients gives $u_0^{(3)} = N_\beta / D_\beta$.

Substitution and collection of factors give

$$N_\beta = (2 Z_1^7 Z_2^3)^2 N_*, \qquad D_\beta = (2 Z_1^7 Z_2^3)^2 D_*. \tag{59}$$

These are polynomial identities over $\mathbb{Z}$, valid for every integer tuple; the sparse certificate checks coefficient equality. Their interpretation as divisor addition requires valid degree-two inputs with $Z_1 Z_2 \mathcal{R}_* \mathcal{E}_* \neq 0$. Under this condition, $D_* = (Z_1 Z_2 \mathcal{E}_*)^2$ is a nonzero square and $\Psi_\gamma = D_*(u_0^{(3)} + \gamma)$ preserves the completed quadratic character, including when $u_0^{(3)} + \gamma = 0$.

B A General Masked Reduction

Theorem B.1 (General masked reduction)**.** *Let G be a finite abelian group and consider an ordered addition tree with fresh random masks R. For every fixed input x, assume that the two true child values at every internal node are independent and uniform in G, as in Proposition 5.1. Let $F(x)$ be the unmasked group sum and T_R the total mask. At node v, let $\delta_v \geq 0$ bound the density of exceptional pairs in G^2, and set*

$$\varepsilon_{\mathrm{exc}} = \min\left\{1, \sum_v \delta_v\right\}.$$

Suppose a total clean diagonal implementation agrees, outside these exceptions, with

$$U_{R,\lambda}|x\rangle = g_\lambda(F(x) + T_R)|x\rangle, \qquad |g_\lambda| = 1,$$

where λ is independent of R. The input and any reference system are independent of the fresh seeds. Then the implemented and ideal randomized phase channels, with the classical seeds retained, satisfy

$$\frac{1}{2}\|\widetilde{\mathcal{E}} - \mathcal{E}\|_\diamond \le \min\{1, 2\sqrt{\varepsilon_{\mathrm{exc}}}\}.$$

For any common quantum channel and failure event, let p_0 and $\widetilde{p}_{\mathrm{fail}}$ be the ideal and implemented failure probabilities averaged over the seeds. Then

$$\widetilde{p}_{\mathrm{fail}} \le \min\left\{1, (\sqrt{p_0} + 2\sqrt{\varepsilon_{\mathrm{exc}}})^2\right\}.$$

Proof. Let $\mathcal{B}_R$ be the set of inputs for which at least one true child pair is exceptional. Uniformity and a union bound give $\Pr_R(x \in \mathcal{B}_R) \le \varepsilon_{\mathrm{exc}}$ for every fixed x. Outside $\mathcal{B}_R$, correctness at each node gives the ideal terminal phase by induction.

Write $\kappa = (R, \lambda)$ for a seed of probability p_κ. The ideal and implemented channels have coupled isometric extensions

$$W|\psi\rangle = \sum_\kappa \sqrt{p_\kappa}|\kappa\rangle_C|\kappa\rangle_E U_\kappa|\psi\rangle,$$
$$\widetilde{W}|\psi\rangle = \sum_\kappa \sqrt{p_\kappa}|\kappa\rangle_C|\kappa\rangle_E \widetilde{U}_\kappa|\psi\rangle.$$

For a normalized input with arbitrary reference, write $|\psi\rangle = \sum_x |x\rangle|\eta_x\rangle$, where $\sum_x \|\eta_x\|^2 = 1$. The two phases agree outside $\mathcal{B}_R$ and differ by at most two on it. Hence

$$\|(\widetilde{W} - W)|\psi\rangle\|^2 \le 4\mathbb{E}_R \sum_{x \in \mathcal{B}_R} \|\eta_x\|^2 \le 4\varepsilon_{\mathrm{exc}}.$$

Tracing out E removes coherence between distinct seeds while retaining their classical record in C. The isometry comparison and contractivity under partial trace give the half-diamond bound.

After an isometric extension of the common postprocessing channel, the ideal failure projection has norm $\sqrt{p_0}$. The triangle inequality bounds its norm on the implemented state by $\sqrt{p_0} + 2\sqrt{\varepsilon_{\mathrm{exc}}}$. Squaring gives the failure-probability bound. □

Independent uniform leaf masks give the required child-pair property for any binary tree; the two-seed construction extends to the tree shapes satisfying the determinant criterion in Section 5.3. Both bounds average over fresh seeds independent of the input. They do not give conditional bounds for every fixed seed or for inputs chosen after observing it.

C Reversible Implementation of the Residue Sampler

C.1 Modular arithmetic and Mersenne alternatives

Modular arithmetic and accumulation. Let $p < 2^b$ be an odd auxiliary modulus, with $b \ge 2$.

Base-field arithmetic is specified in Section 2.2; the following modules evaluate residues at p. Write $C_A(b)$ for the Toffoli cost of clean modular addition or subtraction.

We use the clean modular adders of [32, Propositions 3.4 and 3.10]. On canonical $0 \le x, z < p$ and a control bit g, they realize

$$|g, x, z, 0\rangle \longmapsto |g, x, (z + gx) \bmod p, 0\rangle,$$

preserving g, x and clearing arithmetic scratch. The uncontrolled and controlled bounds are $8b$ and $9b + 1$ Toffolis, with $b + 3$ scratch qubits. A classical addend and the modulus share one operand register, loaded only for the subroutines using them. Loading and unloading use X/CNOT gates, so the Toffoli bound is unchanged.

Doubling uses comparison and conditional subtraction [31, Appendix B]. Extend the target, form $2z$, compute $w = [2z \geq p]$, and subtract wp. Since p is odd, the low bit of $z' = 2z - wp$ is w, allowing a CNOT to clear the flag. With Cuccaro components [15,32], comparison and subtraction each cost at most $2(b+1)$ Toffolis. Thus

$$D_b \leq 4(b+1). \tag{60}$$

The constant operand, target extension, carry, and reduction flag fit within $b+4$ scratch qubits, reused between blocks. The inverse realizes modular halving,

$$H_p(z) = \begin{cases} z/2, & z \text{ even}, \\ (z+p)/2, & z \text{ odd}. \end{cases}$$

Both maps restore their sources and clear their scratch.

