

Redundant CORDIC Timmermann (C)

20170125

Termination Algorithms
Modified CORDIC
CSD (Canonic Sign Digit) Encoding
Booth Encoding

Copyright (c) 2015 - 2016 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Low Latency Time CORDIC Algorithms - Timmermann (1992)
Redundant and on-line CORDIC - Ercegovac & Lang (1990)

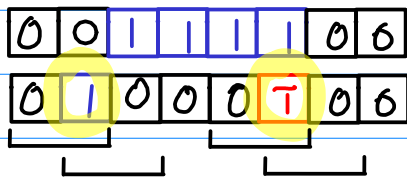
CSD (Canonic Signed Digit)

like Booth encoding (not modified Booth)

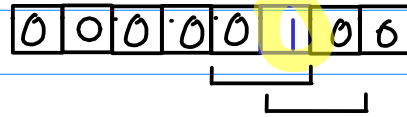
all non-zero digits are separated by zeros

$$\Rightarrow \sigma_i \sigma_{i+1} = 0$$

· 1-rum



$$\begin{array}{ll} 0 \cdot 1 = 0 & 0 \cdot \bar{1} = 0 \\ 1 \cdot 0 = 0 & \bar{1} \cdot 0 = 0 \end{array}$$



$$\begin{aligned}
 x_{i+1} &= x_i - m \sigma_i 2^{-s(m,i)} y_i \\
 y_{i+1} &= y_i + \sigma_i 2^{-s(m,i)} x_i \\
 z_{i+1} &= z_i - \sigma_i \alpha_{m,i}
 \end{aligned}$$

(i+1)

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & -m \sigma_i 2^{-i} \\ \sigma_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

(i+2)

$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 & -m \sigma_{i+1} 2^{-i-1} \\ \sigma_{i+1} 2^{-i-1} & 1 \end{bmatrix} \begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -m \sigma_{i+1} 2^{-i-1} \\ \sigma_{i+1} 2^{-i-1} & 1 \end{bmatrix} \begin{bmatrix} 1 & -m \sigma_i 2^{-i} \\ \sigma_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$= \begin{bmatrix} 1 - m \sigma_{i+1} \sigma_i 2^{-2i-1} & -m \sigma_i 2^{-i} - m \sigma_{i+1} 2^{-i-1} \\ \sigma_{i+1} 2^{-i-1} + \sigma_i 2^{-i} & -m \sigma_{i+1} \sigma_i 2^{-2i-1} + 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$= \begin{bmatrix} 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} & -m (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\begin{bmatrix} 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} & -m (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} \end{bmatrix}$$

$$\sigma_i \neq 0 \rightarrow \sigma_i \sigma_{i+1} = 0$$

property of Booth encoding

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & -m \sigma_i 2^{-i} \\ \sigma_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} & -m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$



$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 & -m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\begin{aligned} x_{i+2} &= x_i - m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) y_i \\ y_{i+2} &= (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) x_i + y_i \end{aligned}$$

$$\begin{aligned} x_{i+2} &= x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i \\ y_{i+2} &= y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i \\ z_{i+2} &= z_i - \sigma_i \alpha_{m,i} - \sigma_{i+1} \alpha_{m,i+1} \end{aligned}$$

$\sigma_i = 0$ \rightarrow inc/dec no rotation,
but compensate the scale factor.

$$\begin{aligned} x_{i+1} &= x_i + m \cdot 2^{-2i-1} x_i && \text{inc/dec} \\ y_{i+1} &= y_i + m \cdot 2^{-2i-1} y_i && \text{inc/dec} \\ z_{i+1} &= z_i && m=+1/m=-1 \end{aligned}$$

$$\begin{aligned} x_{i+1} &= \left(1 + m \cdot 2^{-2i-1} \right) x_i \\ y_{i+1} &= \left(1 + m \cdot 2^{-2i-1} \right) y_i \\ z_{i+1} &= z_i \end{aligned}$$

$$\begin{aligned} x_{i+2} &= \left(1 + m \cdot 2^{-2i-3} \right) x_{i+1} = \left(1 + m \cdot 2^{-2i-3} \right) \left(1 + m \cdot 2^{-2i-1} \right) x_i \\ y_{i+2} &= \left(1 + m \cdot 2^{-2i-3} \right) y_{i+1} = \left(1 + m \cdot 2^{-2i-3} \right) \left(1 + m \cdot 2^{-2i-1} \right) y_i \\ z_{i+1} &= z_i \end{aligned}$$

$$2^{-2i-3} \cdot 2^{-2i-1} = 2^{-4i-4} \ll 1$$

$$\begin{aligned} x_{i+2} &= \left(1 + m \cdot 2^{-2i-3} + m \cdot 2^{-2i-1} \right) x_i \\ y_{i+2} &= \left(1 + m \cdot 2^{-2i-3} + m \cdot 2^{-2i-1} \right) y_i \\ z_{i+1} &= z_i \end{aligned}$$

$$m=1, S(m, i) = i$$

Cond (I) $0 \leq i \leq \frac{1}{4}(n-3)$

$$\begin{aligned} x_{i+1} &= x_i - \sigma_i 2^{-i} y_i \\ y_{i+1} &= y_i + \sigma_i 2^{-i} x_i \\ z_{i+1} &= z_i - \sigma_i \tan^{-1}(2^{-i}) \end{aligned}$$

$$\begin{aligned} x_{i+1} &= x_i - \sigma_i 2^{-i} y_i \\ y_{i+1} &= y_i + \sigma_i 2^{-i} x_i \\ z_{i+1} &= z_i - \sigma_i \tan^{-1}(2^{-i}) \end{aligned}$$

