SmallWood–vSIS: Equations, Packing, and PACS Construction

1 Setting and Notation

Field & ring scope. All polynomial commitments, degrees, and checks live over the base field $\mathbb{F}_q$ (odd characteristic). We model ring objects $R_q = \mathbb{F}_q[X]/(X^\phi + 1)$ *coefficient-wise*: every ring vector/matrix is viewed as a block over $\mathbb{F}_q$ with ϕ coefficients per ring coordinate, and all constraints are enforced coeff.-wise over $\mathbb{F}_q$.

Support & invertibility. We fix an *anchor set* (support)

$$\Omega = \{\omega_1, \dots, \omega_s\} \subset \mathbb{F}_q$$

of distinct points used for packing s column slots per row, and assume $S_0 := |\Omega|$ is invertible in $\mathbb{F}_q$.

We write $\langle a, b\rangle$ for standard inner products over $\mathbb{F}_q$ and use $\odot$ for the Hadamard (componentwise) product; $b \oslash c$ denotes Hadamard division (componentwise) when the denominator coordinates are nonzero.

Radix and bound. We will use a radix $R = 2^W < q$ (any invertible radix works) and choose parameters so that the target norm bound $\beta^2 < q$, avoiding wrap-around in integer comparisons embedded in $\mathbb{F}_q$.

2 From vSIS to a Proof-Friendly BBS-Form Signature

We work over a cyclotomic ring $R_q = \mathbb{F}_q[X]/(X^\phi + 1)$ and view vectors and matrices over R_q *coefficient-wise* as explained above. The vSIS-based signature template signs a message μ by evaluating a rational function $h_{\mu,\chi}$ at public parameters B and sampling a short preimage under a trapdoored matrix A:

$$\begin{aligned}
&\mathsf{Sign}: && \chi \leftarrow \mathcal{X}, \qquad t \leftarrow h_{\mu,\chi}(B) \bmod q, \qquad s \leftarrow \mathsf{Preimage}(A, t; \mathrm{td}_A),\\
& && \sigma := (\chi, s).\\
&\mathsf{Verify}: && \text{Check } As \equiv h_{\mu,\chi}(B) \pmod q \text{ and } \|(s^\top, \mu^\top, \chi^\top)\| \le \beta. && (1)
\end{aligned}$$

This generic pattern (with a message subspace guard $\mu \in \mathcal{M}_{\mathcal{X},H,B}$ to rule out undefined denominators) is standard for vSIS/GenISIS$_f$ signatures.

BBS-type instantiation. Following Dubois–Klooß–Lai–Woo, we pick a keyed rational function

$$h_{\mu,\chi}(B) \;=\; B_0 \cdot (1;\, u;\, x_0) \;\oslash\; \big(b_1 - \mathbf{1}_n \cdot x_1\big),$$

where $B = (B_0, b_1)$ is part of the public key, the message is u, and the signing randomness $\chi = (x_0, x_1)$ consists of ring elements with bounded norm; division and subtraction are taken *componentwise* (i.e. in the Hadamard sense) across the n verification coordinates. With this choice, verification is

$$A\,s \;\equiv\; B_0 \cdot (1;\, u;\, x_0) \;\oslash\; \big(b_1 - \mathbf{1}_n \cdot x_1\big) \pmod q, \qquad \|(s^\top, u^\top, x_0^\top, x_1)\| \le \beta. \qquad (2)$$

vSIS template (signing and verification). Public: B, A with trapdoor td_A; message μ; randomness χ; rational map $h_{\mu,\chi}$. Sign: sample χ, compute $t = h_{\mu,\chi}(B)$, output (χ, s) with $As \equiv t \pmod q$ and $\|s\| \le \beta$. Verify: check $As \equiv h_{\mu,\chi}(B) \pmod q$, validity of μ, χ, and the global norm bound.

Figure 1: The vSIS signing paradigm specialized later to the BBS form via $h_{\mu,\chi}(B) = B_0(1; u; x_0) \oslash (b_1 - \mathbf{1}_n x_1)$.

Security rationale. The tuple (χ, s) is existentially unforgeable under the (strong) hinted vSIS assumption tailored to the keyed rational family $H = \{h_{\mu,\chi}\}$: any forgery $As = h_{\mu,\chi(B_0,b_1) \bmod q}$ with $\|s\| \le \beta$ gives a short linear combination of $\{f(A)\}_{f\in\mathcal{F}}$ that evaluates to $g^{\in G\setminus Q}$ after having seen the Q selective hints precisely the s-Hint-vSIS task. This underpins strong selective-query security for our BBS-instantiation and via the $\mathrm{GenISIS}_f \Rightarrow \mathrm{IntGenISIS}_f$ lifting yields full strong security (CMA) under the interactive version.

Clearing the denominator and Hadamard structure. Equation (2) is equivalent, coordinatewise, to

$$\big(b_1 - \mathbf{1}_n x_1\big) \odot (As) - B_0 \cdot (1;\, u;\, x_0) = 0_n \pmod{q},$$

where $\odot$ denotes the Hadamard product. Distributing the product gives

$$(b_1 \odot A)\, s - (As)\, x_1 - B_0 \cdot (1;\, u;\, x_0) = 0_n \pmod{q}, \tag{3}$$

which is the form used in our notes and throughout this paper. Here, $b_1 \odot A$ means: multiply each column of A by the corresponding coordinate of b_1.

Why this is *proof-friendly*. Equation (17) has two decisive properties:

1. **Bounded degree (quadratic).** Every coordinate is a polynomial of degree ≤ 2 in the secrets (s, u, x_0, x_1): the terms $(b_1 \odot A)s$ and $B_0(1; u; x_0)$ are linear, and the only nonlinear term is the bilinear product $(As)\, x_1$.
2. **Norm-bounded satisfiability.** Apart from (17), verification only asks that the witness (s, u, x_0, x_1) is short:
$$\big\|(s^\top, u^\top, x_0^\top, x_1)\big\|_2 \le \beta, \tag{4}$$
with parameters chosen strictly below q. Consequently, we encode the bound via low-degree digit/carry constraints (for ℓ_2), avoiding bit-decomposing the full witness.

3 Target Relation (vSIS–BBS Form) and Norm Bound

Public data comprise matrices/vectors $(A, B_0, \mathbf{b}_1)$ and a message $\mathbf{u}$; the secret witness is $(\mathbf{s}, \mathbf{x}_0, x_1)$. Coordinate-wise (coeff.-wise in R_q), the verification identity is a bounded-degree (quadratic) equation:

$$(\mathbf{b}_1 \odot A)\, \mathbf{s} - (A\mathbf{s})\, x_1 - B_0 \cdot (1; \mathbf{u}; \mathbf{x}_0) \equiv 0 \pmod{q}. \tag{5}$$

Additionally, we enforce a norm bound on the full witness

$$\big\|(\mathbf{s}, \mathbf{u}, \mathbf{x}_0, x_1)\big\|_2 \le \beta. \tag{6}$$

To keep all constraints of degree ≤ 2, we expand a multiplicative consistency witness $w_3 = w_1 \cdot w_2$ with

$$w_1 := \begin{pmatrix} \mathbf{s} \\ \mathbf{u} \\ \mathbf{x}_0 \end{pmatrix}, \qquad w_2 := x_1, \qquad w_3 := w_1 \cdot w_2,$$

where $\cdot$ is column-wise (per slot) multiplication.

Security rationale. *Completeness* follows from trapdoor preimage sampling for A, which outputs a short s so that (5) and (6) hold. *Soundness* for the norm claim is enforced by our degree-≤ 2 digit/bit/carry constraints (Sec. **??**); any prover that exceeds β must violate at least one parallel bitness identity or an aggregated limb identity, which are caught by the PACS PIOP and PCS binding guarantees.

4 Packing: Rows as Low-Degree Polynomials over Ω

Each *witness row* i produces a polynomial $P_i(X) \in \mathbb{F}_q[X]$ of degree $\leq s + \ell - 1$ such that

$$P_i(\omega_k) = w_{i,k} \quad (k = 1, \ldots, s),$$

and we *blind* it at ℓ extra points $r_1, \ldots, r_\ell \notin \Omega$ by prescribing $P_i(r_j) = y_j$ with random $y_j \leftarrow \mathbb{F}_q$. Public column-dependent objects (rows of A, $\mathbf{b}_1$, blocks of B_0) are interpolated into fixed polynomials $\Theta'(X)$ of degree $\leq s - 1$ that agree with their respective values on Ω.

Notation. We use ℓ for the number of *extra interpolation points per row* (LVCS masks on the grid) and ℓ' for the number of *PCS mask rows per polynomial* in Sec. 7. In our implementation we set $\ell' = \ell$ for simplicity.

A Detailed Witness Matrix

We pack all secrets (and auxiliary values) into a rectangular matrix $W \in \mathbb{F}_q^{r \times s}$ whose columns are indexed by Ω and whose rows are grouped by role:

Block / Row	ω_1	ω_2	$\cdots$	ω_s
Signature (packed coefficients; rows to be squared in Sqs)				
Signature s_1	$s_{1,1}$	$s_{1,2}$	$\cdots$	$s_{1,s}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Signature $s_{m_{\text{sig}}}$	$s_{m_{\text{sig}},1}$	$s_{m_{\text{sig}},2}$	$\cdots$	$s_{m_{\text{sig}},s}$
Public/auxiliary rows (row-constant across columns)				
Message u_1	u_1	u_1	$\cdots$	u_1
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Message u_{m_u}	u_{m_u}	u_{m_u}	$\cdots$	u_{m_u}
Rand $x_{0,1}$	$x_{0,1}$	$x_{0,1}$	$\cdots$	$x_{0,1}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Rand x_{0,m_0}	x_{0,m_0}	x_{0,m_0}	$\cdots$	x_{0,m_0}
Scalar x_1	x_1	x_1	$\cdots$	x_1
Linearized products for $(As)\,x_1$ (only these are needed)				
Product $(x_1 \cdot s_1)$	$x_1 s_{1,1}$	$x_1 s_{1,2}$	$\cdots$	$x_1 s_{1,s}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Product $(x_1 \cdot s_{m_{\text{sig}}})$	$x_1 s_{m_{\text{sig}},1}$	$x_1 s_{m_{\text{sig}},2}$	$\cdots$	$x_1 s_{m_{\text{sig}},s}$
Norm accumulator and base-R expansion				
Accumulator Sqs	$\sum_t s_{t,1}^2$	$\sum_t s_{t,2}^2$	$\cdots$	$\sum_t s_{t,s}^2$
Digits D_0	$D_{0,1}$	$D_{0,2}$	$\cdots$	$D_{0,s}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Digits D_{L_S-1}	$D_{L_S-1,1}$	$D_{L_S-1,2}$	$\cdots$	$D_{L_S-1,s}$
Bits for D_ℓ : $\text{bit}_{\ell,0}$	$\text{bit}_{\ell,0}(1)$	$\text{bit}_{\ell,0}(2)$	$\cdots$	$\text{bit}_{\ell,0}(s)$
$\vdots\ (\ell = 0..L_S-1,\ u = 0..W-1)$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Bits for D_ℓ : $\text{bit}_{\ell,W-1}$	$\text{bit}_{\ell,W-1}(1)$	$\text{bit}_{\ell,W-1}(2)$	$\cdots$	$\text{bit}_{\ell,W-1}(s)$
Remainders T_0	$T_{0,1}$	$T_{0,2}$	$\cdots$	$T_{0,s}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Remainders T_{L_S}	$T_{L_S,1}$	$T_{L_S,2}$	$\cdots$	$T_{L_S,s}$
Column-constant limbs (aggregation)				
Carries C_0	C_0	C_0	$\cdots$	C_0
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Carries C_{L_S-1}	C_{L_S-1}	C_{L_S-1}	$\cdots$	C_{L_S-1}
Slack Δ_0	Δ_0	Δ_0	$\cdots$	Δ_0
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Slack Δ_{L_S-1}	Δ_{L_S-1}	Δ_{L_S-1}	$\cdots$	Δ_{L_S-1}

$W \in \mathbb{F}_q^{r \times s}$

Remarks. (i) The top block contains the m_{sig} signature rows with genuinely different values across columns.
(ii) The message/randomness blocks are *column-constant* (each entry repeats across all ω_k).
(iii) The *product* rows implement $w_3 = w_1 \cdot w_2$ (elementwise).
(iv) The digits D_ℓ *vary across columns* (they encode per-column base-R digits and interpolate

to low-degree rows), while the carries C_ℓ and slack Δ_ℓ are *column-constant* rows (degree 0), as required by the aggregated comparison.

