

OpenAI Math Digest

CATALOGUE 109

Integer multiplication below $n^{\log n}$

Catalogue guide & one proof overview

Independent reading guide · English edition

Edition: 2026-10-22

Created by: Masataka Iwai · Osaka University

AI-generated explanations may contain errors. Check important claims and proofs against the original manuscripts. This booklet collects the website's explanations, not the original papers. Expert review and formal verification have not been performed.

Contents

The catalogue guide is followed by the article. The contents and citations are clickable. Wide tables are arranged as entries for print; the claims and source references are retained.

00 · Catalogue guide	3
----------------------	---

01 · Integer multiplication below $n \log n$	6
--	---

Original-manuscript snapshot: 1581d4cbdd69a0faf34b2089614fe2a814316a6a

[Catalogue 109 · Guide](#)[Read on the website ↗](#)

CATALOGUE 109

Integer multiplication below $n \log n$

This catalogue contains a manuscript claiming exact multiplication of two n -bit integers in worst-case time $O(n(\lg n)^{1-\kappa})$, with $\kappa = 2^{-182}$, on one deterministic Turing machine with a fixed finite alphabet and a fixed finite number of one-dimensional tapes. Savings in address interchange and simultaneous butterflies from linear networks are connected to exact integer recovery from approximate transforms.

Paper in this catalogue

These summaries give the principal scope; consult the cited original results for the complete hypotheses and statements.

Catalogue 109 · 2026-10-22 checked source snapshot

#

01

Manuscript and overview

Integer multiplication below $n \log n \rightarrow$

Short name: Integer multiplication below $n \log n$

September 23, 2026 · 73 pages

Original paper (GitHub) ↗

Principal claim

One deterministic fixed-finite-tape Turing machine outputs the exact product of every pair of n -bit inputs in worst-case $O(n(\lg n)^{(1-\kappa)}), \kappa = 2^{-182}$.

Theorem 1 · p. 3 ↗

Role and available overview

Transfers rank deficits of finite linear networks to volume-normalized recurrences for address interchange and simultaneous butterflies; retained permutations, precision bounds and integer-coefficient recovery complete the theorem.

Read the proof overview →

Entry to the proof

The article follows the construction from radix digits through CRT, Gaussian resampling and synthetic polynomial transforms, then explains two volume-normalized recurrences and the precision and complete cost bounds. Established integer multiplication and resampling analysis of Harvey and van der Hoeven are direct inputs. The polynomial products do not recursively invoke the improved multiplier under construction.

The bound is asymptotic through all input lengths; the manuscript explicitly describes its constants and thresholds as extremely large. No practical implementation benchmark is included.

Theorem 1 · p. 3

Sources and checking scope

The official subject is Theoretical computer science. We use official catalogue 109 and the manuscript dated 23 September 2026. The article checks the main theorem's input/output contract, recursion on fixed tapes, exact recovery from error bounds, and the derivation of κ against the proof text. It is not an independent verification of the complete proof or an implementation test. Consequences for division, square roots and matrix transposition are outside scope. The article records the external source edition and remaining limits.

OpenAI Research Catalog · Theoretical computer science; catalogue 109 · p. 13

Sections 3–9 · pp. 11–67

Integer multiplication below $n \log n$

Fixed linear networks save on both data movement and simultaneous butterflies; controlled errors are then removed by exact recovery.

The manuscript reduces integer multiplication to multidimensional normalized convolutions and accelerates both their tape layout changes and their arithmetic layers. Its central mechanism turns a rank deficit in a fixed linear network into an improved exponent in a recurrence normalized by the stored bit volume. This article explains the author's proof construction; it does not constitute an independent verification of the complete proof.

Manuscript author: OpenAI · September 23, 2026 · 73 pages, including references

[Read the manuscript on GitHub ↗](#)

1. Main results

Theorem 1 — Exact multiplication on a fixed number of one-dimensional tapes

For a binary string x of length n , let $\text{val}(x)$ be its nonnegative value, with the most significant bit on the left. Write $\text{bin}_k(z)$ for its binary representation padded on the left to length k , and put $\lg n = \max\{\lceil \log_2 n \rceil, 1\}$.