Cond (II) $\frac{1}{4}(n-3) < i \leq \frac{1}{2}(n+1)$

$\sigma_i \neq 0$

$$\begin{aligned} x_{i+1} &= x_i - \sigma_i 2^{-i} y_i \\ y_{i+1} &= y_i + \sigma_i 2^{-i} x_i \\ z_{i+1} &= z_i - \sigma_i \tan^{-1}(2^{-i}) \end{aligned}$$

$$\begin{aligned} x_{i+2} &= x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i \\ y_{i+2} &= y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i \\ z_{i+2} &= z_i - \sigma_i \alpha_{m,i} - \sigma_{i+1} \alpha_{m,i+1} \end{aligned}$$

$\sigma_i = 0$

$$\begin{aligned} x_{i+1} &= x_i + m \cdot 2^{-2i-1} x_i \\ y_{i+1} &= y_i + m \cdot 2^{-2i-1} y_i \\ z_{i+1} &= z_i \end{aligned}$$

$$\begin{aligned} x_{i+2} &= (1 + m \cdot 2^{-2i-3} + m \cdot 2^{-2i-1}) x_i \\ y_{i+2} &= (1 + m \cdot 2^{-2i-3} + m \cdot 2^{-2i-1}) y_i \\ z_{i+1} &= z_i \end{aligned}$$

Cond (III) $\frac{1}{2}(n+1) < i$

$\sigma_i \neq 0$

$$\begin{aligned} x_{i+1} &= x_i - \sigma_i 2^{-i} y_i \\ y_{i+1} &= y_i + \sigma_i 2^{-i} x_i \\ z_{i+1} &= z_i - \sigma_i \tan^{-1}(2^{-i}) \end{aligned}$$

$$\begin{aligned} x_{i+2} &= x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i \\ y_{i+2} &= y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i \\ z_{i+2} &= z_i - \sigma_i \alpha_{m,i} - \sigma_{i+1} \alpha_{m,i+1} \end{aligned}$$

$\sigma_i = 0$

$$\begin{aligned} x_{i+1} &= x_i \\ y_{i+1} &= y_i \\ z_{i+1} &= z_i \end{aligned}$$

- $0 \leq i \leq (n-3)/4$:
 - use the prediction algorithm, generate σ_i from z_i using Table 1 in the same manner as in Fig. 1, $\sigma_i \in \{-1,1\}$, execute iterations according to Eqs. 1-3
- $(n-3)/4 < i \leq (n+1)/2$:
 - use the prediction algorithm, generate σ_i from z_i by special recoding (explained later), $\sigma_i \in \{0,1,-1\}$, after each iteration increment i by 2
 - $\sigma_i \neq 0$:
 - $x_{i+2} = x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i$
 - $y_{i+2} = y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i$
 - $z_{i+2} = z_i - \sigma_i \alpha_{m,i} - \sigma_{i+1} \alpha_{m,i+1}$
 - $\sigma_i = 0$:
 - $x_{i+2} = x_i + m 2^{-2i-1} x_i + m 2^{-2i-2} x_i$ (11)
 - $y_{i+2} = y_i + m 2^{-2i-1} y_i + m 2^{-2i-2} y_i$ (12)
 - $z_{i+2} = z_i$ (13)

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & -m \sigma_i 2^{-i} \\ \sigma_i 2^{-i} & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 - m \boxed{\sigma_i \sigma_{i+1}} 2^{-2i-1} & -m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 - m \boxed{\sigma_i \sigma_{i+1}} 2^{-2i-1} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\boxed{\sigma_i = 0} \rightarrow \boxed{\sigma_i \sigma_{i+1} = 0}$$

$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 & -m(\cancel{\sigma_i} 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\cancel{\sigma_i} 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\begin{aligned} x_{i+2} &= x_i - m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) y_i \\ y_{i+2} &= (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) x_i + y_i \end{aligned}$$

$$\begin{aligned} x_{i+2} &= x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i \\ y_{i+2} &= y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i \end{aligned}$$

- $0 \leq i \leq (n-3)/4$:

use the prediction algorithm, generate σ_i from z_i using Table 1 in the same manner as in Fig. 1, $\sigma_i \in \{-1, 1\}$, execute iterations according to Eqs. 1-3

