A Tutorial on Digit Recurrence Algorithms for Division and Square Root

Joonsang Yoon

February 23, 2026

Abstract

Digit recurrence algorithms, especially those in the SRT (Sweeney, Robertson, and Tocher) family, form a standard foundation for high-speed hardware division and square root. Their key idea is to compute the result one digit at a time, representing these digits in a redundant signed-digit set. Redundancy creates overlap between the valid selection regions of adjacent digits. This overlap is the critical performance enabler: it allows the next digit to be selected using fast, truncated estimates of the internal state (the residual), rather than an exact residual value obtained via full-width carry-propagating addition. This tutorial develops the theory and practice of these methods from first principles. We derive the core recurrence relations for division and square root, establish the corresponding convergence/containment requirements, and show how those requirements translate into implementable digit-selection logic. To connect the mathematics to hardware design, we work through detailed Radix-2 and Radix-4 examples, including the determination of key precision parameters and the systematic derivation of final selection tables.

Contents

1 Introduction

Division and square root are fundamental arithmetic operations in modern microprocessors, with broad use in scientific computing, signal processing, and graphics. In contrast to addition and multiplication, division and square root are inherently iterative in hardware. Among the most prevalent high-performance approaches are *digit recurrence algorithms*.

A digit recurrence algorithm produces the result (a quotient or a square root) from most significant digit to least significant digit. Each iteration performs two tightly coupled actions:

1. select the next result digit, and
2. update an internal *residual* (partial remainder).

Because these actions repeat every cycle, the overall performance of the operation is determined by the speed of this loop.

A direct implementation runs into a bottleneck during digit selection. To choose the correct digit, the algorithm must compare the current residual to certain multiples of the divisor (for division) or to quantities derived from the evolving partial root (for square root). These comparisons appear to require the *exact* residual value at full precision. If the residual were maintained in conventional binary form, computing it exactly each cycle would require a full-width carry-propagating addition, thereby slowing the critical path.

SRT algorithms avoid this bottleneck by selecting digits from a *redundant signed-digit set* (for example, using $\{-1, 0, 1\}$ in Radix-2 rather than $\{0, 1\}$). Redundancy introduces overlap between the valid selection regions of neighboring digits. That overlap is the key performance enabler: even when the residual is known only approximately (via truncation), the selection logic can still choose a digit that keeps the recurrence within a safe region. In hardware, this makes it practical to keep the residual in carry-save form and base digit selection on a small number of leading bits, using only a short carry-propagate adder (CPA) and a small lookup table.

This tutorial develops the design methodology systematically:

- section 2 introduces redundant digit sets and the redundancy factor ρ, which quantifies the overlap that makes truncated selection possible.
- section 3 derives digit recurrence division: the residual recurrence, the containment requirement, and digit-selection intervals, followed by Radix-2 and Radix-4 selection logic derived using truncated operands.
- section 4 applies the same methodology to square root. We derive the corresponding recurrence and selection bounds, explain what changes relative to division, and work through Radix-2 and Radix-4 examples (including iteration-dependent startup constraints).
- section 5 summarizes the core principles and practical trade-offs.

2 Redundancy in Digit Sets

To understand how SRT algorithms bypass the exact-comparison bottleneck described in section 1, we must first formalize the concept of redundancy. High-speed digit recurrence algorithms rely on representing the result digits in a redundant signed-digit set. For a given radix B, instead of using a standard non-overlapping representation, SRT-style methods select each digit from a symmetric set:

$$q_{j+1}, s_{j+1} \in \{-a, -a+1, \ldots, a-1, a\}, \tag{2.1}$$

where the integer a is the maximum digit magnitude.

Redundancy is quantified by the *redundancy factor*:

$$\rho = \frac{a}{B-1}. \tag{2.2}$$

To enable fast digit selection, this factor must satisfy

$$\rho > \frac{1}{2}. \tag{2.3}$$

This inequality is essential because it guarantees that the digit-selection intervals for adjacent digits overlap. As derived later (e.g., in section 3), each digit k is valid only when the residual lies within a corresponding interval. When $\rho > \frac{1}{2}$, the interval for k overlaps the interval for $k+1$, so there exist residual values for which *either* digit leads to a valid next residual. That overlap is exactly what allows practical hardware to base digit selection on truncated residual estimates rather than on an exact full-precision residual. At the boundary $\rho = \frac{1}{2}$, overlap disappears (intervals merely touch), and selection becomes sensitive to exact comparisons.

In practice, ρ is typically chosen within

$$\frac{1}{2} < \rho \leq 1, \tag{2.4}$$

which implies the corresponding range for a:

$$\frac{B-1}{2} < a \leq B-1. \tag{2.5}$$

The lower bound ensures overlap. The upper bound is pragmatic. While a larger a increases overlap, it also increases the complexity of forming $q_{j+1}D$ (or analogous terms), often yielding diminishing returns. The case $a = B-1$ (so $\rho = 1$) is *maximally redundant*, while choices near $a \approx \frac{B-1}{2}$ (so $\rho \to \frac{1}{2}$) are *minimally redundant*.

3 Division by Digit Recurrence

Building on the principle of redundant digit sets, we now derive the complete digit recurrence algorithm for division. The fundamental mathematical relationship governing this operation is

$$N = DQ_n + r, \tag{3.1}$$

where N is the dividend, D the divisor, Q_n the final quotient, and r the final (unscaled) remainder.

For fractional arithmetic, it is common to normalize operands. We assume

$$N \in \left(0, \frac{1}{4}\right) \qquad \text{(Dividend)}$$

$$D \in \left[\frac{1}{2}, 1\right) \qquad \text{(Divisor)}$$

which yields a convenient quotient range:

$$Q_n \in \left(\frac{N_{\min}}{D_{\max}}, \frac{N_{\max}}{D_{\min}}\right) = \left(\frac{0}{1}, \frac{1/4}{1/2}\right) = \left(0, \frac{1}{2}\right). \tag{3.2}$$

3.1 Quotient digits and approximation

The quotient is generated over n iterations, indexed by $j = 0, 1, \dots, n-1$. Let Q_j be the current quotient approximation after j digits $(q_1, \dots, q_j)$ have been chosen. At each step, the next digit refines the approximation:

$$Q_{j+1} = Q_j + q_{j+1}B^{-(j+1)}. \tag{3.3}$$

Unfolding this recurrence gives

$$\text{At step } j, \quad Q_{j+1} = Q_0 + \sum_{i=0}^{j} q_{i+1}B^{-(i+1)}. \tag{3.4}$$

After n iterations,

$$\text{At step } j = n-1, \quad Q_n = Q_0 + \sum_{i=0}^{n-1} q_{i+1}B^{-(i+1)}. \tag{3.5}$$

3.2 Error bound and containment

At step j, define the approximation error

$$\varepsilon_{j+1} = \frac{N}{D} - Q_{j+1} = \sum_{i=j+1}^{\infty} q_{i+1}B^{-(i+1)}. \tag{3.6}$$

With $|q_{i+1}| \le a$,

$$\left| \sum_{i=j+1}^{\infty} q_{i+1}B^{-(i+1)} \right| \le \sum_{i=j+1}^{\infty} aB^{-(i+1)} = aB^{-(j+2)} \sum_{k=0}^{\infty} B^{-k} = aB^{-(j+2)} \frac{1}{1-B^{-1}} = \frac{a}{B-1}B^{-(j+1)}. \tag{3.7}$$

Using $\rho = \frac{a}{B-1}$,

$$|\varepsilon_{j+1}| \le \rho B^{-(j+1)}. \tag{3.8}$$

Equivalently,

$$\left| \frac{N}{D} - Q_{j+1} \right| \le \rho B^{-(j+1)}. \tag{3.9}$$

To obtain an "integer-like" quantity more suitable for hardware implementation, we scale this error bound by $B^{j+1}D$ to yield:

$$\left| B^{j+1}(N - DQ_{j+1}) \right| \le \rho D, \tag{3.10}$$

and set

$$R_{j+1} = B^{j+1}(N - DQ_{j+1}). \tag{3.11}$$

This yields the *containment condition*:

$$|R_{j+1}| \le \rho D. \tag{3.12}$$

At the end of the algorithm, the final residual relates to the final remainder:

$$\text{At step } j = n-1, \quad R_n = B^n(N - DQ_n) = B^n r. \tag{3.13}$$

3.3 Residual recurrence

Starting from the residual definition in eq. (3.11) and substituting the quotient update from eq. (3.3),

$$\begin{aligned} R_{j+1} &= B^{j+1}(N - DQ_{j+1}) \\ &= B^{j+1}\left(N - D\left(Q_j + q_{j+1}B^{-(j+1)}\right)\right) \\ &= B^{j+1}(N - DQ_j) - q_{j+1}D \\ &= B\left(B^j(N - DQ_j)\right) - q_{j+1}D \\ &= BR_j - q_{j+1}D. \end{aligned}$$

Thus the central recurrence for division is

$$R_{j+1} = BR_j - q_{j+1}D, \qquad j = 1, 2, \ldots, n-1. \tag{3.14}$$

Each iteration therefore consists of:

1. choosing a digit q_{j+1} from the redundant set, using (possibly truncated) information about R_j and D, and
2. updating the residual using eq. (3.14),

with the requirement that the next residual satisfy eq. (3.12).