There exists **one deterministic Turing machine $\mathcal{A}$** , with a fixed finite alphabet and a fixed finite number of one-dimensional tapes, that, for every $n \geq 1$ and every $x, y \in \{0, 1\}^n$, takes $x \# y$ to exactly

$$\text{bin}_{2n}(\text{val}(x) \text{ val}(y)).$$

Its worst-case time satisfies

$$T_{\mathcal{A}}(n) = O(n(\lg n)^{1-\kappa}), \quad \kappa = 2^{-182}.$$

Leading zeroes are allowed. Including the separator, the input has length $2n + 1$, and the output has exactly $2n$ bits. One step reads and writes at most one cell on each tape and moves each head by at most one cell. The statement does not supply a different circuit or advice for each n .

Theorem 1 · p. 3

The bound is asymptotic through all input lengths. Against an eventual $\Omega(n \lg n)$ lower bound for every fixed multiplication machine, this machine satisfies $T_A(n)/(n \lg n) \rightarrow 0$. The manuscript explicitly describes the constants and thresholds as extremely large and makes no claim of practical speed. Its intermediate XOR computations also distinguish it from transposition models restricted to moving data.

Sections 1.7 and 2 · pp. 7–8

References on the arrows

Unprefixed theorem, lemma, section and equation numbers refer to [Integer multiplication below \$n \log n\$](#) . External inputs use the following keys.

- [HvH21](#) David Harvey and Joris van der Hoeven — Integer multiplication in time $O(n \log n)$: Independent established multiplier, Gaussian resampling analysis and numerical routines, phase approximations, and prime intervals. Statements and applications are checked, not complete external proofs.. HAL hal-02070778v2 (submitted 28 November 2020; 45-page manuscript plus cover). Byte identity with the author-site file specified by the manuscript is unconfirmed..

Each citation gives the result and pages. When printed pagination differs from the PDF position, both are shown. GitHub previews do not automatically jump to the cited page.

2. Proof of Theorem 1 — From the input to the exact product

Proof target : [Theorem 1 · p. 3](#)

Proof route. A convolution-preserving CRT map places radix digits on a product of prime cyclic axes; Gaussian resampling moves the source transforms to power-of-two tensor transforms. Retained frequency permutations are handled inside the chirp and convolution identities. The final axis becomes the coefficient list of $R_r = \mathbb{C}[y]/(y^r + 1)$. Synthetic-root transforms, polynomial products using an independent existing multiplier, and an opposite transform

evaluate the convolution. Known scales are then removed to recover integer coefficients and the product. The next section explains the recurrences responsible for the saving; Section 4 treats errors and total cost.

1 Encode the radix digits exactly

$$x \# y, \quad b = \lceil \log_2 n \rceil, \quad p = 6b, \quad a_j \mapsto a_j 2^{-b-2} \\ q = \lceil n/b \rceil, \quad 2q - 2 < S, \quad T/2 < S < T, \quad Tp = \Theta(n)$$

Avoid wraparound; place the same coefficients in CRT order

Sections 9.2–9.3 and 9.6 · pp. 54–58, 60–61

2 Preserve convolution on prime axes

$$\Phi : \mathbb{C}[x]/(x^S - 1) \simeq \mathbb{C}[x_1, \dots, x_d]/(x_i^{s_i} - 1)_i$$

Triangular controlled shifts and reversible zero padding

Use the Gaussian maps and the chirp identity

Lemmas 19–21 · pp. 48–53

3 Retain permutations inside the transforms

$$RF_s = 2^\gamma B_0 Q F_t A, \quad Q F_t z = \bar{a} \cdot (a * (\bar{a} \cdot z))/T$$

A, B_0 are contractions; no separate sort for Q

Twist the last axis into coefficients; multiply in common LB order

Section 9.4 · pp. 58–59

Lemma 17; Proposition 18 · pp. 44–47

4 Use synthetic transforms and an independent multiplier

$$R_r = \mathbb{C}[y]/(y^r + 1), \quad x_d \mapsto e^{\pi i/r} y$$

$$LBF_t^- \rightarrow \text{signed packed products} \rightarrow F_t^+ B^{-1} L^{-1}$$