- $(n-3)/4 < i \leq (n+1)/2$:

use the prediction algorithm, generate σ_i from z_i by special recoding (explained later), $\sigma_i \in \{0, 1, -1\}$, after each iteration increment i by 2

$$\sigma_i \neq 0: \quad x_{i+2} = x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i$$

$$y_{i+2} = y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i$$

$$z_{i+2} = z_i - \sigma_i \alpha_{m,i} - \sigma_{i+1} \alpha_{m,i+1}$$

$$\sigma_i = 0: \quad x_{i+2} = x_i + m 2^{-2i-1} x_i + m 2^{-2i-2} x_i \quad (11)$$

$$y_{i+2} = y_i + m 2^{-2i-1} y_i + m 2^{-2i-2} y_i \quad (12)$$

$$z_{i+2} = z_i \quad (13)$$

$$\lambda(-1) = 0$$

$$\lambda(1) = 0$$

$$\lambda(0) = 1$$

$$\lambda(\sigma_i) = 0 \quad \sigma_i \in \{-1, 1\}$$

$$= 1 \quad \sigma_i = 0$$

Modified CORDIC

Timmermann 1989 Electronics Letters

$$x_n = k_m \{ x_0 \cos[\sqrt{(m)} \alpha] - \sqrt{(m)} y_0 \sin[\sqrt{(m)} \alpha] \}$$

$$y_n = k_m \{ 1/\sqrt{(m)} x_0 \sin[\sqrt{(m)} \alpha] + y_0 \cos[\sqrt{(m)} \alpha] \}$$

$$z_n = z_0 + \alpha$$

k_m : the scaling factor

m : the coordinate system (0, 1, +1)

α : the rotation angle

$\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix}$: the initial values depends on the iteration goal

Data dependency across iteration

→ CSA no benefit

1st half iterations : the most significant contribution

$$\text{the rotation angle } \alpha_i = \frac{1}{\sqrt{(m)}} \tan^{-1} [\sqrt{(m)} 2^{-s(m, i)}]$$

$s(m, i)$ the iteration shift values

α_i decreases with the increasing
iteration index i

2nd half iterations : can improve the accuracy
only by one bit each

rotation $z_n \rightarrow 0$

$$\begin{aligned}x_n &= k_m \{ x_0 \cos[\sqrt{(m)} \alpha] - \sqrt{(m)} y_0 \sin[\sqrt{(m)} \alpha] \} \\y_n &= k_m \{ 1/\sqrt{(m)} x_0 \sin[\sqrt{(m)} \alpha] + y_0 \cos[\sqrt{(m)} \alpha] \} \\z_n &= z_0 + \alpha\end{aligned}$$

Vectoring $y_n \rightarrow 0$

$$\begin{aligned}x_n &= k_m \sqrt{x_0^2 + m y_0^2} \\z_n &= z_0 + 1/\sqrt{(m)} \tan^{-1} [\sqrt{(m)} y_0 / x_0]\end{aligned}$$

2nd half iterations : can improve the accuracy
only by one bit each

replace these iterations by a single rotation
after the remaining rotation angle
has been reduced using a fixed number of
pure CORDIC iterations

this truncation reduces the latency time and saves area
although the truncation requires extra hardware

the necessary minimum number of iterations

Rotation mode ($z \rightarrow 0$)

After j CORDIC rotations have been performed
the z path contains the remaining rotation angle z_j

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \cos [\sqrt{m} z_j] & -\sqrt{m} \sin [\sqrt{m} z_j] \\ 1/\sqrt{m} \sin [\sqrt{m} z_j] & \cos [\sqrt{m} z_j] \end{bmatrix} \begin{bmatrix} x_j \\ y_j \end{bmatrix}$$

$$x_n = k_m \{ x_0 \cos [\sqrt{m} \alpha] - \sqrt{m} y_0 \sin [\sqrt{m} \alpha] \}$$

$$y_n = k_m \{ 1/\sqrt{m} x_0 \sin [\sqrt{m} \alpha] + y_0 \cos [\sqrt{m} \alpha] \}$$

$$z_n = z_0 + \alpha$$

assume $k_m = 1$

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \cos [\sqrt{(m)} \varepsilon_j] & -\sqrt{(m)} \sin [\sqrt{(m)} \varepsilon_j] \\ 1/\sqrt{(m)} \sin [\sqrt{(m)} \varepsilon_j] & \cos [\sqrt{(m)} \varepsilon_j] \end{bmatrix} \begin{bmatrix} x_j \\ y_j \end{bmatrix}$$

Taylor Series expansions to $\sin \theta$, $\cos \theta$
take only the first terms

$$\sin \theta = x - \frac{1}{3!} x^3 + \frac{1}{5!} x^5 - \frac{1}{7!} x^7 + \dots$$

$$\cos \theta = 1 - \frac{1}{2!} x^2 + \frac{1}{4!} x^4 - \frac{1}{6!} x^6 + \dots$$

