Calculations for Redundant Floating-point Decimal CORDIC Algorithm *

Álvaro Vázquez, Julio Villalba**, Elisardo Antelo and Emilio L. Zapata**

University of Santiago, SPAIN

**University of Málaga, SPAIN

alvaro.vazquez@usc.es, jvillalba@uma.es, elisardo.antelo@usc.es,zapata@uma.es

Abstract

This report includes some calculations used as additional material of the paper "Redundant Floating-point Decimal CORDIC Algorithm".

1 Minimum overlap and number of fractional digits

The overlap between angle i and the addition of the remaining angles plus the bound of the final error is:

$$V[i] = \left(\sum_{j=i+1}^{4m} \alpha_{j,1} + \alpha_{4m,1} \right) - \alpha_{i,1}$$

To obtain the number of bits of the estimation we need a lower bound of the scaled overlap, that is

$$10^{-t} \leq \min(10^{\lceil \frac{i}{4} \rceil} V[i])$$

*This work has been partially supported by the Ministry of Science and Innovation of Spain under projects TIN2007-67537-C03-01 and TIN2006-01078.

We have the following possible cases for the overlap:

- i mod 4 =1 ($i = 4k - 3$)

$$V[4k-3] = -\tan^{-1}(5\ 10^{-k}) + 2\ \tan^{-1}(2\ 10^{-k}) + \tan^{-1}(10^{-k}) + R[k]$$

- i mod 4 =2 ($i = 4k - 2$)

$$V[4k-2] = -\tan^{-1}(2\ 10^{-k}) + \tan^{-1}(2\ 10^{-k}) + \tan^{-1}(10^{-k}) + R[k]$$

- i mod 4 =3 ($i = 4k - 1$)

$$V[4k-1] = -\tan^{-1}(2\ 10^{-k}) + \tan^{-1}(10^{-k}) + R[k]$$

- i mod 4 =0 ($i = 4k$)

$$V[4k] = -\tan^{-1}(10^{-k}) + R[k]$$

with

$$R[k] = \left(\sum_{j=k+1}^{m} \tan^{-1}(5\ 10^{-j}) + 2\ \tan^{-1}(2\ 10^{-j}) + \tan^{-1}(10^{-j}) \right) + \alpha_{4m}$$

We use the properties $\tan^{-1}(u\ v\ 10^{-k}) < u \tan^{-1}(v\ 10^{-k})$ for $u \geq 1$ and $\tan^{-1}(u\ v\ 10^{-k}) > u \tan^{-1}(v\ 10^{-k})$ for $u \leq 1$, to demonstrate the following inequalities:

$$\tan^{-1}(2\ 10^{-k}) - \tan^{-1}(10^{-k}) < 2\tan^{-1}(10^{-k}) - \tan^{-1}(10^{-k}) = \tan^{-1}(10^{-k})$$

and

$$\tan^{-1}(5\ 10^{-k}) - 2\tan^{-1}(2\ 10^{-k}) - \tan^{-1}(10^{-k}) <$$

$$\tan^{-1}(5\ 10^{-k}) - 2\tan^{-1}(2\ 10^{-k}) - 0.5\ \tan^{-1}(2\ 10^{-k}) <$$

$$\tan^{-1}(5\ 10^{-k}) - 2.5\tan^{-1}(2\ 10^{-k}) < 0$$

Taking into account these inequalities results in

- i mod 4 =1

$$V[4k-3] - R[k+1] > 0$$

- i mod 4 =2

$$V[4k-2] - R[k+1] = \tan^{-1}(10^{-k})$$

- i mod 4 =3

$$V[4k-1] - R[k+1] > -\tan^{-1}(10^{-k})$$

- i mod 4 =0

$$V[4k] - R[k+1] = -\tan^{-1}(10^{-k})$$

Thus

$$\min(V[4k-3], V[4k-2], V[4k-1], V[4k]) = V[4k]$$

Therefore, the minimum overlap is for i mod 4 =0. We show below that among the possible values of i that verifies this condition, the worst case for convergence is for $i = 4m - 4$. (that is for $k = m - 1$). To demonstrate this, we take into account the following properties (true in our domain, that is $u\ 10^{-k} \leq 0.5$):

$$u\ 10^{-k} - \frac{1}{3}u^3\ 10^{-3k} < \tan^{-1}(u\ 10^{-k}) < u\ 10^{-k} - \frac{1}{4}u^3\ 10^{-3k}$$