3.4 Selection Intervals for a Positive Divisor

To determine the valid choices for the next quotient digit q_{j+1}, we must find the range of residuals that satisfy containment. For a positive divisor $D \in \left[\frac{1}{2}, 1\right)$, the containment condition

$$|R_{j+1}| \leq \rho D$$

is equivalent to

$$-\rho D \leq R_{j+1} \leq \rho D. \tag{3.15}$$

Substitute eq. (3.14):

$$-\rho D \leq BR_j - q_{j+1}D \leq \rho D. \tag{3.16}$$

Adding $q_{j+1}D$ yields the selection interval for digit q_{j+1}:

$$(q_{j+1} - \rho)D \leq BR_j \leq (q_{j+1} + \rho)D. \tag{3.17}$$

These intervals overlap when $\rho > \frac{1}{2}$, which is the property that enables robust selection under truncation.

3.5 Selection Intervals for a Negative Divisor

When the divisor is negative, $D \in \left[-1, -\frac{1}{2}\right)$, we have $|D| = -D$. In this case, the containment condition can be written as

$$|R_{j+1}| \leq \rho|D| \implies |R_{j+1}| \leq -\rho D, \tag{3.18}$$

which expands to

$$\rho D \leq R_{j+1} \leq -\rho D. \tag{3.19}$$

Substitute eq. (3.14):

$$\rho D \leq BR_j - q_{j+1}D \leq -\rho D. \tag{3.20}$$

Adding $q_{j+1}D$ yields

$$(q_{j+1} + \rho)D \leq BR_j \leq (q_{j+1} - \rho)D, \tag{3.21}$$

i.e., the interval endpoints reverse because $D < 0$.

3.6 Initialization

To initialize the recurrence, we must define the first residual R_1 and the first quotient digit q_1. The recurrence for $j \geq 1$ is:

$$R_{j+1} = BR_j - q_{j+1}D, \qquad j = 1, 2, \ldots, n-1 \tag{3.22}$$

with the initial residual given by $R_1 = B(N - DQ_1)$. From the error bound $|\varepsilon_{j+1}| \leq \rho B^{-(j+1)}$, we have for $j = 0, 1, \ldots, n-1$:

$$\left|\frac{N}{D} - Q_{j+1}\right| \leq \rho B^{-(j+1)}. \tag{3.23}$$

For $j = 0$, this implies:

$$Q_1 - \rho B^{-1} \leq \frac{N}{D} \leq Q_1 + \rho B^{-1}. \tag{3.24}$$

Given the normalized operand ranges $N \in \left(0, \frac{1}{4}\right)$ and $D \in \left[\frac{1}{2}, 1\right)$, the quotient is bounded by $\frac{N}{D} \in \left(0, \frac{1}{2}\right)$, and thus $Q_n \in \left(0, \frac{1}{2}\right)$. We define the first quotient approximation as $Q_1 = Q_0 + q_1 B^{-1}$. By setting $Q_0 = 0$, the first digit q_1 is chosen from the non-negative half of the digit set, $q_1 \in \{0, 1, \ldots, a\}$. Recall that $\rho = \frac{a}{B-1}$ and $\frac{1}{2} < \rho \leq 1$. To ensure the quotient range is fully covered, the minimum and maximum possible values of Q_1 must satisfy:

$$\min(Q_1) - \rho B^{-1} \leq 0 \implies Q_0 - \rho B^{-1} \leq 0, \tag{3.25}$$

$$\max(Q_1) + \rho B^{-1} \geq \frac{1}{2} \implies \left(Q_0 + aB^{-1}\right) + \rho B^{-1} \geq \frac{1}{2} \implies Q_0 + \rho \geq \frac{1}{2}. \tag{3.26}$$

These conditions hold since $Q_0 = 0$ and $\rho > \frac{1}{2}$.

For the Radix-2 case ($B = 2$, $\rho = 1$):

- **Select $q_1 = 0$:**
$$0 - \frac{1}{2} \le \frac{N}{D} \le 0 + \frac{1}{2} \implies -\frac{1}{2}D \le N \le \frac{1}{2}D.$$
- **Select $q_1 = 1$:**
$$\frac{1}{2} - \frac{1}{2} \le \frac{N}{D} \le \frac{1}{2} + \frac{1}{2} \implies 0 \le N \le D.$$

For the Radix-4 case ($B = 4$, $\rho = 2/3$):

- **Select $q_1 = 0$:**
$$0 - \frac{1}{6} \le \frac{N}{D} \le 0 + \frac{1}{6} \implies -\frac{1}{6}D \le N \le \frac{1}{6}D.$$
- **Select $q_1 = 1$:**
$$\frac{1}{4} - \frac{1}{6} \le \frac{N}{D} \le \frac{1}{4} + \frac{1}{6} \implies \frac{1}{12}D \le N \le \frac{5}{12}D.$$
- **Select $q_1 = 2$:**
$$\frac{1}{2} - \frac{1}{6} \le \frac{N}{D} \le \frac{1}{2} + \frac{1}{6} \implies \frac{1}{3}D \le N \le \frac{2}{3}D.$$

For a negative divisor, the analysis is analogous.

3.7 Example: Radix-2 SRT Division

Consider a Radix-2 implementation ($B = 2$) using a maximally redundant digit set $\{-1, 0, 1\}$, which means $a = 1$. The redundancy factor is then

$$\rho = \frac{a}{B-1} = \frac{1}{2-1} = 1. \tag{3.27}$$

3.7.1 Positive Divisor

Substituting $\rho = 1$ and $B = 2$ into eq. (3.17), the selection intervals become

$$q_{j+1} = \begin{cases} 1 & \text{if } 0 \le 2R_j \le 2D \\ 0 & \text{if } -D \le 2R_j \le D \\ -1 & \text{if } -2D \le 2R_j \le 0. \end{cases} \tag{3.28}$$

Because $\rho > 1/2$, these intervals overlap (for example, if $2R_j = 0.2D$, then both $q_{j+1} = 1$ and $q_{j+1} = 0$ are valid choices).

To avoid full carry propagation in the main loop, hardware implementations typically maintain R_j in carry-save form. Consequently, digit selection must operate on a comparable quantity derived from $2R_j$. To keep selection fast, we use a truncated estimate based on only the most significant bits.