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & -\sqrt{(m)} \cdot \sqrt{(m)} \varepsilon_j \\ 1/\sqrt{(m)} \cdot \sqrt{(m)} \varepsilon_j & 1 \end{bmatrix} \begin{bmatrix} x_j \\ y_j \end{bmatrix}$$

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & -m \varepsilon_j \\ \varepsilon_j & 1 \end{bmatrix} \begin{bmatrix} x_j \\ y_j \end{bmatrix}$$

for a sufficiently small ε_j

the required precision of n -bit
the upper limit on the maximal remainder

$$\varepsilon_j \leq \frac{1}{\sqrt{(m)}} \tan^{-1} [\sqrt{(m)} 2^{-j+1}]$$

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} 1 & -m \varepsilon_j \\ \varepsilon_j & 1 \end{bmatrix} \begin{bmatrix} x_j \\ y_j \end{bmatrix}$$

for a sufficiently small ε_j

the required precision of n -bit
the upper limit on the maximal remainder

$$\frac{1}{2} \varepsilon_j^2 \leq 2^{-n} \quad \varepsilon_j^2 \leq 2^{-n+1} \quad \varepsilon_j \leq 2^{-\frac{n+1}{2}}$$

$$\varepsilon_j \leq \frac{1}{\sqrt{m}} \tan^{-1} [\sqrt{m} 2^{-j+1}]$$

$$j > \frac{n+1}{2}$$

Rotation mode

$$\begin{aligned}x_n &= x_j - m z_j y_j & (j > (n+1)/2) \\y_n &= z_j x_j + y_j & (j > (n+1)/2)\end{aligned}$$

Vectoring mode

$$\begin{aligned}x_n &= z_j & j > (n+1)/2 \\z_n &= z_j + y_j/x_j & j > (n/3) + 0.4n2\end{aligned}$$

the prediction algorithm : rotation mode (OK)
vectoring mode (X)

2nd half of the n iterations in rotation mode
~ replaced by 2 multiplications in parallel

A fully parallel n -bit Wallace tree multiplier : $2 \log_2(n)$ FA time unit

prediction + termination.

the Truncated. CORDIC Algorithm

- reduces the number of CORDIC iterations

- Multiplication / division hardware

Booth Technique halves the amount of partial products
Carry Save Architecture

$k_m \neq 1 \Rightarrow$ multiplication \Rightarrow multiplier anyway

Modified Booth Encoding

$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} & -m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 & -m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$x_{i+2} = x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i$$

$$y_{i+2} = y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i$$

$$x_{i+2} = (x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i)$$

$$(1 + \lambda(\sigma_i) m 2^{-2i-1} x_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} x_i)$$

$$y_{i+2} = (y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i)$$

$$(1 + \lambda(\sigma_i) m 2^{-2i-1} y_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} y_i)$$

$$\begin{bmatrix} x_{i+2} \\ y_{i+2} \end{bmatrix} = \begin{bmatrix} 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} & -m(\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) \\ (\sigma_i 2^{-i} + \sigma_{i+1} 2^{-i-1}) & 1 - m \sigma_i \sigma_{i+1} 2^{-2i-1} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix}$$

$$x_{i+2} = (x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i) - m \sigma_i \sigma_{i+1} 2^{-2i-1} x_i$$

$$y_{i+2} = (y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i) - m \sigma_i \sigma_{i+1} 2^{-2i-1} y_i$$

$$x_{i+2} = (x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i) (1 + \lambda(\sigma_i) m 2^{-2i-1} x_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} x_i)$$

$$y_{i+2} = (y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i) (1 + \lambda(\sigma_i) m 2^{-2i-1} y_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} y_i)$$

$$\lambda(\sigma_i) = 1 \quad \text{for } |\sigma_i| = 0 \quad \{0\}$$

$$\lambda(\sigma_i) = 0 \quad \text{for } |\sigma_i| = 1 \quad \{1, \bar{1}\}$$

$$\lambda(\sigma_{i+1}) = 1 \quad \text{for } |\sigma_{i+1}| = 0 \quad \{0\}$$

$$\lambda(\sigma_{i+1}) = 0 \quad \text{for } |\sigma_{i+1}| = 1 \quad \{1, \bar{1}\}$$

$$x_{i+2} = (x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i) (1 + \lambda(\sigma_i) m 2^{-2i-1} x_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} x_i)$$

$$y_{i+2} = (y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i) (1 + \lambda(\sigma_i) m 2^{-2i-1} y_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} y_i)$$

rotation by $\alpha_{m,i}$ or $\alpha_{m,i+1}$

multiplex the diff. shifts

⇒ 4-to-2 cells x_{i+2}, y_{i+2}

3-to-2 cells z_{i+2}

Timmermann's constant scaling factor

for n-bit precision.