For arbitrary-target multiplication, apply $b-1$ halvings to z, then scan y from most to least significant bit, adding $y_j x$ and doubling between updates. For the initial target z_0, the final value satisfies

$$2^{b-1}\big(2^{-(b-1)} z_0 \bmod p\big) + x \sum_{j=0}^{b-1} y_j 2^j \equiv z_0 + xy \pmod{p}.$$

The initial halvings compensate for the factor 2^{b-1} introduced by the scan. Including these halvings and the interleaved doublings gives

$$C_M(b) \leq b(9b+1) + 2(b-1)\,4(b+1) = 17b^2 + b - 8.$$

For squaring, a separate clean qubit holds a CNOT copy of the source bit x_j. Clear the copy after the source is restored. These two CNOTs cost no Toffolis, so C_S has the same bound. The extra control qubit fits within the $b+4$ scratch allowance. For classical γ, precompute $k_j = 2^j \gamma \bmod p$. The b controlled updates $z \leftarrow z + x_j k_j \bmod p$ give $C_\gamma(b) \leq b(9b+1)$. The resulting primitive bounds are

$$\begin{aligned} A_{\text{prim}}(b) &\leq b+4, \\ C_M(b),\ C_S(b) &\leq 17b^2 + b - 8, \\ C_A(b) &\leq 8b, \qquad C_\gamma(b) \leq 9b^2 + b. \end{aligned} \tag{61}$$

These maps act on canonical residues and extend to superpositions by linearity. Their coherently reversible inverses have the same bounds. Including sources and target, the widths are at most $3b+4$ for addition, $4b+4$ for multiplication, and $3b+4$ for squaring or classical-constant multiplication. Appendix D.3 explains the Chen inventory's ancilla convention. Residue evaluation and CRT updates reuse workspace, so their allocations contribute through a maximum over stages.

Mersenne auxiliary moduli. For $p = 2^b - 1$, cyclic rotations implement doubling and halving [8, Section 3.2]. The canonical encoding excludes the all-one word and is preserved by rotation. For $z = \sum_{j=0}^{b-1} z_j 2^j$,

$$D(z) = 2z - z_{b-1}(2^b - 1) \equiv 2z \pmod{p}.$$

The reverse rotation is modular halving. Neither operation uses Toffolis, giving

$$\begin{aligned} A^{\text{Mer}}_{\text{prim}}(b) &\leq b+4, \qquad D^{\text{Mer}}_b = 0, \\ C^{\text{Mer}}_M(b),\ C^{\text{Mer}}_S(b) &\leq 9b^2 + b, \quad C^{\text{Mer}}_A(b) \leq 8b, \\ C^{\text{Mer}}_\gamma(b) &\leq 9b^2 + b. \end{aligned} \tag{62}$$

Since $D^b = I$, the $b-1$ initial halvings combine into one left rotation; no controlled rotation is needed. If $p = 2^d - 1$ is stored in a padded word with $d \leq b$, rotations act on the lowest d bits and preserve the high zero bits. This specialization requires a Mersenne auxiliary modulus. Product-based alternatives and their workspace costs follow below.

Canonical Mersenne folding. For $p = 2^b - 1$, writing $c = c_0 + 2^b c_1$ gives $c \equiv c_0 + c_1 \pmod{p}$. This reduction is used in [26, Appendix A, Fig. 17] for $p = 2^{521} - 1$. The following circuit supplies a canonical result and clean workspace.

Lemma C.1 (Resource certificate for canonical folding)**.** *Let $p = 2^b - 1$, $b \geq 2$, and let the preserved inputs satisfy $0 \leq c_0 \leq p$ and $0 \leq c_1 < p$. A clean zero-target circuit computes*

$$|c_0, c_1, 0\rangle \longmapsto |c_0, c_1, r\rangle, \qquad r = (c_0 + c_1) \bmod p \in \{0, \ldots, p-1\},$$

with $C_{\mathrm{fold}} \leq 10b - 2$ Toffolis and $A_{\mathrm{fold}} \leq b + 3$ additional clean qubits, excluding the inputs and result.

Proof. Copy c_0 to the zero result word and compute $e = [c_0 = p]$ by an AND ladder on its b bits. Flip all result bits when $e = 1$, then reverse the ladder. This normalizes the all-one copy to zero and leaves other values unchanged. The ladder and reversal cost $2(b-1)$ Toffolis with $b-1$ clean qubits. Add c_1 to the canonical result using [32, Proposition 3.4], at most $8b$ Toffolis and $b+3$ scratch qubits. The stages reuse scratch, and the reversible circuit preserves both inputs, proving the bounds. □

For $c = xy$ with canonical x, y, the split $c = c_0 + 2^b c_1$ satisfies $c_0 \leq p$ and $c_1 < p$. Thus the lemma applies, including the boundary $(c_0, c_1) = (p, 0)$.

Fused integer-product accumulation. Let a clean integer-product circuit U_{int} have interface

$$|x, y, 0^{2b}, 0^{A_{\mathrm{int}}}\rangle \longmapsto |x, y, xy, 0^{A_{\mathrm{int}}}\rangle,$$

Toffoli cost T_{int} and scratch A_{int}, excluding the preserved sources and the $2b$-bit product word C. Retain C, fold it into a temporary residue R, update an arbitrary canonical target z, and reverse the temporary computations:

$$U_{\mathrm{int}}; \quad \mathrm{Fold}(C, R); \quad z \leftarrow z + R \bmod p; \quad \mathrm{Fold}^\dagger; \quad U_{\mathrm{int}}^\dagger.$$

Lemma C.1 and $C_A \leq 8b$ give

$$\begin{aligned} C_{\mathrm{acc}}^{\mathrm{fold}} &\leq 2T_{\mathrm{int}} + 2C_{\mathrm{fold}} + C_A \leq 2T_{\mathrm{int}} + 28b - 4, \\ A_{\mathrm{acc}}^{\mathrm{fold}} &\leq 3b + \max\{A_{\mathrm{int}}, b + 3\}. \end{aligned}$$

The $3b$ temporary bits hold C, R, excluding x, y, z; the sequential stages reuse scratch.