Let the full-precision shifted residual be $2R_j = r_{j,2\ldots0}.r_{j,-1\ldots-\infty}$. In carry-save form, this is represented as the sum of a sum vector $Y_j = y_{j,2\ldots0}.y_{j,-1\ldots-\infty}$ and a carry vector $Z_j = z_{j,2\ldots0}.z_{j,-1\ldots-\infty}$:

$$r_{j,2\ldots0}.r_{j,-1\ldots-\infty} = y_{j,2\ldots0}.y_{j,-1\ldots-\infty} + z_{j,2\ldots0}.z_{j,-1\ldots-\infty}. \tag{3.29}$$

We form a truncated estimate t_j by adding only the top bits of Y_j and Z_j using a small CPA. Retaining α fractional bits yields:

$$t_{j,2\ldots0}.t_{j,-1\ldots-\alpha} = y_{j,2\ldots0}.y_{j,-1\ldots-\alpha} + z_{j,2\ldots0}.z_{j,-1\ldots-\alpha}. \tag{3.30}$$

This truncation introduces a one-sided error:

$$0 \le 2R_j - t_{j,2\ldots0}.t_{j,-1\ldots-\alpha} < 2 \cdot 2^{-\alpha} = 2^{-(\alpha-1)}. \tag{3.31}$$

Similarly, we use a truncated divisor estimate derived from its first β fractional bits. In what follows, $\mathrm{Int}(\cdot)$ denotes the integer value of the shown fixed-point word. This value is obtained by removing the binary point (equivalently, scaling by 2^α or 2^β and interpreting the result as an integer). Define integer-valued estimates for table-based selection (PLA):

$$\tau_j = \mathrm{Int}(2^\alpha \cdot t_{j,2\ldots0}.t_{j,-1\ldots-\alpha}) \in \left\{-2^{\alpha+2}, \ldots, 2^{\alpha+2}-1\right\} \qquad (\tau_j \le 2^{\alpha+1}R_j < \tau_j + 2), \tag{3.32}$$

$$\delta = \mathrm{Int}(2^\beta \cdot 0.1d_{-2\ldots-\beta}) \in \left\{2^{\beta-1}, \ldots, 2^\beta - 1\right\} \qquad (\delta \le 2^\beta D < \delta + 1). \tag{3.33}$$

The digit selector then chooses $k \in \{-1, 0, 1\}$ based on the integer pair (τ_j, δ).

To guarantee correct selection using these truncated estimates, the overlap between adjacent valid regions must be large enough to absorb the truncation uncertainty. This requirement is formalized by the *continuity condition.* Let $L_k(D) = (k-\rho)D$ and $U_k(D) = (k+\rho)D$ denote the ideal lower and upper bounds for digit k. This condition requires that adjacent regions overlap even under worst-case truncation:

$$\min(U_{k-1,\mathrm{trunc}}) \ge \max(L_{k,\mathrm{trunc}}). \tag{3.34}$$

Truncation of the residual lowers the effective upper bound by up to $2^{-(\alpha-1)}$. Truncation of the divisor yields $D \in \left[2^{-\beta}\delta, 2^{-\beta}(\delta+1)\right)$. Assuming positive coefficients of D:

$$\min(U_{k-1,\mathrm{trunc}}) = \min_D(U_{k-1}(D)) - 2^{-(\alpha-1)} = (k-1+\rho)(2^{-\beta}\delta) - 2^{-(\alpha-1)}, \tag{3.35}$$

$$\max(L_{k,\mathrm{trunc}}) = \max_D(L_k(D)) = (k-\rho)(2^{-\beta}(\delta+1)). \tag{3.36}$$

For negative coefficients, the roles of δ and $\delta + 1$ are swapped.

Applying this to the boundaries between regions:

- For $k = 1$ (U_0, L_1): $(0+1)(2^{-\beta}\delta) - 2^{-(\alpha-1)} \ge (1-1)(2^{-\beta}(\delta+1))$, which simplifies to $2^{-\beta}\delta - 2^{-(\alpha-1)} \ge 0$.
- For $k = 0$ (U_{-1}, L_0): $(-1+1)(2^{-\beta}(\delta+1)) - 2^{-(\alpha-1)} \ge (0-1)(2^{-\beta}\delta)$, which simplifies to $2^{-\beta}\delta \ge 2^{-(\alpha-1)}$.

This condition must hold for the worst case, which corresponds to the smallest possible δ. Since $D \in [1/2, 1)$, we have $\delta_{\min} = 2^{\beta-1}$. Thus,

$$2^{-\beta}(2^{\beta-1}) \ge 2^{-(\alpha-1)} \Rightarrow \alpha \ge 2.$$

Choosing the minimal sufficient precision $\alpha = 2$ and $\beta = 2$ gives:

$$\tau_j = \mathrm{Int}(4 \cdot t_{j,2\ldots0}.t_{j,-1}t_{j,-2}) \in \{-16, \ldots, 15\} \qquad (\tau_j \le 8R_j < \tau_j + 2), \tag{3.37}$$

$$\delta = \mathrm{Int}(4 \cdot 0.1d_{-2}) \in \{2, 3\} \qquad (\delta \le 4D < \delta + 1). \tag{3.38}$$

The final selection logic can then be implemented as the following table:

$$q_{j+1} = \begin{cases} 1 & \text{if } 0 \le \tau_j < 2(\delta+1) \\ 0 & \text{if } -\delta \le \tau_j \le \delta - 2 \\ -1 & \text{if } -2(\delta+1) - 2 < \tau_j \le -2. \end{cases} \tag{3.39}$$

3.7.2 Negative Divisor

For a negative divisor $D \in [-1, -1/2)$, the interval endpoints reverse:

$$q_{j+1} = \begin{cases} 1 & \text{if } 2D \le 2R_j \le 0 \\ 0 & \text{if } D \le 2R_j \le -D \\ -1 & \text{if } 0 \le 2R_j \le -2D. \end{cases} \tag{3.40}$$

The continuity condition becomes $\min(U_{k,\text{trunc}}) \ge \max(L_{k-1,\text{trunc}})$, where $U_k = (k-\rho)D$ and $L_k = (k+\rho)D$. Care is required here, because multiplying inequalities by negative numbers reverses their order. Ultimately, the same precision requirement $\alpha \ge 2$ results. Using $\alpha = 2$ and $\beta = 2$:

$$\tau_j = \text{Int}(4 \cdot t_{j,2\ldots0}.t_{j,-1}t_{j,-2}) \in \{-16, \ldots, 15\} \qquad (\tau_j \le 8R_j < \tau_j + 2), \tag{3.41}$$

$$\delta = \text{Int}(4 \cdot 1.0d_{-2}) \in \{-4, -3\} \qquad (\delta \le 4D < \delta + 1). \tag{3.42}$$

The corresponding selection logic is

$$q_{j+1} = \begin{cases} 1 & \text{if } 2\delta - 2 < \tau_j \le -2 \\ 0 & \text{if } \delta + 1 \le \tau_j \le -(\delta+1) - 2 \\ -1 & \text{if } 0 \le \tau_j \le -2\delta. \end{cases} \tag{3.43}$$

3.8 Example: Radix-4 SRT Division

For a Radix-4 implementation ($B = 4$), a common minimally redundant digit set is $\{-2, -1, 0, 1, 2\}$. This gives $a = 2$, and thus

$$\rho = \frac{a}{B-1} = \frac{2}{4-1} = \frac{2}{3}. \tag{3.44}$$

3.8.1 Positive Divisor

Substituting $\rho = 2/3$ and $B = 4$ into eq. (3.17), the selection intervals are

$$q_{j+1} = \begin{cases} 2 & \text{if } \frac{4}{3}D \le 4R_j \le \frac{8}{3}D \\ 1 & \text{if } \frac{1}{3}D \le 4R_j \le \frac{5}{3}D \\ 0 & \text{if } -\frac{2}{3}D \le 4R_j \le \frac{2}{3}D \\ -1 & \text{if } -\frac{5}{3}D \le 4R_j \le -\frac{1}{3}D \\ -2 & \text{if } -\frac{8}{3}D \le 4R_j \le -\frac{4}{3}D. \end{cases} \tag{3.45}$$

As in the Radix-2 case, the residual is held in carry-save form, and a truncated estimate t_j is computed using α fractional bits. This truncation yields a one-sided error $0 \le 4R_j - t_j < 2^{-(\alpha-1)}$. We define the corresponding integer estimates as:

$$\tau_j = \mathrm{Int}(2^\alpha \cdot t_{j,2\ldots0}.t_{j,-1\ldots-\alpha}) \in \left\{-2^{\alpha+2}, \ldots, 2^{\alpha+2}-1\right\} \qquad (\tau_j \le 2^{\alpha+2}R_j < \tau_j + 2), \tag{3.46}$$
$$\delta = \mathrm{Int}(2^\beta \cdot 0.1d_{-2\ldots-\beta}) \in \left\{2^{\beta-1}, \ldots, 2^\beta - 1\right\} \qquad (\delta \le 2^\beta D < \delta + 1). \tag{3.47}$$

The continuity condition $\min(U_{k-1,\mathrm{trunc}}) \ge \max(L_{k,\mathrm{trunc}})$ must hold for all adjacent digit pairs, meaning $k \in \{-1, 0, 1, 2\}$. Substituting $\rho = 2/3$ and simplifying yields:

- For $k = 2$: $\frac{1}{3} \cdot 2^{-\beta}\delta - \frac{4}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} \ge 0$.
- For $k = 1$: $\frac{1}{3} \cdot 2^{-\beta}\delta - \frac{1}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} \ge 0$.
- For $k = 0$: $\frac{1}{3} \cdot 2^{-\beta}\delta - \frac{1}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} \ge 0$.
- For $k = -1$: $\frac{1}{3} \cdot 2^{-\beta}\delta - \frac{4}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} \ge 0$.