Therefore, a bound for $V[4k]$ is

$$V[4k] \geq \sum_{j=k+1}^{m} (10\ 10^{-j} - \frac{142}{3}10^{-3j}) + 10^{-m} - \frac{1}{3}10^{-3m} - 10^{-k} + \frac{1}{4}10^{-3k}$$

This results in

$$V[4k] \geq \frac{1}{9}10^{-k} + \frac{2}{10}\ 10^{-3k} - \frac{1}{9}10^{-m} - \frac{1}{3}10^{-3m}$$

Note that for $k = m$ this bound gives a negative overlap of $-(2/15)10^{-3m}$ instead 0 due to the bounds used.

We use the above bound of $V[4k]$ to determine the number of truncation bits. Specifically, we need the scaled bound

$$10^k V[4k] \geq \frac{1}{9} + \frac{2}{10}\ 10^{-2k} - \frac{1}{9}10^{-m+k} - \frac{1}{3}10^{-3m+k}$$

The worst case is obtained for the minimum value of the bound. This minimum is achieved for the maximum value of k, i.e. $k = m$. However the case $k = m$ does not allow to obtain information for convergence, since this is in fact the last elementary rotation. The only constraint for this case is that the final error due to a wrong estimation of the sign in the last iteration should be within the bound of the final error. Therefore, to find t we use the bound of the overlap for $k = m - 1$.

Thus,

$$10^{-t} \leq 0.1 + \frac{599}{30}\ 10^{-2m} \leq 10^{m-1} V[4(m-1)] \leq \min(10^k V[4k])$$

2 Number of digits of the integer part

The number of digits of the integer part of the estimation is obtained from the upper bound of $|r[i]|$. We have the following possible cases:

- i mod 4 =1 ($i = 4k - 3$)

$$|r[i]| \leq 10^k \left(\tan^{-1}(5\ 10^{-k}) + 2\ \tan^{-1}(2\ 10^{-k}) + \tan^{-1}(10^{-k}) + R[k+1]\right)$$

- i mod 4 =2 ($i = 4k - 2$)

$$|r[i]| \leq 10^k \left(2\ \tan^{-1}(2\ 10^{-k}) + \tan^{-1}(10^{-k}) + R[k+1]\right)$$

- i mod 4 =3 ($i = 4k - 1$)

$$|r[i]| \leq 10^k \left(\tan^{-1}(2\ 10^{-k}) + \tan^{-1}(10^{-k}) + R[k+1]\right)$$

- i mod 4 =0 ($i = 4k$)

$$|r[i]| \leq 10^k \left(\tan^{-1}(10^{-k}) + R[k+1]\right)$$

The worst case is for i mod 4=1. A simple bound is obtained using the inequality $\tan^{-1}(u\ 10^{-k}) < u\ 10^{-k}$, which results in

$$|r[i]| < 10^k \left(10\ 10^{-k} + \left(\sum_{j=k+1}^{m} 10\ 10^{-j} \right) + 10^{-m} \right) < \frac{100}{9} - \frac{1}{9}\ 10^{k-m} < \frac{100}{9} = 11.111...$$

3 Convergence for hyperbolic vectoring mode

The residual angle for vectoring is bounded by

$$\frac{10^{-t}}{0.4369...}\ 10^{-\lceil \frac{i}{4} \rceil} + \frac{1}{2} \left(\frac{10^{-3t}}{(0.4369...)^3} \right)\ 10^{-3\lceil \frac{i}{4} \rceil} + \alpha_{i,-1}$$

Thus, for convergence in hyperbolic coordinates it is necessary that

$$\frac{10^{-t}}{0.4369...}\ 10^{-\lceil \frac{i}{4} \rceil} + \frac{1}{2} \left(\frac{10^{-3t}}{(0.4369...)^3} \right)\ 10^{-3\lceil \frac{i}{4} \rceil} + \alpha_{i,-1} \leq \left(\sum_{j=i+1}^{4m,-1} \alpha_{j,-1} + \alpha_{4m,-1} \right) \tag{1}$$