For each row i above we interpolate a polynomial $P_i(X)$ with $P_i(\omega_k) = W_{i,k}$ and degree at most $s + \ell - 1$ (plus ℓ blinded points).

Security rationale. Interpolating each row to degree $\leq s + \ell - 1$ and fixing ℓ random *mask* evaluations makes the opened evaluations statistically independent of the true row contents, conditioned on correctness. This is the HVZK guarantee provided at LVCS level and lifted to PCS. Degree boundedness is enforced later by DECS.

5 Constraint Polynomials: Parallel and Aggregated

Let $W_1^{(k)}(X)$ denote the polynomial encoding the k-th column of w_1 (the stacked block $\mathbf{s}, \mathbf{u}, \mathbf{x}_0$), $W_2(X)$ that of $w_2 = x_1$, and $W_3^{(k)}(X)$ the column of w_3.

Parallel (per-column) multiplicativity, degree ≤ 2. For each column k,

$$F_k(X) \;:=\; W_3^{(k)}(X) \;-\; W_1^{(k)}(X)\, W_2(X) \;\equiv\; 0. \tag{7}$$

Aggregated (row-sum) BBS identity, degree ≤ 2. For each relevant public row index j,

$$F'_j(X) \;:=\; \big((\mathbf{b}_1 \odot A)_j(X) \cdot S(X)\big) \;-\; \big(A_j(X) \cdot S(X)\big)\, X_1(X) \;-\; B_{0,j}(X) \;\equiv\; 0, \tag{8}$$

where $S(X)$ is the (formal) stacked polynomial for $\mathbf{s}$ and $X_1(X)$ for x_1, and $B_{0,j}(X)$ is the affine combination of $(1; \mathbf{u}; \mathbf{x}_0)$ routed through B_0.

Remark (Why parallel constraints are needed). Aggregated constraints only enforce that *sums over columns* vanish, e.g. $\sum_k f'_j(\cdot) = 0$. They do not, by themselves, force *local* multiplicative consistency in each column. The parallel constraints $F_k = W_3^{(k)} - W_1^{(k)} W_2 = 0$ are what pin down the bilinear part $(As)\, x_1$ column-by-column while keeping the overall system at degree ≤ 2. This split parallel for local products; aggregated for row-sums (like the BBS cleared form) is exactly how PACS is defined and checked in the SmallWood PIOP.

6 Batching and Masked Polynomials

Let d_Q be the declared degree bound for the batches, large enough for M_i and the linear combinations below. We form ρ batched polynomials

$$Q_i(X) \;:=\; M_i(X) \;+\; \sum_{t=1}^{m_1} \Gamma'_{i,t}\, F_t(X) \;+\; \sum_{u=1}^{m_2} \gamma'_{i,u}\, F'_u(X), \tag{9}$$

where Γ', γ' are verifier challenges (Fiat–Shamir) and M_i are *mask* polynomials chosen by the prover, independent of (F_t, F'_u).

The protocol enforces the Ω-sum condition

$$\sum_{\omega \in \Omega} Q_i(\omega) \;=\; 0 \qquad \text{for each } i = 1, \ldots, \rho. \tag{10}$$

Since $F_t(\omega) = 0$ for all $\omega \in \Omega$ by (7), (10) boils down to canceling the aggregated contribution with M_i. Let $M_i(X) = \sum_{k=0}^{d_Q} a_k X^k$ and $S_k = \sum_{\omega \in \Omega} \omega^k$ for $k = 0, \ldots, d_Q$ with $S_0 = |\Omega|$

invertible in $\mathbb{F}_q$. Choosing $a_1, \dots, a_{d_Q} \leftarrow \mathbb{F}_q$ at random and setting

$$a_0 \;:=\; -\,S_0^{-1}\left(\sum_{k=1}^{d_Q} a_k S_k \;+\; \sum_t \Gamma'_{i,t} \sum_{\omega\in\Omega} F_t(\omega) \;+\; \sum_u \gamma'_{i,u} \sum_{\omega\in\Omega} F'_u(\omega)\right) \tag{11}$$

ensures (10).

Remark (On a_0 in Eq. (10): hiding and what the verifier sees). We sample $a_1, \dots, a_{d_Q} \leftarrow \mathbb{F}_q$ uniformly and set

$$a_0 \;:=\; -S_0^{-1}\left(\sum_{k=1}^{d_Q} a_k S_k \;+\; \sum_t \Gamma'_{i,t} \sum_{\omega\in\Omega} F_t(\omega) \;+\; \sum_u \gamma'_{i,u} \sum_{\omega\in\Omega} F'_u(\omega)\right),$$

which enforces $\sum_{\omega\in\Omega} Q_i(\omega) = 0$. Conditioned on this single linear relation, Q_i (and M_i) are uniformly distributed over the affine subspace $\{\sum_\Omega Q_i = 0\}$, which is exactly the distribution used by the HVZK simulator hence a_0 *is* hiding. The verifier never learns the coefficients $\{a_j\}$; they only see: (i) a few *evaluations* $Q_i(e)$ at random points $e \in E'$, (ii) the corresponding LVCS masked linear-combo vectors for the rows, and (iii) Merkle paths/degree checks. This view is statistically independent of the hidden coefficients beyond the enforced equalities.

7 Commitments and Openings (LVCS $\to$ PCS): detailed for the simulation

We explain how the witness rows $P_i(X)$ are committed and later opened at verifier-chosen points. The stack has three logical layers:

$$\textsf{DECS (degree \& function binding)} \;\Rightarrow\; \textsf{LVCS (linear map on rows)} \;\Rightarrow\; \textsf{PCS}$$

(A) DECS layer: commit degree-bounded rows and bind linear combos

Let $r = \#\{\text{rows}\}$ and η the DECS repetition parameter. The prover holds the row polynomials $P_1, \dots, P_r \in \mathbb{F}_q[X]$ in *coefficient* form, degree $\le D$ and samples η mask polynomials $M_1, \dots, M_\eta \in \mathbb{F}_q[X]$ of degree $\le D$. The leaf at grid index i contains

$$\big(P_1(\omega_i), \dots, P_r(\omega_i);\; M_1(\omega_i), \dots, M_\eta(\omega_i);\; i, \rho_i\big),$$

which is hashed into a Merkle tree with root root. The verifier derives an $\eta \times r$ challenge matrix $\Gamma = (\Gamma_{k,j})$ from root. The prover replies with

$$R_k(X) \;:=\; M_k(X) \;+\; \sum_{j=1}^{r} \Gamma_{k,j}\, P_j(X), \qquad k = 1, \dots, \eta, \tag{12}$$

and the verifier checks $\deg R_k \le D$ and, on a small random opened set $E \subseteq \{\omega_i\}$ (with Merkle paths), the *masked linear relation*

$$R_k(\omega) \;\overset{?}{=}\; M_k(\omega) \;+\; \sum_{j=1}^{r} \Gamma_{k,j}\, P_j(\omega) \quad \text{for all } \omega \in E,\; k = 1, \dots, \eta. \tag{13}$$

Equation (13) is exactly the DECS binding test at the opened grid points and degree $\le D$ is enforced globally on R_k. This layer gives *function-binding for linear combinations at the opened positions* and degree soundness for the committed family $\{P_j\}$.

(B) LVCS layer: commit a *matrix* of rows and open linear maps

We now view the prover's input as a row matrix $R \in \mathbb{F}_q^{r \times n_{\text{cols}}}$ (the first n_{cols} evaluation slots on the grid Ω). To hide the precise values, the prover appends ℓ fresh random *mask coordinates* per row, obtaining $\tilde{R} \in \mathbb{F}_q^{r \times (n_{\text{cols}}+\ell)}$:

$$\tilde{R}_j \;=\; (\, R_j \parallel \bar{R}_j \,), \qquad \bar{R}_j \;\in\; \mathbb{F}_q^{\ell} \;\text{ uniform.}$$

Each $\tilde{R}_j$ is *interpolated* to a polynomial $P_j(X)$ of degree $\le n_{\text{cols}} + \ell - 1$ such that $P_j(\omega_i) = \tilde{R}_j[i]$ for all positions i; these P_j are exactly the rows passed to DECS in Part (A). Importantly, LVCS also remembers the ℓ per-row masks $\bar{R}_j$ for later.

Linear queries on (masked) rows. Let the verifier specify a linear map $C \in \mathbb{F}_q^{m \times r}$ and a set E of ℓ challenge indices. LVCS asks the prover to compute, *for the mask tail only*,

$$\bar{v}_k[i] \;:=\; \sum_{j=1}^{r} C_{k,j}\, \bar{R}_j[i], \qquad k = 1, \ldots, m, \quad i = 0, \ldots, \ell - 1, \tag{14}$$

which hides the true targets $v_k[i] = \sum_j C_{k,j} R_j[i]$. The prover then opens, via the underlying DECS, the values $\{P_j(\omega)\}_{j \le r}$ at the indices in E. The LVCS verifier checks:

$$\sum_{j=1}^{r} C_{k,j}\, P_j(\omega) \;\overset{?}{=}\; \bar{v}_k[i] \quad \text{for each opened } \textit{mask} \text{ index } \;\omega = \omega_{n_{\text{cols}}+i}. \tag{15}$$

By construction, (15) is true iff the committed rows satisfy the linear map on the opened subset; the non-mask coordinates are not used by the linear check. All DECS checks from (A) (Merkle, degree, and (13)) run underneath.

(C) PCS wrapper: arbitrary point openings via block decomposition

To open a polynomial $P(X)$ of degree d at an *arbitrary* field point e, PCS maps P into a small number of LVCS rows by block-grouping its coefficients. Fix a stripe size μ and a mask-row count ℓ'. Let

$$\nu \;:=\; \max\Big(1,\; \big\lceil (d + 1 - \ell')/\mu \big\rceil\Big), \qquad A \in \mathbb{F}_q^{(\mu+\ell') \times \nu},$$

and fill A by placing the coefficients of P into the first μ rows with stride μ across the ν columns, while the last ℓ' rows are fresh, uniform masks:

$$A[r, k] \;=\; \begin{cases} a_{r+k\mu} & \text{if } r < \mu \text{ and } r + k\mu \le d, \\ 0 & \text{if } r < \mu \text{ and } r + k\mu > d, \\ \text{uniform in } \mathbb{F}_q & \text{if } \mu \le r < \mu + \ell'. \end{cases}$$

Stacking one such A_j for every committed polynomial P_j produces the *LVCS row-matrix* R on which Part (B) operates. Define

$$u_\mu(e) \;:=\; (1, e, \ldots, e^{\mu-1}) \in \mathbb{F}_q^{\mu}, \qquad w_\nu(e) \;:=\; (1, e^{\mu}, e^{2\mu}, \ldots, e^{(\nu-1)\mu}) \in \mathbb{F}_q^{\nu},$$

and $u_{\mu+\ell'}(e)$ by extending $u_\mu(e)$ with ℓ' extra monomials used for masking. By construction,

$$P(e) \;=\; \big\langle u_\mu(e),\; A_{\text{top}}\, w_\nu(e) \big\rangle \;=\; \big\langle u_{\mu+\ell'}(e),\; A\, w_\nu(e) \big\rangle \;-\; \big\langle u_{\mu+\ell'}(e)[\mu{:}],\; A[\mu{:}, :]\, w_\nu(e) \big\rangle, \tag{16}$$

so the true value $P(e)$ is an inner product between a vector determined by e and a *linear* combination of rows of A; the second term is precisely what the ℓ' mask rows hide. Therefore PCS reduces the opening $P(e)$ to an LVCS query with coefficient matrix $C = C(e)$ whose rows encode $w_\nu(e)$ and $u_{\mu+\ell'}(e)$.

In the simulation, the PCS helper computes ν_j for each polynomial P_j, constructs the global $(\mu + \ell') \times (\sum_j \nu_j)$ row matrix R by concatenating the A_j blocks, and delegates the commit and open phases to LVCS/DECS exactly as above.

HVZK and binding summary.

- *Zero-knowledge.* The ℓ LVCS masks per row (Part B) and the ℓ' PCS mask rows per polynomial (Part C) ensure that the vectors the verifier sees (the $\bar{v}_k$ and the opened leaves) are statistically independent of the true row values and coefficient blocks, conditioned on the correctness relations.
- *Binding.* DECS enforces (i) global degree $\deg P_j \le D$ via $\deg R_k \le D$ and (ii) the masked linear relation (13) at all opened indices. LVCS adds the linear map checks (15) on the masked tail. PCS translates a point opening $P(e)$ into a collection of LVCS linear queries as in (16), so function-binding for the LVCS rows implies polynomial-binding for P at e.