Cancel R and use known scales and nearest-integer recovery

Section 9.7.2; equation (44) · pp. 63–64

5 Undo scales and output the exact product

$$\frac{u * v}{S^2} \xrightarrow{\times S} w = \frac{u * v}{S} \xrightarrow{\times 2^{2b+4} S} \text{integer coefficients}$$

Round at error $< 1/2$; apply Φ^{-1} , carries, and exactly $2n$ output bits

Reductions from the input and recovery of the exact product

[Open figure ↗](#) [TeX source ↓](#)

1. Radix digits and a convolution-preserving address map

For sufficiently large n , set $b = \lceil \log_2 n \rceil$, precision $p = 6b$, and digit count $q = \lceil n/b \rceil$. Choose a power of two $T \in [4n/b, 8n/b)$ and distinct odd primes s_i just below power-of-two lengths $t_i \in \{r/2, r\}$. Then $S = \prod_i s_i$ and $T = \prod_i t_i$ satisfy $T/2 < S < T$. Writing each operand as $\sum_{j < q} a_j 2^{bj}$, the product has degree $2q - 2 < S$, so cyclic convolution modulo $x^S - 1$ introduces no nonzero wraparound. Its coefficients satisfy $0 \leq c_j < 2^{3b}$.

Put $P_i = \prod_{j < i} s_j$ and $\mu_i = P_i^{-1} \bmod s_i$. The map

$$\Phi : \mathbb{C}[x]/(x^S - 1) \longrightarrow \mathbb{C}[x_1, \dots, x_d]/(x_1^{s_1} - 1, \dots, x_d^{s_d} - 1), \quad x \longmapsto \prod_i x_i^{\mu_i}$$

is an algebra isomorphism permuting the monomial bases. A mixed-radix index $k = \sum_i a_i P_i$ moves to $b_i = a_i + \mu_i \sum_{j < i} a_j P_j \pmod{s_i}$. The triangular updates are ordered so that their lower-index controls have not yet changed. An extension fixing invalid padding intervals preserves zero padding, and the inverse map restores ordinary coefficient order. Each real input digit is stored exactly as $a_j 2^{-b-2}$, so the arrays u, v have norm less than $1/4$.

Sections 9.2–9.3 and 9.6 · pp. 54–58, 60–61

2. Resampling and chirps with frequency permutations retained

Write $F_s = \bigotimes_i F_{s_i}$ for the normalized Fourier transform. Lemma 19 implements the one-dimensional Gaussian maps with ordered input and output; Lemma 20 tensors them. The resulting identity is

$$RF_s = 2^\gamma B_0 Q F_t A, \quad \gamma = 2d\alpha^2.$$

Here A, B_0 are contractions, and R, Q are explicit coordinate permutations. Instead of separately executing the two label-and-sort operations in the usual resampling algorithm, the manuscript converts $Q F_t$ into a normalized cyclic convolution by a chirp identity. For its chirp array a , one has $Q F_t z = \bar{a} \cdot (a * (\bar{a} \cdot z))/T$. The remaining R cancels across the two forward transforms, their pointwise product, and the opposite-sign transform:

$$(RF_s^+) [(RF_s u) \cdot (RF_s v)] = \frac{u * v}{S^2}.$$

The transform F_s^+ is itself normalized by $1/S$; it is not the inverse matrix of F_s . These normalizations explain the two later multiplications by S .

Lemmas 19–20 · pp. 48–52

Lemma 21; equations (36)–(37) · p. 53

3. Synthetic transforms, retained order, and polynomial products

The substitution $x_d \mapsto \zeta y$, with $\zeta = e^{\pi i/r}$, moves the last cyclic axis into R_r . Since $(\zeta y)^r = 1$, this twist carries cyclic multiplication exactly into negacyclic polynomial multiplication. The sign from $y^r = -1$ cancels on untwisting. Powers of y are signed cyclic shifts of coefficient lists, so multiplication by the synthetic roots for the other axes uses linear scans.

The forward and opposite procedures of Lemma 16 implement LBF_t^- and $F_t^+ B^{-1} L^{-1}$, respectively. Here B is frequency bit reversal and L is the layout suited to simultaneous layers. Both forward outputs remain in their common LB order for the pointwise product, and the opposite procedure consumes that order directly. No intervening full-array sort is required.