The most restrictive conditions occur for $k = 2$ and $k = -1$. To satisfy these, a common choice of precision parameters is $\alpha = 5$ and $\beta = 4$:

$$\tau_j = \mathrm{Int}(32 \cdot t_{j,2\ldots0}.t_{j,-1\ldots-5}) \in \{-128, \ldots, 127\} \qquad (\tau_j \le 128R_j < \tau_j + 2), \tag{3.48}$$
$$\delta = \mathrm{Int}(16 \cdot 0.1d_{-2\ldots-4}) \in \{8, \ldots, 15\} \qquad (\delta \le 16D < \delta + 1). \tag{3.49}$$

The resulting selection logic is

$$q_{j+1} = \begin{cases} 2 & \text{if } \frac{8}{3}(\delta+1) \le \tau_j < \frac{16}{3}(\delta+1) \\ 1 & \text{if } \frac{2}{3}(\delta+1) \le \tau_j \le \frac{10}{3}\delta - 2 \\ 0 & \text{if } -\frac{4}{3}\delta \le \tau_j \le \frac{4}{3}\delta - 2 \\ -1 & \text{if } -\frac{10}{3}\delta \le \tau_j \le -\frac{2}{3}(\delta+1) - 2 \\ -2 & \text{if } -\frac{16}{3}(\delta+1) - 2 < \tau_j \le -\frac{8}{3}(\delta+1) - 2. \end{cases} \tag{3.50}$$

3.8.2 Negative Divisor

For a negative divisor $D \in \left[-1, -\frac{1}{2}\right)$, the interval endpoints reverse:

$$q_{j+1} = \begin{cases} 2 & \text{if } \frac{8}{3}D \le 4R_j \le \frac{4}{3}D \\ 1 & \text{if } \frac{5}{3}D \le 4R_j \le \frac{1}{3}D \\ 0 & \text{if } \frac{2}{3}D \le 4R_j \le -\frac{2}{3}D \\ -1 & \text{if } -\frac{1}{3}D \le 4R_j \le -\frac{5}{3}D \\ -2 & \text{if } -\frac{4}{3}D \le 4R_j \le -\frac{8}{3}D. \end{cases} \tag{3.51}$$

The continuity analysis proceeds analogously using $\min(U_{k,\mathrm{trunc}}) \ge \max(L_{k-1,\mathrm{trunc}})$. The same precision parameters $(\alpha = 5, \beta = 4)$ are sufficient to guarantee overlap:

$$\tau_j = \mathrm{Int}(32 \cdot t_{j,2\ldots0}.t_{j,-1\ldots-5}) \in \{-128, \ldots, 127\} \qquad (\tau_j \le 128R_j < \tau_j + 2), \tag{3.52}$$
$$\delta = \mathrm{Int}(16 \cdot 1.0d_{-2\ldots-4}) \in \{-16, \ldots, -9\} \qquad (\delta \le 16D < \delta + 1). \tag{3.53}$$

The selection logic is

$$q_{j+1} = \begin{cases} 2 & \text{if } \frac{16}{3}\delta - 2 < \tau_j \le \frac{8}{3}\delta - 2 \\ 1 & \text{if } \frac{10}{3}(\delta+1) \le \tau_j \le \frac{2}{3}\delta - 2 \\ 0 & \text{if } \frac{4}{3}(\delta+1) \le \tau_j \le -\frac{4}{3}(\delta+1) - 2 \\ -1 & \text{if } -\frac{2}{3}\delta \le \tau_j \le -\frac{10}{3}(\delta+1) - 2 \\ -2 & \text{if } -\frac{8}{3}\delta \le \tau_j \le -\frac{16}{3}\delta. \end{cases} \tag{3.54}$$

4 Square Root by Digit Recurrence

Having established the methodology for division, we now apply the same digit recurrence principles to square root extraction. The fundamental relationship for this operation is

$$X = S_n^2 + r, \tag{4.1}$$

where X is the radicand, S_n the final root, and r the final (unscaled) remainder.
We normalize

$$X \in \left[\frac{1}{4}, 1\right) \qquad \text{(Radicand)}$$
$$S_n \in \left[\sqrt{\frac{1}{4}}, \sqrt{1}\right) = \left[\frac{1}{2}, 1\right) \qquad \text{(Square Root).}$$

4.1 Root digits and approximation

The root is generated over n iterations with $j = 0, 1, \ldots, n-1$. Let S_j be the approximation after j digits $(s_1, \ldots, s_j)$.

$$S_{j+1} = S_j + s_{j+1} B^{-(j+1)}. \tag{4.2}$$

Unfolding,

$$\text{At step } j, \quad S_{j+1} = S_0 + \sum_{i=0}^{j} s_{i+1} B^{-(i+1)}. \tag{4.3}$$

After n iterations,

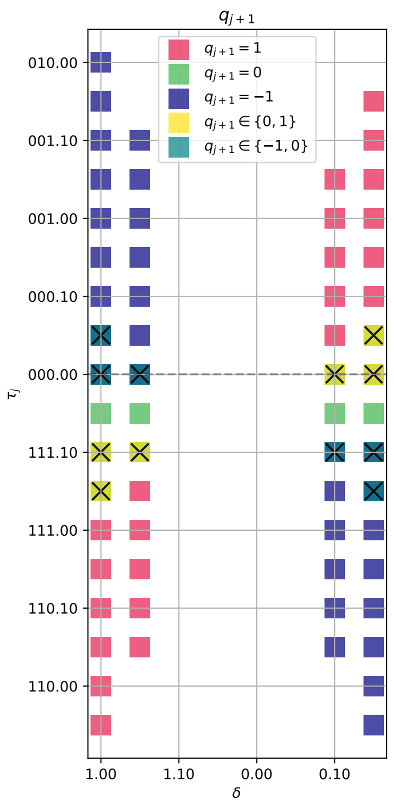
$$\text{At step } j = n-1, \quad S_n = S_0 + \sum_{i=0}^{n-1} s_{i+1} B^{-(i+1)}. \tag{4.4}$$

4.2 Error bound and containment

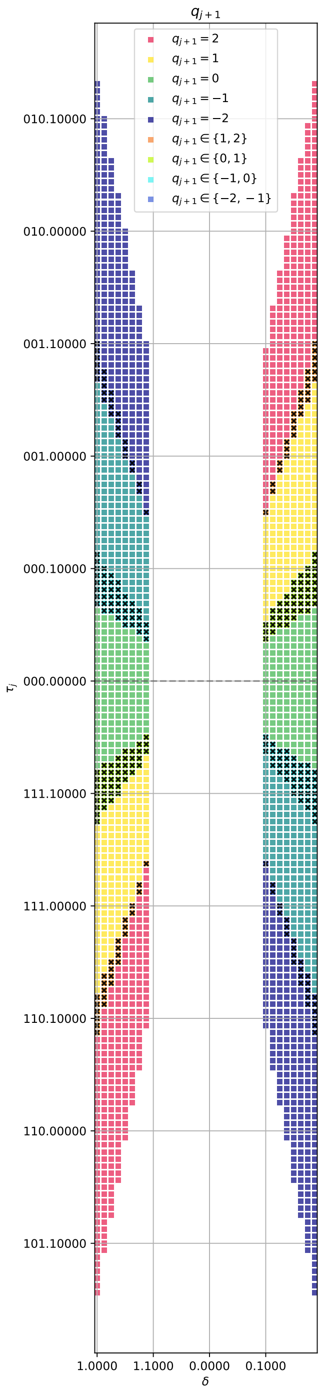
Define the error $\varepsilon_{j+1} = \sqrt{X} - S_{j+1}$. As in division,

