# Fractions in the Canonical-Signed-Digit Number System

*Jeffrey O. Coleman*
Naval Research Laboratory
Washington DC, USA
*jeffc@alum.mit.edu*

*Arda Yurdakul*
Kadir Has University
İstanbul, Türkiye
*a.yurdakul@ieee.org*

## ABSTRACT

**Canonical-signed-digit (CSD) coefficient representations lead to efficient add/subtract networks for hardwired DSP multiplies of two's complement signals. CSD has always been considered a fixed-point system, and available conversion algorithms operate on integers. Using methods applicable to many simple number systems, we rederive CSD for fractional numbers, derive a simple floating-point recursion for converting fractions to CSD, and briefly examine the associated truncation error.**

## 1 INTRODUCTION

In hardware DSP a signal can be scaled by a coefficient using hardwired shifts and adds, but arithmetic operations are fewer when subtractions are permitted also [1, 2]. The standard approach uses coefficients in CSD, a radix-two number system with ternary coefficient set $\{-1, 0, 1\}$, denoted in CSD digit strings in this paper by $\{\underline{1}, 0, 1\}$, and having the "canonical" property that $\underline{1}$ and $1$ are always followed by $0$ in CSD strings. CSD string $100.\underline{1}01$, for example, represents 3.625, the dot product of ternary coefficient vector $(1, 0, 0, -1, 0, 1)$ with powers $(2^2, 2^1, 2^0, 2^{-1}, 2^{-2}, 2^{-3})$. So a CSD coefficient specifies which input-signal shifts to add to the output, which to subtract from it, and which to ignore.

CSD is quite old, but we boldly create it anew, justified by the widespread assumption elsewhere that CSD is a shifted-integer system. CSD here is inherently fractional. We begin generally, in radix $r \in \mathbb{R}$ with $|r| > 1$ and a bounded alphabet $\mathcal{A} \subset \mathbb{R}$ containing zero. Then using $a_k \in \mathcal{A}$,

| sequence | represents |
|---|---|
| $0.a_1\, a_2 \ldots$ | $x = \sum_{k=1}^{\infty} a_k \; r^{-k}$ |
| $a_1.a_2\, a_3 \ldots$ | $rx = a_1 + \sum_{k=1}^{\infty} a_{k+1}\, r^{-k}.$ |

Let translated set $\alpha + \mathcal{F}_\alpha \subset \mathbb{R}$ contain all products $rx$ when $a_1 = \alpha \in \mathcal{A}$ and the sequence is finite. Then

$$r\mathcal{F}_0 = \bigcup_{\alpha\in\mathcal{A}} \alpha + \mathcal{F}_\alpha. \tag{1}$$

This work was supported by the AMRFC program (ONR 31) of the Office of Naval Research, Arlington VA, USA.

Certain properties are desired in sets $\mathcal{F}_\alpha$. First, make them disjoint to make finite sequence representations unique. Second, keep sequence memory finite by structuring the sequence following any $\alpha$ into an initial $\alpha$-dependent component of some length $N_\alpha$ and an $\alpha$-independent tail sequence: $\mathcal{F}_\alpha = \mathcal{I}_\alpha + r^{-N_\alpha}\mathcal{T}$. Third, require $\mathcal{I}_\alpha = 0$ so that every CSD digit of value $\alpha$ is followed immediately by $N_\alpha$ (highly desirable) zero digits. Of course $N_0 = 0$ is essential, else sequences will get "stuck at zero" forever, so $\mathcal{F}_0 = \mathcal{T}$. For $\alpha \neq 0$ then,

$$\mathcal{F}_\alpha = r^{-N_\alpha}\mathcal{F}_0 \tag{2}$$

Fourth and finally, there should be no gaps among numbers representable to arbitrary precision, so require (topological) closure $\overline{\mathcal{F}_0}$, just $\mathcal{F}_0$ with limit points appended, to be a (nontrivial) closed interval. This governs design of such number systems, because, with $\mu$ the (nonzero) width of interval $\overline{\mathcal{F}_0}$, disjoint union (1) and constraint (2) now imply $r\mu = \sum_{\alpha\in\mathcal{A}} r^{-N_\alpha}\mu$, or

$$\sum_{\alpha\in\mathcal{A}} r^{-(N_\alpha+1)} = 1. \tag{3}$$

The terms of this sum give the relative proportions of $\overline{\mathcal{F}_0}$ represented by sequences beginning with each $\alpha \in \mathcal{A}$.