Lemma 17 handles each polynomial product. After removing the denominator 2^p , signed integer coefficients of the real and imaginary parts are packed in radix $B_0 = 2^{4p}$ (a local packing notation, distinct from the compression map). Four integer products yield the complex product. Every ordinary product coefficient has absolute value at most $r2^{2p} < B_0/2$, so centered residues and a carry uniquely recover the signed coefficients. Subtracting coefficient $j + r$ from coefficient j gives the negacyclic product; truncation occurs only after normalization by r . The $O(N \log N)$ multiplier used here is the established theorem of [HvH21], not a recursive invocation of the improved multiplier under construction.

Lemma 16 · pp. 40–44

Lemma 17; Proposition 18 · pp. 44–47

Section 9.4 · pp. 58–59

3. Proof of Theorem 1 — Turning rank deficits into tape-time savings

Proof target : Theorem 1 · p. 3

Proof route. The same representation principle accelerates two expensive operations in the overall construction. Arrays are split equally among W roles in a fixed network, so s child calls contribute the factor s/W , rather than s , to the normalized time. A strict deficit $s < Wm$ is used for both address interchange and numerical layers.

1 Align the endpoints of fixed networks

$$D_{\text{out}} S_{\Omega} D_{\text{in}}^{-1}, \quad m = 10^6$$

$$s_b < W_b m, \quad s_c < W_c m, \quad s_b/W_b < m^{\tau}, \quad s_c/W_c < m^{\sigma}$$

The identity covers every role and arbitrary auxiliary inputs

Keep the same complete suffix to apply pointwise gates

Propositions 7–8; equation (10) · pp. 18–20

2 Split whole rows equally among roles

Parent volume V ; s children, each of volume V/W

$$\text{recursive cost} = s(V/W)F(e/m)$$

Use lower-triangular factorization, row padding, and width splitting

Proposition 11; Lemma 12 · pp. 22–25

3a Address-interchange recurrence

$$F_k \leq (s_b/W_b)F_{k-1} + O(1)$$

A rank- a shear requires a smaller interchanges

arbitrary binary width u : $O(Vu^{\tau})$

Use chunk interchanges in packed basis changes for the numerical recurrence

Lemma 13 · pp. 28–31

3b Pay for basis changes in numerical layers

$$F(e) \leq (s_c/W_c)F(e/m) + O((eK)^{\tau} + 1)$$

$$F(e) = O(e) \quad (e < d^{\beta}) \implies O(Vd^{\lambda'})$$

Every exceptional address in packed XOR is repaired deterministically

Guard widths use arithmetic depth without a factor $1/W_c$

Proposition 14 · pp. 36–38

4 Control precision and fixed-tape scheduling

$$\Delta = \lceil C_0 d^{20} \rceil = o(p), \quad H_0 = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Evaluate the entire layer exactly, then truncate once

$$\|Q_p(H_0^{\otimes D} z) - H_0^{\otimes D} z\|_{\infty} < \sqrt{2} 2^{-p}$$

Two volume-normalized recurrences supplied by fixed networks

Open figure ↗ TeX source ↓

1. Changing frames while restoring arbitrary auxiliary values

Section 3 directly constructs two fixed networks from triples of three-element subsets of $[100]$. Side wires cancel unwanted intersection-size coefficients created by gather/scatter operations; three stages exchange two data banks. Auxiliary wires may start with arbitrary values: their old contribution is first subtracted, and their contents are restored after the computation. Lemma 5 proves this restoration, while Lemma 6 supplies nested subspace labels and counts their dimension losses.

At a vertex v , store a logical array g as $D_v g$. On an edge $u \rightarrow v$, apply $D_v D_u^{-1}$, and use the same D_v on every incidence of one gate. The pointwise linear gate then commutes with the frame. The full operation is $D_{\text{out}} S_{\Omega} D_{\text{in}}^{-1}$. The inverse input frame specifies a logical interpretation and needs no separate preprocessing of the supplied input.