8 Target Relation and ℓ_2 Norm Bound

Setting. All constraints are enforced over the base field $\mathbb{F}_q$ (odd characteristic). Ring objects in $R_q = \mathbb{F}_q[X]/(X^{\phi}+1)$ are handled coefficient-wise: each ring coordinate corresponds to a length-ϕ block over $\mathbb{F}_q$. We fix a packing grid $\Omega = \{\omega_1, \ldots, \omega_s\} \subset \mathbb{F}_q$ of distinct points. We use a radix $R = 2^W < q$ and choose $\beta^2 < q$ to avoid wrap-around.

Packing choice for the ℓ_2 bound. To target the *coefficient-domain* norm, we adopt **coefficient packing**: for each signature row t, the associated row polynomial P_t satisfies $P_t(\omega_j) = a_{t,j}$ for $\omega_j \in \Omega$ (and is blinded on extra points). With this choice,

$$\sum_{\omega\in\Omega} \sum_{t=1}^{m_{\text{sig}}} P_t(\omega)^2 \;=\; \sum_{t=1}^{m_{\text{sig}}} \sum_{j=0}^{\phi-1} a_{t,j}^2.$$

BBS-form verification over $\mathbb{F}_q$. Given public (A, B_0, b_1) and message u, the witness is (s, x_0, x_1) and the (coefficient-wise) quadratic identity is

$$(b_1 \odot A)\, s \;-\; (As)\, x_1 \;-\; B_0 \cdot (1; u; x_0) \;\equiv\; 0 \pmod{q}, \tag{17}$$

together with the global bound

$$\big\|(s^\top, u^\top, x_0^\top, x_1)\big\|_2 \;\le\; \beta. \tag{18}$$

All variables are encoded coefficient-wise over $\mathbb{F}_q$ and packed across the s columns indexed by Ω.

8.1 Packing and Row Polynomials

Each row of the witness matrix is interpolated to a polynomial of degree $\le s + \ell - 1$ that agrees with its column values at Ω and is blinded on ℓ extra points. For the ℓ_2 bound, let the top m_{sig} rows be the signature coefficients; define

$$Sqs(X) \;:=\; \textstyle\sum_{t=1}^{m_{\text{sig}}} P_t(X)^2.$$

Then for every $\omega \in \Omega$, $Sqs(\omega) = \sum_t P_t(\omega)^2$ is the per-column sum of squares, and $\sum_{\omega\in\Omega} Sqs(\omega)$ equals the coefficient-domain squared norm of the signature block.

8.2 Parallel Constraints (per column): Digit/Bit and Remainder Chains

Fix $L_S := \lceil \log_R(\beta^2) \rceil$. Introduce (column-varying) digit polynomials $D_\ell(X)$ with $0 \le D_\ell(\omega) < R$ and a remainder chain $T_0, \ldots, T_{L_S}$. The entire system is degree ≤ 2.

Bit-decomposition of each digit (deg ≤ 2). For $\ell = 0, \dots, L_S - 1$ enforce

$$D_\ell(X) - \sum_{u=0}^{W-1} 2^u \, \mathrm{bit}_{\ell,u}(X) = 0, \tag{19}$$

$$\mathrm{bit}_{\ell,u}(X)^2 - \mathrm{bit}_{\ell,u}(X) = 0. \tag{20}$$

Remainder chain tying Sqs to its base-R digits (deg ≤ 2).

$$T_0(X) - Sqs(X) = 0, \tag{21}$$

$$T_\ell(X) - D_\ell(X) - R\,T_{\ell+1}(X) = 0 \quad (\ell = 0, \dots, L_S - 1), \tag{22}$$

$$T_{L_S}(X) = 0. \tag{23}$$

Chain view (one column ω):

$$\boxed{\begin{array}{rcl} T_0(\omega) & = & Sqs(\omega) \\ T_1(\omega) & = & \dfrac{T_0(\omega) - D_0(\omega)}{R} \\ T_2(\omega) & = & \dfrac{T_1(\omega) - D_1(\omega)}{R} \\ & \vdots & \\ T_{L_S}(\omega) & = & 0 \end{array}} \quad \Longleftrightarrow \quad Sqs(\omega) \;=\; \sum_{\ell=0}^{L_S-1} D_\ell(\omega)\, R^\ell.$$

8.3 Aggregated Constraints (across columns): Limb Summation vs. β^2

Write the public expansion $\beta^2 = \sum_{\ell=0}^{L_S-1} \beta_\ell^2 R^\ell$. Fix public constants $(\alpha_\omega)_{\omega\in\Omega} \subset \mathbb{F}_q$ such that $\sum_{\omega\in\Omega} \alpha_\omega = 1$ (e.g., set $\alpha_{\omega^\star} = 1$ for one distinguished $\omega^\star$ and 0 otherwise). Introduce *column-constant* carries C_ℓ and slacks Δ_ℓ (both range-encoded), and enforce, for each limb ℓ,

$$\sum_{\omega\in\Omega} \Big(D_\ell(\omega) + C_\ell + \Delta_\ell - \alpha_\omega \, \beta_\ell^2 - R\, C_{\ell+1} \Big) \;=\; 0, \qquad C_0 = C_{L_S} = 0. \tag{24}$$

Range (bitness) for carries and slacks (deg ≤ 2). Since $\sum_\omega D_\ell(\omega) \le s\,(R-1)$, choose $U_C := \lceil \log_2 s \rceil + 1$, $U_\Delta := W$, and encode

$$C_\ell = \sum_{u=0}^{U_C-1} 2^u \, \mathrm{bit}^C_{\ell,u}, \quad \Delta_\ell = \sum_{u=0}^{U_\Delta-1} 2^u \, \mathrm{bit}^\Delta_{\ell,u}, \quad b^2 - b = 0 \text{ for all bits.}$$

Why (24) **proves the inequality.** Multiply (24) by R^ℓ and sum over ℓ; carries telescope and the boundaries $C_0 = C_{L_S} = 0$ yield

$$\sum_{\omega\in\Omega} Sqs(\omega) \;+\; s \sum_\ell \Delta_\ell R^\ell \;=\; \sum_\ell \beta_\ell^2 R^\ell \;=\; \beta^2,$$

hence $\sum_\omega Sqs(\omega) \le \beta^2$, with equality iff all $\Delta_\ell = 0$.

8.4 PACS Split and Batching

All digit/bit/remainder and range constraints above are *parallel* (checked per column), while the limb sums (24) are *aggregated* (checked via the $\sum_{\omega\in\Omega}(\cdot) = 0$ test with the selector weights (α_ω)). We batch them with mask polynomials so that each batch Q_i satisfies $\sum_{\omega\in\Omega} Q_i(\omega) = 0$, and verify degree and point identities at random evaluation points outside Ω. All constraints are linear or quadratic, keeping the system within degree ≤ 2 throughout.

9 PIOP Checks and Fiat–Shamir

Let $E' \subset F \setminus \Omega$ be the verifier's random evaluation set for pointwise checks (via LVCS/PCS openings), and let the Ω-sum test be used for aggregated equalities. The verifier accepts iff all of the following hold:

(i) **Degree bounds (parallel).** Each committed row polynomial (witness and auxiliary) respects its declared degree bound; masks used in the HVZK reductions also respect their degrees.

(ii) **Pointwise identities at E' (parallel).** All *parallel* constraints are enforced at every $e \in E'$, including:

- digit bitness for D_ℓ, remainder-chain constraints for T_ℓ,
- bitness for C_ℓ and Δ_ℓ with the fixed bitlengths (U_C, U_Δ),
- the constant carry boundaries $C_0 = 0$ and $C_{L_S} = 0$,
- any other local (degree ≤ 2) relations bundled into the F_t constraints.

(iii) **Ω-sum equalities (aggregated).** All *aggregated* linear relations most notably the limb-wise sums (24) implemented as constant rows via S_0^{-1} pass the $\sum_{\omega\in\Omega}(\cdot) = 0$ checks. This includes any other batched linear constraints F'_u.

All verifier challenges (the randomizers for batching, the sets E' and any additional query set E, and the masking coefficients) are derived in the Random Oracle Model by a Fiat–Shamir transform *with grinding.* Concretely, the prover samples a public salt and derives the four challenge stages and their counters $(\mathrm{ctr}_1, \dots, \mathrm{ctr}_4)$; the proof records these counters, and the verifier checks the required κ_i zero-prefix conditions without re-grinding. This pins the domains of all challenges and accounts for the knowledge-soundness slack in the usual way.

10 End-to-End Algebraic Recipe

1. *Interpolate/blind rows:* for all i, $P_i(\omega_k) = W_{i,k}$ and $P_i(r_j) = y_j$ with $\deg P_i \leq s + \ell - 1$.
2. *Build constraints:* $F_k = W_3^{(k)} - W_1^{(k)} W_2$ (parallel), F'_j as in (8) (aggregated).
3. *Batch:* form $Q_i = M_i + \sum_t \Gamma'_{i,t} F_t + \sum_u \gamma'_{i,u} F'_u$ with declared degree d_Q.
4. *Mask for Ω-sum zero:* choose M_i by (11) to enforce $\sum_{\omega\in\Omega} Q_i(\omega) = 0$.
5. *Open/verify:* use PCS/LVCS to open at random points and verify degree and the identities from step 3.
6. *Norm bound:* enforce the digit/carry/slack system of Sec. **??**; existence of a witness is equivalent to $\|(\mathbf{s}, \mathbf{u}, \mathbf{x}_0, x_1)\|_2 \leq \beta$.

11 Images — Annexe

11.1 Worked Numerical Example (Coefficient Packing, Base-R Chains, and Aggregation)

Field, ring, and packing. Let $q = 101$ and $R_q = \mathbb{F}_{101}[X]/(X^5 + 1)$ (so $\phi = 5$). Fix the packing grid $\Omega = \{0, 1, 2, 3, 4\}$ (so $s = |\Omega| = 5$). We use radix $R = 2^4 = 16$ and take $\beta^2 = 83 < q$, hence $L_S = \lceil \log_{16}(83) \rceil = 2$.

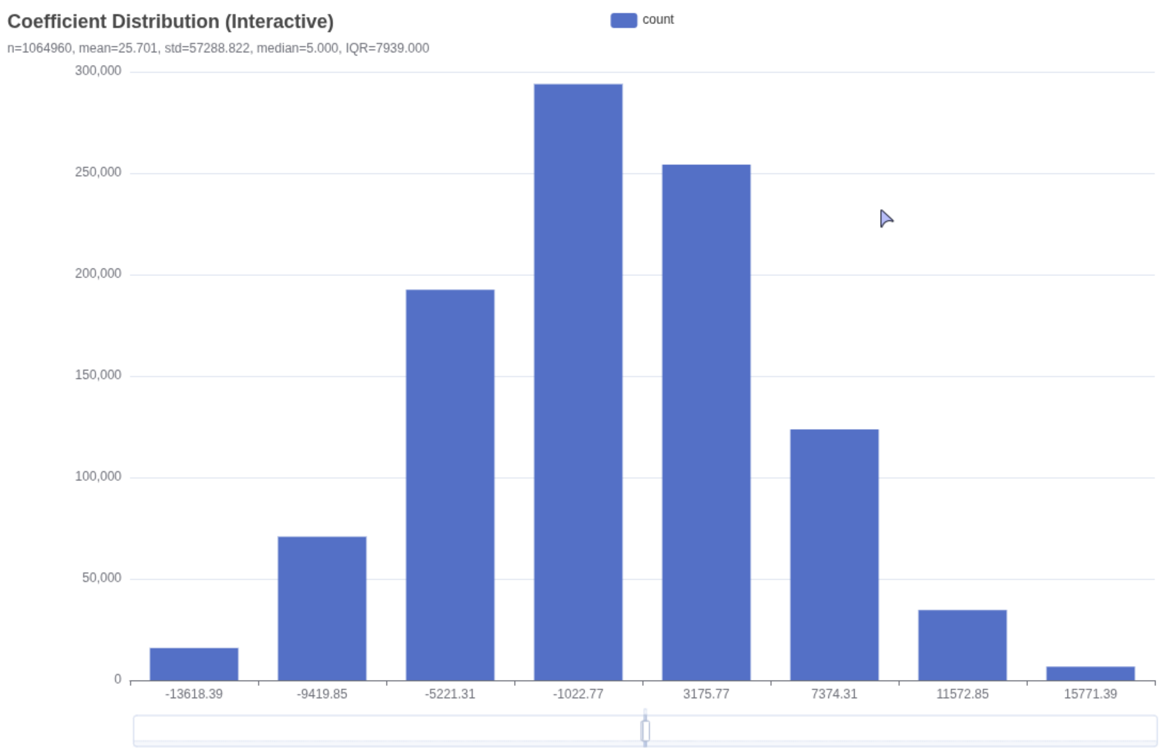

Figure 2: Coefficient distribution of samples from the discrete Gaussian sampler.