Schoolbook multiplication and squaring. Initialize a $2b$-bit product word $C = 0$. For $j = 0, \ldots, b-1$, load $y_j x$ into a clean $(b+1)$-bit operand, add it modulo 2^{b+1} to positions $j, \ldots, j+b$ of C, and unload it. Before round j,

$$C = x(y \bmod 2^j) < 2^{b+j}.$$

Thus the higher bits vanish and the current slice is below 2^b. Adding $y_j x < 2^b$ cannot overflow its $b+1$ bits; the round adds exactly $2^j y_j x$, preserving lower positions. Induction gives $C = xy$, including zero inputs, and the final slice ends at position $2b - 1$.

Loading and unloading cost at most $2b$ Toffolis per round, and clean ripple addition costs $2b - 1$. The operand and one carry give

$$T_{\mathrm{int}} \leq 4b^2 - b, \qquad A_{\mathrm{int}} \leq b + 2.$$

For squaring, the same physical source supplies both operands: a CNOT loads each diagonal term $x_j x_j = x_j$, saving two Toffolis per round including unloading. Hence $T_{\mathrm{int,S}} \leq 4b^2 - 3b$ without a source-copy word. The folded canonical accumulators therefore satisfy

$$\begin{aligned} C_M^{\mathrm{fold,sb}}(b) &\leq 8b^2 + 26b - 4, \\ C_S^{\mathrm{fold,sb}}(b) &\leq 8b^2 + 22b - 4, \\ A_{\mathrm{acc}}^{\mathrm{fold,sb}}(b) &\leq 4b + 3. \end{aligned}$$

Including sources and target, total widths are at most $7b + 3$ for multiplication and $6b + 3$ for squaring. For integer $b \geq 25$, both gate bounds are below $9b^2 + b$, using $3b - 1$ more scratch than the rotation-based implementation.

A clean zero-target modular product U_P producing the canonical residue $R = xy \bmod p$ can instead be adapted by $U_P;\ z \leftarrow z + R \bmod p;\ U_P^\dagger$. If C_P, A_P are its Toffoli cost and scratch, excluding x, y, R, then

$$C_{\mathrm{acc}} \le 2C_P + C_A, \qquad A_{\mathrm{acc}} \le b + \max\{A_P, A_A\},$$

where A_A excludes the adder source and target. For $U_P = U_{\mathrm{int}}; \mathrm{Fold}; U_{\mathrm{int}}^\dagger$, the product module satisfies

$$C_P \le 2T_{\mathrm{int}} + 10b - 2, \qquad A_P \le 2b + \max\{A_{\mathrm{int}}, b + 3\}.$$

Its arbitrary-target accumulator therefore has cost

$$C_{\mathrm{acc}} \le 2C_P + C_A \le 4T_{\mathrm{int}} + 28b - 4.$$

The fused schedule saves two integer-product calls at the same stated peak width. A cited T-count or depth does not give an ordinary-Toffoli count, and total width alone does not specify clean scratch.

Use in the residue sampler. The rotation bounds in (62) substitute directly into Table 3, preserving the $b + 4$ scratch cap. Folded products require the larger $A_{\mathrm{acc}}^{\mathrm{fold}}$; their use requires recomputing the stage maximum in (43). These optimizations apply to the active auxiliary modulus p_i. A Mersenne base field q does not make the CRT primes Mersenne. The finite estimates in Section 6.1 use generic primitives at every auxiliary prime.

C.2 Local evaluation and scalar phase correction

Use the clean accumulation primitives and cost notation of Section 5.1. At auxiliary modulus p, the word length is at most b; source and target words are excluded from $A_{\mathrm{prim}}(b) \le b + 4$. Multiplication realizes

$$|x, y, z, 0\rangle \longmapsto |x, y, z + xy \bmod p, 0\rangle$$

for arbitrary canonical z, preserving x, y and clearing scratch. The finite primitive costs are given in (61) and (62).

Proposition C.2 (A cached clean evaluator)**.** *The polynomial map* (36) *has a clean arbitrary-target accumulation with ten preserved input words, five output words, eight local words, and primitive scratch. Its cost is*

$$C_{\mathrm{node}} \le 274C_M + 262C_S. \tag{63}$$

The scalar terminal uses the same eight-word local allocation and has cost $130C_M + 138C_S + C_\gamma$*. A measured five-coordinate node can be phase-corrected with the ten inputs, one scalar correction word, and the same eight local words at cost*

$$C_{\mathrm{ghost}} \le 638C_M + 776C_S. \tag{64}$$

All local words and primitive scratch are returned to zero.

Proof. Compute $\Delta U_0, \Delta U_1, \Delta V_0, \Delta V_1, \mathcal{C}_*$ in five words, then $\mathcal{R}_*, \mathcal{E}_*, \mathcal{L}_*$ in the remaining three. Erase $\mathcal{C}_*$ followed by the four differences, using fresh scratch for each reversal, and compute $\mathcal{K}^2$ in a freed word. This prepares four caches at cost $43C_M + 50C_S$. Retain them while processing A_*, N_*, B_1, and B_0. The first two quantities are shared across their output contributions and cleared before the next stage. Distributing the terms of N_* and B_i limits additional workspace to four words. Including reversal, the stage costs are

stage	M	S
prepare and clear the four caches	86	100
A_* contributions	28	36
N_* contributions	95	82
B_1 contribution	33	24
B_0 contribution	27	20
Z' contribution	5	0

The terminal accumulates N_* and γD_*, then clears the caches. Ghost correction computes one completed coordinate at a time into a zero scalar word, applies the recorded binary parity, and erases the word. Its scalar interface cannot assume output sharing and therefore needs a separate count. The detailed associations, allocations, and scalar correction counts follow below. □

A uniform local-move bound is thus $G_* = 2\max\{Q_w, 707A_b\}$ when $C_M, C_S, C_\gamma \le A_b$, where Q_w bounds leaf-lookup cost and A_b bounds the selected arithmetic primitives. Section 6.1 gives their finite values. The scalar ghost correction determines this constant, so twice the vector-evaluation cost alone would not suffice. The executable certificate checks the word traces as polynomial identities over $\mathbb{Z}$, including arbitrary-target reversal and recorded-phase correction.