For the bit network with rational subspace labels, the endpoint matrix difference is the identity on every role, and the edge-rank sum $s_b < W_b m$ supplies an address shear. For the complex network with binary subspace labels, the sum $s_c < W_c m$ of orthogonal residual dimensions supplies phase kernels. Both have $m = 10^6$, with $s_b/W_b < m^{\tau}$, $s_c/W_c < m^{\sigma}$, and $\tau = \sigma = 1 - 2^{-50}$. The networks are enormous but independent of the input length.

Lemmas 5–6; equation (5) · pp. 11–17

Propositions 7–8; equation (10) · pp. 18–20

2. Exact address interchanges and deterministic exception repair

For the bit network, whole rows retaining their complete suffixes are distributed cyclically among W_b roles. Each child has the same address set and $1/W_b$ of its parent's bit volume. For a width $e = m^k$, time divided by volume satisfies

$$F_k \leq (s_b/W_b)F_{k-1} + O(1), \quad F_0 = O(1).$$

A lower-triangular factorization reduces a rational shear of rank a to exactly a smaller interchanges; this is why the number of children is the rank sum. Lemma 12 obtains rows from high digits, pads them, partitions widths in base m , and pads binary ranges to a fixed odd-prime radix. It thereby interchanges arbitrary nonadjacent width- u chunks in $O(Vu^{\tau})$ time.

Intermediate XOR strings are not interpreted as numerical coefficients; arithmetic resumes only after the encodings have been exactly permuted.

The numerical network also needs simultaneous XORs on selected bits of many chunks. Lemma 13 uses eight controlled cyclic shifts with a third chunk as auxiliary space. Addresses near a guard boundary constitute at most $80(f-1)2^{-K}$ of the array. These exceptions are extracted, assigned their correct destinations, stably sorted by binary keys, and reinserted. Every exception

is repaired deterministically, independently of payload values; there is no allowed failure probability. The conditions $K/\log p \rightarrow \infty$ and a long polynomial payload absorb the extra repair and address-computation costs.

Proposition 11; Lemma 12 · pp. 22–25

Lemma 13 · pp. 28–31

3. Simultaneous butterflies and recursion on fixed tapes

The complex network produces tensor kernels of $C = H \text{diag}(1, i)H$, where

$H = 2^{-1/2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. Diagonal phases and a dyadic scale convert them into a simultaneous layer of $H_0 = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. For e active axes the normalized time satisfies

$$F(e) \leq (s_c/W_c)F(e/m) + O((eK)^\tau + 1), \quad F(e) = O(e) \quad (e < d^\beta), \quad \beta = \frac{1}{2}.$$

Separating internal costs from leaf costs and choosing $\tau(1 + c/\beta) < \lambda$ and

$\max\{\lambda, \sigma + \beta(1 - \sigma)\} < \lambda' < 1$ yields $O(Vd^{\lambda'})$ for the complete layer, including the base- m grouping.

Intermediate frames need not be contractions. The arithmetic dependency depth used for precision satisfies $A(e) \leq s_c A(e/m) + E$; the volume factor $1/W_c$ does not enter this bound. Section 5.4 bounds denominator and magnitude growth by $\Delta = \lceil C_0 d^{20} \rceil$. The entire layer is evaluated exactly using $p + \Delta$ fractional bits and $\Delta + 3$ integer/sign bits. Since $20\epsilon < 1$, this is still an $O(p)$ -bit format. Only after the completed contraction layer are components truncated toward zero to $2^{-p}\mathbb{Z}$, producing error less than $\sqrt{2}2^{-p}$.

A fixed family of W role tapes is reused at every recursion depth. Inactive streams are parked at the top of a stack, then popped back in their original order after the child returns. Ancestors' parked data are not rescanned; copies, rewinds, and cleanup are charged to the current node's volume. Growing axis counts and recursion depths appear as descriptors stored on tapes, not as additional tapes or alphabet symbols.

Section 5.3; Proposition 14 · pp. 32–38

Proposition 11 proof; Section 5.3 · pp. 23, 33–34

4. Proof of Theorem 1 — Errors, complete cost, and all input lengths

Proof target : Theorem 1 · p. 3

Proof route. At precision $p = 6b$, preserve the disk conditions of every numerical call and undo both normalization factors and the input scales exactly. Error below $1/2$ uniquely recovers integer coefficients; a carry invariant then gives the product. Finally compare every cost exponent, including layout and setup, to obtain κ .