For $t = 1$ (the number of fractional digits used for the sign estimation in circular coordinates)

$$\frac{10^{-1}}{0.4369...}\ 10^{-\lceil \frac{i}{4} \rceil} + \frac{1}{2} \left(\frac{10^{-3}}{(0.4369...)^3} \right)\ 10^{-3\lceil \frac{i}{4} \rceil} < 0.25\ 10^{-\lceil \frac{i}{4} \rceil}$$

and then, expression (1) results in the following condition of convergence:

$$0.25 < 10^{\lceil \frac{i}{4} \rceil} V[i] = 10^{\lceil \frac{i}{4} \rceil} \left(\sum_{j=i+1}^{4m,-1} \alpha_{j,-1} + \alpha_{4m,-1} - \alpha_{i,-1} \right) \tag{2}$$

We have checked that, for hyperbolic coordinates, with the angles derived from the 5221 decimal code, it is not possible to assure the convergence of the algorithm with an estimation with one decimal digit, as it is done for circular coordinates.

The alternative, for hyperbolic coordinates, is to use angles derived from the decimal code 5421, which has more redundancy. Specifically, we show in following subsections that convergence is achieved using the following scheme for hyperbolic coordinates:

- To use angles derived from the code 5221 for $i \leq 4$, that is, to use the following sequence of angles for the leading four iterations: $\tanh^{-1}(5\ 10^{-1})$, $\tanh^{-1}(2\ 10^{-1})$, $\tanh^{-1}(2\ 10^{-1})$ and $\tanh^{-1}(10^{-1})$

- To use the code 5421 for $i > 4$, that is angles of the form $\tanh^{-1}(S[i])$, with $S[i] = C[i]\ 10^{-\lceil \frac{i}{4} \rceil}$, and $C[i] = R[i \bmod 4]$ with $R[0:3] = \{1, 5, 4, 2\}$

Convergence for the 5421 code and $i > 4$

$V[i]$ is the overlap between angle i and the addition of the remaining angles plus the bound of the final error:

$$V[i] = \left(\sum_{j=i+1}^{4m} \alpha_{j,-1} + \alpha_{4m,-1} \right) - \alpha_{i,-1}$$

We show below that the worst case for convergence (minimum overlap) for hyperbolic coordinates corresponds to i mod 4 =2.

We have the following possible cases for the overlap:

- i mod 4 =1 ($i = 4k - 3$)

$$V[4k-3] = -\tanh^{-1}(5\ 10^{-k}) + \tanh^{-1}(4\ 10^{-k}) + \tanh^{-1}(2\ 10^{-k}) + \tanh^{-1}(10^{-k}) + R[k] \quad (3)$$

- i mod 4 =2 ($i = 4k - 2$)

$$V[4k-2] = -\tanh^{-1}(4\ 10^{-k}) + \tanh^{-1}(2\ 10^{-k}) + \tanh^{-1}(10^{-k}) + R[k] \quad (4)$$

- i mod 4 =3 ($i = 4k - 1$)

$$V[4k-1] = -\tanh^{-1}(2\ 10^{-k}) + \tanh^{-1}(10^{-k}) + R[k] \quad (5)$$

- i mod 4 =0 ($i = 4k$)

$$V[4k] = -\tanh^{-1}(10^{-k}) + R[k] \quad (6)$$

with

$$R[k] = \left(\sum_{j=k+1}^{m} \tanh^{-1}(5\ 10^{-j}) + \tanh^{-1}(4\ 10^{-j}) + \tanh^{-1}(2\ 10^{-j}) + \tanh^{-1}(10^{-j}) \right) + \alpha_{4m,-1} \quad (7)$$

In what follows we use the properties:

$$\tanh^{-1}(u10^{-k}) > u\tanh^{-1}(10^{-k}) \quad if \quad u>1 \tag{8}$$

$$\tanh^{-1}(u10^{-k}) < u\tanh^{-1}(10^{-k}) \quad if \quad u<1 \tag{9}$$

$$\tanh^{-1}(u10^{-k}) > u10^{-k}+\frac{1}{3}u^3 10^{-3k} \tag{10}$$

$$\tanh^{-1}(u10^{-k}) < u10^{-k}+\frac{1}{2}u^3 10^{-3k} \tag{11}$$

to look for bounds for $V[4k-3]$, $V[4k-2]$, $V[4k-1]$ and $V[4]$:

- i mod 4 =1 ($i=4k-3$)

$$\begin{aligned}
V[4k-3]-R[k] &= \\
-\tanh^{-1}(5\ 10^{-k})+\tanh^{-1}(4\ 10^{-k})+\tanh^{-1}(2\ 10^{-k})+\tanh^{-1}(10^{-k}) &> (Eq.8)\\
-\tanh^{-1}(5\ 10^{-k})+4\tanh^{-1}(10^{-k})+2\tanh^{-1}(10^{-k})+\tanh^{-1}(10^{-k}) &= \\
-\tanh^{-1}(5\ 10^{-k})+7\tanh^{-1}(10^{-k}) &> (Eq.11\ \&\ 10)\\
-5\ 10^{-k}-\frac{1}{2}5^3 10^{-3k}+7\ 10^{-k}+\frac{7}{3}10^{-3k}=2\ 10^{-k}-\frac{361}{6}10^{-3k} &> 0\ (\ k\geq 2)
\end{aligned}$$

- i mod 4 =2 ($i=4k-2$)

$$\begin{aligned}
V[4k-2]-R[k] &= \\
-\tanh^{-1}(4\ 10^{-k})+\tanh^{-1}(2\ 10^{-k})+\tanh^{-1}(10^{-k}) &< (Eq.\ 9)\\
-\tanh^{-1}(4\ 10^{-k})+\frac{1}{2}\tanh^{-1}(4\ 10^{-k})+\frac{1}{4}\tanh^{-1}(4\ 10^{-k}) &= \\
-\frac{1}{4}\tanh^{-1}(4\ 10^{-k}) &< (Eq.\ 8)\\
-\tanh^{-1}(\ 10^{-k}) &
\end{aligned}$$

- i mod 4 =3 ($i=4k-1$)

$$\begin{aligned}
V[4k-1]-R[k] &= \\
-\tanh^{-1}(2\ 10^{-k})+\tanh^{-1}(10^{-k}) &< (Eq.\ 8)\\
-2\tanh^{-1}(10^{-k})+\tanh^{-1}(10^{-k})=-\tanh^{-1}(\ 10^{-k}) &
\end{aligned}$$

- i mod 4 =0 ($i = 4k$)

$$V[4k] - R[k] = -\tanh^{-1}(10^{-k})$$

Thus, the worst cases are $V[4k-2]$ and $V[4K-1]$. We subtract both expressions to find the smallest:

$$\begin{aligned} V[4k-2] - V[4k-1] = -\tanh^{-1}(4\ 10^{-k}) + 2\tanh^{-1}(2\ 10^{-k}) &< (Eq.\ 8) \\ -2\tanh^{-1}(2\ 10^{-k}) + 2\tanh^{-1}(2\ 10^{-k}) &= 0 \end{aligned}$$

Thus, $V[4k-2] < V[4k-1]$ and thus

$$\min(V[4k-3], V[4k-2], V[4k-1], V[4k]) = V[4k-2]$$

Therefore, the minimum overlap is for i mod $4 = 2$.

Now, we look for a lower bound for $V[4k-2]$, by obtaining a bound for the different terms of Equation (4).

$$\begin{aligned} -\tanh^{-1}(4\ 10^{-k}) + \tanh^{-1}(2\ 10^{-k}) + \tanh^{-1}(10^{-k}) &> (eq.\ 10\ \&\ 11) \\ -4\ 10^{-k} - \frac{1}{2}\ 4^3 10^{-3k} + 2\ 10^{-k} + \frac{1}{3}\ 2^3 10^{-3k} + 10^{-k} + \frac{1}{3}10^{-3k} & \qquad (12) \end{aligned}$$

For $R[k]$ we have (see Eq. (7) and (10)):

$$\begin{aligned} R[k] > \sum_{j=k+1}^{m} 5\ 10^{-k} + \frac{1}{3}\ 5^3 10^{-3k} + 4\ 10^{-k} + \frac{1}{3}\ 4^3 10^{-3k} + 2\ 10^{-k} + \frac{1}{3}\ 2^3 10^{-3k} + 10^{-k} + \frac{1}{3}10^{-3k} + \\ +10^{-m} + \frac{1}{3}10^{-3m} = \frac{4}{3}10^{-k} - \frac{4}{3}10^{-m} + \frac{22}{333}10^{-3k} + \frac{22}{333}10^{-3m} + 10^{-m} + \frac{1}{3}10^{-3m} \qquad (13) \end{aligned}$$