$$\varepsilon_{j+1} = \sum_{i=j+1}^{\infty} s_{i+1} B^{-(i+1)}, \tag{4.5}$$

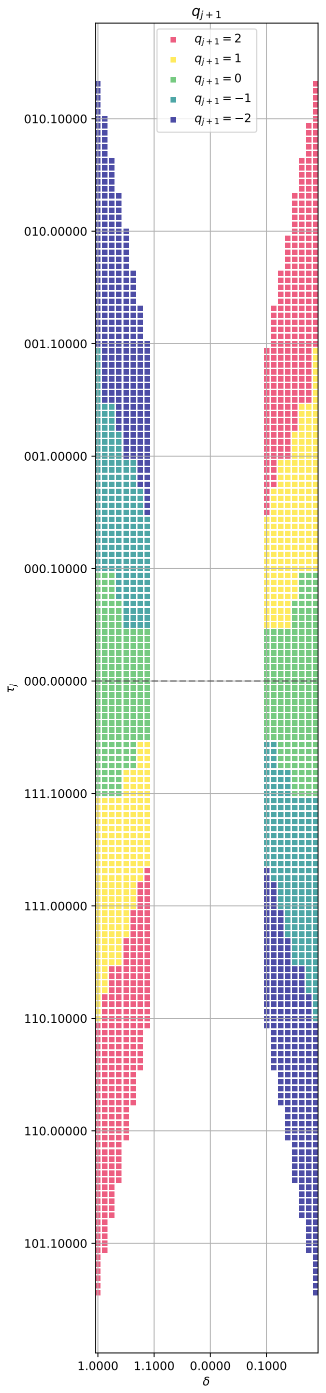
and with $|s_{i+1}| \le a$,



(a) Radix-2 Basic QDS



(b) Radix-4 Basic QDS



(c) Radix-4 Optimized QDS

Figure 1: Selection regions for the Quotient Digit Selector (QDS). The y-axis is the truncated residual estimate τ_j and the x-axis is the truncated divisor estimate δ. The plots show how the choice of quotient digit q_{j+1} depends on these estimates.

$$\left|\sum_{i=j+1}^{\infty} s_{i+1}B^{-(i+1)}\right| \leq \sum_{i=j+1}^{\infty} aB^{-(i+1)} = aB^{-(j+2)}\sum_{k=0}^{\infty} B^{-k} = aB^{-(j+2)}\frac{1}{1-B^{-1}} = \frac{a}{B-1}B^{-(j+1)}. \tag{4.6}$$

Thus

$$|\varepsilon_{j+1}| \leq \rho B^{-(j+1)} \implies \left|\sqrt{X} - S_{j+1}\right| \leq \rho B^{-(j+1)}. \tag{4.7}$$

Rearranging this inequality to bound $\sqrt{X}$ gives:

$$S_{j+1} - \rho B^{-(j+1)} \leq \sqrt{X} \leq S_{j+1} + \rho B^{-(j+1)}. \tag{4.8}$$

Squaring all terms then yields:

$$S_{j+1}^2 - 2S_{j+1}\rho B^{-(j+1)} + \rho^2 B^{-2(j+1)} \leq X \leq S_{j+1}^2 + 2S_{j+1}\rho B^{-(j+1)} + \rho^2 B^{-2(j+1)}. \tag{4.9}$$

Next, we isolate the term $X - S_{j+1}^2$:

$$\rho^2 B^{-2(j+1)} - 2S_{j+1}\rho B^{-(j+1)} \leq X - S_{j+1}^2 \leq \rho^2 B^{-2(j+1)} + 2S_{j+1}\rho B^{-(j+1)}. \tag{4.10}$$

Define the scaled residual

$$R_{j+1} = B^{j+1}(X - S_{j+1}^2).$$

Multiplying eq. (4.10) by B^{j+1} yields

$$\rho^2 B^{-(j+1)} - 2\rho S_{j+1} \leq B^{j+1}(X - S_{j+1}^2) \leq \rho^2 B^{-(j+1)} + 2\rho S_{j+1}, \tag{4.11}$$

i.e.,

$$\rho^2 B^{-(j+1)} - 2\rho S_{j+1} \leq R_{j+1} \leq \rho^2 B^{-(j+1)} + 2\rho S_{j+1}. \tag{4.12}$$

At the end,

$$\text{At step } j = n-1, \quad R_n = B^n(X - S_n^2) = B^n r. \tag{4.13}$$

4.3 Residual recurrence

From the definition of the scaled residual,

$$\begin{aligned}
R_{j+1} &= B^{j+1}(X - S_{j+1}^2) \\
&= B^{j+1}\left(X - \left(S_j + s_{j+1}B^{-(j+1)}\right)^2\right) \\
&= B^{j+1}\left(X - \left(S_j^2 + 2S_j s_{j+1}B^{-(j+1)} + s_{j+1}^2 B^{-2(j+1)}\right)\right) \\
&= B^{j+1}(X - S_j^2) - B^{j+1}\left(2S_j s_{j+1}B^{-(j+1)} + s_{j+1}^2 B^{-2(j+1)}\right) \\
&= B\left(B^j(X - S_j^2)\right) - \left(2S_j s_{j+1} + s_{j+1}^2 B^{-(j+1)}\right) \\
&= BR_j - s_{j+1}\left(2S_j + s_{j+1}B^{-(j+1)}\right).
\end{aligned}$$

Thus

$$R_{j+1} = BR_j - s_{j+1}\left(2S_j + s_{j+1}B^{-(j+1)}\right), \qquad j = 1, 2, \ldots, n-1. \tag{4.14}$$

4.4 Selection intervals

To find the valid selection intervals for the next root digit s_{j+1}, we substitute the residual recurrence from eq. (4.14) into the containment condition in eq. (4.12):

$$\rho^2 B^{-(j+1)} - 2\rho S_{j+1} \le BR_j - s_{j+1}\left(2S_j + s_{j+1}B^{-(j+1)}\right) \le \rho^2 B^{-(j+1)} + 2\rho S_{j+1}. \tag{4.15}$$

Using $S_{j+1} = S_j + s_{j+1}B^{-(j+1)}$:

$$\rho^2 B^{-(j+1)} - 2\rho\left(S_j + s_{j+1}B^{-(j+1)}\right) \le BR_j - s_{j+1}\left(2S_j + s_{j+1}B^{-(j+1)}\right) \le \rho^2 B^{-(j+1)} + 2\rho\left(S_j + s_{j+1}B^{-(j+1)}\right). \tag{4.16}$$

Rearranging yields the selection interval for BR_j:

$$2(s_{j+1} - \rho)S_j + (s_{j+1} - \rho)^2 B^{-(j+1)} \le BR_j \le 2(s_{j+1} + \rho)S_j + (s_{j+1} + \rho)^2 B^{-(j+1)}. \tag{4.17}$$

Unlike in division, these bounds depend on the partial root S_j, which changes with each iteration. This dynamic dependence is the primary reason square-root digit selection is more involved.

4.5 Initialization

To initialize the square root recurrence, we must define the first residual R_1 and the first root digit s_1. The recurrence for $j \ge 1$ is:

$$R_{j+1} = BR_j - s_{j+1}\left(2S_j + s_{j+1}B^{-(j+1)}\right), \qquad j = 1, 2, \ldots, n-1 \tag{4.18}$$

with the initial residual given by $R_1 = B(X - S_1^2)$. From the error bound $|\varepsilon_{j+1}| \le \rho B^{-(j+1)}$, we have for $j = 0, 1, \ldots, n-1$:

$$\left|\sqrt{X} - S_{j+1}\right| \le \rho B^{-(j+1)}. \tag{4.19}$$

For $j = 0$, this implies:

$$S_1 - \rho B^{-1} \le \sqrt{X} \le S_1 + \rho B^{-1}. \tag{4.20}$$

Given the normalized radicand range $X \in \left[\frac{1}{4}, 1\right)$, we define the first root approximation as $S_1 = S_0 + s_1 B^{-1}$. By setting $S_0 = 1$, the first digit s_1 is chosen from the non-positive half of the digit set, $s_1 \in \{0, -1, \ldots, -a\}$. Recall that $\rho = \frac{a}{B-1}$ and $\frac{1}{2} < \rho \le 1$. To ensure the root range is fully covered, the minimum and maximum possible values of S_1 must satisfy:

$$\min(S_1) - \rho B^{-1} \le \frac{1}{2} \implies \left(S_0 - aB^{-1}\right) - \rho B^{-1} \le \frac{1}{2} \implies S_0 - \rho \le \frac{1}{2}, \tag{4.21}$$

$$\max(S_1) + \rho B^{-1} \ge 1 \implies S_0 + \rho B^{-1} \ge 1. \tag{4.22}$$

With $S_0 = 1$, these conditions are satisfied.