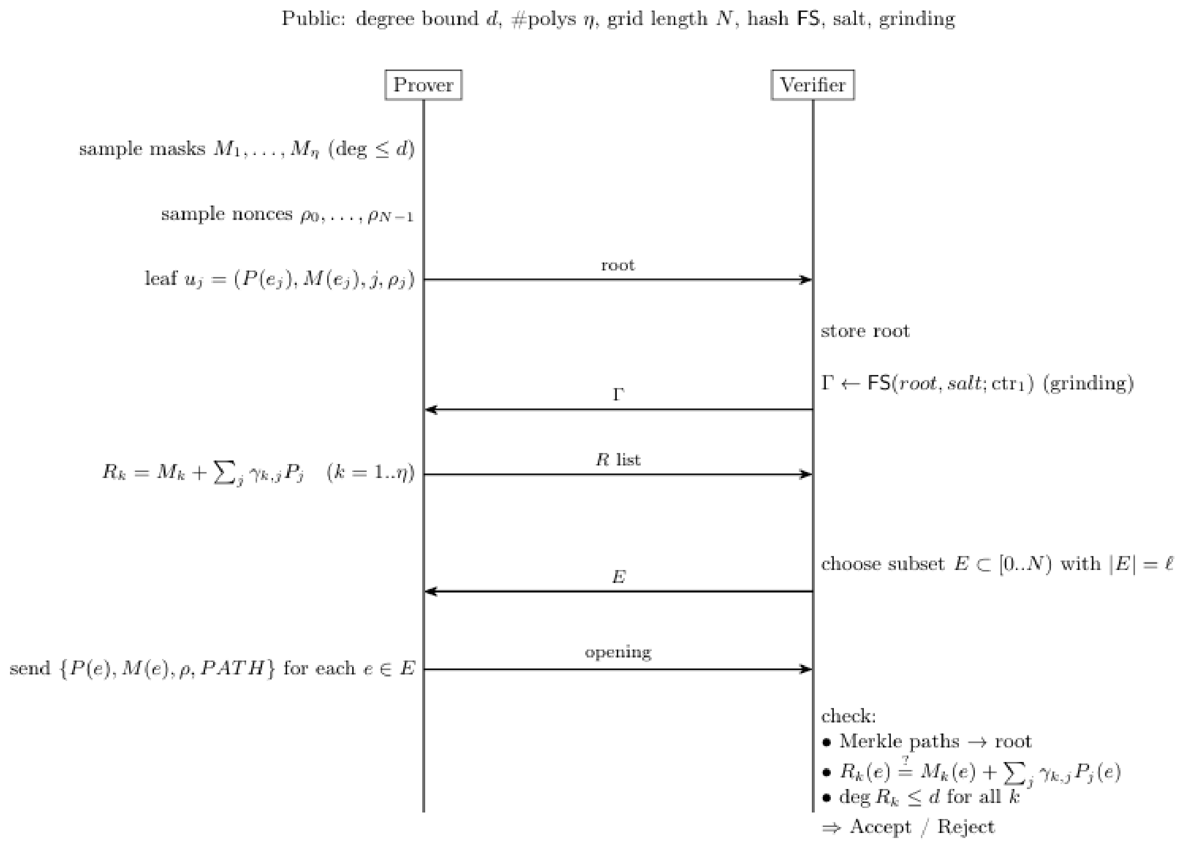
Public: degree bound d, #polys η, grid length N, hash FS, salt, grinding
Prover
Verifier
sample masks $M_1, \dots, M_\eta$ (deg $\leq d$)
sample nonces $\rho_0, \dots, \rho_{N-1}$
leaf $u_j = (P(e_j), M(e_j), j, \rho_j)$
root
store root
$\Gamma \leftarrow \mathsf{FS}(root, salt; \mathrm{ctr}_1)$ (grinding)
Γ
$R_k = M_k + \sum_j \gamma_{k,j} P_j \quad (k = 1..\eta)$
R list
choose subset $E \subset [0..N)$ with $|E| = \ell$
E
send $\{P(e), M(e), \rho, PATH\}$ for each $e \in E$
opening
check:
• Merkle paths $\to$ root
• $R_k(e) \stackrel{?}{=} M_k(e) + \sum_j \gamma_{k,j} P_j(e)$
• deg $R_k \leq d$ for all k
$\Rightarrow$ Accept / Reject

Figure 3: DECS: three-round commit and masked linear-relation check.

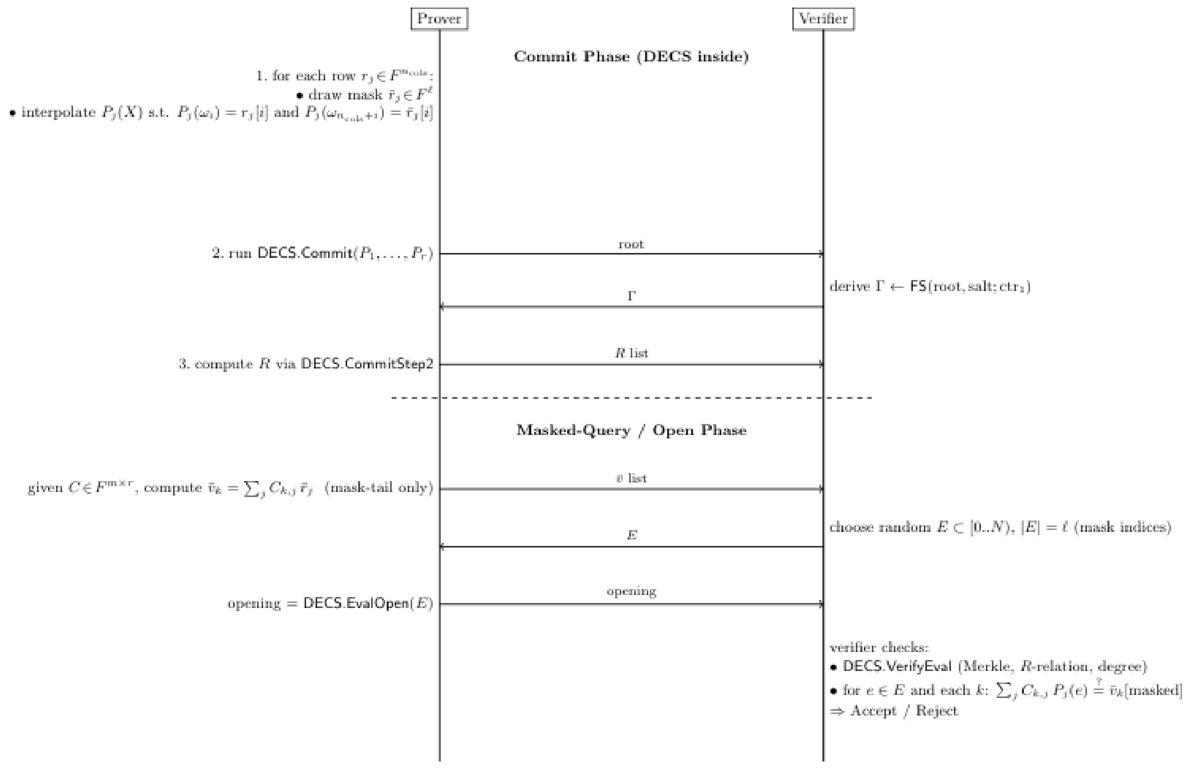
Prover
Verifier
Commit Phase (DECS inside)
1. for each row $r_j \in F^{n_{\text{cols}}}$:
• draw mask $\bar{r}_j \in F^{\ell}$
• interpolate $P_j(X)$ s.t. $P_j(\omega_i) = r_j[i]$ and $P_j(\omega_{n_{\text{cols}}+i}) = \bar{r}_j[i]$
2. run $\mathsf{DECS.Commit}(P_1, \ldots, P_r)$
root
derive $\Gamma \leftarrow \mathsf{FS}(\text{root}, \text{salt}; \text{ctr}_1)$
Γ
3. compute R via $\mathsf{DECS.CommitStep2}$
R list
Masked-Query / Open Phase
given $C \in F^{m \times r}$, compute $\bar{v}_k = \sum_j C_{k,j}\, \bar{r}_j$ (mask-tail only)
$\bar{v}$ list
choose random $E \subset [0..N)$, $|E| = \ell$ (mask indices)
E
opening = $\mathsf{DECS.EvalOpen}(E)$
opening
verifier checks:
• $\mathsf{DECS.VerifyEval}$ (Merkle, R-relation, degree)
• for $e \in E$ and each k: $\sum_j C_{k,j}\, P_j(e) \overset{?}{=} \bar{v}_k[\text{masked}]$
$\Rightarrow$ Accept / Reject

Figure 4: LVCS: commit rows and open masked linear maps on tail coordinates.

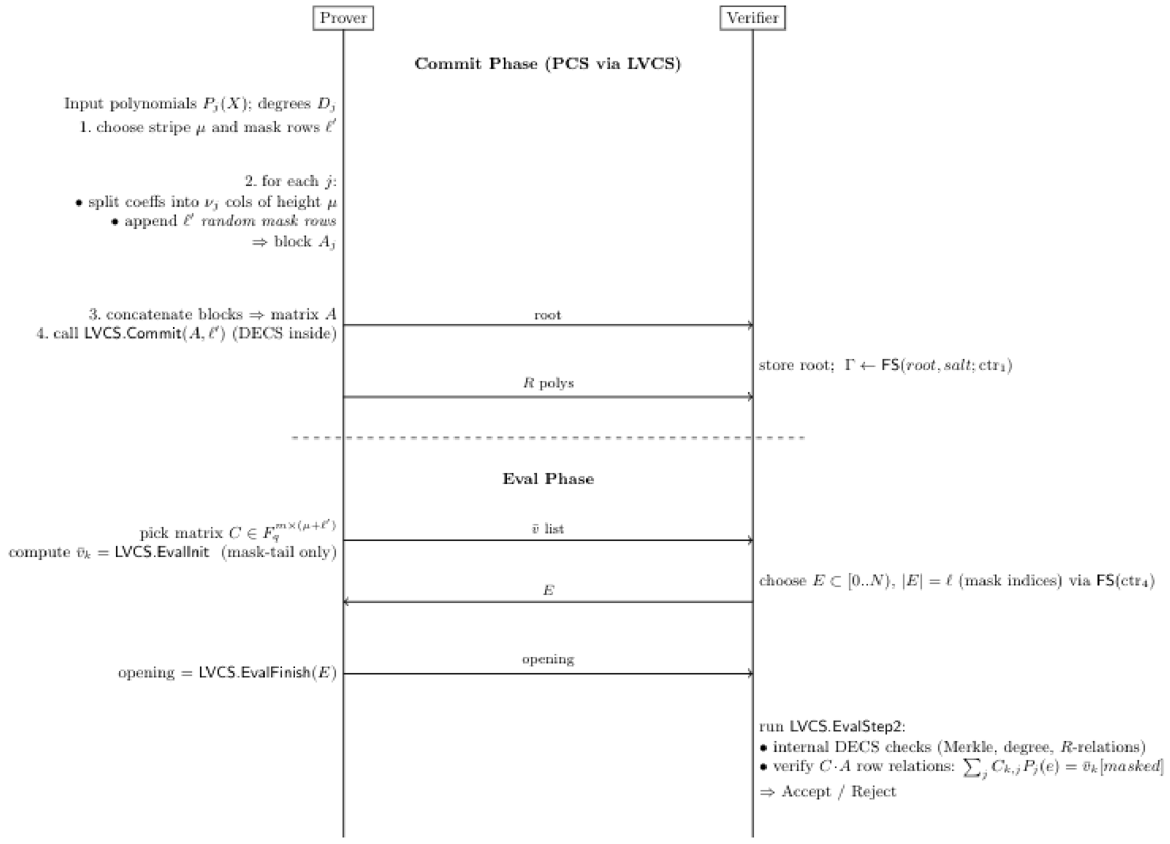
Prover
Verifier
Commit Phase (PCS via LVCS)
Input polynomials $P_j(X)$; degrees D_j
1. choose stripe μ and mask rows ℓ'
2. for each j:
• split coeffs into ν_j cols of height μ
• append ℓ' *random mask rows*
$\Rightarrow$ block A_j
3. concatenate blocks $\Rightarrow$ matrix A
4. call $\mathsf{LVCS.Commit}(A, \ell')$ (DECS inside)
root
store root; $\Gamma \leftarrow \mathsf{FS}(root, salt; \text{ctr}_1)$
R polys
Eval Phase
pick matrix $C \in F_q^{m \times (\mu + \ell')}$
compute $\bar{v}_k = \mathsf{LVCS.EvalInit}$ (mask-tail only)
$\bar{v}$ list
choose $E \subset [0..N)$, $|E| = \ell$ (mask indices) via $\mathsf{FS}(\text{ctr}_4)$
E
opening = $\mathsf{LVCS.EvalFinish}(E)$
opening
run $\mathsf{LVCS.EvalStep2}$:
• internal DECS checks (Merkle, degree, R-relations)
• verify $C \cdot A$ row relations: $\sum_j C_{k,j} P_j(e) = \bar{v}_k[masked]$
$\Rightarrow$ Accept / Reject

Figure 5: PCS wrapper: coefficient blocking with ℓ' mask rows and point evaluation.