Write the two inputs as $(a, c, s, u, z_1, b, d, t, v, z_2)$. Use local cache names $R = \mathcal{R}_*$, $E = \mathcal{E}_*$, $L = \mathcal{L}_*$, and $Q = \mathcal{K}^2$ from (27). Here the cache L is distinct from the tree leaf count.

Reproducible association rule. Let $\rho(e)$ count temporary words beyond the accumulation target and $\boldsymbol{c}(e) = (M, S)$ count multiplication and squaring calls. Available sources have rank and cost zero; a sum takes the maximum rank and adds costs. Set $\epsilon_e = 0$ for a source and 1 otherwise. For a product, compute operands not already available, multiply into the target, and clear them. Thus

$$\begin{aligned}\rho(ef) = \min\{\,&\max(\epsilon_e + \epsilon_e\rho(e), \epsilon_e + \epsilon_f + \epsilon_f\rho(f)),\\ &\max(\epsilon_f + \epsilon_f\rho(f), \epsilon_e + \epsilon_f + \epsilon_e\rho(e))\},\\ \boldsymbol{c}(ef) = {}&2\epsilon_e\boldsymbol{c}(e) + 2\epsilon_f\boldsymbol{c}(f) + (1, 0).\end{aligned}$$

A square has rank $\epsilon_e(1 + \rho(e))$ and cost $2\epsilon_e\boldsymbol{c}(e) + (0, 1)$. For an integer coefficient, repeat only the final primitive with the coefficient's sign, then clear the operands. All reversals are included in these recurrences.

Caches and associations. Compute $\Delta U_0, \Delta U_1, \Delta V_0, \Delta V_1, \mathcal{C}_*$ in five words. Their (M, S) costs are $(2, 4), (2, 4), (6, 8), (6, 8), (2, 0)$, with extra ranks $1, 1, 2, 2, 0$, respectively.

Compute R, E, L in the other three words at cost $(5, 1)$, then erase $\mathcal{C}_*$, both ΔU_i, and both ΔV_i. Each reversal is compiled for the scratch then available. Compute $Q = (z_1 z_2)^2$ at cost $(2, 1)$. Preparation costs $(43, 50)$, peaks at eight words, and leaves four caches. The erasure order prevents overwriting a live cache, which a direct reversal of preparation could do.

The following terms of N_* specify the order of multiplication. The ranks exclude the target and the four caches. The costs include cleanup.

term	ρ	M	S
$Q(L^2)$	1	1	2
$2(z_2^2)(a(EL))$	2	8	2
$(ab)(E^2)$	2	3	2
$-(z_1^2)(d(E^2))$	2	3	6
$(z_2^2)(c(E^2))$	2	3	6
$-2(z_2(z_2^2))(s(RE))$	3	10	4
$(z_1^2)((bQ)(R^2))$	3	7	6
$(z_2^2)((aQ)(R^2))$	3	7	6
total	3	42	34

We expand $B_1 = (z_2(z_2^2))(sR) - (z_2^2)(aL) - a(bE) - (z_2^2)(cE)$ and $B_0 = (z_2(z_2^2))(uR) - (z_2^2)(cL) - c(bE)$ as the displayed sums. Their costs are $(\rho, M, S) = (2, 14, 8)$ and $(2, 11, 6)$, respectively. The displayed A_* has $(2, 10, 14)$, and $T_* = LE - Q(R^2)$ has $(1, 2, 2)$. For any additional cached word W, the output associations are

expression	ρ	M	S
$(R^2)W$	1	1	2
$(WT_*)(R^2)$	2	3	2
$(R^2)(W(E^2))$	2	3	6
$(W(R^2))(E(E^2))$	3	5	8
$(z_1 z_2)(RE)$	2	5	0

These rules give the stages of Proposition C.2. Cache and clear A_*, T_* first, then N_*, T_*, and finally B_1, B_0 separately. With N_* cached, T_* and two product temporaries bring the local allocation to eight words. No vector output is used as a source. The terminal accumulates N_* directly, computes $D_* = Q(E^2)$ in a fifth word, accumulates γD_*, and clears the fifth word. Including cache preparation and reversal, the vector cost is $274C_M + 262C_S$ and the terminal cost is $130C_M + 138C_S + C_\gamma$.

Independent scalar correction. Scalar correction keeps the four caches and one initially zero scalar target. For each coordinate, compute its completed value, apply the recorded parity, and reverse the computation. The forward costs are

coordinate	U_1'	U_0'	V_1'	V_0'	Z'
maximum local words	7	7	8	8	6
M	21	56	76	118	5
S	30	78	136	94	0

Use the factored expression for U_1'. For U_0', V_1', Z', expand in the original coordinates and four cache sources, collect monomials, and enumerate all nonempty bipartitions of distinct variable-power factors. Apply the recurrence above, minimizing rank first and total cost second. For powers, enumerate splits into smaller positive powers, including a half-power square when the exponent is even. For V_0', cache N_*, T_*, accumulate their contribution, and clear them; then do the same with B_0. This finite association certificate is supplied with full traces in the code. Each scalar value has no garbage when its parity is applied. Including reversal, the total is

$$(86, 100) + 2(276, 338) = (638, 776),$$

which certifies 707 in $G_* = 2\max\{Q_w, 707A_b\}$. Twice the shared vector cost alone does not cover this scalar interface.

C.3 Measured tree evaluation

Every auxiliary modulus evaluates the same integer terminal polynomial, with the height bound of Section 5.1. The guarded CRT procedure of Section 5.2 therefore reconstructs its residue modulo q exactly. Its interpretation as group arithmetic holds on the good set of Theorem 5.3; no group-law branches are selected separately at the auxiliary moduli.

For (43), instantiate the balanced-tree measured-pebbling schedule of [11,28] with five-word vertices, a scalar terminal, and the certified local correction costs. A pebble stores five b-bit residue words; a ghost is the classical parity record of a measured vertex. Records for the same vertex combine by XOR. Both scalar inputs remain available.