## 2 CSD FOR FRACTIONS

Define $\mathcal{F} = \overline{\mathcal{F}_0}$ for brevity. In a radix-two system the $\alpha = 0$ term in (3) is $r^{-1} = \frac{1}{2}$. Each nonzero $\alpha$ of a ternary system is msd (most significant digit) to the same proportion of $\mathcal{F}$ if both such terms in (3) are set to $\frac{1}{4}$. If these $\alpha$ are set to $\pm 1$, then the closure of (1) after substituting (2) and dividing out radix $r = 2$ really defines CSD:

$$\mathcal{F} = \left(\frac{1}{4}\mathcal{F} - \frac{1}{2}\right) \cup \left(\frac{1}{2}\mathcal{F}\right) \cup \left(\frac{1}{4}\mathcal{F} + \frac{1}{2}\right) \tag{4}$$

Each number in $\mathcal{F}$ is decomposed into an msd value of $-\frac{1}{2}$, $0$, or $\frac{1}{2}$ and a remainder represented by the other digits. The union was disjoint before closure, so the three closed subintervals just touch and the only solution is $\mathcal{F} = \left[-\frac{2}{3}, \frac{2}{3}\right]$, shown in Fig. 1 with its recursive decomposition according to (4), the full meaning of which is now apparent.

Two representations are possible at endpoints common to two subintervals of interval $\mathcal{F}$, so we arbitrarily open it on (either end but) the right (is conventional) before deriving a conversion algorithm. Now (4) with $\mathcal{F} = \left[-\frac{2}{3}, \frac{2}{3}\right)$ implies

$$c = \begin{cases} \frac{1}{4}c_3 + \frac{1}{2} & \text{if } c \in \left[\frac{1}{3}, \frac{2}{3}\right) \\ \frac{1}{2}c_2 & \text{if } c \in \left[-\frac{1}{3}, \frac{1}{3}\right) \\ \frac{1}{4}c_1 - \frac{1}{2} & \text{if } c \in \left[-\frac{2}{3}, -\frac{1}{3}\right), \end{cases} \tag{5}$$

for some $c_i \in \mathcal{F}$, and so the associated CSD digit string is

$$D(c) = \begin{cases} \mathbf{1}\,\mathbf{0}\; D(c_3) & \text{if } c \in \left[\frac{1}{3}, \frac{2}{3}\right) \\ \mathbf{0}\; D(c_2) & \text{if } c \in \left[-\frac{1}{3}, \frac{1}{3}\right) \\ \underline{\mathbf{1}}\,\mathbf{0}\; D(c_1) & \text{if } c \in \left[-\frac{2}{3}, -\frac{1}{3}\right). \end{cases} \tag{6}$$

Substituting for the $c_i$ solved from (5) recursively relates any $c \in \left[-\frac{2}{3}, \frac{2}{3}\right)$ to its CSD representation :[1]

$$D(c) = \begin{cases} \mathbf{1}\,\mathbf{0}\; D(4(c-0.5)) & \text{if } c \in \left[\frac{1}{3}, \frac{2}{3}\right) \\ \mathbf{0}\; D(2c) & \text{if } c \in \left[-\frac{1}{3}, \frac{1}{3}\right) \\ \underline{\mathbf{1}}\,\mathbf{0}\; D(4(c+0.5)) & \text{if } c \in \left[-\frac{2}{3}, -\frac{1}{3}\right). \end{cases} \tag{7}$$

Recursion (7) and the "CSD ruler" in Fig. 1 made from it show that existing more-significant digits do not change as new less-significant digits are produced. Thus, transforming to an iterative form that stops after some desired number of digits yields a simple real-to-CSD conversion algorithm.