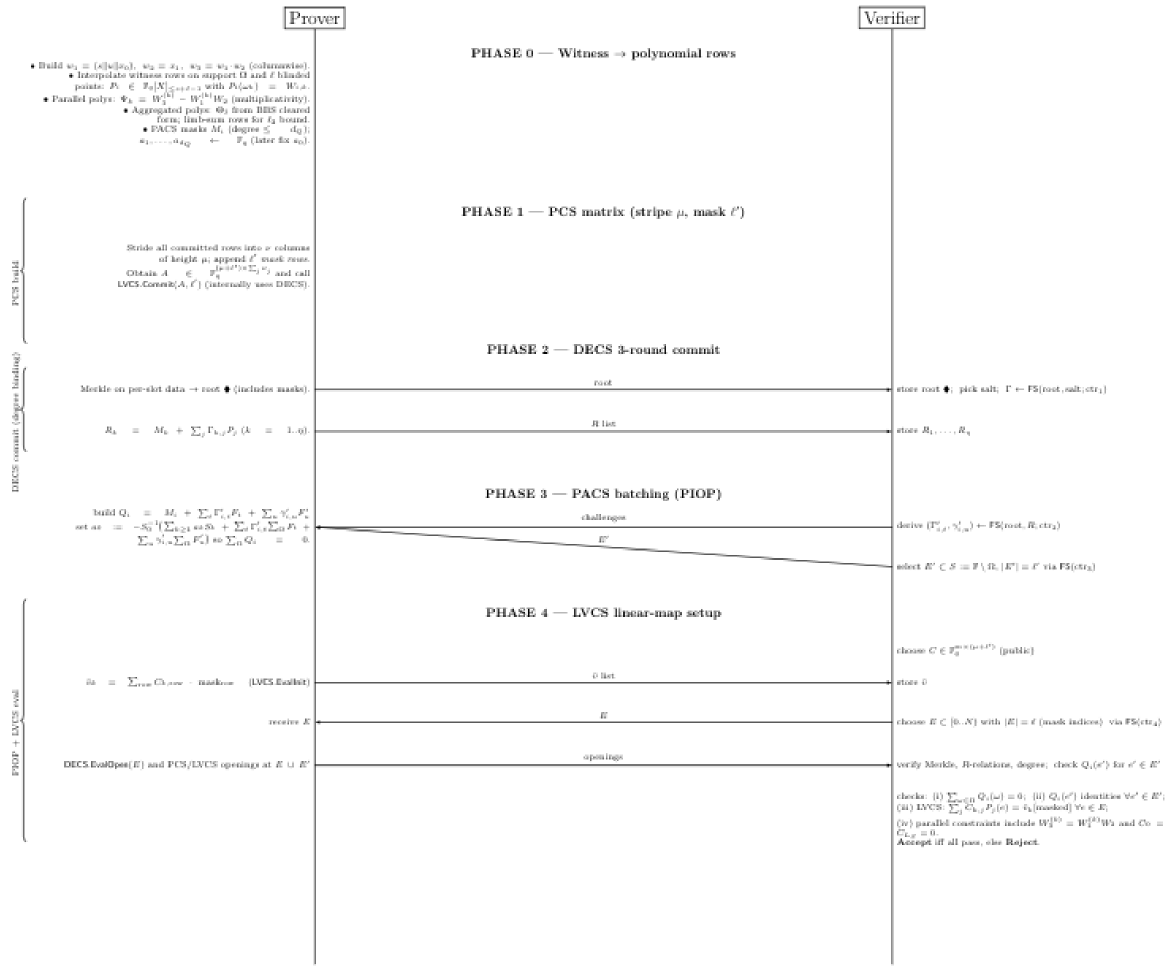
Prover
Verifier
PHASE 0 — Witness ⇒ polynomial rows
• Build $w_1 = (s\|u\|x_0)$, $w_2 = x_1$, $w_3 = w_1 \cdot w_2$ (columnwise).
• Interpolate witness rows on support Ω and ℓ blinded points: $P_i \in \mathbb{F}_q[X]_{\leq s+\ell-1}$ with $P_i(\omega_k) = W_{i,k}$.
• Parallel polys: $\Psi_k = W_3^{(k)} - W_1^{(k)} W_2$ (multiplicativity).
• Aggregated polys: Θ_j from BBS cleared form; limb-sum rows for ℓ_2 bound.
• PACS masks M_i (degree ≤ d_Q); $a_1, \ldots, a_{d_Q} \leftarrow \mathbb{F}_q$ (later fix a_0).
PHASE 1 — PCS matrix (stripe μ, mask ℓ')
PCS build
Stride all committed rows into ν columns of height μ; append ℓ' mask rows. Obtain $A \in \mathbb{F}_q^{(\mu+\ell') \times \sum_j \nu_j}$ and call LVCS.Commit(A, ℓ') (internally uses DECS).
PHASE 2 — DECS 3-round commit
DECS commit (degree binding)
Merkle on per-slot data → root ◆ (includes masks).
root
store root ◆; pick salt; $\Gamma \leftarrow$ FS(root, salt; ctr$_1$)
$R_k = M_k + \sum_j \Gamma_{k,j} P_j$ ($k = 1..\eta$).
R list
store $R_1, \ldots, R_\eta$
PHASE 3 — PACS batching (PIOP)
build $Q_i = M_i + \sum_s \Gamma'_{i,s} F_s + \sum_u \gamma'_{i,u} F'_u$; set $a_0 := -S_0^{-1}(\sum_{k\geq 1} a_k S_k + \sum_s \Gamma'_{i,s} \sum_\Omega F_s + \sum_u \gamma'_{i,u} \sum_\Omega F'_u)$ so $\sum_\Omega Q_i = 0$.
challenges
derive $(\Gamma'_{i,s}, \gamma'_{i,u}) \leftarrow$ FS(root, R; ctr$_2$)
E'
select $E' \subset S := \mathbb{F} \setminus \Omega$, $|E'| = \ell'$ via FS(ctr$_3$)
PHASE 4 — LVCS linear-map setup
PIOP + LVCS eval
choose $C \in \mathbb{F}_q^{m \times (\mu+\ell')}$ (public)
$\bar{v}_k = \sum_{row} C_{k,row} \cdot$ mask$_{row}$ (LVCS.EvalInit)
$\bar{v}$ list
store $\bar{v}$
receive E
E
choose $E \subset [0..N)$ with $|E| = \ell$ (mask indices) via FS(ctr$_4$)
DECS.EvalOpen(E) and PCS/LVCS openings at $E \cup E'$
openings
verify Merkle, R-relations, degree; check $Q_i(e')$ for $e' \in E'$
checks: (i) $\sum_{\omega\in\Omega} Q_i(\omega) = 0$; (ii) $Q_i(e')$ identities $\forall e' \in E'$; (iii) LVCS: $\sum_j C_{k,j} P_j(e) = \bar{v}_k$(masked) $\forall e \in E$; (iv) parallel constraints include $W_3^{(k)} = W_1^{(k)} W_2$ and $C_0 = C_{1,y} = 0$.
Accept iff all pass, else Reject.

Figure 6: PACS/PIOP batching: build Q_i and enforce $\sum_{\omega\in\Omega} Q_i(\omega) = 0$ with checks on E'.

BBS-form relation (one-row toy instance). Take a one-row instance so the relation

$$(b_1 \odot A)s \;-\; (As)\,x_1 \;-\; B_0 \cdot (1; u; x_0) \equiv 0 \pmod q$$

is an equality in R_q. To isolate the norm subsystem in this example, set $b_1 = 1$, $x_1 = 0$, $B_0 = 0$, and $A = 0$, so the identity holds trivially. (*Remark.* In a full instance you would plug nontrivial A, B_0, b_1, x_1; the norm proof below is independent of that choice.)

Signature rows and coefficient packing. Let $m_{\text{sig}} = 3$ and choose the (small) coefficient vectors

$$\begin{aligned} s_1(X) &= \underline{1} + 2X + 0X^2 + 1X^3 + 3X^4, \\ s_2(X) &= \underline{0} + 1X + 1X^2 + 2X^3 + 0X^4, \\ s_3(X) &= \underline{2} + 1X + 0X^2 + 1X^3 + 1X^4, \end{aligned}$$

so that, for $j = 0, \dots, 4$, the packed row values satisfy $P_t(\omega_j) = a_{t,j}$ (*coefficient packing*). The per-column sum of squares is

j	0	1	2	3	4
$(a_{1,j}, a_{2,j}, a_{3,j})$	$(1,0,2)$	$(2,1,1)$	$(0,1,0)$	$(1,2,1)$	$(3,0,1)$
$Sqs(\omega_j) = \sum_t a_{t,j}^2$	5	6	1	6	10

Hence

$$\sum_{\omega \in \Omega} Sqs(\omega) = 5 + 6 + 1 + 6 + 10 = 28 \quad \Rightarrow \quad \|\mathbf{s}\|_2^2 = 28 \le \beta^2 = 83.$$

Parallel constraints (per column): digits and remainder chain. With $R = 16$ and $L_S = 2$, write at each column ω:

$$Sqs(\omega) = D_0(\omega) + 16\, D_1(\omega), \qquad 0 \le D_\ell(\omega) < 16.$$

For our numbers (all $Sqs(\omega) < 16$) we get

ω	0	1	2	3	4
$Sqs(\omega)$	5	6	1	6	10
$D_0(\omega)$	5	6	1	6	10
$D_1(\omega)$	0	0	0	0	0

Bitness (degree-2) enforces $D_\ell = \sum_{u=0}^{3} 2^u \, \mathrm{bit}_{\ell,u}$, with $\mathrm{bit}_{\ell,u}^2 - \mathrm{bit}_{\ell,u} = 0$. The remainder (degree-2) chain is

$$T_0(\omega) - Sqs(\omega) = 0, \qquad T_\ell(\omega) - D_\ell(\omega) - 16\, T_{\ell+1}(\omega) = 0, \quad T_2(\omega) = 0,$$

which here yields $T_1(\omega) = 0$ for every column since $Sqs(\omega) < 16$.

Aggregated constraints (across columns) with selector weights. Write β^2 in base 16: $\beta^2 = 83 = 5 \cdot 16 + 3$, i.e., $(\beta_1^2, \beta_0^2) = (5, 3)$. Fix selector weights $(\alpha_\omega)_{\omega \in \Omega}$ with $\sum_\Omega \alpha_\omega = 1$ (e.g., set $\alpha_0 = 1$ and $\alpha_{\omega \neq 0} = 0$). Introduce *column-constant* carries C_0, C_1, C_2 (with $C_0 = C_2 = 0$) and slacks Δ_0, Δ_1 (both range-encoded in $[0, 15]$), and enforce for each limb $\ell \in \{0, 1\}$:

$$\sum_{\omega \in \Omega} \Big(D_\ell(\omega) + C_\ell + \Delta_\ell - \alpha_\omega\, \beta_\ell^2 - 16\, C_{\ell+1} \Big) = 0.$$

With $\sum_\Omega D_0 = 28$ and $\sum_\Omega D_1 = 0$, a valid assignment is

$$C_1 = 1, \quad \Delta_0 = 11, \quad \Delta_1 = 0 \qquad (C_0 = C_2 = 0).$$

Indeed,

$$\begin{aligned} \ell = 0: &\quad 28 + 5\,(0 + 11 - 16 \cdot 1) - \beta_0^2 = 28 + 5 \cdot (-5) - 3 = 0, \\ \ell = 1: &\quad 0 + 5\,(1 + 0 - 16 \cdot 0) - \beta_1^2 = 5 - 5 = 0. \end{aligned}$$

(*All* constraints used are linear or quadratic.)