Let F_j compute a height-j subtree root, leaving ghosts below it; F_0 loads a leaf. For $j \geq 1$, compute the left root by F_{j-1}, retain it while computing the right, evaluate the parent, and Hadamard-measure both children. If a recomputed vertex already has a ghost, apply its recorded parity to the vector and clear the record before retaining it. This cancels the old phase, so a vertex never carries both a pebble and a ghost.

Count a clean vertex evaluation, Hadamard erasure, or complete ghost correction as one move. The move counts satisfy

$$m_0 = 1, \qquad m_j = 2m_{j-1} + 3 = 2^{j+2} - 3 \quad (j \geq 1).$$

The same recursion requires at most $j + 2$ vector pebbles. At the full root, run F_{h-1} on both children, compute the scalar terminal, copy it to the arbitrary external XOR target, and uncompute the scalar. This combined terminal action, including both evaluations, is counted as one move. Erasing the children then gives $2m_{h-1} + 3 = 4L - 3$ moves and leaves only ghosts in the tree.

Ghosts are cleared in decreasing height order. For a height-j ghost, $j \geq 1$, compute both children with F_{j-1}, correct the parent phase using the scalar schedule in Appendix C.2, and erase the children. This takes $2m_{j-1} + 3 = 4 \cdot 2^j - 3$ moves and creates ghosts only at lower heights. A leaf ghost is corrected by lookup–parity–unlookup. Each correction begins and ends without vector pebbles, so the worst-case total is

$$(4L - 3) + L + \sum_{j=1}^{h-1} \frac{L}{2^j}(4 \cdot 2^j - 3) = (4h - 2)L + 3 \leq 8(h + 1)L.$$

At most $h + 2$ vector pebbles are live. The external target, eight local words, and scalar correction word require another $10b$ qubits, giving (43) after adding $A_{\mathrm{prim}}(b)$ scratch qubits. Each move costs at most $G_* = 2\max\{Q_w, 707A_b\}$, including the scalar ghost cost $638C_M + 776C_S$, the combined terminal action, and leaf correction. Hadamards, recorded parity phases, and resets add no Toffolis.

By the measured-erasure calculation in Section 5.1, every measurement outcome is uniform conditional on the earlier transcript. Clearing all ghosts cancels their phases, also for inputs entangled with a reference. For the root residue $R_p(x)$ and any binary target z, the resulting operation is

$$|x\rangle|z\rangle|0\rangle \longmapsto |x\rangle|z \oplus R_p(x)\rangle|0\rangle.$$

Both reconstruction passes in Section 5.2 use this clean interface. A fresh call clears a computed residue without reversing any measurement. After the character phase, the quotient correction is undone first; reversing the per-modulus updates then clears both accumulators.

C.4 State preparation, Fourier readout and character phase

Preparation and readout. For Theorem 3.3, combine cyclic-shift eigenvalue measurement [27], semiclassical feedback [21], and the modular updates above. Write $m_r = \lceil \log_2 r \rceil$, let J_{prep} be the trial cap, and take t from (14).

For classical $0 \le c_0 < r$ and quantum control d, use the controlled adder with addend and modulus sharing one operand word. Each subroutine restores the word before reloading. A controlled constant shift therefore costs at most $9m_r + 1$ Toffolis and $m_r + 4$ scratch qubits. If $r = 2^{m_r}$, ordinary controlled binary addition satisfies the same bounds.

Apply Hadamards to each input register and measure the cleanly computed predicate $x < r$. Acceptance leaves the exact uniform state on $\mathbb{Z}_r$; on rejection, reset and retry. Each trial succeeds with probability $r/2^{m_r} > 1/2$, so either register exhausting its cap has probability at most $2^{1-J_{\mathrm{prep}}}$. The comparator subtracts r, copies the high bit to the predicate flag, and undoes the subtraction, costing $2(2m_r - 1)$ Toffolis. Both registers together need fewer than four trials in expectation and at most $2J_{\mathrm{prep}}$, giving the preparation bounds.

Define

$$T_r|x\rangle = |x - 1 \bmod r\rangle, \qquad |\phi_j\rangle = r^{-1/2} \sum_{x=0}^{r-1} \omega_r^{jx} |x\rangle.$$

Since $T_r|\phi_j\rangle = \omega_r^j |\phi_j\rangle$, Fourier labels are cyclic-shift eigenphases. Perform t-bit phase estimation with one recycled control and semiclassical feedback [27,21]. For $0 \le h < t$, a controlled power shifts by the classical constant $-2^h \bmod r$. Store measured phase bits classically. The inputs use $2m_r$ qubits, shift workspace $m_r + 4$, and the control one, for $3m_r + 5$ in total. The $2t$ controlled shifts cost at most $2(9m_r + 1)t$ Toffolis.

The rounds on each register use at most $0, 1, \ldots, t-1$ individual dyadic feedback rotations, giving $2\sum_{j=0}^{t-1} j = t(t-1)$ across both registers. Alternatively, add the angles classically within each noninitial round and apply a single rotation, for at most $2(t-1)$ overall. Phase outcomes and retry counters use classical storage; rotation synthesis is counted separately.

On eigenstate ϕ_j, the outcome $z \in \{0, \ldots, 2^t - 1\}$ has probability

$$p_j(z) = \left| 2^{-t} \sum_{a=0}^{2^t-1} e^{2\pi i a (j/r - z/2^t)} \right|^2 .$$

Round $rz/2^t$ to the nearest integer modulo r. For positive circular distance $d_z = \mathrm{dist}(2^t j/r, z)$, the geometric-sum estimate gives $p_j(z) \le 1/(4d_z^2)$. A wrong label requires $d_z \ge D := 2^{t-1}/r$. By (14), $D \ge 2 + 1/\varepsilon_{\mathrm{F}} > 2$, so the two circular tails give

$$\Pr(\text{incorrect label}) \le \frac{1}{2} \sum_{a=\lfloor D \rfloor}^{\infty} a^{-2} \le \frac{1}{2(D-2)} \le \frac{\varepsilon_{\mathrm{F}}}{2}.$$

The measurement effects are diagonal in the Fourier basis. For any joint input, including an entangled state, decoding therefore acts as a classical noisy channel on the ideal joint labels. Couple each decoded label to its ideal value and apply a union bound to obtain total variation distance at most ε_{F}.