parallelizable

late evaluation

after all iterations

Wallace Tree

σ_i 's are recoded in parallel
 # of non-zero σ_i 's at most half of max value w/o recoding

$$\sigma_i \sigma_{i+1} = 0$$

$$\sigma_i \sigma_{i+1}$$

case ① 0 1

case ② 1 0

case ③ 0 0

$$(1 + m 2^{-2i-1})(1 + m 2^{-2i-3}) = 1 + m 2^{-2i-1} + m 2^{-2i-3}$$

$$\prod_{j=0}^n (1 + m 2^{-2i-2j-1}) = 1 + \sum_{j=0}^n m 2^{-2i-2j-1}$$

$$\begin{aligned} & (1 + m 2^{-2i-0-1})(1 + m 2^{-2i-2-1})(1 + m 2^{-2i-4-1})(1 + m 2^{-2i-6-1}) \dots \\ = & 1 + m 2^{-2i-0-1} + m 2^{-2i-2-1} + m 2^{-2i-4-1} + m 2^{-2i-6-1} \dots \end{aligned}$$

Modified Booth Encoding

$$\lambda(t) = 1 \quad \text{for } |t| = 0$$

$$\lambda(t) = 0 \quad \text{for } |t| = 1 \quad \{1, \bar{1}\}$$

$$\begin{aligned}x_{i+2} &= (x_i - m \sigma_i 2^{-i} y_i - m \sigma_{i+1} 2^{-i-1} y_i) \\ &= (1 + \lambda(\sigma_i) m 2^{-2i-1} x_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} x_i)\end{aligned}$$