1 One fixed parameter choice

$$p = 6b, \quad d = \lfloor b^\epsilon \rfloor, \quad K = \lfloor d^c \rfloor, \quad Tp = \Theta(n)$$

$$\epsilon = 2^{-75}, \quad c = 2^{-56}, \quad 1 - \tau = 2^{-50}, \quad 1 - \lambda' = 2^{-54}$$

$$\gamma = 2d\alpha^2 \leq b/4, \quad r = 2^{\Theta(p^{1-\epsilon})}$$

Check contraction and norm margins at every call

Section 9.7.1; equation (43) · pp. 61–62

2 Propagate errors while preserving disk conditions

$$E_s = 9 \cdot 2^\gamma T L, \quad \|\tilde{w} - (u * v)/S\| < 28 \cdot 2^\gamma T^2 L \cdot 2^{-p}$$

Includes phases, layers, convolution, and the source scale

Remove the second S and input scales; use the carry invariant

Section 9.7.2; equation (44) · pp. 63–64

3 Round coefficients after exact scaling

$$\times 2^{2b+4}S: \quad \text{final error} < 448b \cdot 2^{-3b/4} < 1/2$$

$$\Phi^{-1}, \quad c_j + k_j = d_j + 2^b k_{j+1}, \quad 0 \leq k_j < 2^{2b}$$

Count every operation in time normalized by $V = Tp$

Table 1; equation (45) · p. 65

4 Compare all layout, arithmetic, and setup exponents

$$\min_i g_i = \epsilon c(1 - \tau) = 2^{-181} = 2\kappa$$

$$\kappa = 2^{-182}, \quad (\log p)^C \leq p^\kappa \quad \text{eventually}$$

Reserve half the smallest margin for the complete bound

Charge prime search, descriptors, stacks, and input/output

Section 9.8; Table 1; equation (45) · pp. 64–67

5 One machine for all input lengths

$$O(Tp p^{1-\kappa}) = O(n(\lg n)^{1-\kappa})$$

Use schoolbook multiplication below one fixed cutoff; output is exact

Finite alphabet; a fixed finite number of one-dimensional tapes

Exact recovery from approximations and the final cost bound

[Open figure ↗](#) [TeX source ↓](#)

1. Computable parameters and the hypotheses of each call

Use the fixed rational choices of Section 9.1:

$$\tau = \sigma = 1 - 2^{-50}, \quad \beta = \frac{1}{2}, \quad \delta = \frac{1}{16}, \quad \lambda = 1 - 2^{-52}, \quad c = 2^{-56}, \quad \lambda' = 1 - 2^{-54}, \quad \epsilon = 2^{-75}.$$

Set $d = \lfloor b^\epsilon \rfloor$, $K = \lfloor d^c \rfloor$, $r = 2^{\lceil (\log_2 T)/d \rceil}$, and $\ell = \log_2 r$. Choose the d axes from $r/2, r$ so that their product is T and the last axis has length r . Then

$$Tp = \Theta(n), \quad d = \Theta(p^\epsilon), \quad K = \Theta(p^{\epsilon c}), \quad \ell = \Theta(p^{1-\epsilon}), \quad r = 2^{\Theta(p^{1-\epsilon})}.$$

Take $\alpha = \lceil (12d^2b)^{1/4} \rceil$ and $\eta = 1/(4d)$, and choose distinct odd primes

$(1 - 2\eta)t_i < s_i \leq (1 + \eta)t_i$. For sufficiently large inputs, [HvH21, Lemma 5.1] supplies enough primes in each interval. These choices ensure $\gcd(s_i, t_i) = 1$, $2 \leq \alpha < \sqrt{p}$, and

$\alpha^4(t_i/s_i - 1) > p$, as required by resampling. Even testing every candidate by trial division costs only $O(dr^{3/2} \text{poly}(p)) = n^{o(1)} = o(n)$. Fixed rational-power comparisons, inverses, and descriptors are also computed by the machine. Preprocessing is not excluded as free advice or a free table.