Finite Jacobi decomposition. The binary Jacobi algorithm [39,11] implements the clean completed-character phase (31). We give a fixed schedule and gate count for (32) and (33).

Initialize $(a, b, j) = (Y, q, 0)$, using the input register for a. For odd $d > 0$, set $e_2(d) = [d \bmod 8 \in \{3, 5\}]$ and $e_{\mathrm{rec}}(a, b) = [a \equiv b \equiv 3 \pmod 4]$. Run $T = 2n + 1$ macros. Each records a two-bit branch code and updates as follows, evaluating the sign correction on the old operands.

Condition	New (a, b)	$j \leftarrow j \oplus \delta$
$a = 0$	(a, b)	$\delta = 0$
$a > 0$ even	$(a/2, b)$	$\delta = e_2(b)$
a odd, $a \geq b$	$((a - b)/2, b)$	$\delta = e_2(b)$
a odd, $a < b$	$((b - a)/2, a)$	$\delta = e_{\mathrm{rec}}(a, b) \oplus e_2(a)$

The denominator b remains positive and odd. Every update with positive new first operand at least halves ab. Since $ab < 2^{2n}$ initially, the schedule reaches $a = 0$. Quadratic reciprocity and the supplementary law for 2 give

$$\left(\frac{a}{b}\right) = (-1)^{\delta} \left(\frac{a'}{b'}\right)$$

on each active branch, with $(0/1) = 1$. For $Y > 0$, the gcd stays one, so termination gives $b = 1$ and $(-1)^j = \chi_q(Y)$. For $Y = 0$, every macro is stationary and $j = 0$.

Each recorded update extends reversibly to all bit strings. An even word is halved by cyclic right rotation. The third branch subtracts b modulo 2^n and rotates; the fourth subtracts a from b, exchanges the words, and rotates the first. On valid inputs the differences are nonnegative and even. The inverse macro restores the old operands and sign bit, then recomputes the branch code by XOR to clear its record. After the forward schedule, apply Z to j, reverse all macros, and clear the initialization of b.

The two operands occupy $2n$ bits and the records $2T = 4n + 2$ bits. A sign bit, two flags, a ripple carry, a branch-enable bit, and a decomposition bit give total allocation $6n + 8$. Zero tests borrow $n - 2$ dirty bits from the other operand and restore them [1, Lemma 7.2], requiring no further linear workspace. Ripple comparison and controlled arithmetic use $O(n)$ Toffolis per macro with constant clean scratch [15]. The $O(n)$ forward and reverse macros give the quadratic gate bound below.

Encode the stationary, halving, subtraction, and exchange branches by $00, 01, 10, 11$. With $z = [a = 0]$, $\ell = [a < b]$, and low bit a_0, the code bits are a_0 and $(\neg a_0 \wedge \neg z) \oplus (a_0 \wedge \ell)$. Compute the flags, copy the code, and erase the flags before updating the operands.

For $n \geq 3$, a zero test costs $4n - 8$ Toffolis using the borrowed $n - 2$ bits [1, Lemma 7.2]. A comparator costs $2n - 1$, and gate-by-gate controlled ripple subtraction costs at most $3(2n - 3) + (5n - 7) = 11n - 16$ [15]. One forward macro therefore costs

$$\underbrace{12n - 18}_{\text{flags}} + \underbrace{11}_{\text{code/sign}} + \underbrace{22n - 32}_{\text{subtractions}} + \underbrace{2n - 1}_{\text{exchange/rotation}} + \underbrace{6}_{\text{enables}} = 36n - 34.$$

For $n = 2$, a controlled adder costs five Toffolis. A four-Toffoli comparator stores $(1 - a_0)b_0$ in carry c, XORs $(1 - a_1)b_1$ and $(1 - (a_1 \oplus b_1))c$ into the target, and clears c. With a one-Toffoli zero test, the macro costs at most $10 + 11 + 19 = 40 \leq 36n$. Missing high bits in the sign rule are zero. Both passes have $2n + 1$ macros, giving $C_\chi \leq 72n(2n + 1)$ and $W_\chi \leq 6n + 12$.

D Verification and Resource Comparisons

D.1 Height, parameters, and verification scope

Exact-integer checks expand the terminal and compact polynomials. Table 6 gives their degrees, coefficient norms, and weighted bidegrees. The terminal square-factor relations are verified as polynomial identities, beyond evaluation at sampled points. Each compact-output norm is below its corresponding bound in (37), preserving the height-growth estimate.

The sparse expansions in Table 6 are computed directly from (28) and (36). Represent a polynomial by a dictionary of integer coefficients, add termwise, and multiply by convolution. Combine equal exponent vectors and remove zero coefficients before computing $\|P\|_1 = \sum_\alpha |c_\alpha|$. The weighted bidegrees and coefficient norms

Table 6. Exact sparse-polynomial certificates. Coefficient norms are sums of absolute integer coefficients; bidegrees use weights $(2, 2, 3, 3, 1)$ on each input tuple.

Polynomial	Monomials	Degree	Coefficient norm	Weighted bidegree
N_*	115	19	232	$(12, 12)$
D_*	34	18	64	$(12, 12)$
U_1'	743	32	7512	$(20, 20)$
U_0'	837	31	9816	$(20, 20)$
V_1'	6623	48	577524	$(30, 30)$
V_0'	7603	47	621964	$(30, 30)$
Z'	44	15	60	$(10, 10)$

certify the recurrence in (40) and the terminal bound in (41) for all integer tuples, including exceptional tuples and zero scales.

An independent Cantor implementation checks all 1,296 and 4,356 ordered reduced pairs on the specified curves over $\mathbb{F}_5$ and $\mathbb{F}_7$, and 2,000 sampled pairs over each of $\mathbb{F}_{11}$ and $\mathbb{F}_{13}$. All generic outputs agree, including under independent input rescalings. The shared-expression certificate records the eight-word workspace and the separate node, terminal, and scalar-correction costs.