From (12) and (13) we have:

$$V[4k-2] > \frac{1}{3}10^{-k} - 28.94\ 10^{-3k} - \frac{1}{3}10^{-m} + 0.26\ 10^{-3m}$$

To determine the number of truncation digits we need the scaled overlap:

$$\begin{aligned} 10^k\ V[4k-2] > \frac{1}{3} - 28.94\ 10^{-2k} - \frac{1}{3}10^{k-m} + 0.26\ 10^{k-3m} > \\ \frac{1}{3} - \max(28.94\ 10^{-2k}) - \max(\frac{1}{3}10^{k-m}) \qquad (14) \end{aligned}$$

The highest contribution of terms $28.94\ 10^{-2k}$ and $\frac{1}{3}10^{k-m}$ is for $k = 2$ and $k = m - 1$ respectively[1]. Taking into account this, Expression (14) becomes:

$$10^k\ V[4k-2] > \frac{1}{3} - 28.94\ 10^{-4} - \frac{1}{3}10^{-1} = 0.297 \tag{15}$$

Therefore, the value of the overlap is higher than 0.25 (see (2)), which is the bound of the error in the angle, and then the convergence of the algorithm is assured.

Convergence for the 5221 and $i \leq 4$

For the case $k = 1$ and the angles derived from the 5221 code, the four bounds are

- $i = 1$

$$V[1] - R[1] = -\tanh^{-1}(5\ 10^{-1}) + 2\tanh^{-1}(2\ 10^{-1}) + \tanh^{-1}(10^{-1}) = -0.0435 \tag{16}$$

- $i = 2$

$$V[2] - R[1] = -\tanh^{-1}(2\ 10^{-1}) + \tanh^{-1}(2\ 10^{-1}) + \tanh^{-1}(10^{-1}) = 0.1003 \tag{17}$$

- $i = 3$

$$V[3] - R[1] = -\tanh^{-1}(2\ 10^{-1}) + \tanh^{-1}(10^{-1}) = -0.1024 \tag{18}$$

- $i = 4$

$$V[k] - R[1] = -\tanh^{-1}(10^{-1}) = -0.1003 \tag{19}$$

with

$$R[1] = \left(\sum_{j=2}^{m} \tanh^{-1}(5\ 10^{-j}) + \tanh^{-1}(4\ 10^{-j}) + \tanh^{-1}(2\ 10^{-j}) + \tanh^{-1}(10^{-j})\right) + \alpha_{4m,-1}$$

Thus,

$$\min(V[1], V[2], V[3], V[4]) = V[3]$$

[1]For $k = m$ there is not overlap, but the additional error, $0.25\ 10^{-m}$ can be accommodated with the other sources of error (truncation errors).

Therefore, the minimum overlap is for $i = 3$. Now we look for a lower bound of $V[3]$. From Eq. (18) and (13) we have:

$$V[3] = -\tanh^{-1}(2\ 10^{-1}) + \tanh^{-1}(10^{-1}) + R[1] >$$
$$-0,10239 + \frac{4}{30} - \frac{4}{3}10^{-m} + \frac{22}{333000} + \frac{22}{333}10^{-3m} + 10^{-m} + \frac{1}{3}10^{-3m}$$

For all practical purposes $m \geq 16$. Thus, tanking into account this value we conclude that:

$$V[3] > 0.031$$

The scaled overlap is

$$10\ V[3] > 0.31$$

The value of the overlap is higher than 0.25 (see (2)), which is the bound of the error in the angle, and then the convergence of the algorithm is assured also for this case.

4 High-Level Range Reduction Methods for Floating-Point

We consider the computation of the following transcendental functions: $\cos(F)$, $\sin(F)$, $\tan^{-1}(F/G)$, $\sinh(F)$, $\cosh(F)$, $\tanh^{-1}(F/G)$, e^F, 10^F, $\ln(F)$, $\log_{10}(F)$ and $\sqrt{F}$ where $F = S_A\ A\ 10^{E_A}$ and $G = S_B\ B\ 10^{E_B}$, with, S_A, S_B the sign bits, $A,\ B \in [1, 10)$ coded in BCD and $E_A,\ E_B$ the exponents. Although, according to the IEEE-754 2008 standard, the input operands may not be normalized, for transcendental functions the preferred exponent is the minimum possible, so a normalization stage is necessary.