Sections 9.1–9.2 · pp. 54–56

2. Undoing both normalizations and recovering coefficients and carries

Put $h_p = 2^{-p}$ and $L = \log_2 T$. Contraction of each synthetic-transform round lets its errors add; Lemma 16 gives error less than $\sqrt{2}\ell h_p$. The ring convolution's error includes its final factor $M = T/r$. After twists and chirps, the source-transform error is bounded by

$$E_s h_p, \quad E_s = 9 \cdot 2^\gamma T L, \quad \gamma = 2d\alpha^2 \leq b/4.$$

Section 9.7.1 checks norms as well as errors: source inputs of norm at most .5, chirp inputs at most .6, and convolution inputs with one operand at most .7 give successive margins. Thus every call stays within its specified disk grid, without clipping its output.

After truncating the product of the two forward outputs, $\|z\| < .5$. The exact counterpart of the opposite transform is $(u * v)/S^2$. The first multiplication by S produces an approximation $\tilde{w}$ to $w = (u * v)/S$, with error less than $28 \cdot 2^\gamma T^2 L h_p$. Multiplication by $2^{2b+4}S$ then removes the remaining S and both input scales. These integer products and shifts act exactly on numerators. The final error is

$$2^{2b+4}S \cdot 28 \cdot 2^\gamma T^2 L \cdot 2^{-6b} < 448b \cdot 2^{-b+\gamma} \leq 448b \cdot 2^{-3b/4} < \frac{1}{2}.$$

The nearest integer to the real component is therefore the correct c_j . Serialize it in a known $O(p)$ -bit format and restore ordinary coefficient order with Φ^{-1} . For $Q_0 = 2^b$, scan the recurrence

$$k_0 = 0, \quad d_j = (c_j + k_j) \bmod Q_0, \quad k_{j+1} = \left\lfloor \frac{c_j + k_j}{Q_0} \right\rfloor.$$

The bounds $0 \leq k_j < Q_0^2$ and the invariant $\sum_{h < j} c_h Q_0^h = \sum_{h < j} d_h Q_0^h + k_j Q_0^j$ ensure bounded carry width and exactness. One final reverse copy, removing only redundant zero padding, produces exactly $2n$ bits in most-significant-first order. Zero inputs and leading zeroes obey the same rules.

Section 9.7.1; equation (43) · pp. 61–62

Section 9.7.2; equation (44) · pp. 63–64

3. Dominant costs and the exponent $\kappa = 2^{-182}$

Dividing by $V = Tp = \Theta(n)$, the principal costs are dK for prefix layout and individual rounds, $pK^{\tau-1}$ for chunk exchanges, $\ell d^{\lambda'}$ for simultaneous layers, $d^2(1 + \ell^\tau)$ for CRT and axis layouts, $dp^{1/2+\delta}\alpha$ for Gaussian maps, dp^δ for chirps and related scalar operations, and $\log(rp)$ for polynomial products. The margins of their powers of p below one are

$$\begin{aligned} g_1 &= 1 - \epsilon(1 + c), & g_2 &= \epsilon c(1 - \tau), & g_3 &= \epsilon(1 - \lambda'), \\ g_4 &= 1 - \tau - \epsilon(2 - \tau), & g_5 &= \frac{1}{4} - \delta - \frac{3}{2}\epsilon, & g_6 &= 1 - \delta - \epsilon, & g_7 &= \epsilon. \end{aligned}$$

Their minimum is $g_2 = 2^{-75}2^{-56}2^{-50} = 2^{-181} = 2\kappa$. Absorbing any remaining fixed power of $\log p$ into p^κ gives $O(Vp^{1-\kappa})$. The smallest margin thus comes from the product of the very small chunk-grouping exponent and the saving in chunk interchange itself.

The accounting includes linear input, padding, recovery, carry and output scans; phase generation; signed-format conversions; opposite layout restoration; exception sorting; and parking/restoration on fixed tapes. There are T/r polynomial products on $O(rp)$ -bit integers, costing $O(Tp \log(rp))$ altogether. Per-node setup is $\text{poly}(p) = o(V)$, while per-record and per-Gaussian-line setup is absorbed because r exceeds every fixed polynomial. Choose one fixed cutoff satisfying all large-input conditions and use schoolbook multiplication below it. Enlarging the constant for the finitely many remaining lengths gives one machine with the stated worst-case bound for every $n \geq 1$.