For $n = 127$, the general weighted height bound gives

L	h	t_{h-1}	$H = n + 10 + 12t_{h-1}$
16	4	1,022,736	12,272,969
32	5	20,454,736	245,456,969
64	6	409,094,736	4,909,136,969

These values determine the auxiliary-modulus allocations in Section 6.1. An independent height checker verifies the CRT margin using rational bounds for $\ln 2 = 2\sum_{k\ge 0}((2k+1)3^{2k+1})^{-1}$. After m terms, the remainder is at most $2\,3^{-(2m+1)}/((2m+1)(1-1/9))$; $m = 100$ suffices.

A complete small-field example. For $y^2 = x^5 + 2x^3 + x^2 + 2x + 1$ over $\mathbb{F}_3$, the pair $P = (x^2, x + 1)$ generates a subgroup of order 11. With $Q = [2]P$, a two-bit window and four leaves, `checks/check_complete_sampler.py` exhausts 121 scalar inputs, 121 group-seed pairs and three field shifts, giving 43,923 phase cases. It connects the masked tables, integer tree, streamed CRT, Jacobi phase, measured cleanup and Fourier output. The averaged useful probabilities are 256/363 for the ideal translated phase and 102544/5314683 for the polynomial implementation. Here $\varepsilon_{\mathrm{tree}} = 1$, so the example tests integration rather than a positive theoretical success bound. Integer and rational checks are exact; numerical Fourier comparisons use tolerance 2×10^{-12}. Word traces also check two fresh residue-XOR calls on inputs entangled with a two-dimensional reference, arbitrary targets, and all 2^{15} outcomes of one vector erasure.

Finite-field experiment and Fourier moments. The curve $y^2 = x^5 + 2x^3 + x^2 + 2x + 10$ over $\mathbb{F}_{503}$ has Jacobian of prime order $r = 252737$. Take

$$P = (x^2 + 501x + 1,\ 442x + 65), \qquad Q = [2]P,$$

and use $w = 9$ with $L = 4$. Enumeration gives $e = 505$; the exact fibre counts and Theorem 4.3 give $V_G = 0.99602369\ldots$. The masked-tree bound is positive for this instance. Table 7 compares the rigorous bounds with classical evaluation of the actual polynomial phase. The probabilities exclude preparation, Fourier approximation, and synthesis errors.

The estimator uses an exact Fourier moment identity rather than an $r \times r$ state-vector simulation. It evaluates field polynomials on sampled scalar inputs and independent public seeds. The Fourier identity and confidence bounds are given below. This experiment tests the finite-field phase and the exceptional-input estimate.

The program `check_moderate_sampler.py` verifies square-freeness, primality of r, $P \neq 0_G$, and $[r]P = 0_G$ for the $q = 503$ example. Direct point counts give

$$\#C(\mathbb{F}_q) = 505, \qquad \#C(\mathbb{F}_{q^2}) = 251455,$$

Table 7. Finite-field phase verification for $q = 503$, $r = 252737$, $w = 9$, $L = 4$. The two estimated quantities use 100000 independent trials. Their Hoeffding intervals hold simultaneously with probability at least 0.99; displayed half-widths are rounded upward.

Quantity	Value
Exact ideal useful probability	$0.9999760588\ldots$
Tree exception upper bound	$0.0836244792\ldots$
Polynomial useful-probability lower bound	$0.5885856443\ldots$
Estimated bad-input probability	0.02503 ± 0.00548
Estimated polynomial useful probability	0.95702 ± 0.02190

and hence $\#\operatorname{Jac}(C)(\mathbb{F}_q) = (505^2 + 251455)/2 - 503 = 252737$. The program enumerates the subgroup, checks that all r entries are distinct, and compares 1024 independently selected additions with Cantor arithmetic. It also compares 256 arbitrary-tuple vectorized evaluations with the integer polynomial formulas. The exact shift-averaged ideal useful probability is

$$1 - \frac{1}{qr^2} \sum_{\gamma \in \mathbb{F}_q} \left(e + \sum_a m_a s_q(a + \gamma) \right)^2 = \frac{32128854338528}{32129623558007}.$$

For a fixed public seed $\sigma = (U, V, \gamma)$, let $f_\sigma(a, b) \in \{-1, 1\}$ be the implemented polynomial phase, including dummy leaf tuples. Fourier orthogonality gives the mass on the useful line, excluding its origin, as

$$p_{\mathrm{use}}(\sigma) = \frac{1}{r} \sum_{t \in \mathbb{Z}_r} \left(\frac{1}{r} \sum_{b \in \mathbb{Z}_r} f_\sigma(t - kb, b) \right)^2 - \left(\frac{1}{r^2} \sum_{a,b \in \mathbb{Z}_r} f_\sigma(a, b) \right)^2 .$$

Draw $t, b, b', a_3, b_3, a_4, b_4$ independently and uniformly in $\mathbb{Z}_r$. The random variable

$$X = f_\sigma(t - kb, b) f_\sigma(t - kb', b') - f_\sigma(a_3, b_3) f_\sigma(a_4, b_4)$$

lies in $[-2, 2]$ and has conditional mean $p_{\mathrm{use}}(\sigma)$. Resampling both the public seed and these variables on every trial therefore gives an unbiased estimate of the averaged useful probability, even when the polynomial phase is not exactly periodic. The identity is independently checked against a complete 11×11 Fourier transform of a random sign function.

With N independent trials, Hoeffding's inequality gives half-width $\sqrt{8 \log(2/\delta)/N}$ for the mean of X. The indicator of a bad true-child pair has half-width $\sqrt{\log(2/\delta)/(2N)}$. Taking $N = 100000$ and $\delta = 0.005$ for each estimate gives simultaneous confidence at least 0.99 by a union bound. The program checks phase agreement with the translated ideal function on every sampled input outside the true-tree bad set.

Classical parameter certificates. The parameters are from the Gaudry–Schost instance used by μKummer [36]. Independent certificates verify primality of q, r, $\operatorname{ord}(P) = r$, and $\#\operatorname{Jac}(C')(\mathbb{F}_q) = 16r$.