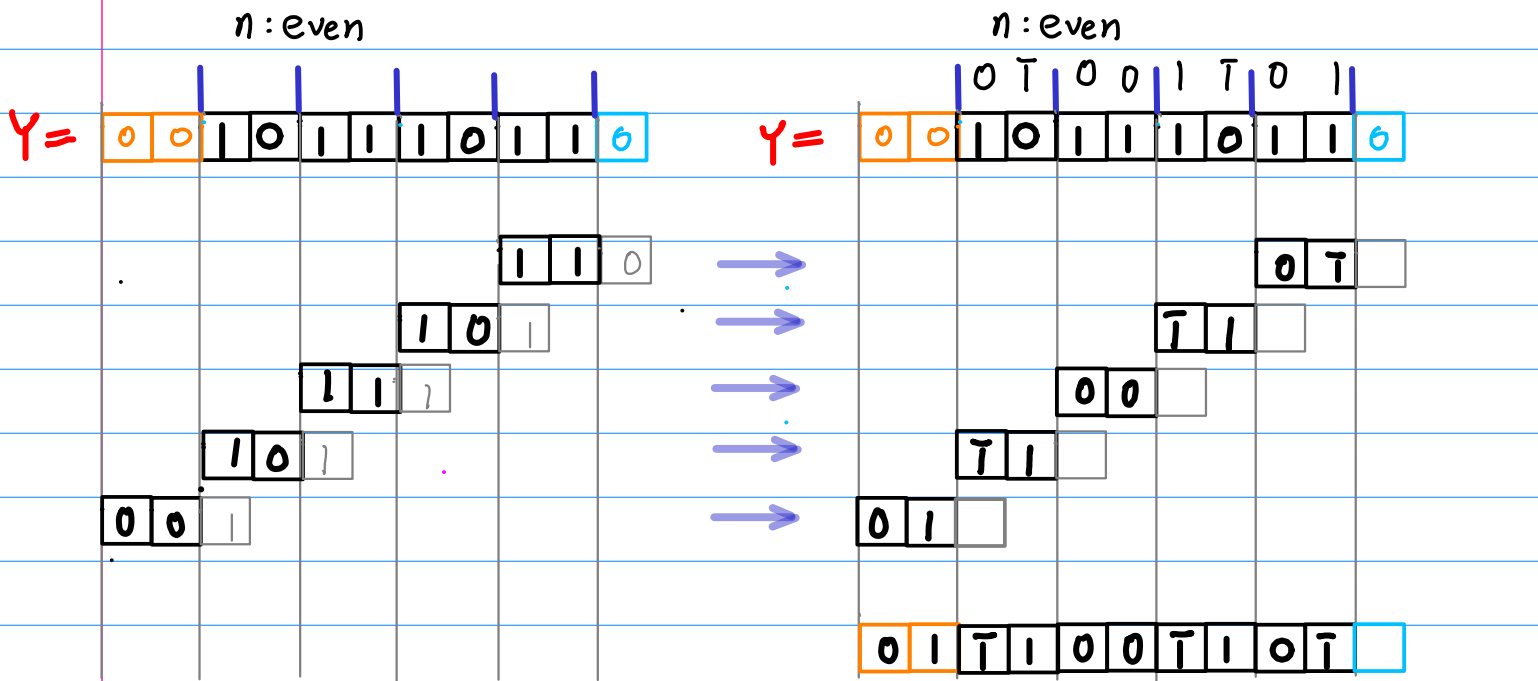
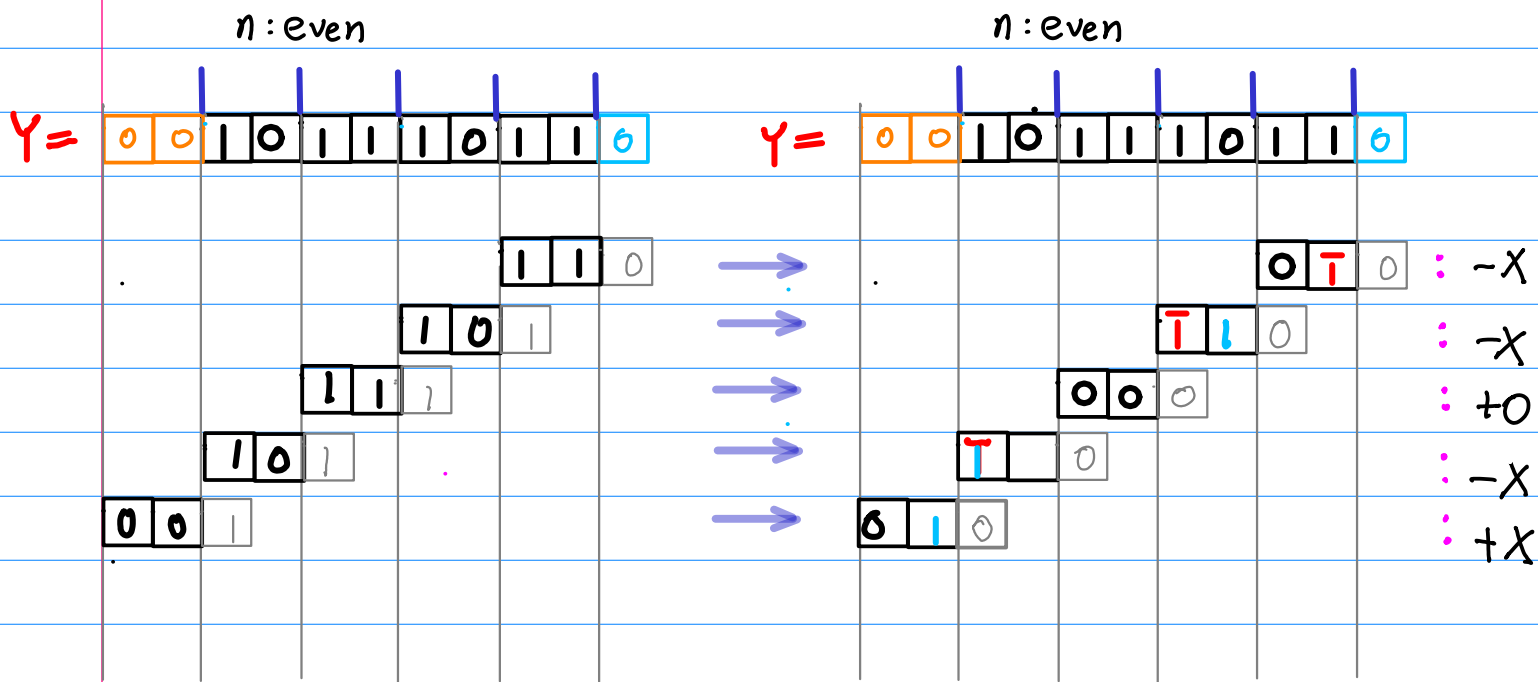
$$\begin{aligned}y_{i+2} &= (y_i + \sigma_i 2^{-i} x_i + \sigma_{i+1} 2^{-i-1} x_i) \\ &= (1 + \lambda(\sigma_i) m 2^{-2i-1} y_i + \lambda(\sigma_{i+1}) m 2^{-2i-3} y_i)\end{aligned}$$

2-bit encoding

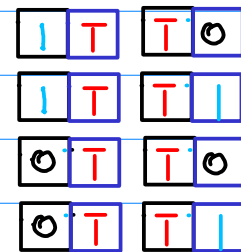
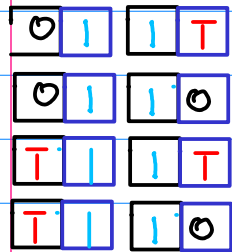
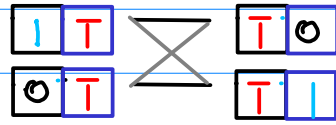
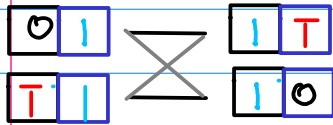
		2^1 2^0 \downarrow \downarrow	scale factor	
all zero's	0 0 0	→ 0 0 0	$0 \cdot 2 + 0 = +0$	+0
end of 1's	0 0 1	→ 0 1 0	$0 \cdot 2 + 1 = +1$	+X
isolated 1	0 1 0	→ 1 T 0	$1 \cdot 2 + T = +1$	+X
end of 1's	0 1 1	→ 1 0 0	$1 \cdot 2 + 0 = +2$	+2X
start of 1's	1 0 0	→ T 0 0	$T \cdot 2 + 0 = -2$	-2X
isolated 0	1 0 1	→ T 1 0	$T \cdot 2 + 1 = -1$	-X
start of 1's	1 1 0	→ 0 T 0	$0 \cdot 2 + T = -1$	-X
all 1's	1 1 1	→ 0 0 0	$0 \cdot 2 + 0 = 0$	+0