Section 9.8; Table 1; equation (45) · pp. 64–67

5. External inputs and where they are used

The principal external input followed here is [HvH21]. The manuscript directly proves its finite networks, frame connections, shear decomposition, chirp identities, and signed packing. References cited as antecedents of those constructions are therefore not automatically treated as proof-dependency arrows.

Supplied result

[HvH21] Theorem 1.1

Use in this manuscript

Lemma 17, Sections 9.5 and 9.8

Content and hypotheses

Established fixed-tape $O(N \log N)$ multiplication on $O(rp)$ -bit or $O(p)$ -bit integers

Checking scope

Statement and application checked; complete external proof not independently verified

Supplied result

[HvH21] Theorem 4.2, Lemma 4.6, Proposition 4.7, Lemmas 4.8–4.12

Use in this manuscript

Lemmas 19 and 20

Content and hypotheses

Gaussian identity, convergence of the inverse, disk-grid approximations and line costs; Section 9.2 ensures $\theta > p/\alpha^4$ and $2 \leq \alpha < \sqrt{p}$

Checking scope

Statements, algorithm boundaries and the manuscript's connection checked; complete Gaussian analysis not independently verified

Supplied result

[HvH21] Lemma 2.12, Lemma 5.1

Use in this manuscript

Sections 9.2 and 9.4–9.5

Content and hypotheses

Exponential approximations for prepared phases and existence of prime lengths; arguments have $O(p)$ bits and prime-interval hypotheses are checked in Section 9.2

Checking scope

Statements and hypotheses checked; proofs in further external references not pursued

[HvH21, Theorem 1.1, print p. 1 / PDF p. 2]

[HvH21, Theorem 4.2; Lemma 4.6; Proposition 4.7; Lemmas 4.8–4.12, print pp. 24, 28–33 / PDF pp. 25, 29–34]

[HvH21, Lemmas 2.12 and 5.1, print pp. 17, 37–38 / PDF pp. 18, 38–39]

6. Return to the sources

- Theorem 1 · p. 3 — Read the input/output contract, quantifiers and worst-case bound.
- Lemmas 5–6; equation (5) · pp. 11–17 — Read restoration of arbitrary auxiliaries and the frame invariant.
- Proposition 11; Lemma 12 · pp. 22–25 — Follow the volume-normalized recurrence and fixed-tape schedule.
- Section 5.3; Proposition 14 · pp. 32–38 — Track basis-change costs, stopping rules and precision guards.
- Lemma 17; Proposition 18 · pp. 44–47 — Check unique signed coefficient extraction and normalization.
- Lemmas 19–20 · pp. 48–52 — Read the connection from external inputs to transforms retaining permutations.
- Section 9.7.2; equation (44) · pp. 63–64 — Check the two S factors, error below 1/2, and the carry invariant.
- Section 9.8; Table 1; equation (45) · pp. 64–67 — Inspect the complete costs, slack yielding kappa, and one small-input cutoff.

[HvH21] was checked in the author-deposited HAL version submitted on 28 November 2020. Its manuscript has 45 pages; the HAL cover shifts PDF positions one page beyond printed page numbers. The author-site file specified in reference [15] was inaccessible, so byte identity with that file remains unconfirmed; the cited result numbers and statements were checked. The original Proposition 4.7 costs $O(tp^{3/2+\delta}\alpha + tp \log t)$; removing the latter term is the modification achieved by this manuscript's Lemma 19 through retained permutations.

The reading covers the main theorem and machine model; the frames and rank deficits of Section 3; the recurrences, row splitting, exception repair and guard estimates of Sections 4–5; retained layouts and signed coefficient recovery in Sections 6–7; the resampling connections in Section 8; and the parameters, precision and total cost in Section 9. This explains the connections to every conclusion of the main theorem, but does not independently recompute every finite-network edge, exhaust all guard-boundary cases of Lemma 13, verify every low-level descriptor implementation, or test an executable multiplication program. The complete proofs of [HvH21] and its references have not been independently verified.

Corollary 2 on division and square roots, Section 11 on matrix transposition, and the separate manuscript's exact DFT application are outside this article's proof scope. Results in a different computational model are not transferred to the present bit-complexity claim.