4.7 CORDIC for Phase Rotation

The phase compensations due to carrier frequency offset as well as ADC frequency offset are done using the CORDIC technique. The CORDIC algorithm takes in the real and imaginary values of a vector and rotates the vector by Φ radians. The rotation operation is performed iteratively, with the number of iterations denoted as n.

The CORDIC algorithm is described below:

Step 0: Let z = Φ;

Step 1: Compare $abs(z)$ with $> (\pi / 2))$

```
    if abs(z) > (π / 2))
            d = sign(z)
            x = - d * yc;
            y = d * xc;
            z = z - d * pi/2;
    else
            x = xc;
            y=yc;
    end
```

Step 2:

```
    Set A=1
    for i=0:n-1
            d = sign(z);
            x =x – y * d * 2 ^ (-i);
            y =y + x * d * 2 ^ (-i);
            z =z - d* atan(2 ^ (-i);
            A= A * sqrt(1 + 2 ^ (-2i));
    end
```

Step 3:

```
    [x y] = [x y] / A;
```

Values for the parameters used in the CORDIC algorithm are listed in Table 13.

i	(2 ^ i)	atan(1/s)	A
0	1	0.7854	1.4142
1	2	0.4636	1.5811
2	4	0.245	1.6298
3	8	0.1244	1.6425
4	16	0.0624	1.6457
5	32	0.0312	1.6465
6	64	0.0156	1.6467
7	128	0.0078	1.6467
8	256	0.0039	1.6468
9	512	0.002	1.6468
10	1024	0.001	1.6468
11	2048	0.0005	1.6468
12	4096	0.0002	1.6468

Table 13: Values for each iterative step of CORDIC algorithm

4.8 CORDIC for Phase Calculation

A variation on the CORDIC algorithm is used to obtain the phase of a complex number. Here, this is needed in the acquisition mode when coarse and fine frequency offset is estimated as well as in the tracking mode, when the phase of each pilot in each OFDM symbol is found.

This algorithm is very similar to the CORDIC technique that is used for phase rotation. It takes in the real and imaginary values of a vector. The rotation operation is performed iteratively, with the number of iterations denoted as n.

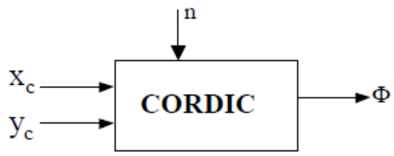

Figure 28: CORDIC algorithm

This CORDIC algorithm is described below. Va lues of the parameters are listed in Table 14

Step 0: Let z = 0;

Step 1: if $x_C < 0$

$$d_0 = sign(y_0)$$
$$x_0 = |y_C|$$
$$y_0 = -d_0 * x_0$$
$$z_0 = +d_0 * \frac{1}{2}$$

else

$$x_0 = x_C$$
$$y_0 = y_C$$
$$z_0 = 0$$

end

Step 2:

for $i = 0 : n-1$

$$d_{i+1} = sign(y_i)$$
$$x_{i+1} = x_i - d_{i+1} * y_i * 2^{-i}$$
$$y_{i+1} = y_i + d_{i+1} * x_i * 2^{-i}$$
$$z_{i+1} = z_i - d_0 * \frac{\arctan(2^{-i})}{\pi}$$

end

Step 3:

$$\Phi = \frac{1}{\pi} z_n$$

i	2^i	$\alpha_i = \frac{1}{\pi}\arctan(2^{-i})$	$round(2^{12} * \alpha_i)$
0	1	0.25000000000000	1024
1	2	0.14758361765043	605
2	4	0.07797913037737	319
3	8	0.03958342416057	162
4	16	0.01986852430554	81
5	32	0.00994394782359	41
6	64	0.00497318727895	20
7	128	0.00248674539367	10
8	256	0.00124339166871	5
9	512	0.00062169820592	3
10	1024	0.00031084939941	1

Table 14: Values for each iterative step of CORDIC algorithm

For implementation the CORDIC algorithm in a digital system, we usually need to assign at least n+2 bits for *arctan* lockup table.