For the Radix-2 case ($B = 2$, $\rho = 1$):

- **Select $s_1 = 0$:**

$$1 - \frac{1}{2} \le \sqrt{X} \le 1 + \frac{1}{2} \implies \frac{1}{2} \le \sqrt{X} \le \frac{3}{2} \implies \frac{1}{4} \le X \le \frac{9}{4}.$$

- **Select $s_1 = -1$:**

$$\frac{1}{2} - \frac{1}{2} \le \sqrt{X} \le \frac{1}{2} + \frac{1}{2} \implies 0 \le \sqrt{X} \le 1 \implies 0 \le X \le 1.$$

For the Radix-4 case ($B = 4$, $\rho = 2/3$):

- **Select $s_1 = 0$:**

$$1 - \frac{1}{6} \le \sqrt{X} \le 1 + \frac{1}{6} \implies \frac{5}{6} \le \sqrt{X} \le \frac{7}{6} \implies \frac{25}{36} \le X \le \frac{49}{36}.$$

- **Select $s_1 = -1$:**

$$\frac{3}{4} - \frac{1}{6} \le \sqrt{X} \le \frac{3}{4} + \frac{1}{6} \implies \frac{7}{12} \le \sqrt{X} \le \frac{11}{12} \implies \frac{49}{144} \le X \le \frac{121}{144}.$$

- **Select $s_1 = -2$:**

$$\frac{1}{2} - \frac{1}{6} \le \sqrt{X} \le \frac{1}{2} + \frac{1}{6} \implies \frac{1}{3} \le \sqrt{X} \le \frac{2}{3} \implies \frac{1}{9} \le X \le \frac{4}{9}.$$

4.6 Example: Radix-2 SRT Square Root

For a Radix-2 implementation, we use $B = 2$ and $a = 1$, which gives $\rho = 1$ and a digit set of $s_{j+1} \in \{-1, 0, 1\}$. Substituting these parameters into eq. (4.17) yields the selection intervals:

$$s_{j+1} = \begin{cases} 1 & \text{if } 0 \le 2R_j \le 4S_j + 4 \cdot 2^{-(j+1)} \\ 0 & \text{if } -2S_j + 2^{-(j+1)} \le 2R_j \le 2S_j + 2^{-(j+1)} \\ -1 & \text{if } -4S_j + 4 \cdot 2^{-(j+1)} \le 2R_j \le 0. \end{cases} \tag{4.23}$$

Because selection depends on both $2R_j$ and S_j, we must derive truncated estimates for both quantities. The residual estimate t_j has a truncation error of $0 \le 2R_j - t_j < 2^{-(\alpha-1)}$. We define the integer estimates as:

$$\tau_j = \text{Int}(2^\alpha \cdot t_{j,3\ldots0}.t_{j,-1\ldots-\alpha}) \in \left\{-2^{\alpha+3}, \ldots, 2^{\alpha+3} - 1\right\} \quad (\tau_j \le 2^{\alpha+1} R_j < \tau_j + 2), \tag{4.24}$$

$$\sigma_j = \text{Int}(2^\beta \cdot 0.1u_{j,-2\ldots-\beta}) \in \left\{2^{\beta-1}, \ldots, 2^\beta - 1\right\} \quad (\sigma_j \le 2^\beta S_j \le \sigma_j + 1 - 2^\beta \text{ulp}). \tag{4.25}$$

Here, we write the binary expansion of $S_j \in [1/2, 1)$ as $S_j = 0.1u_{j,-2}u_{j,-3}\ldots$ (where the u variables represent bits of S_j, not SRT digits). Additionally, ulp $= 2^{-j}$, meaning $\sigma_j \le 2^\beta S_j \le \sigma_j + 1 - 2^\beta\text{ulp}$.

The continuity analysis uses the condition $\min(U_{k-1,\text{trunc}}) \ge \max(L_{k,\text{trunc}})$. Substituting $\rho = 1$ and $B = 2$ yields:

- For $k = 1$: $\min(U_{0,\text{trunc}}) \ge \max(L_{1,\text{trunc}})$ gives $2 \cdot 2^{-\beta}\sigma_j - 2^{-(\alpha-1)} \ge 0$. This holds for $\alpha \ge 1$ since $\sigma_j \ge 2^{\beta-1}$.

- For $k = 0$: $\min(U_{-1,\text{trunc}}) \geq \max(L_{0,\text{trunc}})$ gives $2 \cdot 2^{-\beta}\sigma_j - 2^{-(\alpha-1)} - B^{-(j+1)} \geq 0$.

Evaluating this for the worst-case partial root ($\sigma_j = 2^{\beta-1}$):

$$1 - 2^{-(\alpha-1)} - 2^{-(j+1)} \geq 0 \;\Rightarrow\; \alpha \geq 2.$$

Choosing $\alpha = 2$ and $\beta = 2$ satisfies this requirement:

$$\tau_j = \text{Int}(4 \cdot t_{j,3\ldots0}.t_{j,-1}t_{j,-2}) \in \{-32, \ldots, 31\} \qquad (\tau_j \leq 8R_j < \tau_j + 2), \tag{4.26}$$

$$\sigma_j = \text{Int}(4 \cdot 0.1u_{j,-2}) \in \{2, 3\} \qquad (\sigma_j \leq 4S_j \leq \sigma_j + 1 - 4 \cdot 2^{-j}). \tag{4.27}$$

This results in the following selection logic:

$$s_{j+1} = \begin{cases} 1 & \text{if } 0 \leq \tau_j < 4(\sigma_j + 1) \\ 0 & \text{if } -2(\sigma_j - 1) \leq \tau_j \leq 2\sigma_j - 2 \\ -1 & \text{if } -4(\sigma_j + 1) - 2 < \tau_j \leq -2. \end{cases} \tag{4.28}$$

4.6.1 Special Case: Radicand Close to 1

A special condition can arise during early iterations if the first selected digits are zero. If $s_1, \ldots, s_j = 0$, then $S_j = S_0 = 1$. Consequently, the root estimate remains constant, and its truncated estimate σ_j becomes constant as well. This scenario is particularly relevant when the radicand X is close to 1.

For Radix-2 with $\alpha = 2$ and $\beta = 2$, τ_j estimates the scaled residual $2R_j$. Since $R_1 = 2(X - 1) < 0$ and $R_k = BR_{k-1}$ for $k = 2, 3, \ldots, n$ when $s_k = 0$, the residual remains negative as long as only zeros are selected; thus, $\tau_j < 0$. With $S_j = 1$, the estimates become:

$$\tau_j = \text{Int}(4 \cdot 1t_{j,2\ldots0}.t_{j,-1}t_{j,-2}) \in \{-32, \ldots, -1\} \qquad (\tau_j \leq 8R_j < \tau_j + 2), \tag{4.29}$$

$$\sigma_j = \text{Int}(4 \cdot 1.00) = 4 \qquad (4S_j = \sigma_j). \tag{4.30}$$

Under these conditions, the selection logic reduces to:

$$s_{j+1} = \begin{cases} 0 & \text{if } -2(\sigma_j - 1) \leq \tau_j \leq -1 \\ -1 & \text{if } -4\sigma_j - 2 < \tau_j \leq -2. \end{cases} \tag{4.31}$$

4.7 Example: Radix-4 SRT Square Root

For a Radix-4 implementation, we use $B = 4$ and $a = 2$, yielding $\rho = 2/3$ and a digit set of $s_{j+1} \in \{-2, -1, 0, 1, 2\}$. Substituting these into eq. (4.17) provides the selection intervals:

$$s_{j+1} = \begin{cases} 2 & \text{if } \frac{8}{3}S_j + \frac{16}{9}4^{-(j+1)} \leq 4R_j \leq \frac{16}{3}S_j + \frac{64}{9}4^{-(j+1)} \\ 1 & \text{if } \frac{2}{3}S_j + \frac{1}{9}4^{-(j+1)} \leq 4R_j \leq \frac{10}{3}S_j + \frac{25}{9}4^{-(j+1)} \\ 0 & \text{if } -\frac{4}{3}S_j + \frac{4}{9}4^{-(j+1)} \leq 4R_j \leq \frac{4}{3}S_j + \frac{4}{9}4^{-(j+1)} \\ -1 & \text{if } -\frac{10}{3}S_j + \frac{25}{9}4^{-(j+1)} \leq 4R_j \leq -\frac{2}{3}S_j + \frac{1}{9}4^{-(j+1)} \\ -2 & \text{if } -\frac{16}{3}S_j + \frac{64}{9}4^{-(j+1)} \leq 4R_j \leq -\frac{8}{3}S_j + \frac{16}{9}4^{-(j+1)}. \end{cases} \tag{4.32}$$