Scale factor { 0, ±1, ±2 }

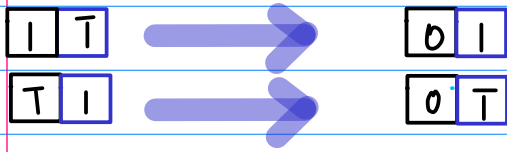
Original Booth Encoding



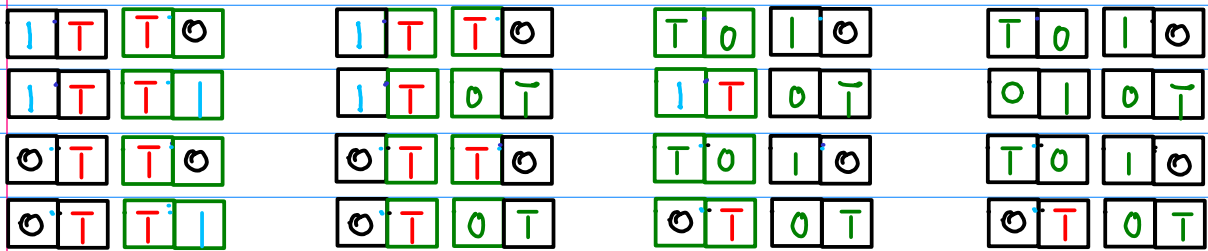
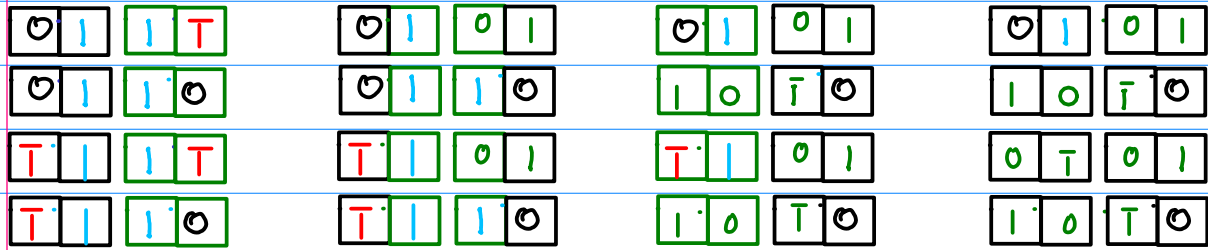
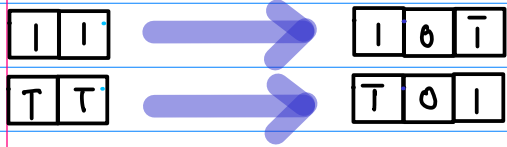
After Pass 1



Pass 2 Operation



iterative application



2 Pass

$\gamma =$

0	0	1	0	1	1	1	0	1	1	0
---	---	---	---	---	---	---	---	---	---	---

$\sigma_i \sigma_{i+1} \neq 0$

0	1	$\bar{1}$	1	0	0	$\bar{1}$	1	0	$\bar{1}$	
---	---	-----------	---	---	---	-----------	---	---	-----------	--

0	$\bar{1}$
---	-----------

0	1	$\bar{1}$	1	0	0	$\bar{1}$	1	0	$\bar{1}$	
---	---	-----------	---	---	---	-----------	---	---	-----------	--

1	0
---	---

0	1	$\bar{1}$	1	0	0	$\bar{1}$	1	0	$\bar{1}$	
---	---	-----------	---	---	---	-----------	---	---	-----------	--

$\bar{1}$	1
-----------	---

0	1	$\bar{1}$	1	0	0	0	$\bar{1}$	0	$\bar{1}$	
---	---	-----------	---	---	---	---	-----------	---	-----------	--

0	0
---	---

0	1	$\bar{1}$	1	0	0	0	$\bar{1}$	0	$\bar{1}$	
---	---	-----------	---	---	---	---	-----------	---	-----------	--

0	0
---	---

0	1	$\bar{1}$	1	0	0	0	$\bar{1}$	0	$\bar{1}$	
---	---	-----------	---	---	---	---	-----------	---	-----------	--

1	0
---	---

0	1	$\bar{1}$	1	0	0	0	$\bar{1}$	0	$\bar{1}$	
---	---	-----------	---	---	---	---	-----------	---	-----------	--