Why this proves the bound (checked numerically). Summing the limb equalities after multiplying by R^ℓ telescopes the carries and yields

$$\sum_{\omega\in\Omega} Sqs(\omega) \; + \; s\cdot\big(\Delta_0 + 16\,\Delta_1\big) \; = \; \beta^2.$$

Plugging our numbers:

$$28 \; + \; 5\cdot(11+16\cdot 0) \; = \; 28+55 \; = \; 83 \; = \; \beta^2,$$

so $\sum_\omega Sqs(\omega) \le \beta^2$ holds (with equality iff all slacks vanish; here the positive slack sits in the least significant limb).

Zero-knowledge and binding (brief). Each witness row is committed as a low-degree polynomial P_t that agrees with its values on Ω and is blinded on extra points; all checks above are enforced as: (i) *parallel* constraints (per column) for digit/bit/remainder, and (ii) *aggregated* constraints (across columns) for the limb equations. The prover opens only the batched Ω-sum tests and random evaluation checks; no column reveals the coefficients $(a_{t,j})$.

This concrete instance exhibits every moving part of the norm subsystem: coefficient packing, per-column digit chains, and the limb-sum aggregation that certifies $\|\mathbf{s}\|_2^2 = \sum_\omega Sqs(\omega) \le \beta^2$.

Block / Row	ω_1	ω_2	ω_3	ω_4	ω_5
Signature (packed coefficients)					
Signature s_1	1	2	0	1	3
Signature s_2	0	1	1	2	0
Signature s_3	2	1	0	1	1
Row-constant public/auxiliary (unused in this toy norm check)					
Message u_1	u_1	u_1	u_1	u_1	u_1
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Rand $x_{0,1}$	$x_{0,1}$	$x_{0,1}$	$x_{0,1}$	$x_{0,1}$	$x_{0,1}$
Scalar x_1	**0**	**0**	**0**	**0**	**0**
Linearized products for $(As)\,x_1$ (here: all zero since $x_1 = 0$)					
Product $(x_1 \cdot s_1)$	0	0	0	0	0
Product $(x_1 \cdot s_2)$	0	0	0	0	0
Product $(x_1 \cdot s_3)$	0	0	0	0	0
Norm accumulator and base-R expansion					
Accumulator Sqs	5	6	1	6	10
Digits D_0	5	6	1	6	10
Digits D_1	0	0	0	0	0
Bits $\text{bit}_{0,0}$ (LSB)	1	0	1	0	0
Bits $\text{bit}_{0,1}$	0	1	0	1	1
Bits $\text{bit}_{0,2}$	1	1	0	1	0
Bits $\text{bit}_{0,3}$ (MSB)	0	0	0	0	1
Bits $\text{bit}_{1,u}$ $(u = 0..3)$	0	0	0	0	0
Remainder T_0	5	6	1	6	10
Remainder T_1	0	0	0	0	0
Remainder T_2	0	0	0	0	0
Column-constant limbs (aggregation)					
Carry C_0	0	0	0	0	0
Carry C_1	1	1	1	1	1
Slack Δ_0	11	11	11	11	11
Slack Δ_1	0	0	0	0	0

W_{example}

11.2 DECS, LVCS, PCS in Our Protocol: What They Are, What They Bind, and What We Open

Objects in our instance. We work over $\mathbb{F}_q$ and the ring $R_q = \mathbb{F}_q[X]/(f)$ with $\deg f = \phi$. The prover's witness consists of

$$(s, x_0, x_1) \quad \text{with} \quad s = (s_1, \ldots, s_{m_{\text{sig}}}),\ s_t(X) = \sum_{j=0}^{\phi-1} a_{t,j} X^j \in R_q.$$

Public data are (A, B_0, b_1) and the message u, with the coefficient-wise quadratic identity

$$(b_1 \odot A)s - (As)\,x_1 - B_0 \cdot (1; u; x_0) \equiv 0 \pmod{q},$$

and the global ℓ_2 bound on the *coefficients* of (s, u, x_0, x_1).

Packing in our instance (what we commit and where). Fix a public packing grid $\Omega = \{\omega_1, \ldots, \omega_s\} \subset \mathbb{F}_q$ of size s, and a disjoint set of ℓ *blinding points* Ω'. We lay the witness as a row matrix $W \in \mathbb{F}_q^{r \times s}$ whose columns are indexed by Ω. For the signature block we adopt **coefficient packing**:

$$\forall\, t \le m_{\text{sig}},\ \forall\, \omega_j \in \Omega: \quad W_1^{(t)}(\omega_j) = a_{t,j}.$$

Every row i is then interpolated to a **row polynomial** $P_i(X)$ with degree bound

$$\deg P_i \ \le\ s + \ell - 1 \quad \text{and} \quad P_i(\omega_j) = W_{i,j} \ \text{ for all } \omega_j \in \Omega,$$

while on Ω' we write random masks. *In our application, these P_i are not the ring polynomials s_t; they are the* <u>*packing polynomials*</u> *whose evaluations on Ω are exactly the per-column data we will manipulate in the proof.*

What "low-degree" means here. "Low-degree" refers to the interpolation degree of each *row polynomial* P_i as a function in the *packing variable* X. The bound $\deg P_i \le s + \ell - 1$ is induced by s packed points (the columns) and ℓ blinded points. This degree bound is what allows succinct polynomial checks and underpins soundness via Schwartz Zippel.

How the BBS relation appears column-wise. Because we packed *coefficients*, each column $\omega_j \in \Omega$ corresponds to the j-th coefficient across all rows. When we form

$$F_{\text{BBS}}(\omega_j) \ :=\ \big[(b_1 \odot A)s\big](\omega_j) \ -\ \big[(As)\,x_1\big](\omega_j) \ -\ \big[B_0 \cdot (1; u; x_0)\big](\omega_j),$$

this is a quadratic expression in the *row values at ω_j*. We enforce $F_{\text{BBS}}(\omega) = 0$ for every $\omega \in \Omega$ through the PACS "parallel" constraints and the batched Ω-sum checks, without ever revealing the per-column entries.

Where the norm lives. Define the *sum-of-squares row*

$$Sqs(X) \ :=\ \sum_{t=1}^{m_{\text{sig}}} P_t(X)^2.$$

By coefficient packing, $\sum_{\omega \in \Omega} Sqs(\omega) = \sum_{t,j} a_{t,j}^2$ is exactly the coefficient-domain squared norm we need to bound by β^2. We encode Sqs in base $R = 2^W$ via digit rows D_ℓ, a remainder chain T_ℓ, and limb-wise aggregated equalities with carries C_ℓ and slacks Δ_ℓ to prove $\sum_\omega Sqs(\omega) \le \beta^2$ with degree-≤ 2 constraints throughout.

Layer 1 — DECS (Degree & Function Binding).

- *What we commit:* the family of row polynomials $\{P_i\}$ plus masking polynomials.
- *What DECS binds:* that every future linear combination of rows corresponds to the evaluation of some low-degree polynomial consistent with the commitment (degree bounded; function-binding at a small opened set).
- *What we open:* a handful of *masked* linear mixes $R_k = \sum_i \Gamma_{k,i} P_i + M_k$ at a tiny set of verifier-chosen points $E \subset \mathbb{F}_q \backslash (\Omega \cup \Omega')$ (or at designated tail indices), together with degree checks.
- *In our protocol:* DECS underlies *all* later checks—parallel BBS equations, digit/bit constraints, remainder chains, and the limb equalities are all expressed as linear relations among rows and are therefore DECS-bound once we batch them.

Layer 2 — LVCS (Linear-Map Vector Commitments).

- *What we view:* the committed rows as a vector over the row index.
- *What LVCS exposes:* only *masked linear maps* across rows, typically on *tail coordinates* (indices in Ω') so that revealed values are statistically independent of the head entries on Ω.
- *What LVCS binds:* if a claimed linear map C is accepted, then there exist rows $\{P_i\}$ consistent with DECS such that the opened vector $v = C \cdot (P_1, \dots, P_r)$ holds at the opened indices. Any deviation breaks DECS with SZ-type probability.
- *In our protocol:* every parallel constraint (bitness, $T_\ell - D_\ell - RT_{\ell+1}$, $T_0 - Sqs$, linearized BBS terms) is enforced by asking LVCS to open the *appropriate linear mix of rows* at blinded coordinates; we never open raw column values on Ω.

Layer 3 — PCS (Polynomial Commitments / Point Openings).

- *What PCS provides:* point openings $P_i(e)$ at verifier-chosen $e \in \mathbb{F}_q$ using LVCS as a subroutine. Concretely, $P_i(e)$ is linear in the coefficient vector, hence is a *linear map across rows*; PCS packages the monomial coefficients into an LVCS query and returns $P_i(e)$ while keeping the rest hidden.
- *What PCS binds:* that the opened value equals the evaluation of a degree-bounded polynomial consistent with the DECS commitment. Any PCS binding break reduces to LVCS/DECS.
- *In our protocol:* PCS is used to check the batched polynomials at random points outside Ω, e.g., the Q_i that encode (i) the Ω-sum vanishing of parallel constraints, and (ii) the aggregated limb equalities for the norm bound.

What is opened when (summary for the audience).

Step	**Verifier learns (and nothing more)**
DECS check	A few masked linear mixes $R_k(e)$ at $e \in E$ and degree certificates.
LVCS parallel checks	Masked evaluations of the linear combinations that encode: $T_0 - Sqs$, $T_\ell - D_\ell - RT_{\ell+1}$, digit bitness, linearized BBS rows.
PCS random-point checks	Values $Q_i(\eta)$ at random $\eta \notin \Omega$ to certify: $\sum_{\omega\in\Omega}$-vanishing of batched parallel constraints and the aggregated limb identities.

Why the imbrication matters (binding flow). Packing fixes *what* the columns mean (here: signature *coefficients* at Ω). DECS guarantees that any accepted evaluation/identity truly comes from low-degree row polynomials. LVCS restricts disclosure to *masked* linear maps on rows, preserving HVZK. PCS turns those linear maps into *point* checks, letting us verify global identities (batched Ω-sums, limb telescoping) succinctly. Thus, when we conclude $\sum_{\omega\in\Omega} Sqs(\omega) \le \beta^2$ and $F_{\mathrm{BBS}}(\omega) = 0$ for every $\omega \in \Omega$, we have revealed *only* what these statements logically imply—no individual coefficient $a_{t,j}$, and no extra linear information about the witness rows.

11.3 Coupling the BBS Relation and the Norm Bound (No Witness Splitting)

Single-source rows. We commit *once* to the signature block as row polynomials $\{P_1, \dots, P_{m_{\mathrm{sig}}}\}$ with coefficient packing: $P_t(\omega_j) = a_{t,j}$ for $\omega_j \in \Omega$. These rows are the *only* source of signature data in the proof.

Both subsystems use the same rows.

- **BBS subsystem.** The coefficient-wise relation $(b_1 \odot A)s - (As)\,x_1 - B_0 \cdot (1; u; x_0) \equiv 0$ is encoded as *parallel* constraints at each $\omega \in \Omega$, using the values $\{P_t(\omega)\}$ and row-constant public rows (messages u, randomness x_0, scalar x_1). All products are linearized via standard "copy product into its own row" tricks, but they still depend only on $\{P_t\}$.
- **Norm subsystem.** The accumulator is $Sqs(X) := \sum_{t=1}^{m_{\text{sig}}} P_t(X)^2$; digits D_ℓ, remainder rows T_ℓ, carries C_ℓ, and slacks Δ_ℓ are auxiliary rows constrained by

  $$T_0 - Sqs = 0, \quad T_\ell - D_\ell - R\,T_{\ell+1} = 0, \quad \text{bitness for } D_\ell, \quad \sum_{\omega \in \Omega} (D_\ell + C_\ell + \Delta_\ell - \alpha_\omega \beta_\ell^2 - R\,C_{\ell+1}) = 0.$$

  Thus the bound $\sum_{\omega \in \Omega} Sqs(\omega) \le \beta^2$ references *exactly the same* $\{P_t\}$ used by the BBS subsystem.