We define the truncated integer estimates as:

$$\tau_j = \mathrm{Int}(2^\alpha \cdot t_{j,3\ldots0}.t_{j,-1\ldots-\alpha}) \in \left\{-2^{\alpha+3}, \ldots, 2^{\alpha+3}-1\right\} \quad (\tau_j \le 2^{\alpha+2}R_j < \tau_j + 2), \tag{4.33}$$
$$\sigma_j = \mathrm{Int}(2^\beta \cdot 0.1u_{j,-2\ldots-\beta}) \in \left\{2^{\beta-1}, \ldots, 2^\beta - 1\right\} \quad (\sigma_j \le 2^\beta S_j \le \sigma_j + 1 - 2^\beta 4^{-j}). \tag{4.34}$$

The continuity condition must hold for all adjacent digit pairs ($k \in \{-1, 0, 1, 2\}$). Substituting $\rho = 2/3$ gives:

- For $k = 2$: $\frac{2}{3} \cdot 2^{-\beta}\sigma_j - \frac{8}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} \ge 0$.
- For $k = 1$: $\frac{2}{3} \cdot 2^{-\beta}\sigma_j - \frac{2}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} \ge 0$.
- For $k = 0$: $\frac{2}{3} \cdot 2^{-\beta}\sigma_j - \frac{2}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} - \frac{4}{9}B^{-(j+1)} \ge 0$.
- For $k = -1$: $\frac{2}{3} \cdot 2^{-\beta}\sigma_j - \frac{8}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} - \frac{25}{9}B^{-(j+1)} \ge 0$.

A key complication here—one not present in division—is the explicit $B^{-(j+1)}$ dependence in the bounds. This makes the overlap iteration-dependent, being most restrictive for small j and for negative digits with larger squared terms. The tightest boundary occurs between selecting $k = -1$ and $k = -2$:

$$\frac{2}{3} \cdot 2^{-\beta}\sigma_j - \frac{8}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} - \frac{25}{9}B^{-(j+1)} \ge 0.$$

For example, using $\alpha = 4$, $\beta = 4$, $B = 4$, and the worst-case partial root $\sigma_j = 8$, the constant part evaluates to $1/24$, and the condition becomes:

$$\frac{1}{24} - \frac{25}{9}4^{-(j+1)} \ge 0,$$

which holds only for $j \ge 3$. The failure for $j = 1$ and $j = 2$ indicates the need for a special startup mechanism (or increased precision) during the earliest iterations.

For $j \ge 3$, using $\alpha = 4$ and $\beta = 4$ yields the estimates:

$$\tau_j = \mathrm{Int}(16 \cdot t_{j,3\ldots0}.t_{j,-1\ldots-4}) \in \{-128, \ldots, 127\} \quad (\tau_j \le 64R_j < \tau_j + 2), \tag{4.35}$$
$$\sigma_j = \mathrm{Int}(16 \cdot 0.1u_{j,-2\ldots-4}) \in \{8, \ldots, 15\} \quad (\sigma_j \le 16S_j \le \sigma_j + 1 - 16 \cdot 4^{-j}). \tag{4.36}$$

This leads to the following selection logic for $j \ge 3$:

$$s_{j+1} = \begin{cases} 2 & \text{if } \frac{8}{3}(\sigma_j + 1) \le \tau_j < \frac{16}{3}(\sigma_j + 1) \\ 1 & \text{if } \frac{2}{3}(\sigma_j + 1) \le \tau_j \le \frac{10}{3}\sigma_j - 2 \\ 0 & \text{if } -\frac{4}{3}\left(\sigma_j - \frac{1}{48}\right) \le \tau_j \le \frac{4}{3}\sigma_j - 2 \\ -1 & \text{if } -\frac{10}{3}\left(\sigma_j - \frac{5}{96}\right) \le \tau_j \le -\frac{2}{3}(\sigma_j + 1) - 2 \\ -2 & \text{if } -\frac{16}{3}(\sigma_j + 1) - 2 < \tau_j \le -\frac{8}{3}(\sigma_j + 1) - 2. \end{cases} \tag{4.37}$$

4.7.1 Startup Issues and Iteration-Dependent Selection

As observed in section 4.7, under the absolute worst-case assumption for the partial root ($\sigma_j = 8$), the continuity condition fails for iterations $j = 1$ and $j = 2$. This explicit dependence on $B^{-(j+1)}$ is a hallmark of square root digit recurrence, making the valid selection intervals narrowest during the earliest iterations.

However, the worst-case σ_j does not occur in all execution paths. The actual value of S_j depends on the digits selected so far. If S_j is larger, the overlap between adjacent selection regions widens, which can compensate for the large $B^{-(j+1)}$ term even during early iterations.

To illustrate this, consider the special case where the radicand is close to 1, leading to the selection of $s_1, \dots, s_j = 0$. In this path, $S_j = S_0 = 1$, and thus $\sigma_j = 16$. Let us re-evaluate the most restrictive continuity condition (between selecting -1 and -2) for this specific scenario:

$$\frac{2}{3} \cdot 2^{-\beta}\sigma_j - \frac{8}{3} \cdot 2^{-\beta} - 2^{-(\alpha-1)} - \frac{25}{9}B^{-(j+1)} \geq 0.$$

Substituting $B = 4$, $\alpha = 4$, $\beta = 4$, and $\sigma_j = 16$ gives:

$$\begin{aligned} \frac{2}{3} \cdot 2^{-4}(16) - \frac{8}{3} \cdot 2^{-4} - 2^{-(4-1)} - \frac{25}{9}4^{-(j+1)} &\geq 0 \\ \frac{2}{3} - \frac{1}{6} - \frac{1}{8} - \frac{25}{9}4^{-(j+1)} &\geq 0 \\ \frac{3}{8} - \frac{25}{9}4^{-(j+1)} &\geq 0. \end{aligned}$$

Testing this inequality for the earliest iterations yields:

- **For $j = 0$:** $\frac{3}{8} - \frac{25}{36} = -\frac{23}{72} < 0$. (Fails, but $j = 0$ is handled by the initialization step).
- **For $j = 1$:** $\frac{3}{8} - \frac{25}{144} = \frac{29}{144} > 0$. (Passes).

Because this condition holds for $j \geq 1$, the standard precision ($\alpha = 4$, $\beta = 4$) is sufficient for this specific path, starting from the very first standard iteration.

For this path ($s_1, \dots, s_j = 0 \implies S_j = 1$), the residual remains negative, meaning $\tau_j < 0$. The truncated estimates then become:

$$\tau_j = \mathrm{Int}(16 \cdot 1t_{j,2\ldots0}.t_{j,-1\ldots-4}) \in \{-128, \dots, -1\} \qquad (\tau_j \leq 64R_j < \tau_j + 2), \tag{4.38}$$