Lucas–Lehmer certifies q, and a recursive Pocklington certificate certifies r. Exact checks of square-freeness, membership, $P \neq 0_G$, and $[r]P = 0_G$ establish $\operatorname{ord}(P) = r$. For $s = \lfloor \sqrt{q} \rfloor + 1$, the genus-two Weil interval lies in $[B_-, B_+]$ with $B_\pm = q^2 + 6q + 1 \pm 4(q + 1)s$. Exact division gives $\lceil B_-/r \rceil = \lfloor B_+/r \rfloor = 16$. Since r divides the group order, the order is $16r$. The certificate contains 35 recursively certified primes, with leaves checked by trial division.

The computational supplement supplies the sparse-polynomial, modular-word, sampling and pebbling checks through `run_checks.py`, together with exact certificates and the finite resource ledger. Its `VERIFICATION_MAP.md` associates these checks with the relevant sections and numerical claims. The resource checks cover arbitrary-target products, canonical folds, Mersenne rotations and workspace restoration. They verify integer identities and arithmetic-word traces, including reference-state phases. The finite-field phase experiment uses classical sampling; these checks do not compile the full elementary-gate sampler. Adder correctness and gate decompositions come from [32]. A cryptographic-size fault-tolerant layout and Clifford+T depth require a separate compilation and schedule [38].

D.2 Polynomial gate and storage bounds

For the modulus choice used in Theorem 6.1, $w = \Theta(\log_2 n)$ gives $L = O(n/\log_2 n)$, $h = O(\log_2 n)$ and $H = O(nL^{\log_2 20})$ by (41). Choose all odd primes up to 2^b, excluding q, with $b = O(\log_2(H+n)+w) = O(\log_2 n)$ sufficiently large for the prime-product bound in Section 6.1 and $w - 1 \le 8b$. Then K is polynomial in n and $a_{\mathrm{fp}} = O(\log_2 n)$. The resulting success and width bounds are proved in Theorem 6.1.

An explicit polynomial gate bound. For a concrete asymptotic choice, take $w = \lceil \log_2 n \rceil$ and write $\alpha = \log_2 20$. Then

$$L = O(n/\log n), \qquad H = O\left(\frac{n^{1+\alpha}}{(\log n)^{\alpha}}\right).$$

Choose $b = \lceil \log_2(cH) \rceil$ for a sufficiently large fixed constant c. The prime-product estimate supplies the required CRT margin, and the prime-count estimate gives

$$K = O(H/\log H) = O\left(\frac{n^{1+\alpha}}{(\log n)^{1+\alpha}}\right).$$

The local arithmetic cost is $O(b^2) = O(\log^2 n)$, while the explicit lookup bound is $Q_w = 4w2^w = O(n \log n)$. Thus $G_* = O(n \log n)$ and $(h+1)L = O(n)$. In the finite ledger, tree evaluation costs $O(n^2 \log n)$ per auxiliary prime; the other reconstruction terms are smaller. Consequently the arithmetic Toffoli bound is

$$O\left(\frac{n^{3+\log_2 20}}{(\log n)^{\log_2 20}}\right), \qquad 3 + \log_2 20 \approx 7.322.$$

This bound includes reversible table selection and cleanup. With fixed or inverse-polynomial preparation and Fourier error budgets, their additional costs do not change this exponent; rotation synthesis retains its separately stated cost. Classical storage is

$$O(nL2^w + Kb) = O\left(\frac{n^3}{\log n} + \frac{n^{1+\log_2 20}}{(\log n)^{\log_2 20}}\right) \quad \text{bits.}$$

The large exponents quantify the cost behind the polynomial-time statement; they are upper bounds for this schedule, not lower bounds.

D.3 The reported Chen arithmetic inventory

For prime $q = 2^n - 1$, the table reproduces the local costs in [8, Table 1], with $\ell_n = \lceil \log_2 n \rceil$. It retains the source's primitive names; extra controls are counted separately.

Primitive	Total qubits	Ancillas	Toffolis
Addition/subtraction	$3n + 1$	$n + 1$	$6n - 4$
Constant addition/subtraction	$3n + 1$	$2n + 1$	$6n - 4$
Doubling/halving	$n + 1$	1	0
Multiplication	$4n + 2$	$2n + 2$	$n(6n - 4)$
Constant multiplication	$4n + 2$	$3n + 2$	$n(6n - 4)$
Squaring	$3n + 3$	$2n + 3$	$n(6n - 4)$
Inversion	$7n + 2\ell_n + 9$	$5n + 2\ell_n + 9$	$2n(8n - 3)$

Ancillas and register allocation. The Ancillas column uses the source's terminology, whereas A_{prim} in Section 5.1 excludes source and result words. For multiplication, $(4n + 2) - (2n + 2) = 2n$ accounts for the two inputs, although the out-of-place interface also has an n-bit result. Hence this column cannot be added as clean scratch when the result is already counted. At $b = n$, our total $3n + (n + 4) = 4n + 4$ alone establishes neither interface equivalence with the reported $4n + 2$ nor an n-qubit saving. The source also describes its inversion quantity as total width in the table and auxiliary space in the allocation discussion. Any reallocation therefore needs a live-register schedule; sequential peak width depends on simultaneous live registers.

Canonical zero and controls. End-around carry in [8, Section 3.1, Fig. 2] uses $2^n \equiv 1 \pmod{q}$, but leaves the all-one word when $x + y = q$ without binary overflow, as for $q = 7, x = 3, y = 4$. Bitwise complementation likewise maps zero to the all-one word. This word represents zero modulo q but lies outside our canonical set $\{0, \ldots, q-1\}$. Direct use in our interface therefore requires zero normalization, coherent cleanup and the required controls. We retain the reported $6n - 4$ cost in the comparison; our Mersenne primitives use the clean canonical adder in Appendix C.1.

Operation inventory. Each outer transform uses $(A, M, S, I) = (40, 42, 2, 4)$ and the middle transform uses $(5, 10, 4, 6)$. The total $(85, 94, 8, 14)$ gives

$$85(6n - 4) + 102n(6n - 4) + 28n(8n - 3) = 836n^2 + 18n - 340.$$

For the nominal $4n$ additions, this expands to $3344n^3 + 72n^2 - 1360n$; the source prints a negative quadratic term. Section 6.2 uses the actual $2m_r$ additions and retains the reported width $20n + 2\ell_n + 10$ and generic domain. The correction concerns the inventory expansion and leaves the reported circuit interfaces unchanged.