Some important CSD properties are now apparent. Zeros abound, as (7) never places nonzeros consecutively. The CSD ruler shows that conversion of a random variable distributed uniformly on $\mathcal{F}$ yields nonzero-digit probability $\frac{1}{2}$ for the msd and then $\frac{1}{2}, \frac{1}{4} \ldots$, oscillating about and converging to $\frac{1}{3}$. As the number $N$ of kept digits grows, the distribution of the normalized truncation error asymptotically approaches that of the Fig. 2 pdf. Truncation adds no bias, and the asymptotic standard error deviation is $\frac{1}{3}2^{-N}$, exceeding rounded binary's by 1.25 dB. But the range of representation is higher for CSD than two's complement by twice this, so CSD as presented has 1.25 dB more dynamic range (SNR).[2]

## 3 A FINAL APPLICATION NOTE

CSD should be used for coefficients but not signals, because of overflow. A large input in a typical FIR-filter stopband, for example, results in large intermediate sums internal to the filter, with stopband cancellation only effective at the filter output. Using two's complement for signal paths renders this intermediate overflow harmless, as Fig. 3 shows. CSD is for coefficients only and only at design time, when it guides the design of add/subtract/shift systems for signals.

[1] Or for unambiguous storage efficiency (adapting Hashemian [3]), use trailing sign bits and replace **10**, **0**, and **10** here with binary 10, 0, and 11.

[2] Conventional integer-to-CSD conversion has 1.25 dB less error, but only at the cost of giving up the stability of existing digits as more digits are converted, a property useful in common-subexpression optimization[4].

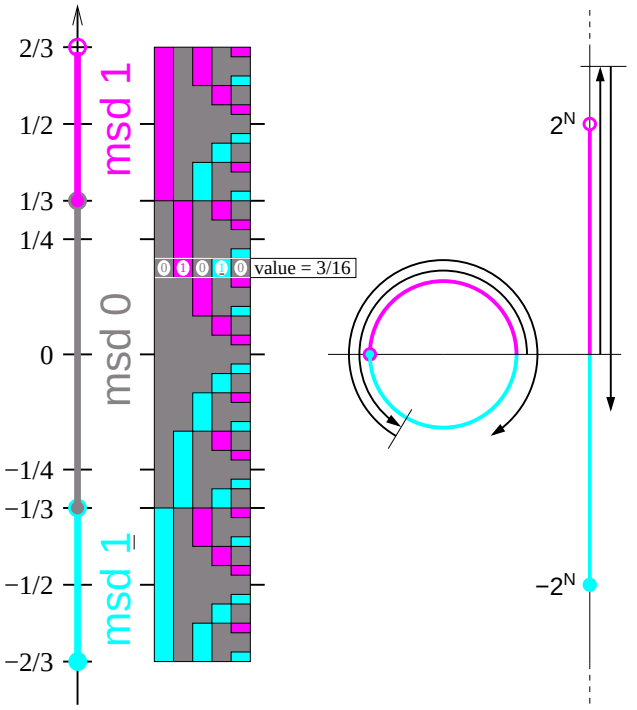


**Figure 1: Partitioning the CSD interval recursively using the msb makes a simple "CSD ruler."**

**Figure 3: Two's complement arithmetic is modular and so renders intermediate overflows harmless.**

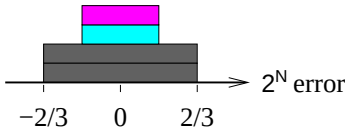


**Figure 2: The asymptotic (in the number of digits kept) error pdf for CSD conversion of a uniform random variable.**

## REFERENCES


[1] A. Avizienis, "Signed-digit number representation for fast parallel arithmetic," *IRE Trans. Electron. Comp.*, vol. EC-10, pp. 389–400, 1961.

[2] H. L. Garner, "Number systems and arithmetic," *Advanced Computers*, vol. 6, pp. 131–194, 1965.

[3] R. Hashemian, "A new method for conversion of a 2's complement to canonic signed digit number system and its representation," in *Proc. 1997 Asilomar Conf. on Signals, Systems and Computers*, pp. 904–907.

[4] A. Yurdakul and G. Dündar, "Multiplierless realization of linear DSP transforms by using common two-term expressions," *J. VLSI Signal Processing 22*, pp. 163–172, 1999.