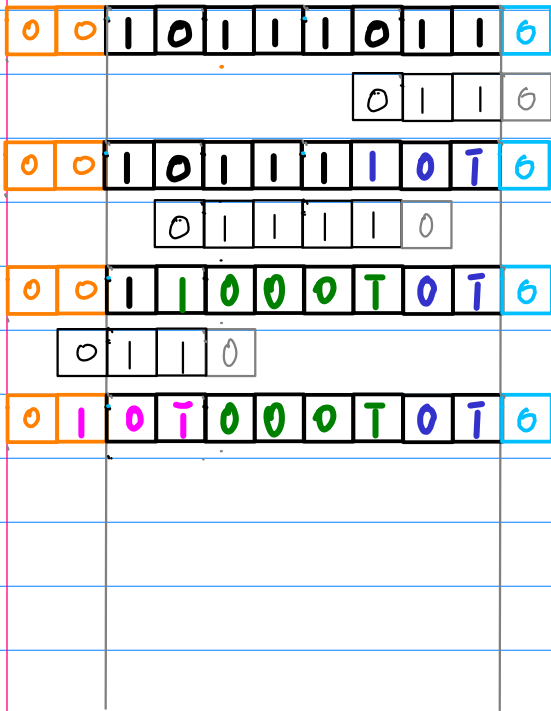
$\bar{1}$	
-----------	--

0	1	0	$\bar{1}$	0	0	0	$\bar{1}$	0	$\bar{1}$	
---	---	---	-----------	---	---	---	-----------	---	-----------	--

1	0
---	---

0	1	0	$\bar{1}$	0	0	0	$\bar{1}$	0	$\bar{1}$	
---	---	---	-----------	---	---	---	-----------	---	-----------	--

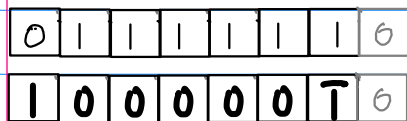
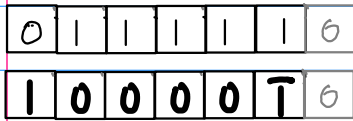
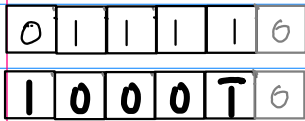
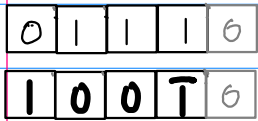
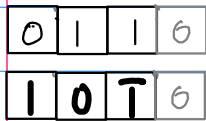
$\sigma_i \sigma_{i+1} = 0$



Iterative Reduction of 1-runs

$$\sigma_i \sigma_{i+1} = 0$$

Unique encoding



Verification

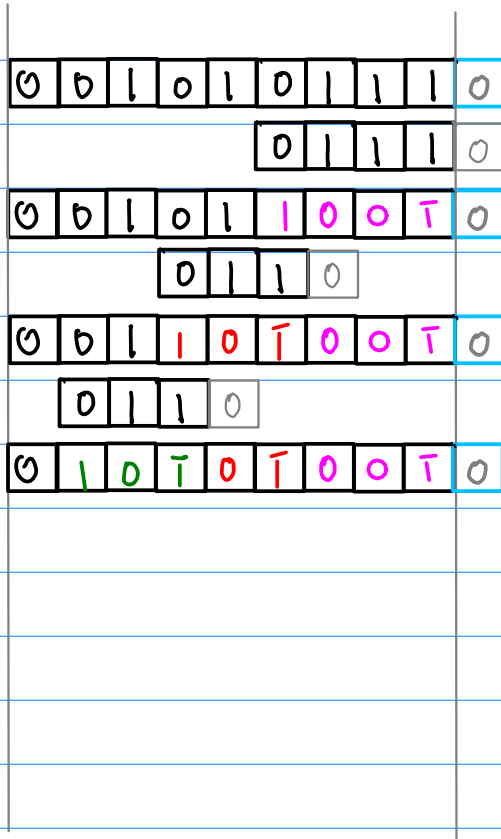
	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0			
	0	0	1	0	1	1	1	0	1	1	6
	0	1	$\bar{1}$	1	0	0	$\bar{1}$	1	0	$\bar{1}$	
	0	1	0	$\bar{1}$	0	0	0	$\bar{1}$	0	$\bar{1}$	
	0	1	0	$\bar{1}$	0	0	0	$\bar{1}$	0	$\bar{1}$	6

$$2^9 + 2^5 + 2^4 + 2^3 + 2^1 + 2^0 = 128 + 32 + 16 + 8 + 2 + 1 = 187$$

$$2^8 - 2^7 + 2^4 - 2^3 + 2^2 - 2^0 = 256 - 128 + 64 - 8 + 4 - 1 = 187$$

$$2^8 - 2^6 - 2^2 - 2^0 = 256 - 64 - 4 - 1 = 187$$

$$2^8 - 2^6 - 2^2 - 2^0 = 256 - 64 - 4 - 1 = 187$$



Canonical Signed Digit

(CSD)