Why a prover cannot split witnesses. All prover openings are masked linear combinations of the *committed* rows, checked at a few grid/tail indices (DECS/LVCS) and at random field points (PCS). If the prover tried to use different rows for the norm (say $\{P_t^{\text{norm}}\}$) than for the BBS relation (say $\{P_t^{\text{bbs}}\}$), they would need to respond to *one* set of challenges with values consistent with $\{P_t^{\text{bbs}}\}$ and simultaneously to *another* set consistent with $\{P_t^{\text{norm}}\}$. Since there is only *one* commitment and every check reduces to openings of the *same* committed polynomials, DECS function-binding forces $P_t^{\text{norm}} \equiv P_t^{\text{bbs}}$ as polynomials (up to negligible SZ error). In other words, a single post-commitment family $\{P_t\}$ must satisfy *both* subsystems at once.

Implementation invariant (explicit coupling). To make the coupling obvious and auditable, we include the explicit degree-≤ 2 equation

$$\boxed{Sqs(X) \;-\; \sum_{t=1}^{m_{\text{sig}}} P_t(X)^2 \;=\; 0}$$

as a *parallel* constraint, and we ensure every BBS constraint references the same row indices $t \in \{1, \ldots, m_{\text{sig}}\}$. No duplicate "signature rows" are allowed. If, for engineering reasons, an alias is introduced (e.g., a copy row P_t^{alias} for a subroutine), we add the linear *copy constraint*

$$P_t^{\text{alias}}(X) \;-\; P_t(X) \;=\; 0,$$

checked via the same DECS/LVCS/PCS machinery. This prevents any accidental or malicious divergence between subsystems.

If different domains are unavoidable. If a subroutine needs a linear transform of the signature rows (e.g., an NTT-like view), we add *linear linking constraints*:

$$\widehat{P}_t(X) \;=\; \sum_{k=1}^{s} H_{t,k}\, P_k(X),$$

for a public matrix H representing the transform on the packing grid. Because the relation is linear and degree-preserving, DECS/LVCS enforce it with the same soundness, thereby tying the transformed view to the original rows.

11.4 Main Algorithms: PIOP Package (SmallWood Branch)

Algorithm 1 BuildWitness — assemble (w_1, w_2, w_3) for the quadratic gate

1

```
Require: Ring context ringQ, public $A \in (\mathbb{F}_q[X]/\langle X^N + 1\rangle)^{n\times m}$, $b_1$, $B_0^{\text{const}}$, $B_0^{\text{msg}}$, $B_0^{\text{rnd}}$ (NTT), private $s \in (\cdot)^m$,
    $x_1$, $u$, $x_0$ (NTT)
Ensure: $w_1 = (s, u, x_0)$, $w_2 = x_1$, $w_3 = (w_{1,i} \cdot x_1)_i$ (NTT)
 1: Assert $\dim(A), \dim(b_1), \dim(B_0^\star)$ are consistent
 2: for $j = 1..n$ do                                  ▷ check proof-friendly equation row-wise
 3:     $L_1 \leftarrow \sum_{t=1}^{m} (b_{1,j} \odot A_{j,t}) \cdot s_t$
 4:     $L_2 \leftarrow \sum_{t=1}^{m} (A_{j,t} \cdot s_t) \cdot x_1$
 5:     $R \leftarrow B_{0,j}^{\text{const}} + \sum_i B_{0,i,j}^{\text{msg}} \cdot u_i + \sum_i B_{0,i,j}^{\text{rnd}} \cdot x_{0,i}$
 6:     assert $L_1 - L_2 - R = 0$ in NTT domain
 7: end for
 8: $w_1 \leftarrow (s, u, x_0)$;  $w_2 \leftarrow x_1$;  $w_3[i] \leftarrow w_1[i] \cdot w_2$
 9: return $(w_1, w_2, w_3)$
```

Algorithm 2 RunPACSSimulation — end-to-end SmallWood demo

2

```
 1: Load params; build ring $(N, q)$
 2: $(w_1, w_2, w_3) \leftarrow$BuildWitnessFromDisk(); $(A, b_1, B_0^{\text{c}}, B_0^{\text{m}}, B_0^{\text{r}}) \leftarrow$loadPublicTables()
 3: Build $\Omega = \{\omega_j\}_{j=1}^{s}$ from NTT grid; check $0 < s < q$ and distinctness
 4: Repack top $m_{\text{sig}}$ rows: $P_t(\omega_j) \leftarrow a_{t,j}$ (coefficient packing)
 5: Repack message and $x_0$ rows as column-constant packing
 6: Recompute $w_3[i] \leftarrow w_1[i] \cdot w_2$
 7: Choose $R = 2^W$, $\beta$, build BoundSpec; allocate decomposition/carry/slack columns
 8: Append new columns to $w_1$; (Sqs) $\leftarrow$ProverFillIntegerL2(...); append Sqs
 9: LVCS: commit rows $\rightarrow$ root; degree check (EvalStep2); sample FS $(\Gamma', \gamma')$
10: Build $F_{\text{par}}$: core gate; integer decomp; Sqs-coupling; carry/slack bits
11: Build $F_{\text{agg}}$: BBS-on-$\Omega$ via $\Theta'$; integer limb-sum+$\Delta$
12: Compute $\sum_\Omega F$; $M \leftarrow$BuildMaskPolynomials$(d_Q)$; $Q \leftarrow$BuildQ(...)
13: Open indices $E$; check Eq.(4) at openings; VerifyQ$(Q, \Omega)$; return verdict
```

Algorithm 3 appendDecompositionColumns

3

```
Require: Ring $r$, limbs $LS$, width $W$
Ensure: Struct DecompCols{D[0..LS-1], T[0..LS], Bit[LS][W]} (NTT zero-polys)
 1: for $\ell = 0..LS - 1$ do allocate $D[\ell]$ and $Bit[\ell][u]$ for $u = 0..W - 1$ (NTT-lifted zeros)
 2:     for $\ell = 0..LS$ do allocate $T[\ell]$ (NTT-lifted zeros)
 3:         return DecompCols
```

Algorithm 4 ProverFillIntegerL2 — digits/bits/remainders, carries & slack

4

```
Require: $w_1[0..m_{\text{sig}}-1]$ (NTT), spec $(q, R, W, LS)$, $\Omega$, $\ell$, $S_0 = |\Omega|$, $S_0^{-1}$
Ensure: Row poly Sqs; fills $D_\ell$, $Bit_{\ell,u}$, $T_\ell$, global carries $C_\ell$, slack $\Delta_\ell$
 1: Convert top $m_{\text{sig}}$ rows to coeff domain
 2: for $j = 1..s$ do                                                    ▷ per column
 3:     $a_t \leftarrow P_t(\omega_j) \in \mathbb{F}_q$; $S_{\mathbb{Z}} \leftarrow \sum_t \widetilde{a_t}^2 \in \mathbb{Z}$ (centered lift)
 4:     $\text{Sqs}(\omega_j) \leftarrow \sum_t a_t^2 \pmod{q}$
 5:     Base-$R$ split $S_{\mathbb{Z}} = \sum_{\ell=0}^{LS-1} D_\ell(\omega_j) R^\ell$ with $D_\ell(\omega_j) \in [0, R)$
 6: end for
 7: Interpolate $D_\ell, Bit_{\ell,u}$ and Sqs over $\Omega$ to degree $\leq s + \ell - 1$
 8: Build integer remainders: $T_0(\omega_j) \leftarrow S_{\mathbb{Z}} \bmod q$, $T_{\ell+1} = \lfloor (T_\ell - D_\ell)/R \rfloor$; interpolate rows
 9: $B \leftarrow \beta^2$ as big-int from limbs; $\Delta \leftarrow \max\{0, B - \sum_j S_{\mathbb{Z}}^{(j)}\}$
10: Split $\Delta$ in base-$R$: $\Delta_\ell \in [0, R)$; compute carry chain $C_0 = 0$, $C_{\ell+1} = \lfloor \frac{\sum_j D_\ell + C_\ell + \Delta_\ell - \beta_\ell^2}{R} \rfloor$
11: Embed as constant rows: write $\Delta_\ell \cdot S_0^{-1}$ and bits; $C_\ell \cdot S_0^{-1}$ and bits (all NTT)
12: return Sqs =0
```

Algorithm 5 BuildFparIntegerDecomp — degree-2 parallel constraints

5

```
Require: $Sqs$, DecompCols $(D_\ell, T_\ell, Bit_{\ell,u})$, spec $(LS, W, R)$
Ensure: List $F_{\text{par}}$
 1: for $\ell = 0..LS - 1$ do
 2:     for $u = 0..W - 1$ do                                   ▷ bitness in coeff domain
 3:         $F_{\text{par}} \mathrel{+}= \mathsf{InvNTT}(Bit_{\ell,u})^2 - \mathsf{InvNTT}(Bit_{\ell,u})$ then NTT
 4:     end for
 5:     $F_{\text{par}} \mathrel{+}= D_\ell - \sum_{u=0}^{W-1} 2^u \, Bit_{\ell,u}$                         (NTT linear combo)
 6: end for
 7: $F_{\text{par}} \mathrel{+}= T_0 - Sqs$
 8: for $\ell = 0..LS - 1$ do
 9:     $F_{\text{par}} \mathrel{+}= T_\ell - D_\ell - R\,T_{\ell+1}$
10: end for
11: $F_{\text{par}} \mathrel{+}= T_{LS}$                                                    ▷ must be 0
12: return $F_{\text{par}}$
```

Algorithm 6 BuildFparSqsCoupling — enforce $Sqs - \sum_t P_t^2 = 0$

6

```
Require: $Sqs$, signature rows $\{P_t\}$ (NTT)
Ensure: Single row $F_{\text{par}}^{\text{sqs}}$
 1: $S \leftarrow 0$;
 2: for $t$ do $S \leftarrow S + \mathsf{NTT}\big(\mathsf{InvNTT}(P_t)^{\circ 2}\big)$
 3: end for
 4: return $Sqs - S$
```

Algorithm 7 buildFparGlobCarryBits — range-encode carries

7

```
Require: $C_\ell$ and bit rows $CBit_{\ell,u}$, width $W_c$, $S_0^{-1}$
 1: for $\ell = 0..LS$ do
 2:     for $u = 0..W_c - 1$ do
 3:         add bitness for $CBit_{\ell,u}$
 4:     end for
 5:     add $C_\ell - \sum_u (S_0^{-1} 2^u) CBit_{\ell,u} = 0$
 6: end for
 7: add boundary constraints: $C_0 = 0$, $C_{LS} = 0$
```

Algorithm 8 buildFparGlobSlackBits — range-encode slacks

8

```
Require: $\Delta_\ell$ and bits $\Delta Bit_{\ell,u}$, width $W$, $S_0^{-1}$
1: for $\ell$ do bitness on each $\Delta Bit_{\ell,u}$; add $\Delta_\ell - \sum_u (S_0^{-1} 2^u)\Delta Bit_{\ell,u} = 0$
2: end for
```

Algorithm 9 buildFaggIntegerSumDelta — aggregated limb-sum with slack

9

```
Require: $D_\ell$ rows, $C_\ell$, $\Delta_\ell$, $R$, $\Omega$, $S_0^{-1}$
1: for $\ell = 0..LS-1$ do
2:     $S_\ell \leftarrow S_0^{-1} \cdot \sum_{\omega\in\Omega} D_\ell(\omega)$          ▷ as constant row
3:     $B_\ell \leftarrow S_0^{-1} \cdot \beta_\ell^2$          ▷ as constant row
4:     Add constraint:   $S_\ell + C_\ell + \Delta_\ell - B_\ell - R\,C_{\ell+1} = 0$
5: end for
```

Algorithm 10 BuildThetaPrimeSet — interpolate public tables over Ω

10

```
Require: Public $A, b_1, B_0^{\mathrm{const}}, B_0^{\mathrm{msg}}, B_0^{\mathrm{rnd}}$ (NTT), $\Omega$
Ensure: $\Theta' = \{$polys of degree $\le s-1\}$ s.t. $P(\omega_k) = \mathrm{table}[k]$
1: for each entry varying with column $k$ do
2:     evaluate NTT poly in coeff domain at each $\omega_k$ to get values
3:     interpolate values on $\Omega$ to degree $\le s-1$; NTT-lift result
4: end for
5: return structured $\Theta'$
```