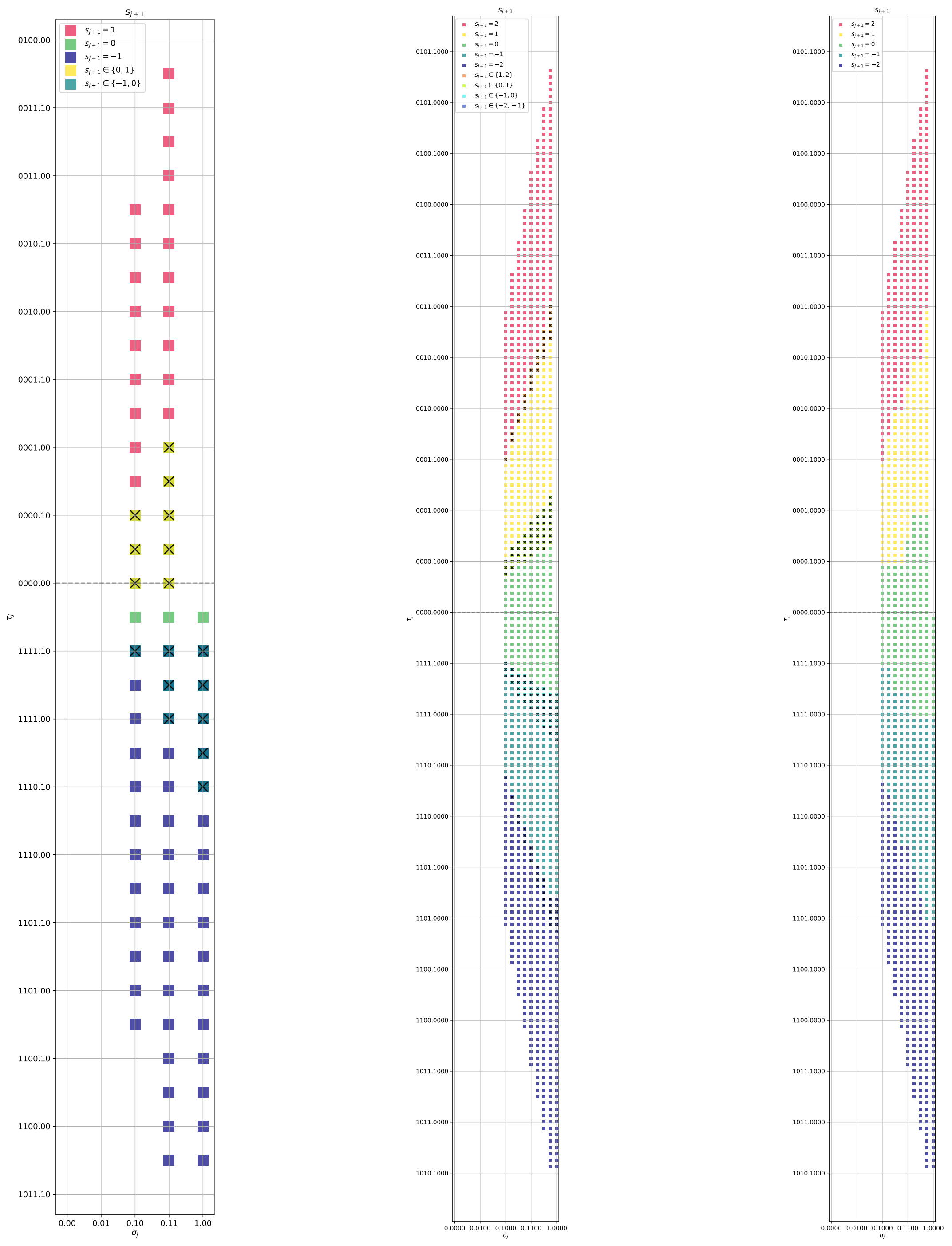
$$\sigma_j = \mathrm{Int}(16 \cdot 1.0000) = 16 \qquad (16S_j = \sigma_j). \tag{4.39}$$

Substituting $\sigma_j = 16$ into the general Radix-4 bounds yields a simplified, specialized selection logic for $j \geq 1$:

$$s_{j+1} = \begin{cases} 0 & \text{if } -\frac{4}{3}\left(\sigma_j - \frac{1}{3}\right) \leq \tau_j \leq -1 \\ -1 & \text{if } -\frac{10}{3}\left(\sigma_j - \frac{5}{6}\right) \leq \tau_j \leq -\frac{2}{3}\sigma_j - 2 \\ -2 & \text{if } -\frac{16}{3}\sigma_j - 2 < \tau_j \leq -\frac{8}{3}\sigma_j - 2. \end{cases} \tag{4.40}$$

5 Conclusion

In summary, digit recurrence algorithms provide an efficient framework for implementing division and square root in hardware. They address a central tension in iterative arithmetic: exact comparisons and full-precision state updates can limit speed, but overly aggressive approximation can break correctness. The core concept that resolves this tension is *redundancy*. By selecting digits from a redundant set (with $\rho > \frac{1}{2}$), SRT-style methods create overlapping selection regions. This overlap permits the next digit to be chosen using fast, truncated estimates of the residual (and, for square root, the partial root), avoiding a full-width carry-propagating addition in the critical path.



(a) Radix-2 Basic RDS

(b) Radix-4 Basic RDS

(c) Radix-4 Optimized RDS

Figure 2: Selection regions for the Root Digit Selector (RDS). The y-axis is the truncated residual estimate τ_j and the x-axis is the truncated partial root estimate σ_j. The plots show how the choice of root digit s_{j+1} depends on these estimates.

Table 1: Example selection logic constants for a Radix-4 implementation ($j \geq 1$). The table shows the required integer range for the truncated residual estimate τ_j to select a given digit, based on the truncated divisor/root estimate (δ or σ_j).

τ_j	$q_{j+1}, s_{j+1} = 2$	$q_{j+1}, s_{j+1} = 1$	$q_{j+1}, s_{j+1} = 0$	$q_{j+1}, s_{j+1} = -1$	$q_{j+1}, s_{j+1} = -2$
$\delta, \sigma_j = 8$	24 to 49	8 to 23	-8 to 7	-26 to -9	-49 to -27
$\delta, \sigma_j = 9$	28 to 53	8 to 27	-8 to 7	-28 to -9	-55 to -29
$\delta, \sigma_j = 10$	32 to 58	8 to 31	-12 to 7	-32 to -13	-60 to -33
$\delta, \sigma_j = 11$	32 to 64	8 to 31	-12 to 7	-34 to -13	-65 to -35
$\delta, \sigma_j = 12$	36 to 71	12 to 35	-12 to 11	-36 to -13	-71 to -37
$\delta, \sigma_j = 13$	40 to 74	16 to 39	-16 to 15	-40 to -17	-76 to -41
$\delta, \sigma_j = 14$	40 to 80	16 to 39	-16 to 15	-44 to -17	-81 to -45
$\delta, \sigma_j = 15$	48 to 85	16 to 47	-16 to 15	-48 to -17	-87 to -49

Table 2: Example selection logic constants for a Radix-4 implementation ($j \geq 2$). The table shows the required integer range for the truncated residual estimate τ_j to select a given digit, based on the truncated divisor/root estimate (δ or σ_j).

τ_j	$q_{j+1}, s_{j+1} = 2$	$q_{j+1}, s_{j+1} = 1$	$q_{j+1}, s_{j+1} = 0$	$q_{j+1}, s_{j+1} = -1$	$q_{j+1}, s_{j+1} = -2$
$\delta, \sigma_j = 8$	24 to 48	8 to 23	-8 to 7	-26 to -9	-49 to -27
$\delta, \sigma_j = 9$	28 to 53	8 to 27	-8 to 7	-28 to -9	-55 to -29
$\delta, \sigma_j = 10$	32 to 58	8 to 31	-12 to 7	-32 to -13	-60 to -33
$\delta, \sigma_j = 11$	32 to 64	8 to 31	-12 to 7	-34 to -13	-65 to -35
$\delta, \sigma_j = 12$	36 to 69	12 to 35	-12 to 11	-36 to -13	-71 to -37
$\delta, \sigma_j = 13$	40 to 74	16 to 39	-16 to 15	-40 to -17	-76 to -41
$\delta, \sigma_j = 14$	40 to 80	16 to 39	-16 to 15	-44 to -17	-81 to -45
$\delta, \sigma_j = 15$	48 to 85	16 to 47	-16 to 15	-48 to -17	-87 to -49

This tutorial derived the fundamental recurrence relations for division and square root, established the containment conditions that guarantee convergence, and showed how those conditions drive practical digit-selection design. The continuity condition provides the key bridge from theory to implementation: it determines how much operand truncation can be tolerated while still guaranteeing that overlap covers uncertainty. Furthermore, the Radix-2 and Radix-4 case studies illustrated a complete design workflow, from interval derivation through parameter selection to final selection logic.

Square root follows the same broad methodology as division but introduces additional complexity because its bounds depend on the evolving partial root S_j and include iteration-dependent terms. As the Radix-4 example showed, these terms can shrink overlap in early iterations, requiring either greater precision or special startup logic. This dynamic highlights the trade-offs among radix (performance), selection complexity, and implementation cost.

Ultimately, digit recurrence methods exemplify a practical design paradigm in computer arithmetic: exploiting algorithmic structure (redundancy and overlap) to simplify and accelerate hardware. They are not merely of theoretical interest, but remain a cornerstone of high-performance arithmetic units in modern processors.