For range reduction we consider the methods described in [1] [2]. The operations performed are dependent on the function computed:

[sin(F)/cos(F)]: the angle is decomposed as $A\ 10^{E_A} = N\ \pi/2 + z_{in}$, with N an integer, $z_{in} = M_{zin}\ 10^{-E_{zin}} \in [-\pi/4 - \gamma, \pi/4 + \gamma]$ and $\pi/4 + \gamma \leq 1.069....$ The parameter γ allows certain

redundancy that may simplify the range reduction implementation [1]. The functions are computed in the circular rotation mode with input arguments (no scale factor compensation is necessary since we use scaled input arguments) $x_{in} = K_1$, $y_{in} = 0.0$ and $z_{in} = M_{zin}\ 10^{-E_{zin}}$ with $|M_{zin}| \in [1, 10)$ and $E_{zin} \geq 0$. After computing the sine or cosine of z_{in} (the final result of the y or x iteration), the sine or cosine of the input angle may be obtained by simple trigonometric identities [1].

[tan^{-1}(F/G)]: if $F \geq G$ the algorithm computes $\tan^{-1}(G/F)$ and then by trigonometric identities $\tan^{-1}(F/G)$ is obtained. In this way we assure that the angle to be computed is within the range of convergence of the algorithm, which is larger than $\pi/4$. The function is computed in the circular vectoring mode with $(x_{in}, y_{in}) = (A, B\ 10^{-E_{yin}})$ if $F \geq G$ or $(B, A\ 10^{-E_{yin}})$ in other case, with $E_{yin} = |E_A - E_B| \geq 0$. The final result is obtained in the z coordinate.

[sinh(F)/cosh(F)]: the following decomposition is performed $S_A\ A\ 10^{E_A} = N \ln(10) + z_{in}$, with N an integer, $z_{in} = M_{zin}\ 10^{-E_{zin}} \in [-\ln(10)/2 - \gamma, \ln(10)/2 + \gamma]$, and $|\ln(10)/2 + \gamma| \leq 1.166...$ As before, γ provides some redundancy to simplify the range reduction. Following the method of [2], the functions are computed in the hyperbolic rotation mode with $(x_{in}, y_{in}) = (0.5\ (1 + 10^{-2N})\ K_{-1}, 0.5\ (1 - 10^{-2N})\ K_{-1})$ and $z_{in} = M_{zin}\ 10^{-E_{zin}}$ with $|M_{zin}| \in [1, 10)$ and $E_{zin} \geq 0$. The result is obtained in the x or y coordinate and does not require scale factor compensation (due to the initialization of the x and y input arguments already scaled). N should be added to the exponent of the result.

[e$^{\mathbf{F}}$]: the same range reduction as for sinh/cosh is performed. The function is computed in the hyperbolic rotation mode with $x_{in} = y_{in} = K_{-1}$ and $z_{in} = M_{zin}\ 10^{-E_{zin}}$. Since the initial values of x and y are the same, for the hyperbolic rotation the resultant final values of both coordinates are also the same (both the x and y iterations perform the same effective addition or subtraction operation). Therefore it is only necessary to implement one of the iterations, x or y.

[10$^{\mathbf{F}}$]: the following decomposition is performed $10^{S_A\ A\ 10^{E_A}} = 10^{N+r} = 10^N\ e^{r\ \ln(10)}$, with $-0.5 - \gamma/\ln(10) \leq r \leq 0.5 + \gamma/\ln(10)$ and $|0.5 + \gamma/\ln(10)| \leq 1.166...$ Then a base e exponential is computed.