Algorithm 11 buildFaggOnOmega — BBS aggregated rows using Θ'

11

```
Require: $w_1, w_2$ (NTT), $\Theta'$, $m_{\mathrm{sig}}$
1: for each row $j$ do
2:     $L_1 \leftarrow \sum_{t<m_{\mathrm{sig}}} \Theta'(b_1)_j \odot \Theta'(A)_{j,t} \cdot w_{1,t}$
3:     $L_2 \leftarrow \sum_{t<m_{\mathrm{sig}}} \Theta'(A)_{j,t} \cdot w_{1,t} \cdot w_2$
4:     $R \leftarrow \Theta'(B_0^{\mathrm{const}})_j + \sum_i \Theta'(B_0^{\mathrm{msg}})_{i,j} \cdot w_{1,m_{\mathrm{sig}}+i} + \sum_i \Theta'(B_0^{\mathrm{rnd}})_{i,j} \cdot w_{1,\cdot}$
5:     $F_j'(X) \leftarrow L_1 - L_2 - R$
6: end for
7: return $\{F_j'\}$
```

Algorithm 12 BuildMaskPolynomials — choose M_i s.t. $\sum_\Omega Q_i = 0$

12

```
Require: $\rho$, degree bound $d_Q$, $\Omega$, $\Gamma'$, $\gamma'$, $\{\sum_\Omega F_{\mathrm{par}}\}$, $\{\sum_\Omega F_{\mathrm{agg}}\}$
1: Check $0 < s < q$ and distinct $\Omega$
2: Precompute power-sums $S_k = \sum_{\omega\in\Omega} \omega^k$ for $k = 0..d_Q$
3: for $i = 1..\rho$ do
4:     Sample $a_{i,1}, \ldots, a_{i,d_Q} \xleftarrow{\$} \mathbb{F}_q$
5:     $\Sigma \leftarrow \sum_j \Gamma'_{i,j} \sum_\Omega F_{\mathrm{par},j} + \sum_j \gamma'_{i,j} \sum_\Omega F_{\mathrm{agg},j}$
6:     $a_{i,0} \leftarrow -\Sigma \cdot S_0^{-1}$          ▷ since $\sum_\Omega X^0 = S_0$ and $\sum_\Omega X^k$ known
7:     $M_i(X) \leftarrow \sum_{k=0}^{d_Q} a_{i,k} X^k$ (NTT-lift)
8: end for
9: return $\{M_i\}$
```

Algorithm 13 BuildQ — assemble batched Q_i

13

```
Require: $M_i$, $F_{\text{par}}$, $F_{\text{agg}}$, $\Gamma'_{i,\cdot}$, $\gamma'_{i,\cdot}$ (all NTT)
1: for $i = 1..\rho$ do
2:     $Q_i \leftarrow M_i + \sum_j \Gamma'_{i,j} F_{\text{par},j} + \sum_j \gamma'_{i,j} F_{\text{agg},j}$
3: end for
4: return $\{Q_i\}$
```

Algorithm 14 VerifyQ — check $\sum_{\omega\in\Omega} Q_i(\omega) = 0$

14

```
Require: $Q_i$ (NTT), $\Omega$
1: Check $0 < s < q$ and distinct $\Omega$
2: for $i = 1..\rho$ do
3:     $q_i(X) \leftarrow \mathsf{InvNTT}(Q_i)$
4:     $S \leftarrow \sum_{\omega\in\Omega} q_i(\omega)$                ▷ Horner eval mod $q$
5:     if $S \neq 0$ then return False
6: end for
7: return True
```

Algorithm 15 columnsToRows — row view for LVCS exposure

15

```
Require: $w_1[0..s-1], w_2, w_3[0..s-1]$ (NTT), $\Omega$
Ensure: Matrix of evaluations: each witness column as a row
1: for $k = 0..s-1$ do $rows[k][j] \leftarrow \mathsf{EvalPoly}(\mathsf{InvNTT}(w_1[k]), \omega_j)$
2: end for
3: $rows[s][j] \leftarrow \mathsf{EvalPoly}(\mathsf{InvNTT}(w_2), \omega_j)$
4: $rows[s+1][j] \leftarrow \mathsf{EvalPoly}(\mathsf{InvNTT}(w_3[j]), \omega_j)$
5: return $rows$
```

Algorithm 16 buildFpar — core parallel multiplicative gate

16

```
Require: Ring $r$, signature block $w_1[0..m_{\text{sig}}-1]$ (NTT rows), scalar row $w_2$ (NTT), product block $w_3[0..m_{\text{sig}}-1]$ (NTT)
Ensure: List $F^{\text{core}}_{\text{par}}$ of degree-$\leq 2$ constraints
1: $F^{\text{core}}_{\text{par}} \leftarrow [\,]$
2: for $i = 0..m_{\text{sig}}-1$ do
3:     $P \leftarrow r.\mathsf{NewPoly}()$
4:     $P \leftarrow w_3[i] \; - \; (w_1[i] \odot w_2)$        ▷ In NTT: multiply $w_1[i]$ by $w_2$ pointwise
5:     append $P$ to $F^{\text{core}}_{\text{par}}$
6: end for
7: return $F^{\text{core}}_{\text{par}}$
```

Algorithm 17 CHECKEQ4ONOPENING — Eq.(4) batched identity on opened points

17

```
Require: Ring $r$, batched polynomials $Q[1..\rho]$ (NTT), masks $M[1..\rho]$ (NTT), DECS opening bundle open, lists $F_{\text{par}}, F_{\text{agg}}$ (NTT), matrices $\Gamma', \gamma'$, indices $E \subseteq \{1..|\Omega|\}$
Ensure: TRUE/FALSE
 1: for each $e \in E$ do                                  ▷ work pointwise using opened evaluations
 2:     Extract from open all needed $P(\omega_e)$ values to evaluate each $F_{\text{par},t}(\omega_e)$ and $F_{\text{agg},u}(\omega_e)$
 3:     for $i = 1..\rho$ do
 4:         $lhs \leftarrow Q_i(\omega_e)$                  ▷ evaluate by InvNTT+Horner or directly from opening
 5:         $rhs \leftarrow M_i(\omega_e)$
 6:         $rhs \leftarrow rhs + \sum_t \Gamma'_{i,t} F_{\text{par},t}(\omega_e) + \sum_u \gamma'_{i,u} F_{\text{agg},u}(\omega_e)$
 7:         if $lhs \neq rhs$ then return FALSE
 8:         end if
 9:     end for
10: end for
11: return TRUE
```

Algorithm 18 EVALPOLY — Horner evaluation in $\mathbb{F}_q$

18

```
Require: Coefficients $c[0..d] \in \mathbb{F}_q$, point $x \in \mathbb{F}_q$, modulus $q$
Ensure: $y = c(x) \bmod q$
 1: $y \leftarrow 0$
 2: for $i = d..0$ do
 3:     $y \leftarrow (y \cdot x + c[i]) \bmod q$
 4: end for
 5: return $y$
```

Algorithm 19 INTERPOLATE — Lagrange on distinct grid $x_0, \ldots, x_{n-1}$

19

```
Require: Points $x[0..n-1]$ distinct in $\mathbb{F}_q$, values $y[0..n-1] \in \mathbb{F}_q$
Ensure: Coeffs $c[0..n-1]$ of $P(X)$ with $P(x_i) = y_i$
 1: $c \leftarrow$ zero polynomial
 2: for $i = 0..n-1$ do
 3:     // basis numerator  $N_i(X) = \prod_{j \neq i}(X - x_j)$
 4:     $N \leftarrow [1]$                                                    ▷ poly 1
 5:     for $j = 0..n-1$, $j \neq i$ do
 6:         $N \leftarrow N \cdot [(-x_j), 1] \pmod q$
 7:     end for
 8:     // basis denominator  $D_i = \prod_{j \neq i}(x_i - x_j) \in \mathbb{F}_q^{\times}$
 9:     $D \leftarrow 1$;
10:     for $j = 0..n-1$, $j \neq i$ do $D \leftarrow D \cdot (x_i - x_j) \bmod q$
11:     end for
12:     $c \leftarrow c + y[i] \cdot D^{-1} \cdot N \pmod q$
13: end for
14: return $c$
```

Algorithm 20 SUMEVALS — $\sum_{\omega\in\Omega} P(\omega)$

20

```
Require: Ring $r$, NTT-poly $P$, grid $\Omega = \{\omega_j\}$, scratch coeff-poly $S$
Ensure: $S_\Omega = \sum_{\omega\in\Omega} P(\omega) \in \mathbb{F}_q$
 1: $q \leftarrow r.\mathsf{Modulus}[0]$; $r.\mathsf{InvNTT}(P, S)$
 2: $s \leftarrow 0$;
 3: for $\omega \in \Omega$ do
 4:     $s \leftarrow s + \text{EVALPOLY}(S.\mathsf{Coeffs}[0], \omega, q) \bmod q$
 5: end for
 6: return $s$
```

Algorithm 21 SUMPOLYLIST — vectorized $\sum_\Omega$ over a list

21

```
Require: Ring $r$, list $\mathcal{P} = [P_1, \dots, P_m]$, grid $\Omega$
Ensure: $v[i] = \sum_\Omega P_i(\omega)$
 1: $v \leftarrow$ length-$m$ zeros; $S \leftarrow r.\mathsf{NewPoly}()$
 2: for $i = 1..m$ do
 3:     $v[i] \leftarrow \text{SUMEVALS}(r, P_i, \Omega, S)$
 4: end for
 5: return $v$
```

Algorithm 22 CHECKOMEGA — precondition $0 < |\Omega| < q$ and distinctness

22

```
Require: Grid $\Omega = \{\omega_j\}$, prime $q$
 1: if $|\Omega| = 0$ or $|\Omega| \geq q$ then
 2:     return ERROR
 3: end if
 4: Seen $\leftarrow \emptyset$
 5: for $\omega \in \Omega$ do
 6:     if $\omega \in$ Seen then
 7:         return ERROR
 8:     end if
 9:     Insert $\omega$ into Seen
10: end for
11: return OK
```

Algorithm 23 MAKECONSTROW — constant polynomial row (degree 0), NTT-lifted

23

```
Require: Ring $r$, scalar $c \in \mathbb{F}_q$
Ensure: NTT-poly $P$ with $P(\omega) = c$ for all $\omega$
 1: $P \leftarrow r.\mathsf{NewPoly}()$; set $P.\mathsf{Coeffs}[0][0] \leftarrow c$ and all other coeffs 0
 2: $r.\mathsf{NTT}(P, P)$; return $P$
```

Algorithm 24 SCALEPOLYNTT / MULSCALARNTT / ADDINTO — NTT primitives

24

```
Require: Ring $r$
 1: SCALEPOLYNTT$(P, \lambda, T)$: set $T \leftarrow \lambda \cdot P$ (component-wise mod $q$)
 2: MULSCALARNTT$(r, P, \lambda, T)$: alias of SCALEPOLYNTT
 3: ADDINTO$(r, Acc, Q)$: set $Acc \leftarrow Acc + Q$ (component-wise mod $q$)
```

Algorithm 25 BoundSpecFromParams — digits of β^2 and widths

25

```
Require: $\beta^2 \in \mathbb{Z}_{\geq 0}$, radix $R = 2^W$, grid size $s$
 1: $LS \leftarrow \lceil \log_R(\beta^2 + 1) \rceil$
 2: Split $\beta^2 = \sum_{\ell=0}^{LS-1} \beta_\ell^2 R^\ell$, with $0 \leq \beta_\ell^2 < R$
 3: $U_C \leftarrow \lceil \log_2(s) \rceil + 1$                    ▷ carry width
 4: $U_\Delta \leftarrow W$
 5: return BoundSpec$\{LS, W, R, \{\beta_\ell^2\}, U_C, U_\Delta\}$
```

Algorithm 26 bytesU64Mat/Vec & newFSRNG & sampleFSMatrix — FS sampling (sketch)

26

```
Require: Transcript bindings: Merkle root, hash($R$), matrix $C$, grid $\Omega$
 1: seed $\leftarrow$ Hash(“GammaPrime” ∥ root ∥ hashR ∥ bytes($C$) ∥ bytes($\Omega$))
 2: sampleFSMatrix($\rho, k, q,$ seed): PRF-expand to $\Gamma' \in \mathbb{F}_q^{\rho \times k}$
 3: Similarly derive $\gamma'$ with domain separation label “gammaPrime”
 4: return $(\Gamma', \gamma')$
```