[tanh^{-1}(F/G)]: The domain of the function is defined for $|F| < |G|$. Since $\tanh(1.166..) = 0.8229...$, we may perform the direct computation of the function for $|F|/|G| \leq 0.8229....$ We use the range reduction method proposed in [2]. For the cases i) $E_A \leq E_B - 2$, or ii) $A\ 10^{E_A - E_B} < 0.5\ B$ when $E_A = E_B$ or $E_A = E_B - 1$, the function is computed directly in the hyperbolic vectoring mode with $(x_{in}, y_{in}) = (B, A\ 10^{E_{yin}})$ with $E_{yin} = EA - EB$. For $A\ 10^{E_A - E_B} \geq 0.5\ B$ with $E_A - E_B = 0$ or -1 the following transformation is performed:

$$\tanh^{-1}\left(1 - \frac{(B-A)\ 10^{E_A - E_B}}{B}\right) = \tanh^{-1}\left(1 - \frac{S\ 10^{-E_s}}{B}\right) = \tanh^{-1}(T) + \frac{E_s}{2}\ \ln(10)$$

with $T = \frac{(B+A)-(B-A)\ 10^{E_s}}{(B+A)+(B-A)\ 10^{E_s}} = \frac{Y^*}{X^*}$. This transformation assures that $T < 0.8229....$ Then the function is computed in the hyperbolic vectoring mode with $(x_{in}, y_{in}) = (X^*, Y^*)$.

[ln(F)]: we use the transformation $\ln(A\ 10^{E_A}) = E_A\ \ln(10) + \ln(A)$. Then the following computation is performed: $\ln(A) = 2\ \tanh^{-1}((A-1)/(A+1))$. Since $A < 10$ we have that $(A-1)/(A+1) < 0.8229..$ and the function is computed directly in the hyperbolic vectoring mode with $(x_{in}, y_{in}) = (A+1, A-1) = (A+1, M_{yin}\ 10^{-E_{yin}})$ (note that $A \geq 1$ and that $A - 1$ may have leading zeros, which we express as a normalized significand and an exponent).

[log$_{10}$(F)]: the following transformation is used $\log_{10}(A\ 10^{E_A}) = \log_{10}(e)\ \ln(A) + E_A$. Then $\ln(A)$ is computed as in the previous case.

[Square root]: we compute $\sqrt{A}$ for even exponent, and $\sqrt{A/10}$ for odd exponent. The square root is computed using the hyperbolic vectoring mode that allows the computation of $\sqrt{x_{in}^2 - y_{in}^2}/K_{-1}$ (obtained in the final value of the x coordinate). Specifically, for even exponent we compute $\sqrt{(A + K_{-1}^2)^2 - (A - K_{-1}^2)^2}/K_{-1} = 2\sqrt{A}$ (the final result have to be multiplied by 0.5). To avoid an overflow in the x coordinate we use $(x_{in}, y_{in}) = (A, A)$, perform the first CORDIC iteration to obtain $(x[2], y[2])$ and add to $x[2]$ (subtract to $y[2]$) the constant correction term $K_{-1}^2\ (1 + 0.5\ K_{-1}^2)$. For odd exponent we use $(x_{in}, y_{in}) = ((A/10 + K_{-1}^2/4), (A/10 - K_{-1}^2/4))$, which results in $\sqrt{A}$ in the range $[\sqrt{0.1}, 1)$, so that a final decimal left shift is needed. The hyperbolic modulus is computed with the required accuracy in about half of the iterations required for the other functions. Therefore, about $2m$ hyperbolic rotations are necessary.

The most complex part of the range reduction is the accurate computation of the remainder given by the difference between the input argument and the highest integer multiple of a constant C ($C = \pi/2$ for the circular functions, and $C = \ln(10)$ for the hyperbolic functions) lower than the input argument. For a p–digit format (significand), this remainder should be computed with an accuracy of p digits. Methods used for binary floating point such as those presented in [3] [4], among others, might be adapted for decimal floating point.

References

[1] J.M. Muller, "Elementary Functions: Algorithms and Implementation". Birkhauser Verlag AG, second edition, 2007.

[2] H. Hahn, D. Timmermann, B.J. Hosticka, B. Rix, "A Unified and Division-Free CORDIC Argument Reduction Method with Unlimited Convergence Domain Including Inverse Hyperbolic Functions," IEEE Transactions on Computers, vol. 43, no. 11, pp. 1339-1344, Nov., 1994.

[3] M. Payne and R. Hanek, Radian Reduction for Trigonometric Functions, SIGNUM Newsletter, vol. 18, pp. 19-24, 1983.

[4] N. Brisebarre, D. Defour, P. Kornerup, J.-M.Muller and N. Revol, "A New Range-Reduction Algorithm," IEEE Transactions on Computers, vol. 54, no. 3, pp. 331-339, Mar. 2005